Cybersecurity for Data Science

Individual Report

SQL injection:

How to find and exploit SQLi vulnerabilities

Author:
Tariq Baghrous - 904027- t.baghrous@campus.unimib.it

Master's Degree in Data Science
Università degli Studi di Milano-Bicocca
2023/2024 Academic Year

Contents

1 Introduction

SQL injection represents a web security vulnerability enabling attackers to manipulate the queries sent by an application to its database. Exploiting SQL injections, attackers gain unauthorized access to sensitive data beyond their normal reach, potentially compromising user information or any accessible data within the application. Furthermore, attackers can manipulate or delete this data, leading to persistent alterations in the application's content or functionality.

Main SQL injection attacks include:

- **Union-Based SQL Injection:** This attack exploits the UNION SQL operator to combine the result sets of two or more SELECT statements, allowing the attacker to retrieve unauthorized data from the database
- **Blind SQL Injection (Boolean-Based):** With Boolean-Based SQL injection, the attacker sends SQL queries to the database and analyzes the application's response to infer whether the injected query evaluates to true or false, enabling them to extract data or perform unauthorized actions

In this report, we delve into an examination of SQL injection vulnerabilities on a deliberately vulnerable web application named *ginandjuice.shop*, utilizing sophisticated tools such as Burp Suite for comprehensive vulnerability scanning and SQLMap for executing automated attacks.

2 Software and Websites

2.1 Gin & Juice Shop

Gin & Juice Shop[1] is a modern deliberately vulnerable web application provided by PortSwigger, similar to an e-commerce site. Gin & Juice Shop is designed to contain many web vulnerabilities, doesn't require deployment, and is just the right size to quickly assess a scan test.

2.2 Burp Suite

Burp Suite Enterprise Edition[2] is a web vulnerability management platform developed by PortSwigger. It provides a centralized platform to conduct web vulnerability scanning, management and reporting. It allows users to monitor real-time scan results and prioritize vulnerabilities based on severity, in order to efficiently identify and remediate security vulnerabilities.

2.3 SQLmap

SQLmap[3] is an open source penetration testing tool that automates the process of detecting and exploiting SQL injection flaws of database servers. SQLmap is able to detect and exploit a wide range of SQL injection attacks in various database management systems.

SQLmap is highly configurable and allows users to specify different attack options, such as level of aggressiveness and risk. It provides detailed logs of the exploitation process, helping users understand the vulnerabilities and potential impact on the target system.

3 Vulnerability Scan

3.1 Methodology

The first step of the research after choosing the target web application consisted in selecting a web vulnerability scanner to discover SQL injection flaws. As anticipated I used Burp Suite Enterprise Edition as follows:

1. Download and install Burp Suite Enterprise Edition
2. Register and login through the URL *http://localhost:8080/*
3. Run a lightweight scan of the target website using http and https
4. After 10-15 minutes the scan is completed with all the security issues reported in a dashboard

3.2 Results

Burp Suite has shown a SQL Injection vulnerability with Severity 'High' and Confidence 'Tentative' on the target URL, specifically in the path */catalog*. This means that the *category* parameter of the products in the site appears to be vulnerable to SQL injection attacks.

Burp was able to find this flaw by submitting a first request **GET /catalog?category=Juice' HTTP/2** with a single quote in the category parameter, returning a general error message **HTTP/2 500 Internal Server Error**. Then a second request **GET /catalog?category=Juice" HTTP/2** with two single quotes were submitted and the error message disappeared (**HTTP/2 200 OK**).

However, the confidence is 'Tentative' which would indicate that Burp has detected a vulnerability but is uncertain whether it is a false positive. Therefore, this warrants further testing and verification to confirm whether the vulnerability is actually exploitable. To confirm this doubt I will use SQLmap, elaborately explained in the following paragraph.

4 SQL Attack

4.1 Methodology

After downloading SQLmap and making sure to have Python installed, We are ready to see if the vulnerability discovered is actually a true positive:

1. Open the terminal and navigate to the directory containing *sqlmap.py* using the **cd** command
2. Scan the URL target as reported in Burp Suite submitting
   py sqlmap.py -u "https://ginandjuice.shop/catalog?category=Juice" –risk=3 –level=5 –batch
3. After a couple of minutes the command is completed and fetched data are logged to a text file in the directory

The attack was made using the maximum level and risk, which means it is more aggressive and uses advanced attack techniques. The **–batch** flag runs SQLmap in batch mode, allowing the tool to be executed without any user interaction.

4.2 Results

The log file, a text file that records the details of the exploitation process performed by SQLmap, is automatically saved in the SQLmap directory and showed the following informations:

```
sqlmap identified the following injection point(s) with a total of 31 HTTP(s) requests:
---
Parameter: category (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: category=Juice' AND 8539=8539-- cnXj

    Type: UNION query
    Title: Generic UNION query (NULL) - 8 columns
    Payload: category=Juice' UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,CHAR(113)||CHAR(122)||CHAR(106)||CHAR(106)||CHAR(113)||CHAR(105)||CHAR(104)||CHAR(80)||CHAR(77)||CHAR(98)||CHAR(102)||CHAR(119)||CHAR(84)||CHAR(89)||CHAR(77)||CHAR(102)||CHAR(90)||CHAR(121)||CHAR(104)||CHAR(73)||CHAR(119)||CHAR(101)||CHAR(122)||CHAR(104)||CHAR(89)||CHAR(108)||CHAR(84)||CHAR(73)||CHAR(86)||CHAR(86)||CHAR(115)||CHAR(103)||CHAR(76)||CHAR(115)||CHAR(65)||CHAR(113)||CHAR(118)||CHAR(84)||CHAR(67)||CHAR(110)||CHAR(110)||CHAR(74)||CHAR(116)||CHAR(120)||CHAR(87)||CHAR(113)||CHAR(106)||CHAR(113)||CHAR(118)||CHAR(113),NULL,NULL-- pcjQ
---
back-end DBMS: H2
```

SQLmap found two injection points while making a total of 31 HTTP(s) requests on the target URL, a boolean-based blind and a UNION injection:

1. **Boolean-Based Blind**: the attack includes a boolean operation AND in a WHERE or HAVING clause. The Payload field provides an example of the query that SQLmap used to test the vulnerability, where the second condition will always return true even if the category is false, manipulating the database behavior. (”– cnXj” probably identifies the attack).

2. **UNION query**: SQLmap performed a generic UNION query with NULL values across eight columns, that means the number of columns in the original result is 8. UNION ALL SELECT is a SQL operation used to combine the results of two or more SELECT queries into a single result set. The number of columns must match across all SELECT queries involved, so NULL values are used as placeholders to ensure that the injected SELECT statement aligns correctly with the original. The example in the Payload shows that in the sixth column of the injected SELECT statement a set of character codes (ASCII) are concatenated using ”||” to form a string of characters using the CHAR() function. By using CHAR() function calls, the attacker constructs the string character by character (often employed to bypass input filters). So this string is used to extract data from the sixth column of the database.

”back-end DBMS: H2” indicates that the back-end Database Management System used is H2, a relational DBMS written in Java.

5 Conclusions

The exploration of SQL injection vulnerabilities within the Gin & Juice Shop application using Burp Suite Enterprise Edition and SQLmap has provided valuable insights into the importance of web security testing.

It's necessary to specify that SQL injection testing should only be conducted on systems owned by the tester or with explicit permission from the system owners.

Furthermore, leveraging Burp Enterprise in conjunction with SQLmap enhances the validation process for SQL injection vulnerabilities. This combination of tools facilitates result optimization and reduces false positives.

In conclusion, The most effective way to prevent SQL injection attacks is to use parameterized queries, also known as prepared statements, for all database access. Parameterized queries work by separating SQL code from data values, making it secure against SQL injection attacks.

References

[1] Gin & Juice Shop, `https://ginandjuice.shop/`

[2] Burp Suite Enterprise Edition, `https://portswigger.net/burp/enterprise`

[3] sqlmap®, `https://github.com/sqlmapproject/sqlmap`

[4] SQL injection, `https://portswigger.net/web-security/sql-injection`