Bypass testing in Web Application (Client Side)

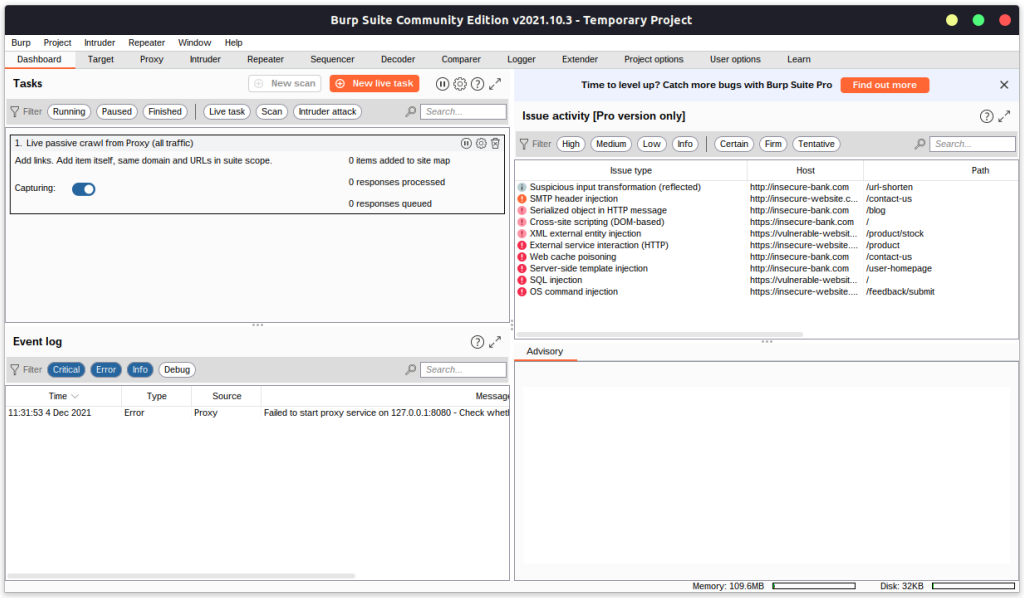
Client Side Bypass Testing

Bypass testing is the act of circumventing the checks and validations at the client side to send the unexpected input values to the server and observe it's behaviour.

Tools Used

Bypass Test Tool: BurpSuite

BurpSuite: Burp will be used to capture the response sent by WebGoat and circumvent the client side javascript validations.

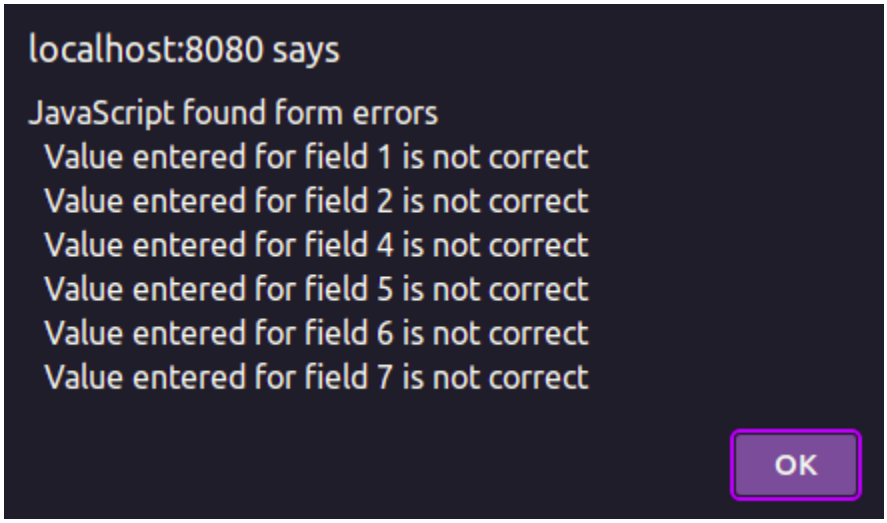


Input Validation Test Cases

Test Case 1: Input exactly three lowercase characters(^[a-z]{3}$)
Test Case 2: Input exactly three digits(^[0-9]{3}$)
Test Case 3: Input letters, numbers, and space only(^[a-zA-Z0-9]*$
Test Case 4: Input enumeration of numbers (^(one|two|three|four|five|six|seven|eight|nine)$)
Test Case 5: Input simple zip code (^\d{5}$)
Test Case 6: Input zip with optional dash four (^\d{5}(-\d{4})?$)
Test Case 7: Input US phone number with or without dashes (^[2-9]\d{2}-?\d{3}-?\d{4}$)

Test Cases	Valid Inputs	Invalid Inputs
Test Case 1	abc	abc!
Test Case 2	123	123!
Test Case 3	abc 123 ABC	abc 123 ABC!
Test Case 4	seven	seven!
Test Case 5	01101	01101!
Test Case 6	90210-1111	90210-1111!
Test Case 7	301-604-4882	301-604-4882!

Client Side Response before Bypassing the input validations -



Bypassing the Input Validation -

Now we will intercept the request sent by the client to the server after the validation of the inputs at the client side.

Intercepting the request -

First we will send the valid inputs to bypass the client side javascript validations and then change the intercepted request to send the invalid inputs.

Intercepted Request -

```
POST /WebGoat/BypassRestrictions/frontendValidation/ HTTP/1.1
Host: localhost:8080
Content-Length: 112
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="96"
Accept: */*
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/96.0.4664.45 Safari/537.36
sec-ch-ua-platform: "Linux"
Origin: http://localhost:8080
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost:8080/WebGoat/start.mvc
Accept-Encoding: gzip, deflate
Accept-Language: en-GB,en-US;q=0.9,en;q=0.8
Cookie: JSESSIONID=nikFh2uNkElyQ153ClVxAWCeRcLNqKmZnW0e_DT4
Connection: close

field1=abc&field2=123&field3=abc+123+ABC&field4=seven&field5=01101&field6=90210-1111&field7=301-604-4882&error=0
```

Now we can modify the fields to send the invalid inputs to the server

Previous Inputs in the Request -
field1=abc!&field2=123!&field3=abc+123+ABC!&field4=seven!&field5=01101!&field6=90210-1111!&field7=301-604-4882!&error=0

Modified Request with Invalid Inputs -
field1=abc!&field2=123!&field3=abc+123+ABC!&field4=seven!&field5=01101!&field6=90210-1111!&field7=301-604-4882!&error=0

Now send the modified request to the server and that's how we can bypass client side validations.