

**Assignment: Fall Semester Juice-Shop Assignments**

Ramon Lopez Jr

Department of Cyber Security: University of Advancing Technology

NTS330: Applied Exploits

Professor Sarah Bunce

December 15<sup>th</sup>, 2024

## **Assignment: Fall Semester Juice-Shop Assignments**

The OWASP Juice Shop is a deliberately vulnerable web application designed to evaluate and enhance users' ethical hacking skills by presenting a variety of security challenges. Throughout these assignments, the focus was on identifying and exploiting vulnerabilities such as hidden pages, broken access control, security misconfigurations, and Cross-Site Scripting (XSS). Each challenge simulates real-world scenarios, requiring users to apply techniques like reconnaissance and exploit development to uncover and resolve security flaws. To begin, the environment was consistently set up using a Kali Linux virtual machine and the Juice Shop service, which was launched locally using the command `sudo npm start` after navigating to the application directory. Challenges ranged from locating the hidden Scoreboard page to addressing broken access controls and security misconfigurations like Error Handling and Deprecated Interface. A recurring issue with service permissions was successfully resolved using administrative commands, highlighting the importance of troubleshooting skills during the process. The culmination of these efforts involved tackling the DOM XSS challenge, highlighting how these vulnerabilities compromise the confidentiality and availability principles of the CIA triad. These tasks provided valuable firsthand experience in identifying, exploiting, and mitigating critical security issues in web applications.

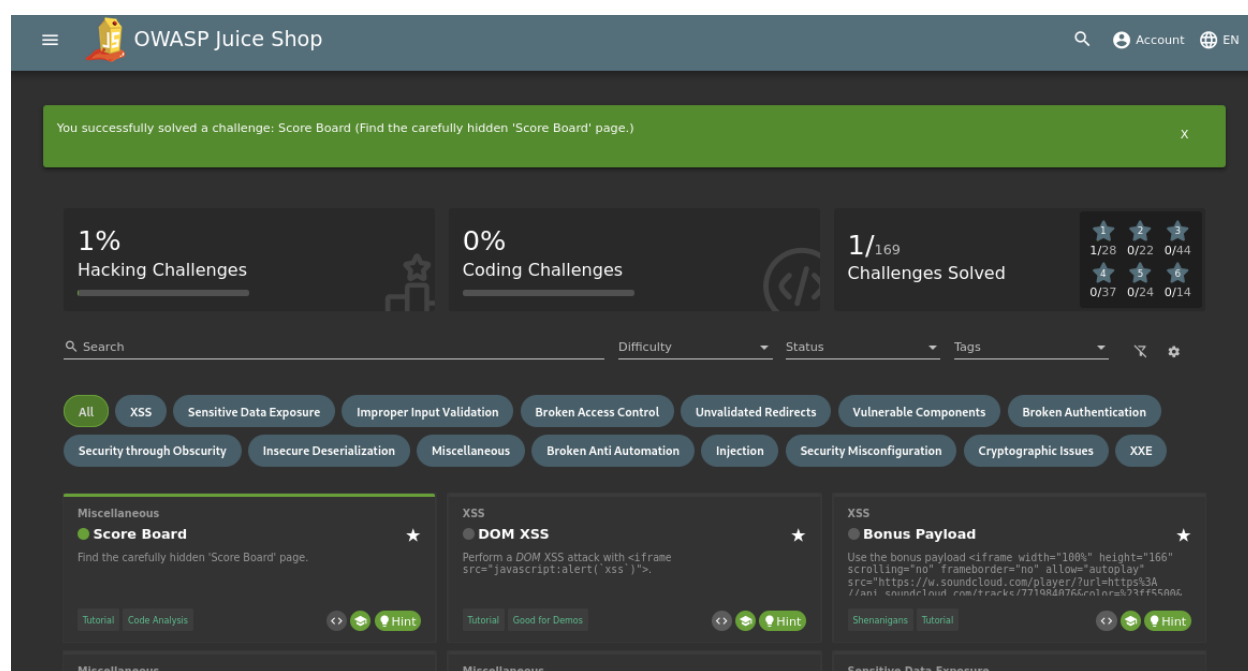
### **Finding the Scoreboard:**

Once my server was running, I opened Firefox and navigated to `http://localhost:3000`, which is the local URL hosting my instance of Juice Shop. On the homepage, I clicked on the orange “Help getting started” button, which triggered a pop-up panel on the left side of the screen with useful hints. One of these hints suggested examining the website’s JavaScript files for clues, pointing me in the direction of client-side resources. Using Firefox’s web development

tools (also known as “Inspect Element”), I began searching through the website’s JavaScript files to find clues related to the Scoreboard page. I consulted ChatGPT to ask, “first I am going to start looking through the client-side JavaScript to see if I could find any clues. I am using Firefox with its web developer tools. What are some keywords I could be searching for?” ChatGPT suggested terms like “scoreboard,” “path,” “URL,” and “challenge.” With these keywords in mind, I started searching the JavaScript files and found references to “score-board” in a file called main.js. I discovered the following code snippet: `path: "score-board", component: Vc`. This appeared to be a clear indication of the route used for the scoreboard page. I attempted to access the page by entering the URL `http://localhost:3000/score-board`, but this only brought me back to the homepage. At this point, I was unsure why the path wasn’t working as expected. I consulted ChatGPT again and learned that the Juice Shop application uses hash-based routing, which is common in single-page applications (SPAs). In hash-based routing, the URL after the `/#/` fragment is used to dynamically load different views without reloading the entire page. ChatGPT suggested trying the URL with the hash fragment: `http://localhost:3000/#/score-board`. By doing this, I was able to successfully load the Scoreboard page.

Now that I had located the Scoreboard, I reflected on the tools and techniques I used to complete this challenge. First, I relied on Firefox’s built-in web development tool, also known as the “Inspect” tool, to begin my reconnaissance. This tool allows users to inspect the structure, scripts, and other resources of a website, making it ideal for uncovering hidden features like the scoreboard. I chose this tool because it required no additional installations and provided immediate access to the client-side resources of the Juice Shop application. Additionally, by leveraging ChatGPT, I was able to refine my search and receive guidance on troubleshooting the URL structure. In terms of techniques from the MITRE ATT&CK framework, the one I applied

during this challenge was **T1590 – Gather Victim Network Information**. This technique involves collecting information about the target network, such as URLs, IP addresses, and other network-related data, which can be used in further attacks or investigations. By analyzing the website’s JavaScript and understanding how the routing system worked, I gathered critical information that allowed me to access the hidden Scoreboard page. This technique is often used in the reconnaissance phase of an attack, where attackers attempt to identify entry points or hidden features of a target system. In this case, I applied similar methods to uncover the Scoreboard page, demonstrating how reconnaissance techniques can be effectively used in web application exploitation.



## Login MC SafeSearch:

Starting with the first challenge, “Login MC SafeSearch,” I began by searching for any leads. Initially feeling stuck, I checked the hints provided within the Juice Shop app. One hint linked to a YouTube video titled “Protect Ya’ Passwordz.” After watching the video but still

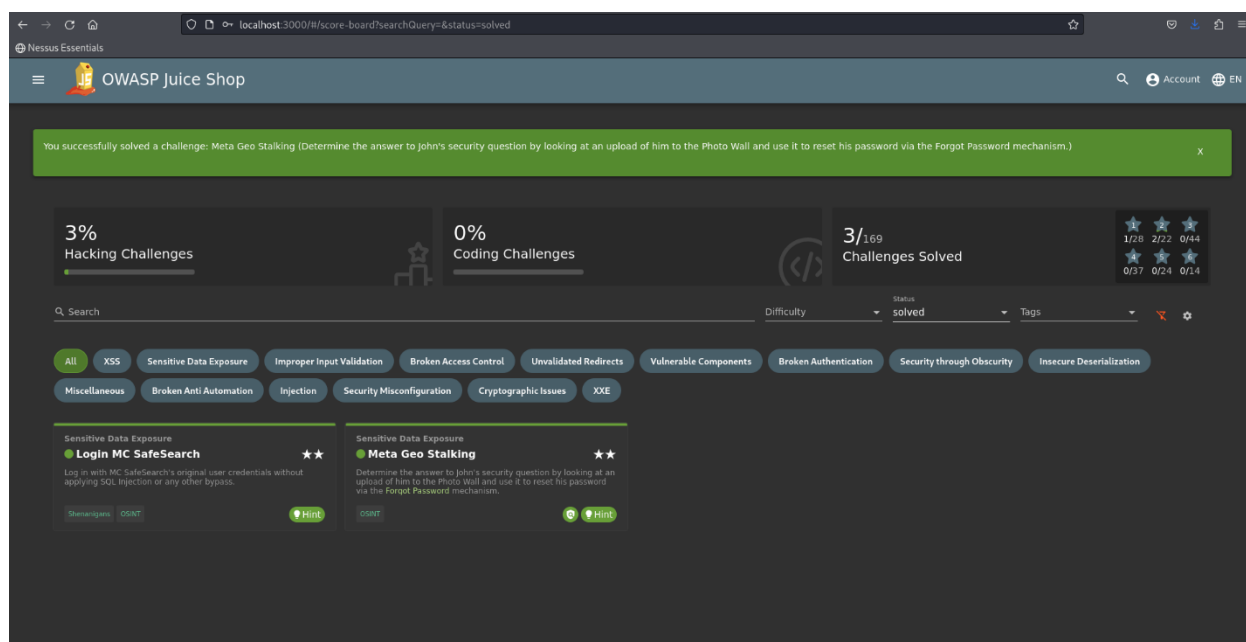
feeling unsure, I consulted ChatGPT again. I provided the lyrics from the video and asked it, “Could you analyze any unusual lines that might hint towards this user’s password?” ChatGPT pointed out the lines, “But then how do you remember is the question that I get. I say why not use the first name of your favorite pet? Mine’s my dog, Mr. Noodles. It doesn’t matter if you know. Because I was tricky and replaced some vowels with zeroes.” Using this information, I looked at "Mr. Noodles" as a potential password, but first, I needed MC Safe’s email address. Switching to the homepage, I browsed the images and user reviews. Under a review for “Juice Shop ‘Permafrost’ 2020 Edition,” I found an email address: mc.safesearch@juice-sh.op. With this email, I navigated to the login page and entered it, followed by "Mr. Noodles" as the password. This attempt failed. Rechecking the hint, I noticed the line about replacing vowels with zeros. Adjusting the password by replacing vowels in "Noodles" with zeros, I tried again. This time, it worked, and the challenge was completed!

For the Login MC SafeSearch challenge, the technique used aligns with T1078: Valid Accounts from the MITRE ATT&CK Framework. This technique involves leveraging valid credentials to access systems or applications without authorization from the account owner. By locating MC Safe’s email address and attempting to use it for login, this challenge exploits an account with existing credentials in the system. Successful exploitation here highlights the risks of credential leakage in publicly accessible areas, such as user reviews, emphasizing the need for better credential management and awareness to prevent unauthorized access.

## **Meta Geo Stalking:**

Moving on to the next challenge, “Meta Geo Stalking,” I reviewed the scoreboard to clarify the goal. The challenge involved tracking a user named "John" to discover his security question. I checked the official Juice Shop guide, which hinted at users inadvertently exposing sensitive information online. This suggested that the challenge might involve metadata in an image. I switched to the “Photo Wall” tab, hoping it contained relevant information. Browsing the images, I found one posted by "J0hNey" showing him hiking with a drink. Unsure how to proceed, I clicked on “Forgot Password” on the login page and entered the email john@juice-sh.op. This triggered a security question: “What’s your favorite place to go hiking?” Recalling the hiking image, I uploaded it to an EXIF metadata viewer to check for GPS data. The metadata included a GPS position, which I entered Google Maps. It led me to a hiking trail in Kentucky, USA, near the "Daniel Boone National Forest." Using this as the answer to the security question, I was able to reset the password to something simple, like "12345," completing the challenge successfully!

For the Meta Geo Stalking challenge, the technique used corresponds to T1602: Data from Local System within the MITRE ATT&CK Framework. This technique involves accessing and extracting metadata embedded within files to gather sensitive information, such as geographic location or device data. By analyzing the EXIF metadata in images, attackers can identify specific locations associated with image files, potentially revealing sensitive or exploitable location data. This exposure underscores the importance of stripping or securing metadata in public-facing images to prevent unauthorized information disclosure and safeguard privacy.



## View Basket:

For this challenge, the objective was to view another user's shopping basket by exploiting access control vulnerabilities. First, I reviewed the challenge details in Juice Shop for an overview of what was required. To start, I consulted ChatGPT, asking, "...using this information, how would I go about starting to find this exploit?" ChatGPT suggested that I begin by signing up to Juice Shop. Using the login credentials from a previous assignment (email: `mc.safesearch@juice-sh.op`, password: `Mr. N00dles`), I signed in and accessed the main page. I explored the available options and felt unsure about the next steps, so I asked ChatGPT, "I have signed into the juice-shop, now where should I start looking?" The AI suggested navigating to the "Your Basket" section and opening the web inspection tool. Following this advice, I accessed "Your Basket" and opened the web inspection tool. After exploring various page elements for about 15 minutes, I achieved success by changing the value in the Storage section under the "bid" category. I modified the "bid" value from "6" to "1" and reloaded the page, which completed the challenge.

The technique used in this challenge aligns with T1078: Valid Accounts, where unauthorized access is gained by manipulating session tokens. In this instance, by changing the “bid” parameter, I accessed another user’s data, demonstrating a violation of access controls. Mitigation for this issue includes enforcing strict session validation and implementing access controls that restrict user access to only their own resources.

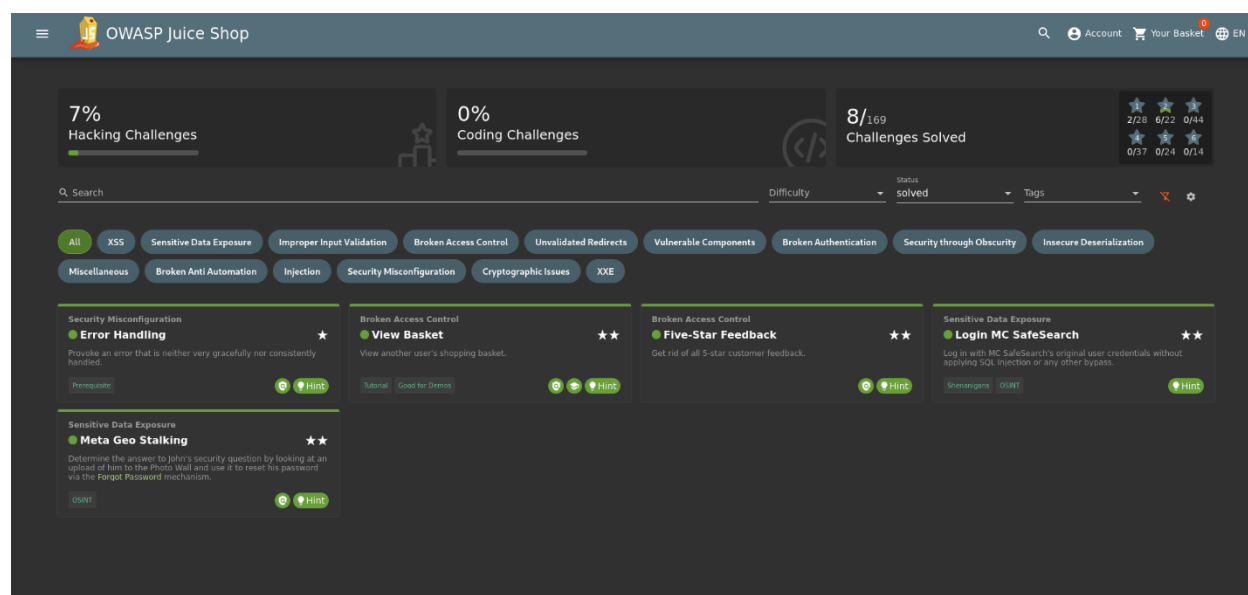
### **Five-Star Feedback:**

For the Five-Star Feedback challenge, the objective was to remove all the five-star customer feedback, leveraging access control weaknesses. I began by examining the challenge details on the scoreboard, which hinted that I might need to perform administrative functions. As a first step, I consulted ChatGPT, asking, “...how would I go about doing this challenge?” It suggested that I log in as an administrator. Since I didn’t know the administrator password, I searched online and found a relevant article at <https://curiositykillscolby.com>, which, combined with ChatGPT’s guidance, recommended trying “0 or 1=1 –” as the login with any password. I entered this into the login page, and it successfully logged me in as an administrator, also completing an unrelated challenge. With administrator access, I continued by exploring potential administrative tools. I consulted ChatGPT again: “Now signed in as an administrator, where do you think administration tools for the website would be hidden?” It suggested searching for hidden pages. I entered keywords like “admin” into the search bar, but it was only after using “administrator” following “/#/” in the URL that I accessed an admin page. Here, I found the necessary options to delete five-star reviews, completing the challenge.

This challenge relates to T1588: Obtain Capabilities. In this instance, I obtained unauthorized administrative capabilities, enabling me to delete user feedback. Mitigation



strategies include robust input validation, ensuring role-based access restrictions, and regularly reviewing access control mechanisms to prevent privilege escalation.



## Error Handling:

Upon opening the Juice Shop scoreboard, I located the “Error Handling” challenge. Interestingly, it was already marked as completed, indicating that I may have inadvertently triggered this challenge during previous work. The challenge description reads: “Provoke an error that is neither very graceful nor consistently handled.” Although I couldn’t recall exactly how I achieved this, I reviewed online resources and found guidance at OWASP Juice Shop Help, which describes ways to intentionally provoke an unhandled error. One common method is to enter a single colon as the username and "999" as the password on the login page. This input triggered the error message "[object Object]", which indicates that the application is not handling errors gracefully, a sign of inadequate error handling.

The MITRE ATT&CK techniques that were used for this challenge were the T1203: Exploitation for Client Execution, triggering unhandled error responses involves inputting

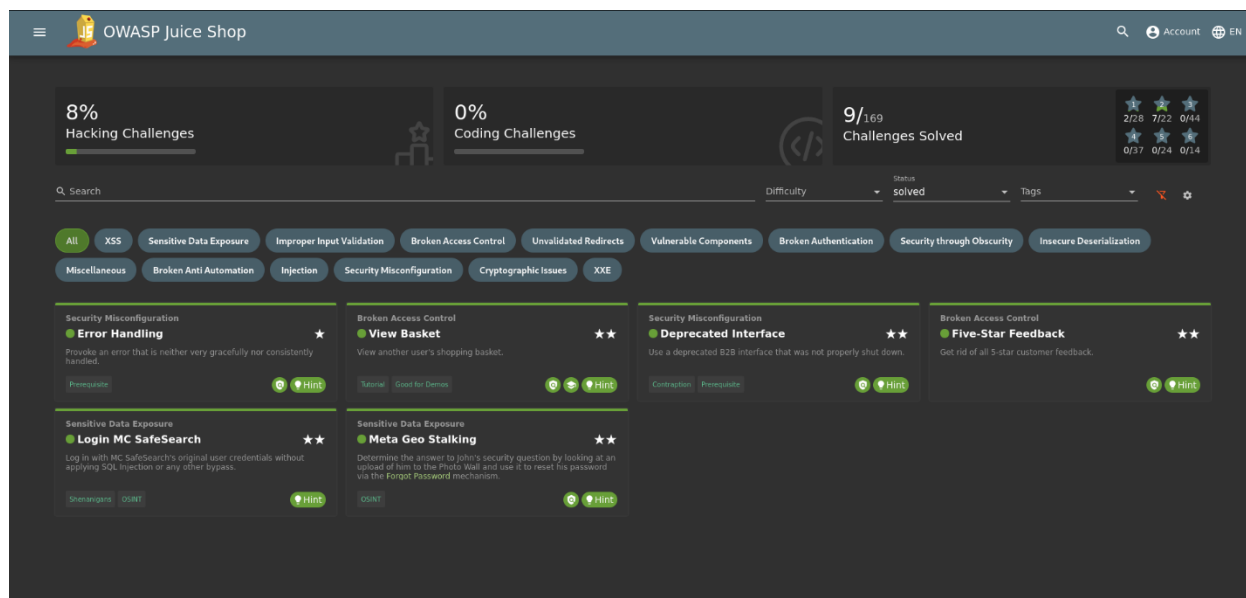
unexpected characters, causing the application to expose internal processing logic. This aligns with T1203 as the error results from manipulation of the user input and client-side interaction. And T1069: Unsecured Protocols, displaying unfiltered error messages may leak information about the application's underlying structure, making it vulnerable to reconnaissance tactics. This also hints at a misconfiguration under T1069. Mitigation for this issue is to improve security; error messages should be user-friendly without disclosing technical details. Restricting error handling to log detailed messages on the server side only, while presenting a generic message to the user, helps mitigate risks of information leakage.

### **Deprecated Interface:**

The "Deprecated Interface" challenge requires exploiting an old B2B interface that wasn't fully disabled. Following guidance from the OWASP Juice Shop Help, I began by logging into Juice Shop with the test account: mc.safesearch@juice-sh.op. Once logged in, I accessed the "Complaint" feature under the "Contact Us" section. Initially, I tried uploading a .deb file, which prompted a rejection message: "Forbidden file type. Only PDF, ZIP allowed." I then examined the "main.js" file in the developer tools to check for allowable file types. Using this information, I tried uploading a sample .xml file found online, along with a generic description "1234". When I clicked submit, the challenge was successfully completed.

The MITRE ATT&CK techniques that were used for this challenge were the T1136: Create or Modify System Process, interacting with an improperly disabled interface potentially allows an attacker to exploit legacy processes not securely removed, aligning with T1136. T1190: Exploit Public-Facing Application, accessing deprecated interfaces, particularly for complaint submission, may reveal older system configurations that lack current security standards, making it a classic T1190 exploitation scenario. Mitigation for this issue is to address

deprecated interfaces, it's critical to decommission outdated services thoroughly and remove old code or endpoint access. Regularly testing all exposed interfaces and reviewing legacy components helps to prevent exploitation through obsolete pathways.





## DOM XSS:

The DOM XSS challenge required gathering additional information to understand the exploitation method. I navigated to the scoreboard via the main menu and searched for "DOM XSS." The challenge description stated the goal: "Perform a DOM XSS attack with `<iframe src='javascript:alert('xss')'>`." To proceed, I consulted ChatGPT for advice, asking, "...based on the information I have provided to you, where would you start looking for exploitation?" ChatGPT suggested testing user input fields, so I examined the search bar in the OWASP Juice Shop interface. Still uncertain, I asked ChatGPT for further clarification: "...Using the search bar, what should I try inserting into that?" It recommended testing a simple HTML string, such as `<h1>text</h1>`, to check if the input was being processed and embedded into the JavaScript code. Testing this, I found that typing a string like "Hello World" dynamically rendered the text

on the page. Using the browser's Inspect Tool, I confirmed that the entered text was directly embedded into the code without proper sanitization. Armed with this information, I attempted the challenge payload: `<iframe src="javascript:alert('xss')">`. Upon hitting Enter, the malicious script executed successfully, displaying the alert box, and completing the challenge.

The DOM XSS attack aligns with the MITRE ATT&CK Framework, specifically technique T1059.007 - Cross-Site Scripting (XSS). This technique exploits vulnerabilities in web applications that improperly handle user input, allowing attackers to inject and execute malicious JavaScript in the victim's browser. The impact of this vulnerability can include unauthorized data exposure, session hijacking, and user impersonation. By targeting user input fields and manipulating the DOM dynamically, the attacker gains control over the browser environment without directly interacting with the server. To prevent DOM XSS vulnerabilities, several mitigation strategies can be implemented. First, user inputs should be validated and sanitized before rendering them in the DOM. This includes escaping special characters to prevent injection attacks. Additionally, adopting a Content Security Policy (CSP) can help restrict which scripts and resources can execute in the browser, reducing the attack surface. Developers should also avoid using dangerous JavaScript functions like `eval()` or `document.write()` and instead adopt safer alternatives. Finally, conducting regular security audits, including code reviews and penetration testing, is essential for identifying and addressing vulnerabilities before attackers exploit them.

  OWASP Juice Shop

<iframe src="javascript:alert('xss')"></iframe> X Account EN

You successfully solved a challenge: DOM XSS (Perform a DOM XSS attack with <iframe src="javascript:alert('xss')">.) X

9%  
Hacking Challenges

0%  
Coding Challenges

10/169  
Challenges Solved

3/28 7/22 0/44  
0/37 0/24 0/14

Q Search Difficulty solved Tags X

All XSS Sensitive Data Exposure Improper Input Validation Broken Access Control Unvalidated Redirects Vulnerable Components Broken Authentication Security through Obscurity Insecure Deserialization Miscellaneous Broken Anti Automation Injection Security Misconfiguration Cryptographic Issues XXE

**Security Misconfiguration**  
**Error Handling** ★  
Provide an error that is neither very gracefully nor consistently handled.  
Prerequisite

**Broken Access Control**  
**View Basket** ★★  
View another user's shopping basket.  
Tutorial Good for Demos

**Security Misconfiguration**  
**Deprecated Interface** ★★  
Use a deprecated 828 interface that was not properly shut down.  
Contraption Prerequisite

**Broken Access Control**  
**Five-Star Feedback** ★★  
Get rid of all 5-star customer feedback.

**Sensitive Data Exposure**  
**Login MC SafeSearch** ★★  
Log in with MC SafeSearch's original user credentials without applying SQL injection or any other bypass.  
Stealingers DONT

**Sensitive Data Exposure**  
**Meta Geo Stalking** ★★  
Determine the answer to John's security question by looking at an upload of him to the Photo Wall and use it to reset his password via the Forgot Password mechanism.  
DONT

**Resources:**

ChatGPT: For guidance on potential exploitation methods and validation of techniques.

Firefox Web Developer Tools: For finding information embedded within the website's code.

EXIF Metadata Viewer: To view the metadata off the files to complete a challenge.

Google Maps: To find location based on what was found on the metadata off a file.

Official OWASP Juice-Shop Help: For finding official guidance on how to complete the challenges.

## References:

- codeblue04. (2020, November 2). *Hacking OWASP's Juice Shop Pt. 5: Login Admin*. Curiosity Kills Colby. <https://curiositykillscolby.com/2020/11/01/pwning-owasps-juice-shop-pt-4/>
- Kimminich, B. (n.d.). *Introduction · Pwning OWASP Juice Shop*. Help.owasp-Juice.shop. Retrieved December 15, 2024, from <https://help.owasp-juice.shop/>
- MITRE. (2024). *MITRE ATT&CK™*. Mitre.org. <https://attack.mitre.org/>
- OWASP. (n.d.). *Running OWASP Juice Shop :: Pwning OWASP Juice Shop*. Pwning.owasp-Juice.shop. Retrieved December 15, 2024, from <https://pwning.owasp-juice.shop/companion-guide/latest/part1/running.html>
- OWASP. (2016). *Sensitive Data Exposure :: Pwning OWASP Juice Shop*. Owasp-Juice.shop. [https://pwning.owasp-juice.shop/companion-guide/latest/part2/sensitive-data-exposure.html#\\_determine\\_the\\_answer\\_to\\_johns\\_security\\_question](https://pwning.owasp-juice.shop/companion-guide/latest/part2/sensitive-data-exposure.html#_determine_the_answer_to_johns_security_question)
- OWASP. (2023). *OWASP foundation, the open-source foundation for application security*. Owasp.org. <https://owasp.org/>
- Rapper Who Is Very Concerned With Password Security*. (2014, October 27).  
Www.youtube.com. <https://www.youtube.com/watch?v=v59CX2DiX0Y>