Ethical Student Hackers

An Introduction to Web Hacking

SHEFFIELD Ethical Student Hackers

Breaking into security.

Welcome to SESH!

Who are we?

- We formed in 2018
- We meet every Monday, usually 18:00 - 20:00 (ish) on Blackboard Collaborate

What do we do?

- A variety of theory lectures and live demos on a range of topics
- Publish worksheets to further your learning
- A range of guest talks with some great industry partners!

SHEFFIELD Ethical Student Hackers
Breaking into security.

Meet the Committee

President

Nick

Nick is a 2nd-year Software Engineering student. He enjoys computer networking, cyber security and sailing

Vice-President

Mac

Mac is a 3rd-year Computer Science Student, currently on a Year in Industry. He likes building tools, applying automation and AI to cybersecurity, and scouting

Secretary

Chloe

Chloe is a Computer Science student currently on her Year in Industry. Her interests include web scraping, event organisation and cycling.

Treasurer

Brooks

Brooks is a 2nd-year Molecular Biology student who enjoys pretending to be a computer scientist and breaking Linux systems (occasionally intentionally)

Inclusion Officer

Sohyun

Sohyun is a final year software engineering student. She enjoys web development and cyber security.

SHEFFIELD Ethical Student Hackers

Breaking into security.

The Legal Bit

- The skills taught in these sessions allow identification and exploitation of security vulnerabilities in systems. We strive to give you a place to practice legally, and can point you to other places to practice. These skills should not be used on systems where you do not have explicit permission from the owner of the system. It is VERY easy to end up in breach of relevant laws, and we can accept no responsibility for anything you do with the skills learnt here.

- If we have reason to believe that you are utilising these skills against systems where you are not authorised you will be banned from our events, and if necessary the relevant authorities will be alerted.

- Remember, if you have any doubts as to if something is legal or authorised, just don't do it until you are able to confirm you are allowed to.

SHEFFIELD Ethical Student Hackers
Breaking into security.

Code of Conduct

- Before proceeding past this point you must read and agree to our Code of Conduct - this is a requirement from the University for us to operate as a society.
- If you have any doubts or need anything clarified, please ask a member of the committee.
- Breaching the Code of Conduct = immediate ejection and further consequences.
- Code of Conduct can be found at https://shefesh.com/downloads/SESH%20Code%20of%20Conduct.pdf

What is Juice Shop?

- Open Web Application Security Project (OWASP) Juice Shop is probably the most modern and sophisticated insecure web application!
- It can be used in security trainings, awareness demos, CTFs and as a guinea pig for security tools!
- Juice Shop encompasses vulnerabilities from the entire OWASP Top Ten along with many other security flaws found in real-world applications!
- It's basically a very insecure application that can be exploited in multiple ways in order to gain an understanding of possible exploit vectors that are common in some websites.

SHEFFIELD Ethical Student Hackers
Breaking into security.

How to connect to the website

https://juice-shop.herokuapp.com/#/

Or simply Google 'OWASP Juice shop'

Using TryHackMe (If you have the setup for it)

1. Create a TryHackMe account
   a. https://tryhackme.com/
2. Download the OpenVPN file
   a. tryhackme.com/access
3. Connect to the OpenVPN server
   a. sudo openvpn [Path To File]
   b. Equally on windows you can download the OpenVPN software from https://openvpn.net/client-connect-vpn-for-windows/
4. Join the TryHackMe Juice Shop room and start an instance
   a. https://tryhackme.com/room/juiceshop

Exploring the Website

- Alright! Now that we're (hopefully) connected, let's have a look around the website!
- There are multiple sections to this website that look interesting at a first look
    - Login form - Default creds? SQL injection?
    - Contact form - Very simple looking captcha
    - /score-board - Depending if you're using the online or TryHackMe
    - About Us - Contains some links
    - The home screen

The Score-Board Page

SHEFFIELD Ethical Student Hackers

Breaking into security.

Let's look around

- Looking at the source of the index page, we can see that in main.js there is a link to some pages that are not visible when you navigate the pages (We first have to beautify it, then the pages should be at the bottom).

- This exposes /administration and /score-board

- Having a look at these we can see that there is an administration panel and also a score board.

- We didn't have to be logged in as a user to see this information, it was given to us when we first loaded up the website.

How do we find the page?

- Let's have a look at the javascript file that is run on the website
- CTRL + SHIFT + I
  - Opens developer console for Chrome and Firefox
- Head over to the debugger and look at some of the .js files
  - You may need to beautify the javascript to make it a bit more readable
  - https://beautifier.io/
- At the bottom of the page we can see some interesting strings
- Now we can see the different type of vulnerabilities we can find on the website, as well as some other information such as the administration panel.
- Also inspect on the online website shows the score-board link is hidden

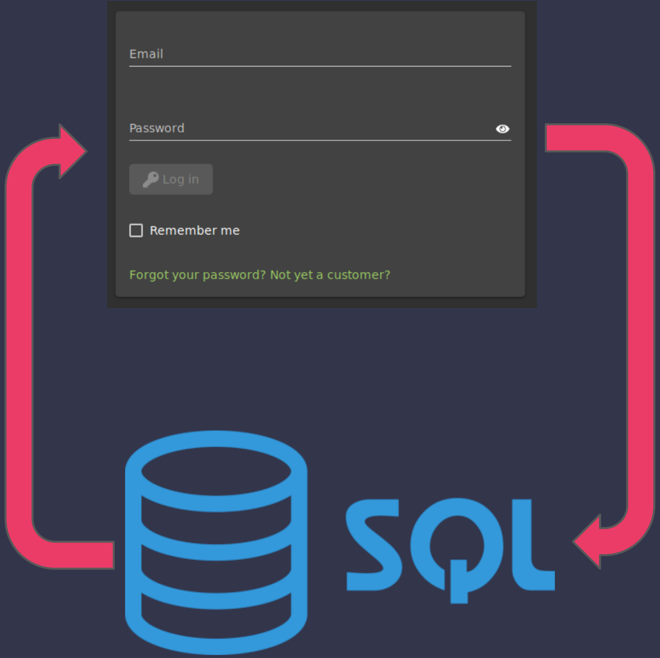
SHEFFIELD Ethical Student Hackers
Breaking into security.

The Login Form

SHEFFIELD Ethical Student Hackers
Breaking into security.

Taking a look at the login form

- Looks like a fairly simple login form
- This website links to an SQL database in the back of the website
- This means that, at some point, a query is sent to the database that asks the database if there is a valid user given the username and the password.
- A default query for this would be:
    - SELECT * FROM users WHERE username = '[username]' AND password = '[password]'
- If some data gets returned by this, then there is a valid user in the database, if not then the user cannot log in.
- We can exploit this if it has not been implemented correctly using SQL injection!



SQL Injection

- First of all, let's see if the login form is vulnerable - to do this, we'll try to provoke an error by submitting a quotation mark (" or ')
- The site returns an SQL error! This is because SQL (the language for talking to databases) uses quotation marks as special characters to 'wrap' strings - adding an extra one breaks the pattern!
- Looking again at the statement from the last slide, we can guess how the database queries its data
  - SELECT * FROM users WHERE username = '[username]' AND password = '[password]'
- If we enter in some SQL code, such as " ' or 1=1;-- " then we could possibly login to the website.
  - This works out, in the database's context, as:
  - SELECT * FROM users WHERE username = " or 1=1;-- ' AND password = '[password]'
  - The red coloured text is the SQL that is actually run, the grey has all been commented out by the -- and is therefore ignored

SHEFFIELD Ethical Student Hackers
Breaking into security.

Logging in as other users

- So we found we can login to the website using some very basic SQL injection as an admin account
  ‘ or 1=1;--

- However, we can also login to other user accounts too. Given that the previous SQL statement will log us into the first account in the database, we can refine the statement down so that is selects a specific account.

- ' or 1=1 and email like('%jim%');--
  This allows us to login to the user account jim, however we can also find other accounts on the /administration page that we can also login as

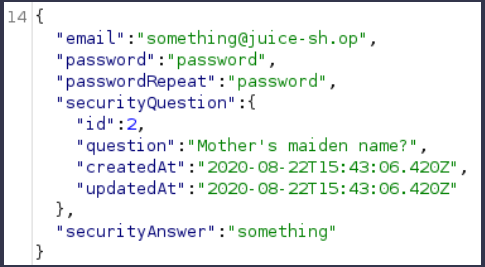
Looking at the user token

- First, let’s have a look around at the cookies that are set when we create an account
- We can see that some form of token is created and set as a cookie. This can be decoded using https://jwt.io, an online tool for decoding json web tokens.
- Once we have decoded the token, we are able so see some information about the user we have logged in as. Some of this information can be very useful to us, for example the id of the user, the email, password, as well as if the account is an admin or not.
- Given that we can now login to any of the user accounts using the SQL injection, we are now able to obtain the password hashes of the user accounts.
- However this also gives us some more information on the backend of the server, that there is some form of isAdmin value.

SHEFFIELD Ethical Student Hackers
Breaking into security.

Creating admin accounts

- Running based on the knowledge we learnt previously, we can see that there is an isAdmin variable that is either true or false.
- Now let’s have a look at what happens when we create a new account in burp, we can see that there is a email, password, passwordRepeat and the security question.
- However we know that there is an isAdmin field that’s in the user token
- If we intercept the request to create the account and add the role field and set it to “admin”, will it create an admin account instead of it being set to a false by default and creating a normal account?



```
{
  "email":"something@juice-sh.op",
  "password":"password",
  "passwordRepeat":"password",
  "securityQuestion":{
    "id":2,
    "question":"Mother's maiden name?",
    "createdAt":"2020-08-22T15:43:06.420Z",
    "updatedAt":"2020-08-22T15:43:06.420Z"
  },
  "securityAnswer":"something"
}
```

Ethical Student Hackers
SHEFFIELD
Breaking into security.

The About Page

FTP Access through /about

- Having a look around the about page, we can see a rather long hyperlink in the middle of the lorem ipsum text
- This hyperlink takes us to an interesting directory of the website as it seems to be hosting FTP
- Navigating to the root of this directory we can see a load of exposed files that we probably shouldn't be able to see
- Not all of the files are readable, however. Only the ones ending in .pdf or .md
- We can bypass this by using a method called null-byte injection - but we'll show you this in a later session!

XSS Injection

What is XSS Injection?

- XSS is an abbreviation of cross-site scripting
- This is a vulnerability where a user is able to inject client-side scripts (typically javascript) into a webpage for it to be loaded by other users when they access the page on the website
- XSS can come in three different forms:
    - Persistent (stored) - The XSS is stored on the servers, such as in a database, and is delivered back to the user whenever they access the website, for example in the form of a comment section of a website.
    - Reflected - The XSS is delivered as part of the HTTP request, for example as a parameter in a search field. When the user visits the given URL, the malicious script is loaded.
    - DOM - The XSS is delivered via the website's Document Object Model (i.e. its elements), for example by modifying an element via a script built into the site.
- https://owasp.org/www-community/attacks/xss/

How can we use XSS on Juice Shop?

- There are a couple of fields that can be used to exploit XSS on the website, both in reflected and also stored forms.
- Heading over to the search box on the home screen, you can see that the search query is rendered back onto the webpage when you search for something. This means that it may be possible to inject some malicious code and for it to be run on the clients browser.
- Using the iframe code that is given to us on the score-board page, we can cause an alert to appear on the page.
- There are a couple more places on the website where an XSS can be performed, such as the customer feedback form (whose results are displayed on the about page). This is a bit tricker to perform though, as it has better sanitisation

That's It!

Any Questions?

Thirsty for more?

- Get yourself a membership!
- Have a go at this week's worksheet!
- Come along to our second Juice Shop session, where we'll take a deeper look at some of the things from today (and some juicy extras)

Thanks for coming!

SHEFFIELD Ethical Student Hackers
Breaking into security.