**Session 1 Solutions**

1) There are lots of automated tools that can discover URLs on a website by testing common URLs and seeing if they return a certain HTTP status code. For example, you might predict that Juice Shop has a login page at *juice-shop.herokuapp.com/login* – if you made a HTTP request to this page and it returned a 200 (OK) status, you would know the page exists!

There are lots of tools that can automatically visit hundreds of these URLs per second, using a list of common URLs. One of our favourite tools for finding these is gobuster [1]. To run gobuster against a common list of URLs, targeting a site at IP *1.2.3.4*, run the following command:

$ gobuster dir -u http://1.2.3.4 -w /path/to/wordlist

Where /path/to/wordlist is the location of your list of URLs. A good wordlist to use is the one stored at /usr/share/wordlists/dirb/common.txt on Kali Linux!

Note: make sure you only do this against a TryHackMe instance of Juice Shop, or a locally hosted one – OWASP advise against running automated tools against the official Heroku instance.

2) In Developer tools (activate these by pressing F12 or Ctrl + Shift + I) you can take a look around the site. In older versions of Juice Shop there is a style attribute that sets the display to ‘none’, hiding the button – find this style option, and simply delete it! The button should now appear and you can click it to navigate to the scoreboard.

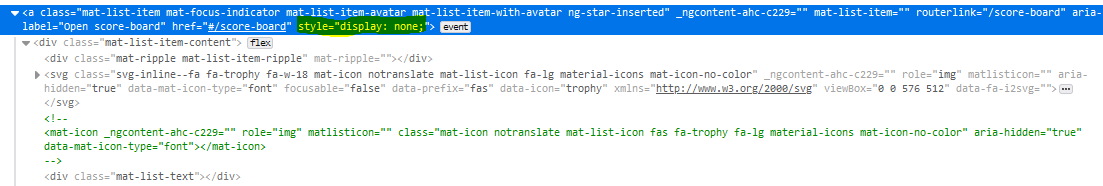


Figure : Displaying the style attribute that hides the scoreboard button

3) In a lot of ethical hacking situations, often the simplest solution to guessing a password is to… well, guess! You will often be able to gain a foothold onto a system by trying common passwords, such as ‘*letmein*’ and ‘*Password123*’, or default credentials (such as *tomcat:tomcat* on Tomcat servers, or blank authentication when authenticating over SMB – two situations you may come across often).

In this case, the password to the [admin@juice-sh.op](mailto:admin@juice-sh.op) account is ‘*admin123*’. In the session we covered two other ways to log into the admin account – by basic SQL Injection, and by stealing the hash from the user token. But guessing is by far the easiest way!

If you’re curious about attacks against common credentials, we will be briefly covering the art of password cracking in our ‘Automation in Python’ session on the 26th October. Attacks such as this often make use of long wordlists, such as the ones published by danielmiessler [2]. There are also several tools built for ‘password spraying’ – you can watch a demo by Ippsec [3], and learn how to defend against password sprays with an article from the NCSC [4].

4.1) First, log in as any user. Then open your developer tools and go to the ‘Storage’ tab. Under ‘Cookies’ in the sidebar the URL for Juice Shop should be listed. Click this to view all cookies, including the token.

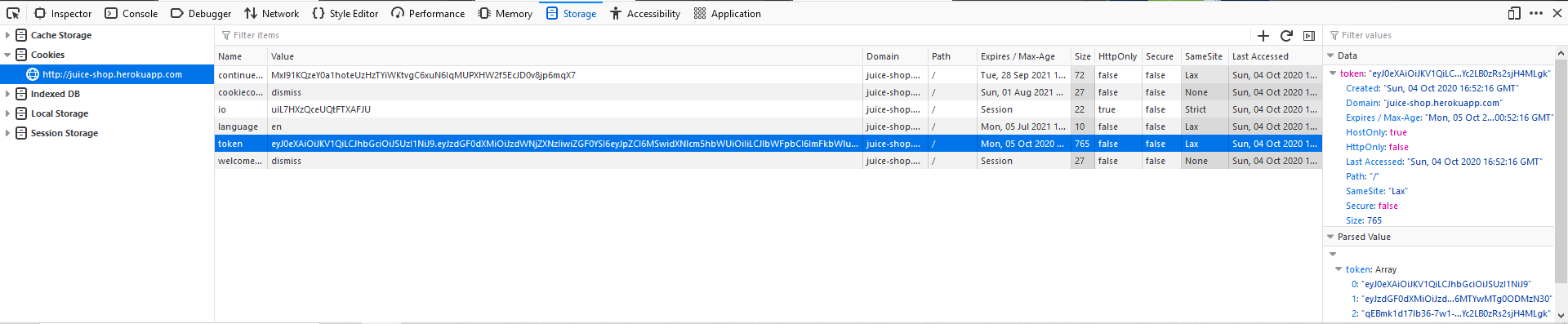


Figure 2: Viewing the user token

4.2) First, go to the registration page and open your developer tools. Go to the Network tab, which shows all the requests being sent by your browser. Then fill out the form as normal and send it. The Network tab should show a request to */api/Users*, which is the Application Programming Interface (API) that handles new user registration. Right-click on this request and press *‘Edit and Resend’*. You should now be able to add the *“role”:“admin”* parameter, as we did in session with Burp Suite (remember to also change the email!) Send this request off, and you’re in!

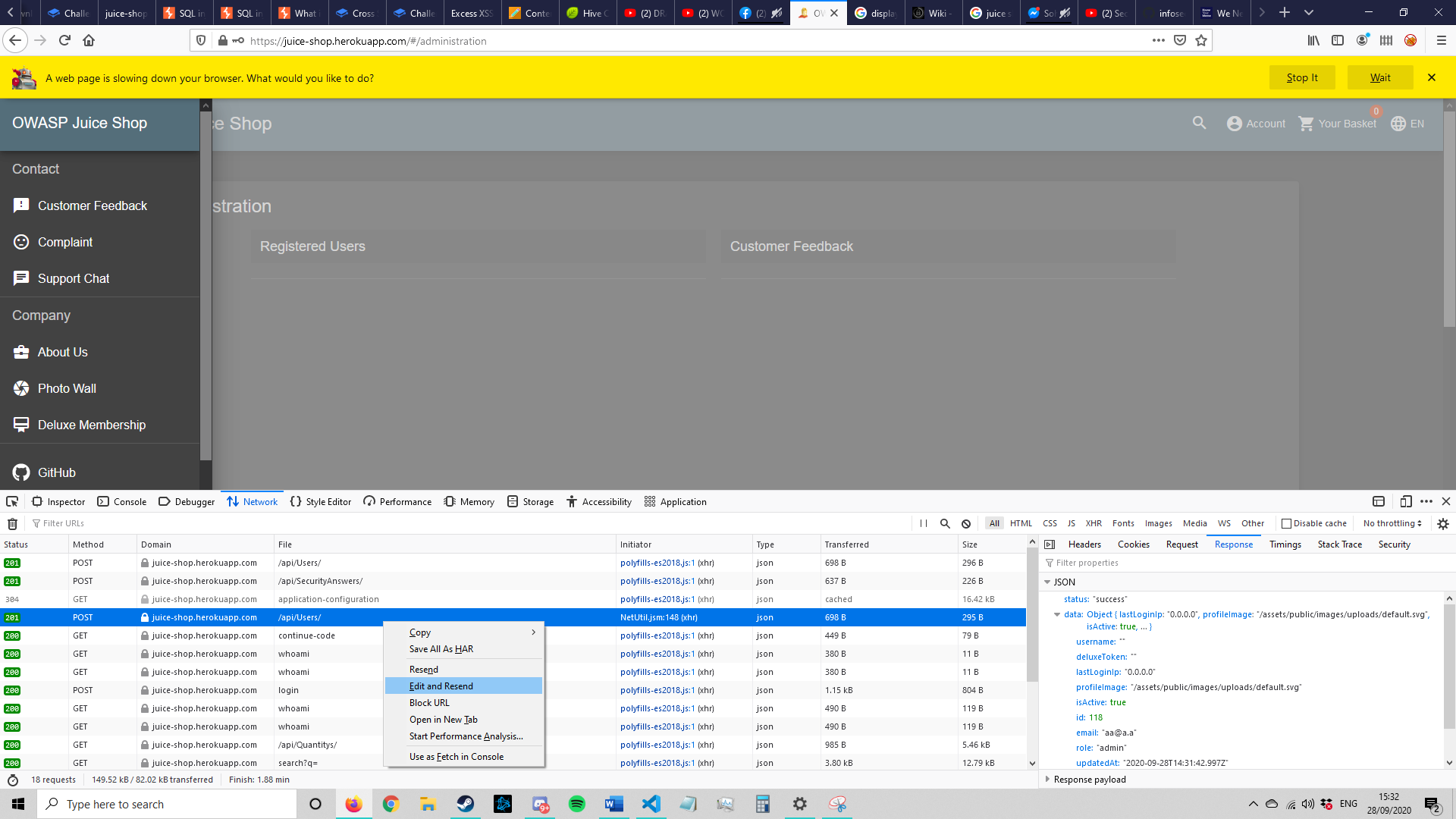


Figure : Editing and resending a request

Example request payload:

{"email":"a@a.a","password":"aaaaa","passwordRepeat":"aaaaa","role":"admin","securityQuestion":{"id":1,"question":"Your eldest siblings middle name?","createdAt":"2020-09-28T11:00:10.999Z","updatedAt":"2020-09-28T11:00:10.999Z"},"securityAnswer":"a"}

5) If you haven’t installed Burp Suite, get it for free here [5]. Once installed, launch it and click ‘Start Temporary Project’ and ‘Use Burp Defaults’ to get to the dashboard.

If this is the first time you have used Burp you will also need to configure the proxy to capture your HTTP traffic. Go to the ‘Proxy’ tab in Burp and click ‘Options’ – from here, configure your proxy listeners to listen on *127.0.0.1:8080* – this address is also known as ‘Localhost’ and lets Burp know to set a listener on your own computer. 8080 is the port that Burp listens on.

Once you have done this, you will need to tell your browser to direct traffic towards the proxy so that Burp can see it. In Firefox, go to settings (*about:preferences* in your URL bar) and scroll to the ‘Network Settings’ at the bottom, then press ‘Settings’. Input the following settings:

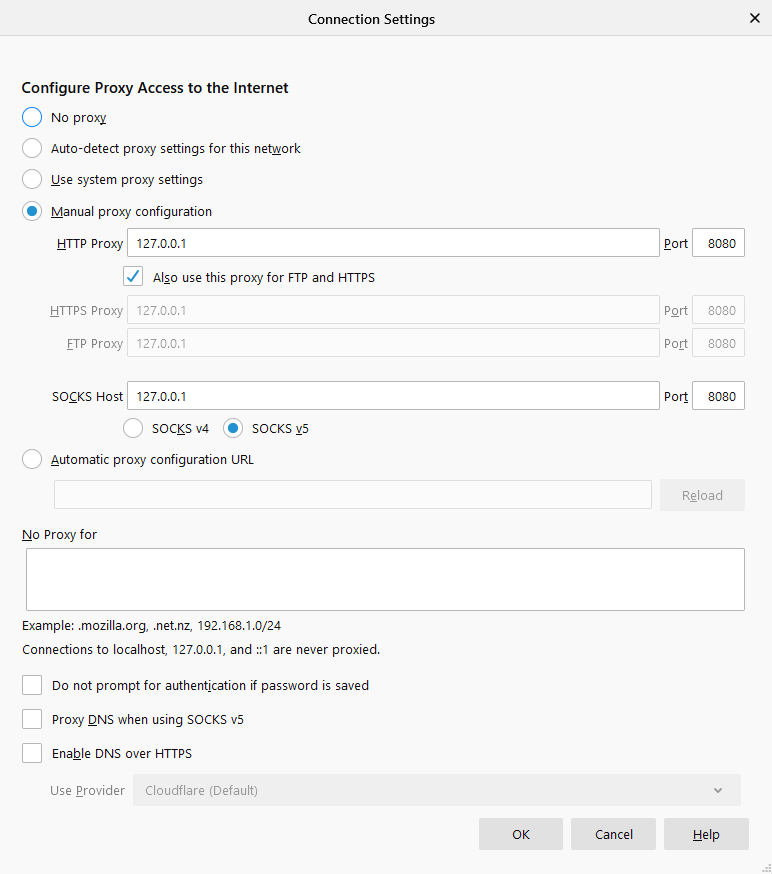


Figure 4: Firefox proxy settings

Alternatively, you can get the FoxyProxy [6] extension for Firefox, and configure it with the following settings:

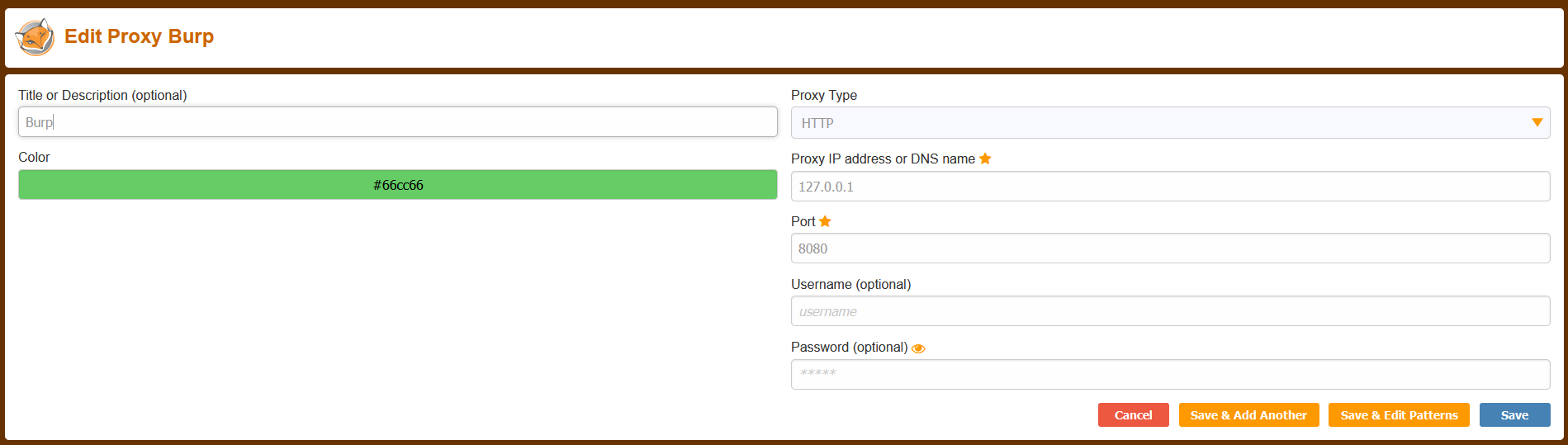


Figure 5: FoxyProxy settings

Then click the FoxyProxy icon in the top-right of your browser and click ‘Burp’.

On Chrome, you cannot set a proxy natively in the browser and instead have to edit your computer’s proxy settings. Follow the instructions at [7] and configure a HTTP proxy for 127.0.0.1:8080 – you will have to look up how to do this for your operating system.

You can now see intercepted traffic in the Proxy > Intercept tab. Make sure Burp says ‘Intercept is on’. Press ‘Forward’ to send a request off, and ‘Action’ to do various other things. If your browser looks like it’s stuck, it might be because you haven’t forwarded a request. Now that’s setup, we can do the challenges!

Note: if your intercept displays several requests to *detectportal.firefox.com* you can click ‘Action’ and ‘Do not intercept requests to this host’ to stop this.

5.1) Now your proxy is configured, click on any product on the home screen of Juice Shop to bring up the reviews. Fill out your review as normal, making sure to give it a rating so it lets you submit.

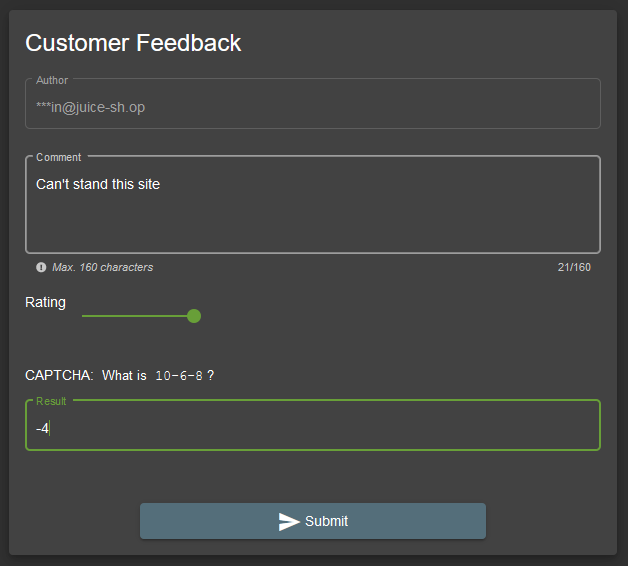


Figure 6: Submitting feedback

When you click submit, Burp should capture a request that looks like this

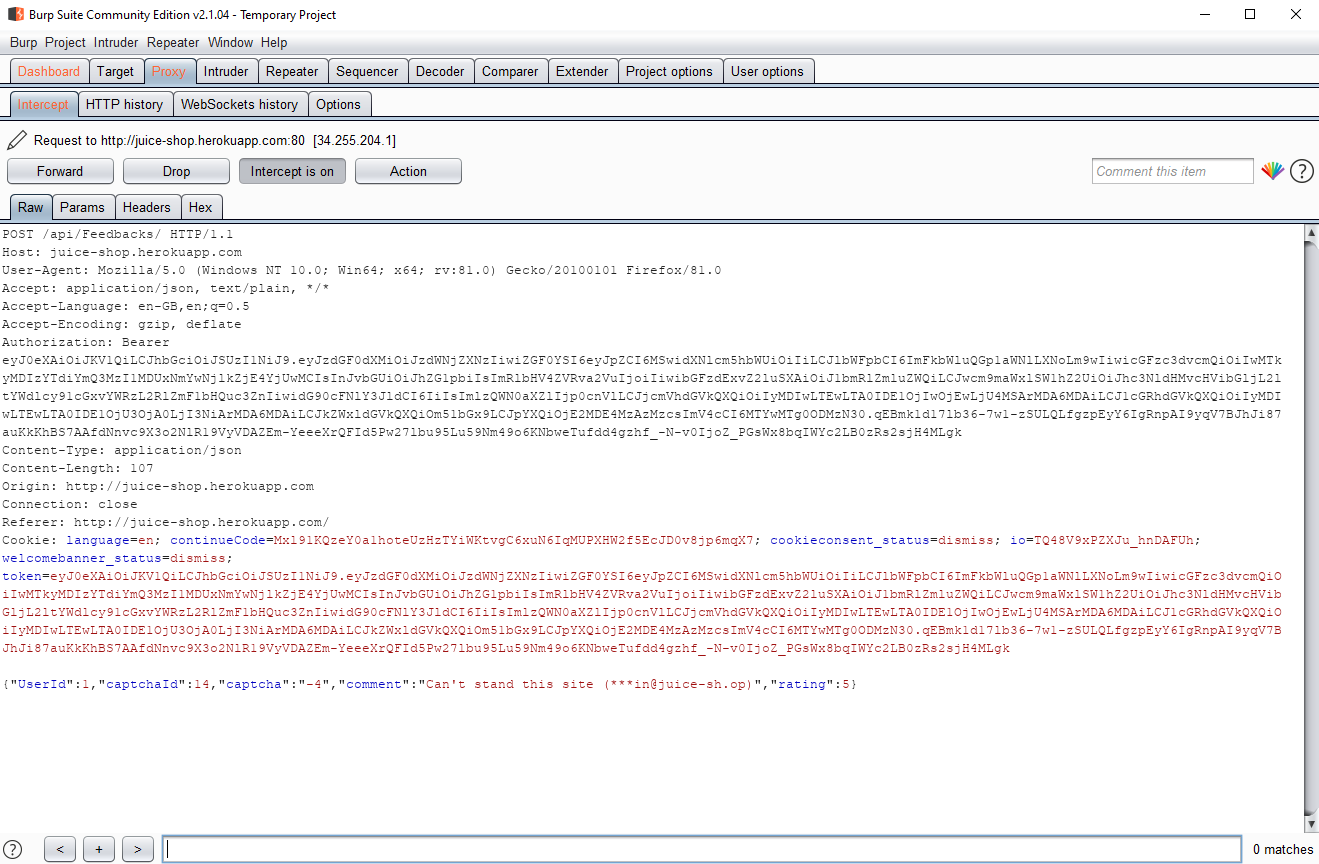


Figure 7: Capturing the feedback request

Let the request go through by forwarding it – this will submit your review. We can then go into the HTTP history tab and find the request we submitted (look for a POST request to /api/Feedbacks/)

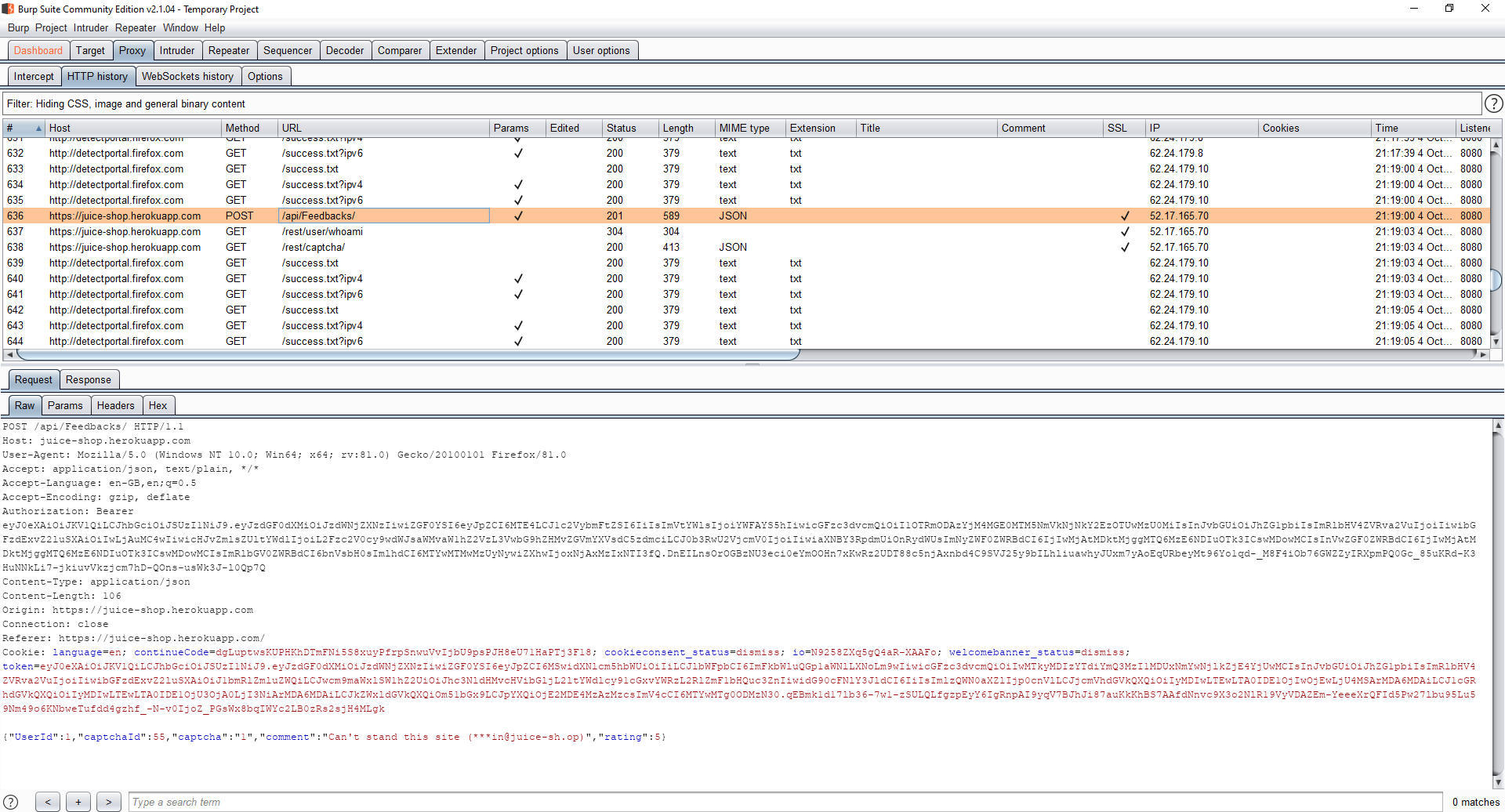


Figure 8: Our request

We can then right-click our request and press ‘Send to Repeater’ (or press Ctrl+R). The Repeater is a powerful tool that lets us edit and resend requests (similarly to how we did in 4.2) with several other powerful options for crafting custom payloads. We will be using the most basic function, which is to manually edit our request with the 0-star value we want to submit.

Click into the request box and change the ‘rating’ parameter to read *“rating”:0*. Then click ‘Send’. A response should appear on the right-hand side

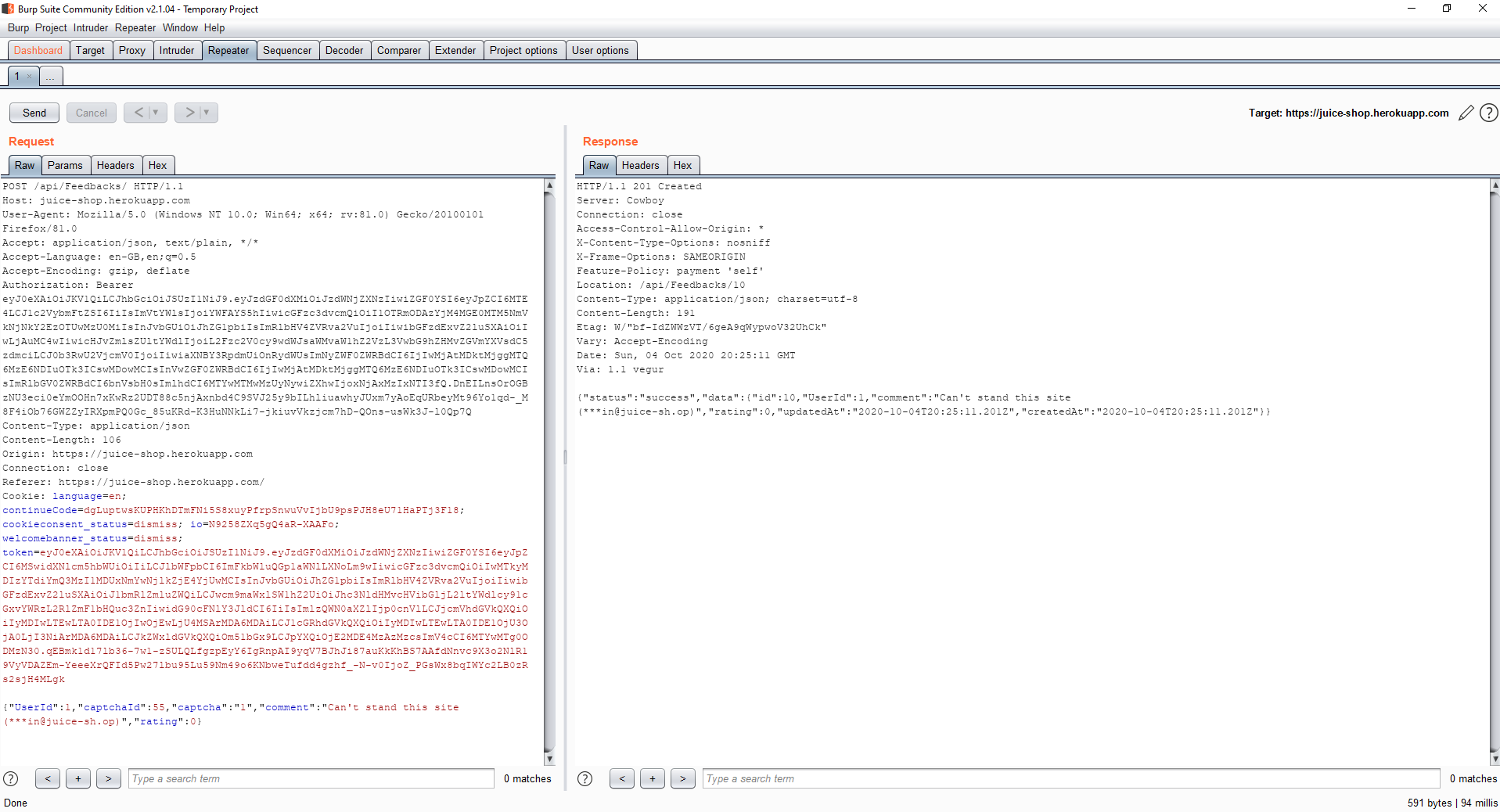


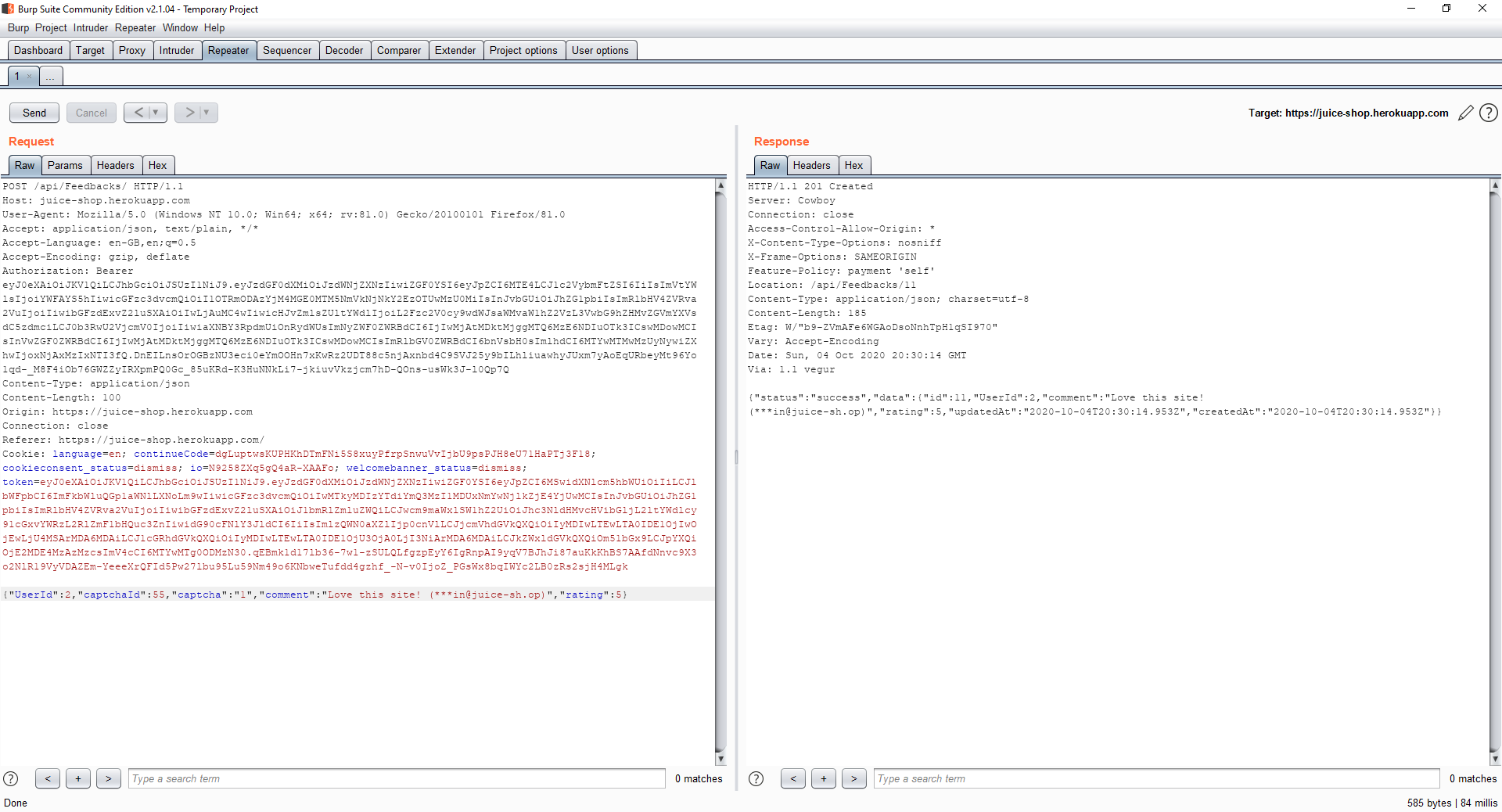
Figure 9: Editing our request in the Repeater

Then turn off intercept and your proxy and go to the ‘About us’ page to see your feedback. It will have a rating of null!



Figure 10: Our feedback

5.2) Using the same request captured in the Repeater, you can now edit the ‘UserId’ parameter in the request to post as another user!



On sites with better authentication, you may also be required to put the user’s token in the request as well. Our method of getting tokens requires being logged in as that user, so this might seem pointless – why submit feedback as a user through Burp when you can just log in as them and do it manually? But in the next Juice Shop worksheet we will take you through stealing a user’s token *without* logging in as them – in this case, you could submit reviews as the user without having to ever log in as them yourself!

Note: It is hard to tell from the reviews screen whether this exploit actually worked – the rating is displayed as ‘null’, and the feedback doesn’t even show who posted it. However, we will show you on the next worksheet how to display data from any table in the site’s database – using this exploit, we can verify that this exploit worked! As a sneak peek, you can see our reviews here:

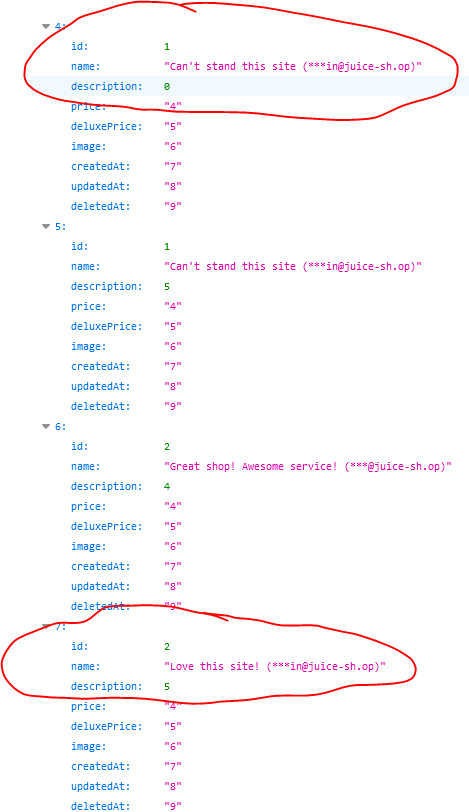


Figure 11: Our fake feedback in the database

Don’t be confused if the columns look a little off – we’ll explain this later. All you need to know is that *id=UserId* and *description=rating* – so the exploit worked!

6) That’s enough Burp Suite for now – time to do some Cross-Site Scripting! In the session we talked about how sanitisation of data is not applied recursively – this can be seen in the Customer Feedback section, where inputting the following text:

<<script>Foo</script>iframe src="javascript:alert(‘xss’)">

causes an alert to appear on the About Us screen. This is because the sanitisation identifies the <script></script> tags and removes everything inside them, believing it to be malicious code – but after doing that, it leaves us with:

<iframe src="javascript:alert(‘xss’)">

Which is still a valid XSS attack! Because this process only happens once, it never checks the result of the first sanitisation, which could be anything… Now let’s do something more interesting than an alert – we’ll make our script grab the site’s cookie using document.cookie and output that instead.

<<script>Foo</script>iframe src="javascript:alert(document.cookie)">

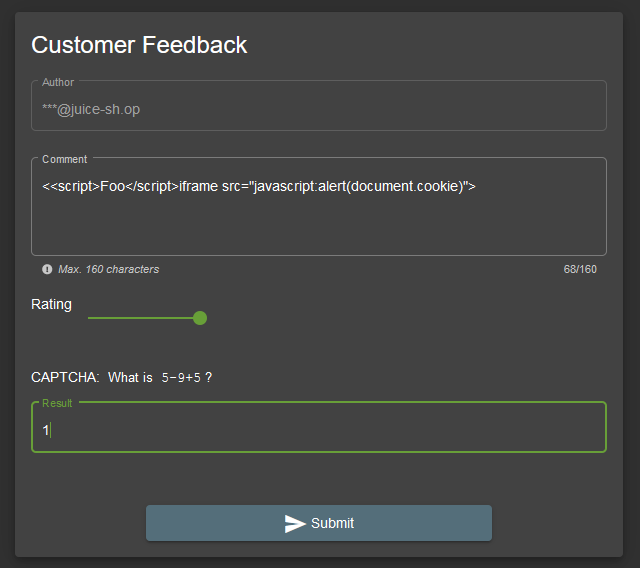


Figure 12: Submitting our script via the customer feedback form

Then when we visit the administration page of our juice shop instance (for example, <https://juice-shop.herokuapp.com/#/administration>), and the token is displayed in the alert box!

Graphical user interface, text

Description automatically generated

Figure : Displaying the token for all to see!

That’s it! Next time we cover Juice Shop we will be going over some more malicious things to do with stealing tokens – but this is good enough for now!

Note: you cannot perform this attack on the [https://juice-shop.herokuapp.com](https://juice-shop.herokuapp.com/#/administration) page, as they have included extra protections against persistent XSS specifically – this is to prevent users having particularly malicious attacks performed against them, as the Heroku instance is available publicly.

We hope you enjoyed this sheet, and now feel more confident with your web hacking skills!

**Appendix**

[1] - <https://github.com/OJ/gobuster>

[2] - <https://github.com/danielmiessler/SecLists>

[3] - <https://www.youtube.com/watch?v=H9FcE_FMZio&t=1110>

[4] - <https://www.ncsc.gov.uk/blog-post/spray-you-spray-me-defending-against-password-spraying-attacks>

[5] - <https://portswigger.net/burp/communitydownload>

[6] - <https://addons.mozilla.org/en-GB/firefox/addon/foxyproxy-standard/>

[7] - https://portswigger.net/burp/documentation/desktop/getting-started/proxy-setup/browser/chrome