**Performance Comparison of SQLmap and Burp Suite in Exploiting SQL Injection on DVWA**

Muhammad Izzat Farahidi¹, Afrizal Dzikri Arifah², Fazza Zevana Maheswara³, Aji Sultan Jagadnata⁴, and Luqman Hakim ⁵

¹Informatics Engineering, Universitas Muhammadiyah Malang, Malang, Indonesia 65145

a) Corresponding author: luqman.hakim@umm.ac.id

b) izzatfarahidi@webmail.umm.ac.id

c) afrizaldzikri@webmail.umm.ac.id

d) fazzazevana87@webmail.umm.ac.id

e) ajisltan31@webmail.umm.ac.id

**Abstract.** This study compares the performance of SQLmap and Burp Suite in exploiting SQL Injection vulnerabilities on DVWA (Damn Vulnerable Web Application) configured at low, medium, and high security levels. Using a comparative experimental approach, the measured parameters include exploitation success rate, execution time, and adaptability in detecting hidden vulnerabilities. The findings show that at low and medium levels, both tools successfully perform exploitation, with SQLmap clearly outperforming in terms of speed (1.66 seconds vs. Burp Suite's manual process which takes around 3 minutes at the low level). However, at the high security level, SQLmap fails to exploit hidden parameters, while Burp Suite succeeds through session variable manipulation. The study concludes that SQLmap is more effective for automation in low to medium security contexts, whereas Burp Suite offers greater flexibility in multifaceted scenarios involving hidden parameters. These results provide practical insights for security practitioners in selecting the appropriate penetration testing tools based on target complexity.

**Keywords**: SQL injection, SQLmap, Burp Suite, DVWA, penetration testing.

# INTRODUCTION

The current development of information technology has significantly driven the adoption of web-based applications in various fields, from public services to large-scale businesses. Unfortunately, this increased use of web applications is not matched by a strengthening of security aspects. One security gap that still frequently occurs is SQL Injection. SQL Injection is an attack technique that exploits weaknesses in input validation within SQL queries to gain illegal access to databases. In previous studies [1], it was confirmed that SQL Injection is included in the list of the most dangerous vulnerabilities according to the OWASP Top Ten. Furthermore, [2] showed that this attack could open access to sensitive data, such as names, email addresses, and population identification numbers (NIK), on web-based service systems developed without the application of input validation and adequate security mechanisms.

In line with this, various penetration testing methods have been developed to identify and measure the level of vulnerability of web applications. Tools such as SQLmap can be considered effective due to their ability to perform SQL Injection exploitation automatically with high accuracy and relatively fast speed [3]. Meanwhile, Burp Suite is known as a powerful web vulnerability scanner and proxy tool because it provides interactive features such as intercept, intruder, and repeater [4]. This is in line with the findings in a study by [5] which stated that SQL Injection is one of the most critical vulnerabilities requiring in-depth testing approaches, both automatically and manually through VAPT tools such as Burp Suite and SQLmap.

Previous research using the Damn Vulnerable Web Application (DVWA) platform showed that testing real attack scenarios can provide more accurate information regarding the difficulty level, time required, and impact of exploiting SQL Injection vulnerabilities with or without Web Application Firewall (WAF) protection [3]. Therefore, comparing the performance of SQLmap and Burp Suite in the same scenario will be an important step to determine their effectiveness in the context of penetration testing.

Based on this background, this research was conducted to analyze and compare the performance of SQLmap and Burp Suite in performing SQL Injection exploitation on DVWA (Damn Vulnerable Web Application). Through testing based on time metrics, the level of expertise required, and the success rate of the attack, the results of this study are expected to provide practical contributions for developers, security practitioners, and other related parties in improving the resilience of web applications against SQL Injection attacks.

## RESEARCH METHODOLOGY

## Research Method

This research uses a comparative experimental approach by conducting direct testing on the DVWA (*Damn Vulnerable Web Application*) using two popular penetration testing tools, namely SQLmap and Burp Suite. The main objective of this experiment is to measure and compare the effectiveness of these two tools in executing SQL Injection attacks at three levels of application security settings: low, medium, and high.

## Test Environment

**TABLE I.** Virtual Environment Configuration

|  |  |
| --- | --- |
| Component | Description |
| Target Platform | DVWA v1.10-dev |
| Operating System | Kali Linux |
| Web Server | Apache 2.4.25 (Debian 9) |
| Database Management | MySQL (MariaDB fork) |
| Browser | Chromium (via Burp Suite proxy) |
| Tools | SQLmap v1.9.2 and Burp Suite Community Edition |
| Test Time | Juni 29, 2025 |

## Techniques and Testing Tools

### SQLmap

SQLmap is an automated penetration tool for exploiting SQL Injection vulnerabilities, highly popular among security practitioners. In this study, SQLmap was used to detect and execute various injection techniques automatically. Previous studies have compared several SQL Injection tools and concluded that SQLmap excels in functionality and automatic extraction speed [6]. Clarke & Furnell (2022) in their research emphasized that SQLmap has a powerful detection engine and various special features that make it the ultimate penetration testing tool with a broad range of switches and options [7]. This tool can exploit databases and obtain sensitive data quickly, even used in simulations of server takeover via SQL Injection [8]. The options and parameters applied include:

* --batch: Used to skip all interactive prompts so the exploitation process can run automatically without manual intervention.
* --dbs and --dump: Function to display a list of databases connected to the target application and extract the contents of related tables as proof of a successful attack.
* *Tamper script* usage: Used if the target has certain input filter or protection mechanisms. Tamper scripts can also help modify payloads to bypass filters.
* *Exploitation mode*: SQLmap will automatically try various injection techniques, such as error-based, Boolean, time-based, and UNION. These techniques allow SQLmap to detect and exploit SQL injection gaps according to the target input conditions.

With this configuration, SQLmap was used to measure automatic exploitation capabilities, including how quickly it can detect and extract sensitive data from DVWA at different security levels [8, 6].

### Burp Suite

Unlike the automatic SQLmap, Burp Suite was used manually to perform SQL Injection exploitation. Burp Suite is a proxy web vulnerability scanner tool with various features for analyzing HTTP/HTTPS traffic. Malik et al. (2022) explain that Burp Suite can be integrated with machine learning algorithms to improve vulnerability detection capabilities, including SQL injection, Cross-Site Request Forgery, and XML External Entity vulnerabilities [9]. The features used in this study include:

* *Repeater*: Functions to test payloads directly, where researchers can repeatedly modify HTTP requests with different parameters and then send them to the server to see the response.
* *Intruder*: Used to automate the sending of payload variations, thus helping to find combinations of parameters that are vulnerable to exploitation.
* *Manual injection techniques*: Exploitation is done by manually inputting payloads into GET or POST parameters. The UNION SELECT technique was used to combine the results of injection queries with original data, while session variable analysis was used to identify hidden parameters that could be exploited at the high-security level.

The use of Burp Suite focuses on human ability to analyze request and response patterns, making it more flexible for detecting gaps that may not be read by automated tools [10]. Recent research also shows that Burp Suite is very effective for detecting security gaps such as SQL Injection and XSS in modern web-based applications.

## Testing Procedure

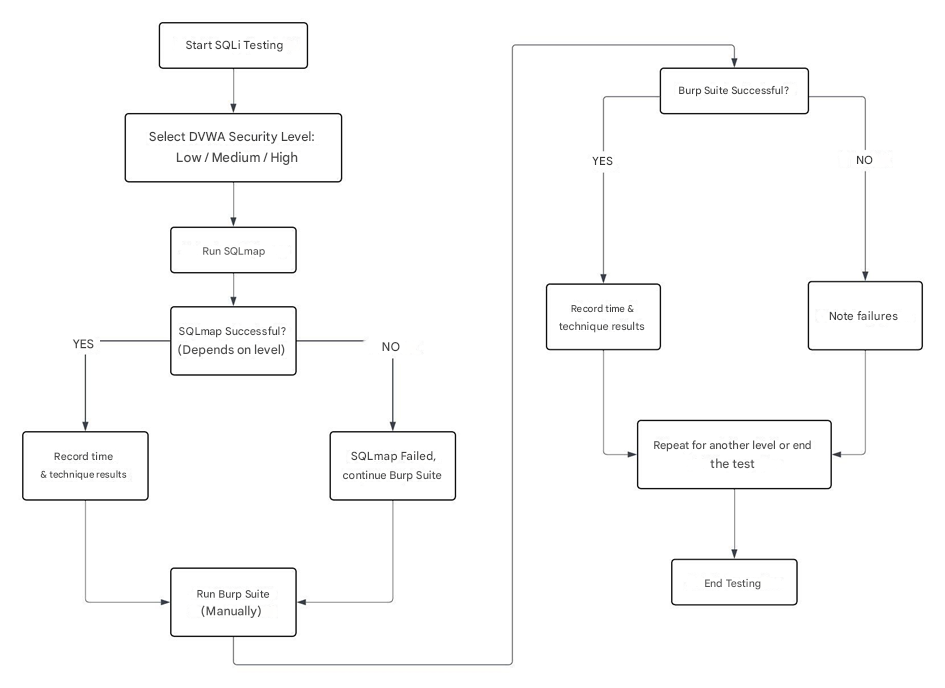
The stages of the testing procedure in this study were carried out systematically as follows:

1. Adjusting the DVWA security level: DVWA was set at three security levels to represent variations in input protection levels commonly found in real web applications.
2. Performing SQL Injection exploitation tests at each level:
   * SQLmap was used to perform exploitation automatically with parameters adjusted to the target, including the use of POST or GET methods.
   * Burp Suite was used manually through the Repeater and Intruder features with payload variations sent iteratively.
3. Recording test results in the form of:
   * Attack success status (successful/unsuccessful)
   * Automatic execution time (SQLmap) and manual process time (Burp Suite)
   * Exploitation techniques used (error-based, time-based, UNION, etc.)
   * Payload or query used
4. Analyzing the performance comparison of the two tools in terms of effectiveness (success/failure), efficiency (speed of time), and flexibility in complex conditions (hidden parameters, session variables, etc.). This analysis was conducted by comparing the two tools in terms of:
   * Effectiveness: The extent to which the tool can exploit vulnerabilities at each security level.
   * Time Efficiency: Execution speed and the need for manual intervention.
   * Flexibility: The ability to detect hidden gaps, such as hidden parameters or session variables.

## Measurement Parameters

To compare the performance of the two tools, several parameters were used as follows:

* Exploitation success: Measured by the tool's ability to display or extract (dump) sensitive data from the database.
* Automatic/Manual execution time: Measured from the time required from payload delivery until the final result is obtained.
* Ability to read hidden/session parameters: Measured by the extent to which the tool can exploit hidden variables or sessions that are not directly accessible.
* Flexibility level: Assesses the tool's ability to adapt the attack approach to different input protection conditions.



**FIGURE 1.** Research flowchart.

# RESULTS AND DISCUSSION

## Test Results

Testing was conducted on three security levels on DVWA, namely Low, Medium, and High, using two tools: SQLmap (automatic) and Burp Suite (manual). The test results focused on three main parameters: exploitation success, execution time, and flexibility in detecting hidden vulnerabilities.

**TABLE II.** Security Level: Low

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tool | Success | ExecTime | ProcTime | Technique | Payload / Brief Output |
| SQLmap | ✅ Yes | **1.66s** | Automatic | Full Auto | Detected 4 techniques: boolean, error, time, UNION; dump dvwa DB |
| Burp Suite | ✅ Yes | **1.34s** | ± 3 min | Manual UNION | 1' OR '1'='1 → successfully displayed all user data |

At this level, input validation is very weak. SQLmap can automatically detect and execute SQL Injection attacks in 1.66 seconds. Four techniques were successfully detected: boolean-based, error-based, time-based, and UNION-based. Burp Suite also successfully exploited the vulnerability with a simple payload (1' OR '1'='1) via the *Repeater* feature. The response time was only 1.34 seconds, but the overall manual process took approximately ±3 minutes.

**TABLE III.** Security Level: Medium

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tool | Success | ExecTime | ProcTime | Technique | Brief Output |
| SQLmap | ✅Yes | ± 2 min total | Automatic | Error, Time, UNION-based | Successfully dump dvwa.users table contents, crack hash |
| Burp Suite | ✅ Yes | 1.28s | ± min | Manual payload via Repeater | UNION query successfully displayed user+hash |

At this level, inputs begin to be filtered using the POST method. SQLmap still managed to exploit the vulnerability by including the parameters --data="id=1&Submit=Submit" and --flush-session. The process took approximately ±2 minutes due to the number of techniques tested (error-based, time-based, UNION-based). SQLmap also successfully extracted data and cracked some password hashes using dictionary attacks. Meanwhile, Burp Suite required a manual approach using *Repeater* and encoded payloads. The initial response time was only 1.28 seconds, but the identification and injection process took ±5 minutes. The UNION payload successfully displayed a combination of usernames and password hashes.

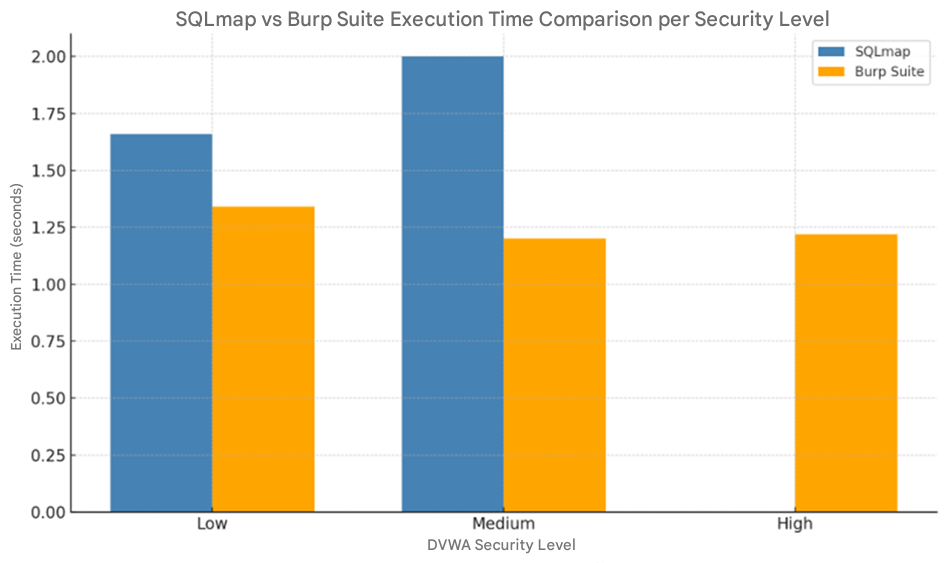
**TABLE IV.** Security Level: High

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tool | Success | ExecTime | ProcTime | Technique | Notes |
| SQLmap | ❌ No | - | - | No param detected | Parameter id not available in GET/POST |
| Burp Suite | ✅ Yes | 1.30s | ± 10 min | Manual UNION in POST | Can inject via session-input.php |

At this level, the parameters used for SQL queries are not directly available in GET or POST. SQLmap failed to perform exploitation because the id parameter is only set via the $\_SESSION['id'] variable on the server side and cannot be detected by automated tools. Conversely, Burp Suite successfully performed injection by modifying hidden parameters on the session-input.php page. The UNION query technique was used and successfully extracted usernames and password hashes through manipulated POST data. The total process took approximately ±10 minutes.

**TABLE V.** Comparative Summary

|  |  |  |
| --- | --- | --- |
| Level | SQLmap (Automatic) | Burp Suite (Manual) |
| Low | ✅ Very Effective (1.66s) | ✅ Very Effective (1.34s) |
| Medium | ✅ Effective (needs POST+flush) | ✅ Effective (needs encoding) |
| High | ❌ Cannot | ✅ Can exploit SESSION |



**FIGURE 2.** Security level comparison graph.

## Discussion

*Effectiveness of Tools*

In general, both tools were able to exploit SQL Injection well at Low and Medium levels. However, at the High level, only Burp Suite succeeded, due to its flexibility in manually tracing and modifying HTTP requests. SQLmap relies on explicit parameters and is unable to handle hidden server-side variables like $\_SESSION.

*Time Efficiency*

SQLmap showed superiority in time efficiency, especially in Low and Medium level scenarios. By running just one command, this tool is capable of automatic detection, exploitation, and data extraction. Conversely, Burp Suite requires more intensive manual involvement, even though its HTTP response time is very fast.

*Exploitation Flexibility*

Burp Suite showed high flexibility especially at the high-security level. The Repeater and Intruder features allow granular testing of each parameter, including hidden ones. This is very useful in cases where SQLmap cannot identify the injection point due to the absence of explicit parameters in the URL or form.

# CONCLUSION

In testing SQL Injection exploitation against DVWA with SQLmap and Burp Suite, these two tools have their respective characteristics and advantages. At low and medium security levels, both tools achieved 100% exploitation success. SQLmap began to show a clear advantage over Burp Suite due to its efficiency in automation. Although Burp Suite's response time was a consistent 1.3 seconds, it required a longer manual time between 3-10 minutes. A significant further difference was observed at the high level where SQLmap failed in exploitation due to undetected session variables. While advanced techniques could not be bypassed by SQLmap, Burp succeeded in exploiting through session variables on session-input.php. This shows that Burp is superior in complex hidden parameter scenarios, whereas for explicit automation parameters, SQLmap proves better. These findings suggest that for lower security levels, SQLmap excels in routine testing and automation, while at higher levels it requires thorough analysis using session mechanisms with deep automation by Burp.

# Acknowledgments

The authors would like to thank Universitas Muhammadiyah Malang for supporting this research.

# References

1. A. Djajadi and N. Sutisna, Internetworking Indonesia Journal 13, 3–9 (2021).
2. Khairul, A. Abdullah, and S. Sucipto, Jurnal Nasional Teknologi Dan Sistem Informasi 11, 78–86 (2025).
3. A. W. Pranowo, A. Widjajarto, and M. Fathinuddin, Kesatria : Jurnal Penerapan Sistem Informasi (Komputer Dan Manajemen) 4, 841–851 (2023).
4. P. Ranjan et al., in SpringerBriefs in Applied Sciences and Technology (Springer International Publishing, Cham, 2025), pp. 125–133.
5. U. Ravindran and R. V. Potukuchi, Review of Computer Engineering Studies 9, 1–22 (2022).
6. P. Gio et al., in Proceedings of the 4221st Conference (2022), pp. 286–292.
7. J. Clarke and S. Furnell, Computers & Security 114, 102298 (2022).
8. R. Hermawan, Jurnal Teknik Informatika dan Sistem Informasi 6, (2021).
9. A. Malik, S. Kumar, and R. Singh, International Journal of Advanced Computer Science and Applications 13, (2022).
10. L. Agustina et al., Journal of Information Security Research 4, 16–21 (2025).
11. M. Baklizi et al., International Journal of Intelligent Systems and Applications in Engineering 10, 75–85 (2022).
12. R. Thaqi, K. Vishi, and B. Rexha, Journal of Applied Security Research 18, 789–807 (2023).