15TH JANUARY 2026

VULNERABILITY ASSESSMENT REPORT FOR DAMN VULNERABLE WEB APPLICATION

Chiluka Varshith Reddy

Table Of Contents

Confidentiality Statement

This document is the exclusive property of the author and is prepared solely for educational and learning purposes as part of the Future Interns Cyber Security Internship Program. The contents of this report must not be redistributed, reproduced, or used for malicious purposes. This document may be shared with academic reviewers or mentors for evaluation and learning purposes only.

Disclaimer

This assessment was conducted on an intentionally vulnerable demonstration application (Damn Vulnerable Web Application - DVWA) hosted on a public training platform. All testing activities were performed ethically within a controlled environment for educational purposes only. The vulnerabilities identified in this report do not represent real-world security flaws in any production system or organization.

Contact Information

Name: Chiluka Varshith Reddy
Role: Cyber Security Intern
Program: Future Interns Cyber Security Internship
Email: chilukavarshith4005@gmail.com

Assessment Overview

This vulnerability assessment was conducted to identify common security weaknesses in a web application environment using safe, controlled, and non-destructive testing techniques.

The objective of this assessment is to provide clear visibility into potential risks, explain why they matter from a business perspective, and suggest practical remediation steps to improve the security posture of the application. The focus of this report is not exploitation, but risk awareness and security improvement.

Assessment Components

The assessment included the following activities:

- Application reconnaissance and navigation
- Identification of vulnerable input points
- Verification of vulnerabilities through controlled testing
- Proof-of-concept demonstration
- Risk classification and reporting

Tools Used:

- Web Browser (Manual Testing)
- Burp Suite Community Edition (optional)
- Browser Developer Tools

External Vulnerability Assessment

An external penetration test simulates the actions of an attacker attempting to identify and exploit vulnerabilities from outside the application without internal system access. This assessment focused on externally accessible functionality within the DVWA application and did not include any internal infrastructure or server-level testing.

Finding Severity Ratings

Severity	Description
Critical	Complete compromise or unauthorized system access
High	Significant impact on confidentiality, integrity, or availability
Medium	Moderate security risk exploitable under certain conditions
Low	Minor security weakness with limited impact

Scope

In Scope:

- Web application: Damn Vulnerable Web Application (DVWA) — training environment
- Publicly accessible application functionality
- Common web application vulnerabilities based on OWASP Top 10 categories

Out of Scope:

- Denial of Service (DoS) testing
- Testing on real business or production systems
- Data extraction beyond minimal proof
- Exploitation for persistence, privilege escalation, or lateral movement

Executive Summary

A vulnerability assessment was conducted on the Damn Vulnerable Web Application (DVWA) to identify common security weaknesses in a controlled and legal environment. Several vulnerabilities were identified, including injection flaws, cross-site scripting, and improper request validation. These vulnerabilities could allow attackers to manipulate application behavior, access unauthorized data, and compromise user sessions. This report documents the identified vulnerabilities along with their potential impact and recommended remediation steps.

Attack Summary

Sl. No	Action	Recommendation	Severity
1	The application is vulnerable to SQL Injection	Implement parameterized queries and input validation	Critical
2	The application is vulnerable to Reflected Cross-Site Scripting (XSS)	Apply output encoding and input validation	High
3	The application is vulnerable to Stored Cross-Site Scripting (XSS)	Encode stored output and sanitize user input	High
4	The application is vulnerable to Command Injection	Avoid executing system commands with user input and validate input	Critical
5	The login functionality allows brute-force attacks	Limit login attempts and implement rate limiting	High

Penetration test findings

1) SQL Injection (Critical)

Description

It was possible to manipulate backend database queries by modifying user input in the SQL Injection module of the application. The application returned multiple unintended database records instead of a single expected result, indicating that user input is directly incorporated into SQL queries without proper validation or sanitization.

Impact

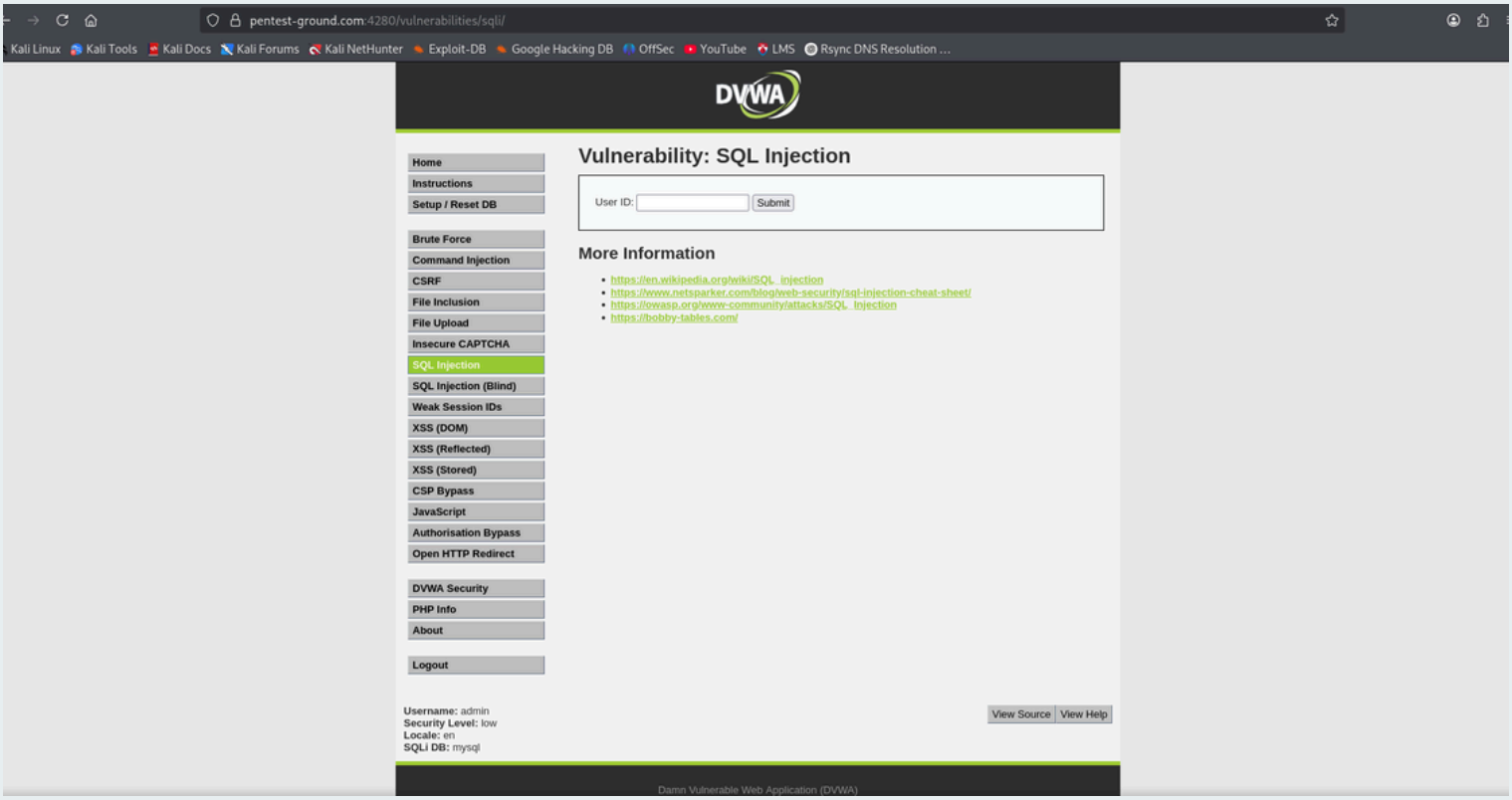
Critical — This vulnerability could allow an attacker to access sensitive database information, bypass application logic, and compromise the confidentiality and integrity of stored data.

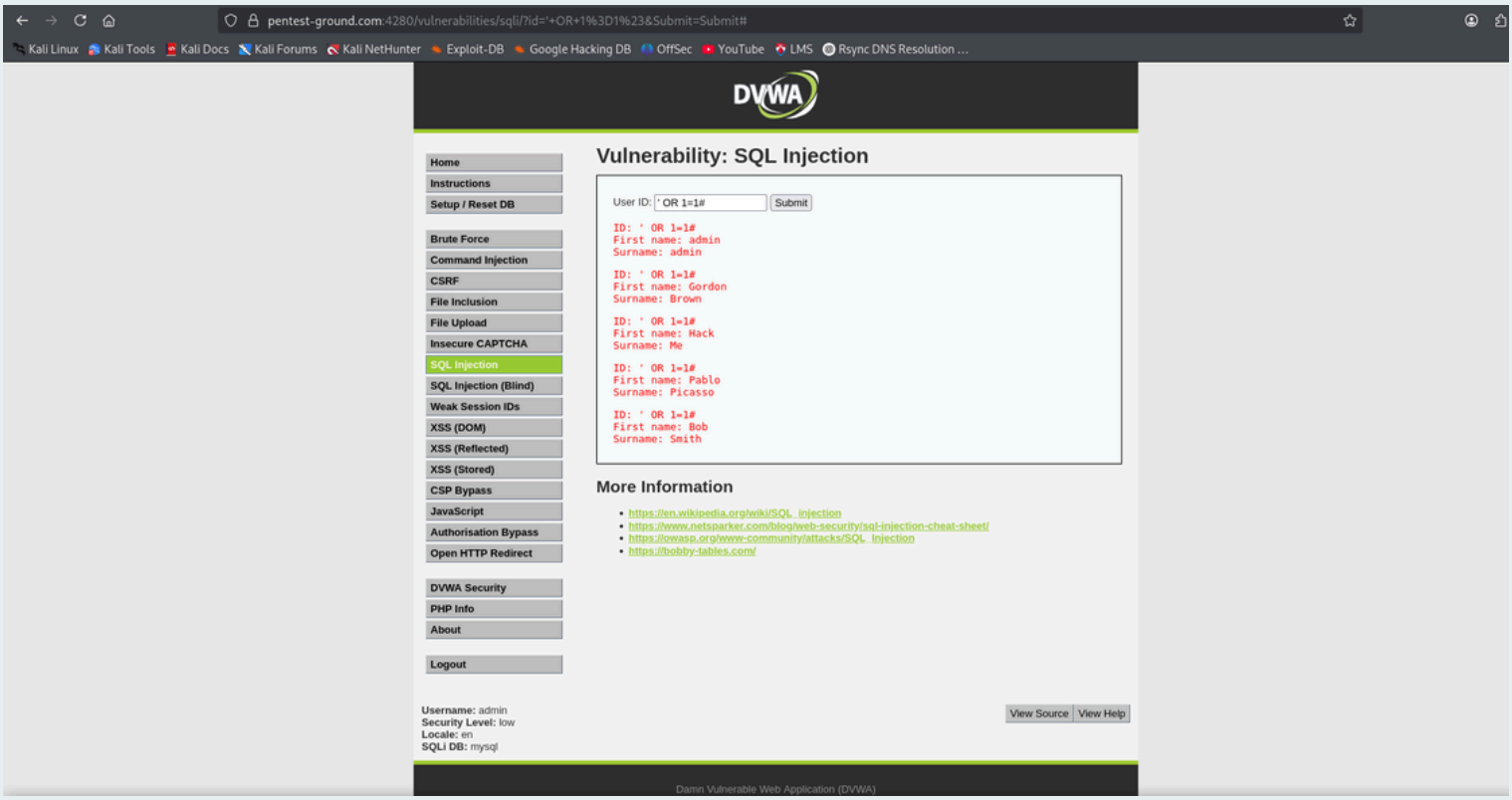
System

Damn Vulnerable Web Application (DVWA) — SQL Injection Module

Proof of concept:

The application returned multiple database records when user input was manipulated, demonstrating that backend SQL query logic is influenced by unsanitized user input.


pentest-ground.com:4280/vulnerabilities/sqli/
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: SQL Injection
User ID: Submit
More Information
https://en.wikipedia.org/wiki/SQL injection
https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
https://owasp.org/www-community/attacks/SQL Injection
https://bobby-tables.com/
Username: admin
Security Level: low
Locale: en
SQLi DB: mysql
View Source View Help
Damn Vulnerable Web Application (DVWA)


pentest-ground.com:4280/vulnerabilities/sqli/?id='+OR+1%3D1%23&Submit=Submit#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: SQL Injection
User ID: ' OR 1=1# Submit
ID: ' OR 1=1#
First name: admin
Surname: admin
ID: ' OR 1=1#
First name: Gordon
Surname: Brown
ID: ' OR 1=1#
First name: Hack
Surname: Me
ID: ' OR 1=1#
First name: Pablo
Surname: Picasso
ID: ' OR 1=1#
First name: Bob
Surname: Smith
More Information
https://en.wikipedia.org/wiki/SQL injection
https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/
https://owasp.org/www-community/attacks/SQL Injection
https://bobby-tables.com/
Username: admin
Security Level: low
Locale: en
SQLi DB: mysql
View Source View Help
Damn Vulnerable Web Application (DVWA)

Recommendation:

- Use parameterized queries or prepared statements for all database operations.
- Validate and sanitize all user input on the server side.
- Restrict database user privileges to the minimum required.

2) Reflected Cross-Site Scripting (XSS) (High)

Description

It was possible to inject and execute client-side script through user input in the reflected XSS module of the application. The input provided by the user was reflected back in the server response without proper output encoding or sanitization, allowing execution of arbitrary script in the user’s browser.

Impact

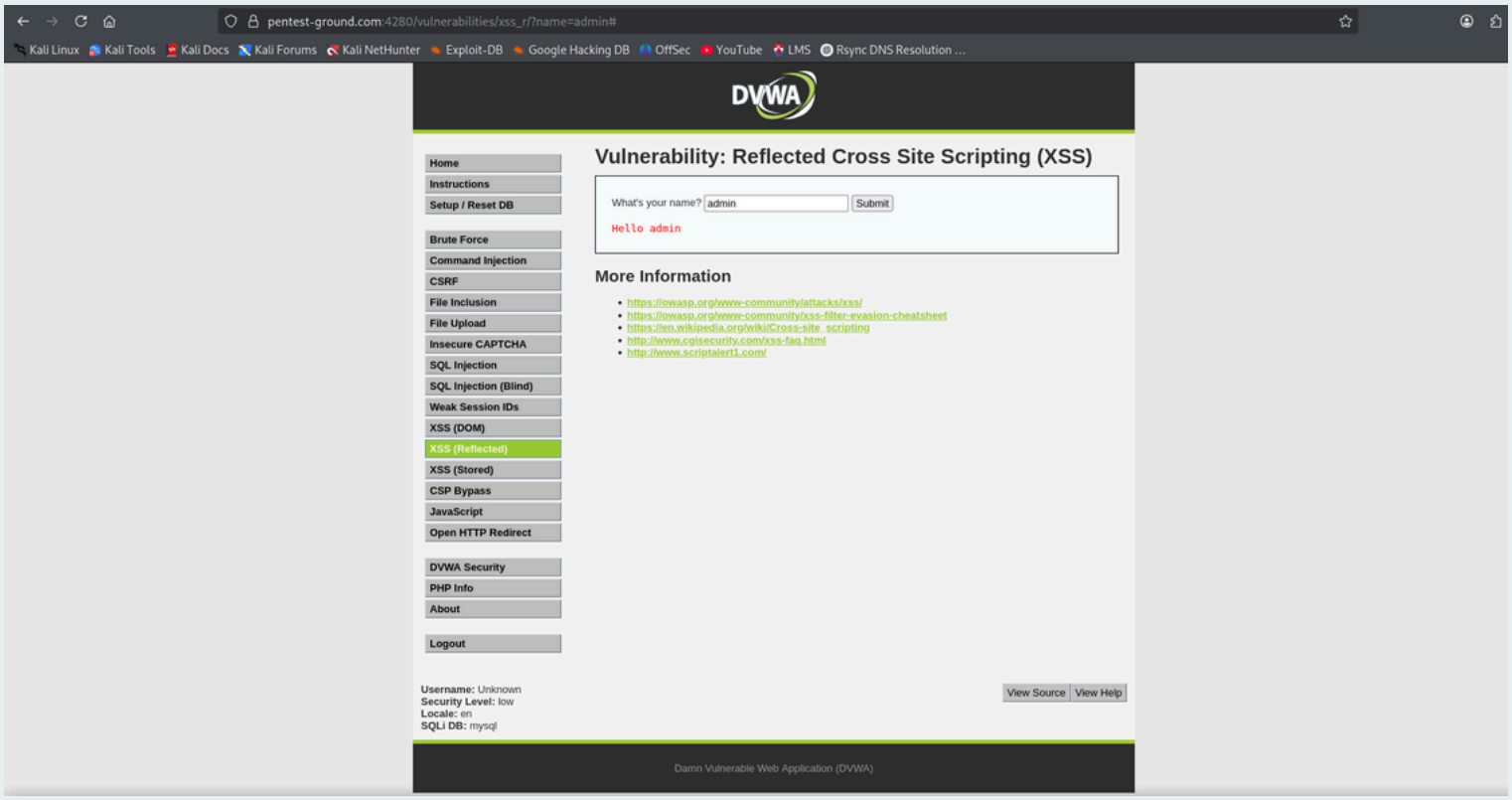
High — An attacker could execute malicious scripts in the context of another user’s browser, potentially allowing theft of session information, manipulation of page content, redirection to malicious sites, or impersonation of users.

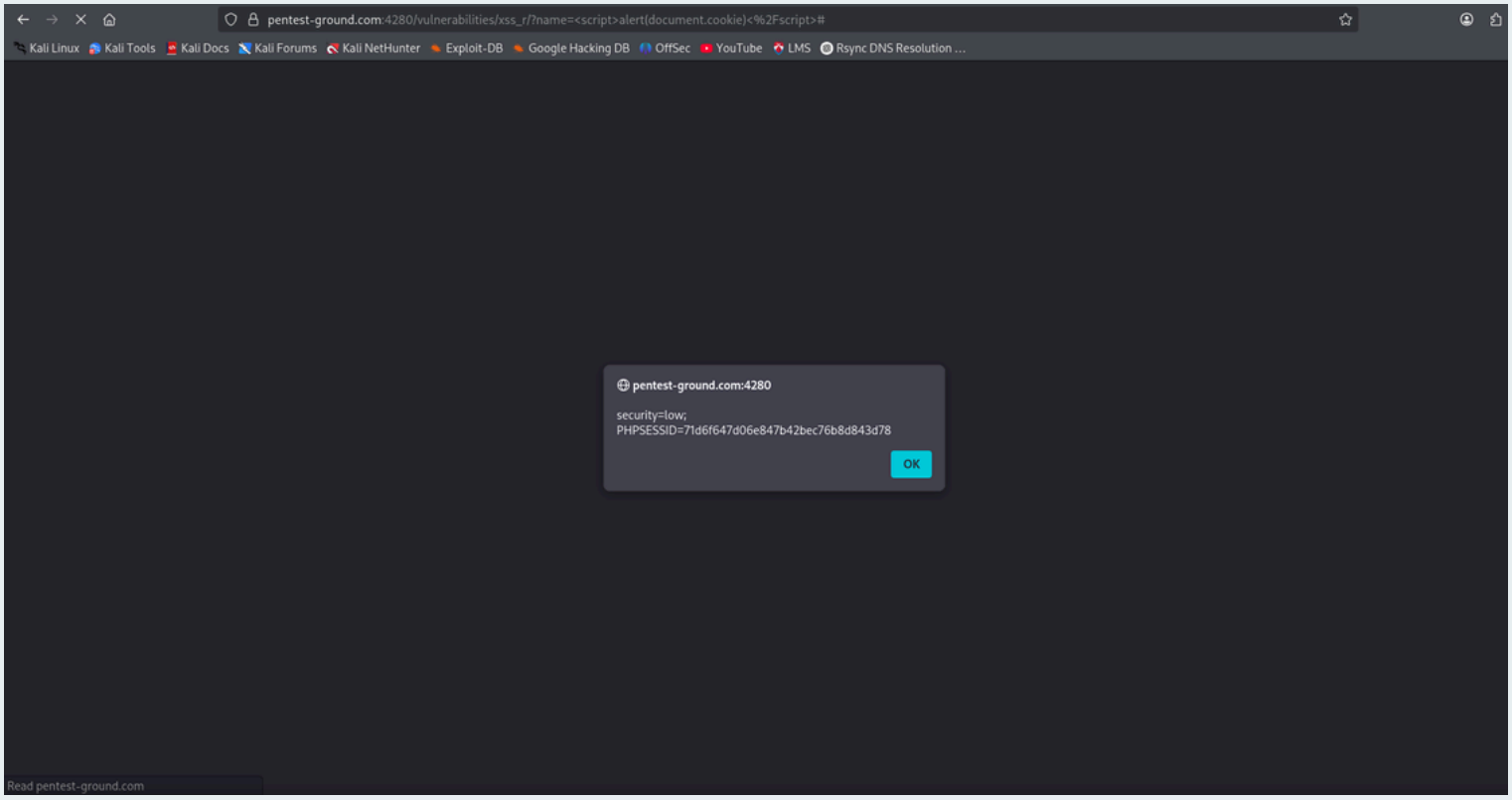
System

Damn Vulnerable Web Application (DVWA) — XSS (Reflected) Module

Proof of concept:

User input was reflected directly into the response and executed as client-side script, confirming that the application does not properly encode or sanitize reflected input.


pentest-ground.com:4280/vulnerabilities/xss_r/?name=admin#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: Reflected Cross Site Scripting (XSS)
What's your name? admin
Submit
Hello admin
More Information
https://owasp.org/www-community/attacks/xss/
https://owasp.org/www-community/xss-filter-evasion-cheatsheet
https://en.wikipedia.org/wiki/Cross-site_scripting
http://www.cgisecurity.com/xss-faq.html
http://www.scriptalert1.com/
Username: Unknown
Security Level: low
Locale: en
SQLi DB: mysql
View Source
View Help
Damn Vulnerable Web Application (DVWA)


pentest-ground.com:4280/vulnerabilities/xss_r/?name=<script>alert(document.cookie)<%2Fscript>#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
pentest-ground.com:4280
security=low;
PHPSESSID=71d6f647d06e847b42bec76b8d843d78
OK
Read pentest-ground.com

Recommendation:

- Encode all user-supplied input before rendering it in HTML output.
- Use context-aware output encoding libraries.
- Implement a Content Security Policy (CSP) to reduce the impact of XSS.
- Validate input on the server side and restrict special characters where appropriate.

3) Stored Cross-Site Scripting (XSS) (High)

Description

It was possible to inject client-side script into the application through the guestbook functionality in the Stored XSS module. The application stores user input on the server and later renders it back to users without proper output encoding or sanitization, allowing persistent execution of arbitrary script.

Impact

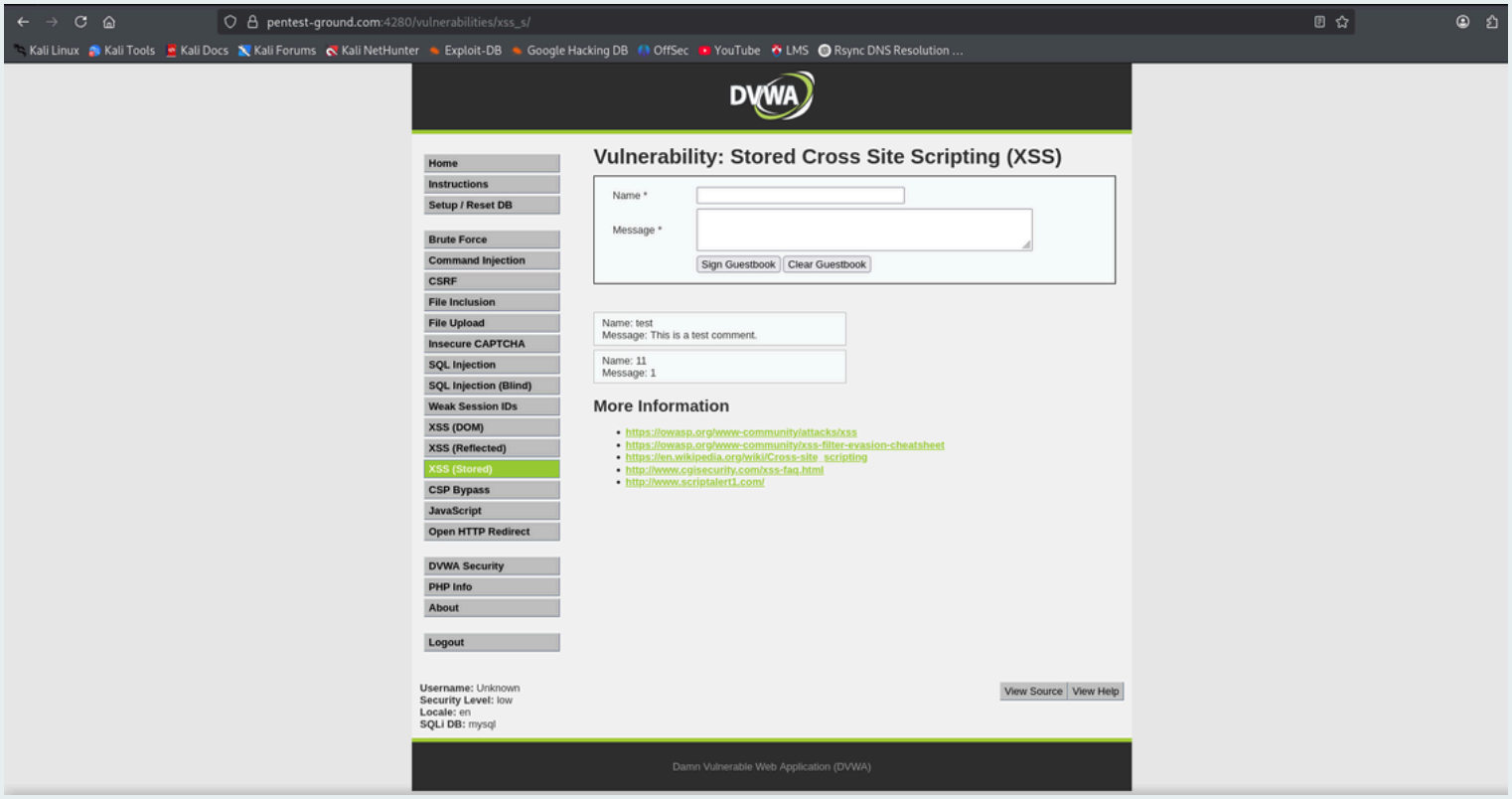
High — An attacker could store malicious scripts that execute in the browsers of all users who view the affected page. This could lead to session hijacking, user impersonation, defacement, or redirection to malicious websites.

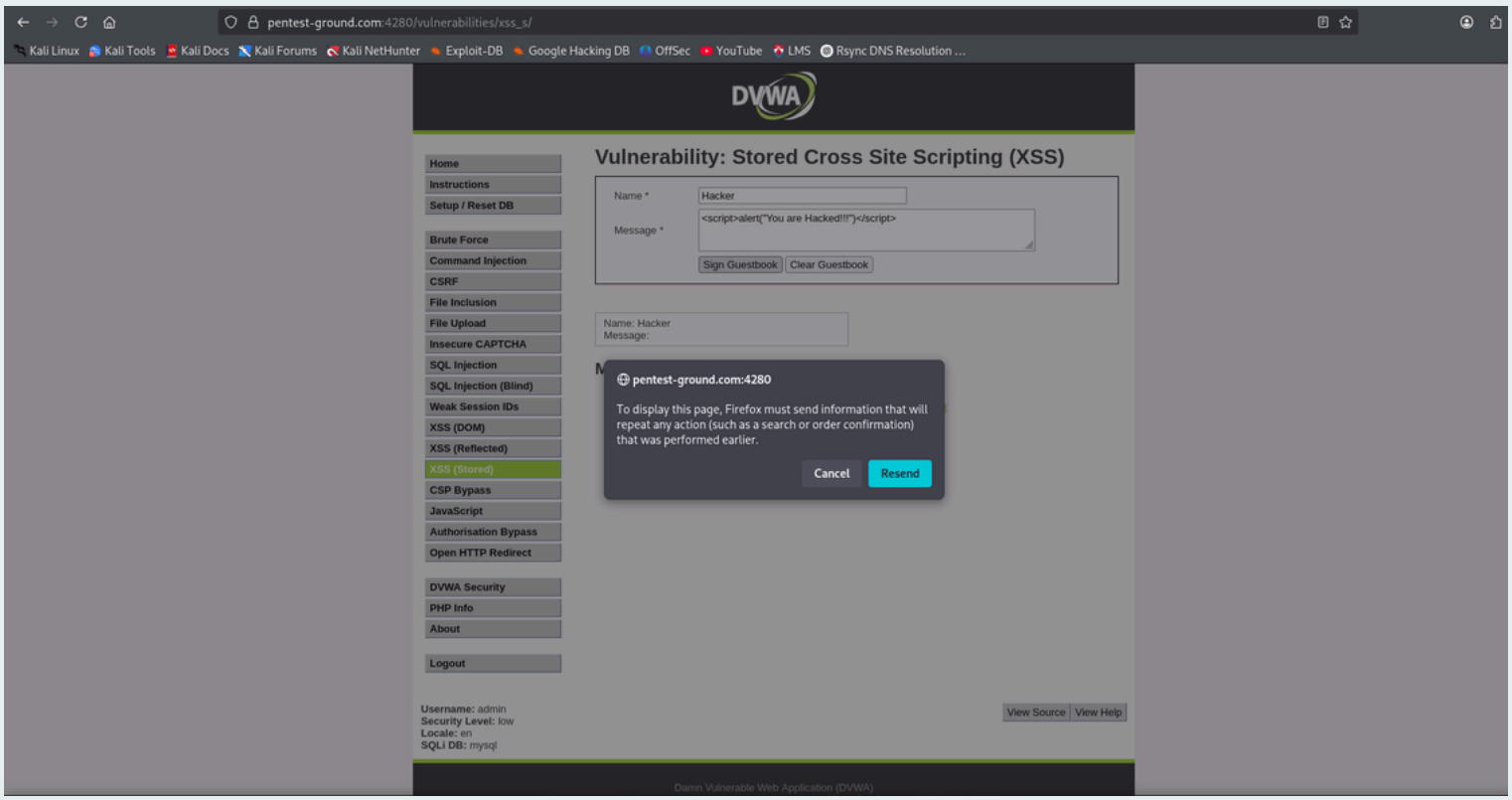
System

Damn Vulnerable Web Application (DVWA) — XSS (Stored) Module

Proof of concept:

User input submitted through the guestbook persisted on the server and was rendered back to the page without encoding, causing the browser to execute the injected script upon viewing the page.


pentest-ground.com:4280/vulnerabilities/xss_s/
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: Stored Cross Site Scripting (XSS)
Name *
Message *
Sign Guestbook
Clear Guestbook
Name: test
Message: This is a test comment.
Name: 11
Message: 1
More Information
https://owasp.org/www-community/attacks/xss
https://owasp.org/www-community/xss-filter-evasion-cheatsheet
https://en.wikipedia.org/wiki/Cross-site_scripting
http://www.cgisecurity.com/xss-faq.html
http://www.scriptalert1.com/
Username: Unknown
Security Level: low
Locale: en
SQLi DB: mysql
View Source
View Help
Damn Vulnerable Web Application (DVWA)


pentest-ground.com:4280/vulnerabilities/xss_s/
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: Stored Cross Site Scripting (XSS)
Name *
Hacker
Message *
<script>alert("You are Hacked!!!")</script>
Sign Guestbook
Clear Guestbook
Name: Hacker
Message:
pentest-ground.com:4280
To display this page, Firefox must send information that will repeat any action (such as a search or order confirmation) that was performed earlier.
Cancel
Resend
Username: admin
Security Level: low
Locale: en
SQLi DB: mysql
View Source
View Help
Damn Vulnerable Web Application (DVWA)

Recommendation:

- Encode all output before rendering it in the browser (HTML entity encoding).
- Validate and sanitize all user input on the server side.
- Implement a Content Security Policy (CSP) to reduce XSS impact.
- Avoid storing untrusted content without proper filtering.

4) Command Injection (Critical)

Description

It was possible to inject operating system commands through user input in the Command Injection module of the application. The application passes user-supplied input directly to a system-level command without proper validation or sanitization, allowing execution of unintended system commands.

Impact

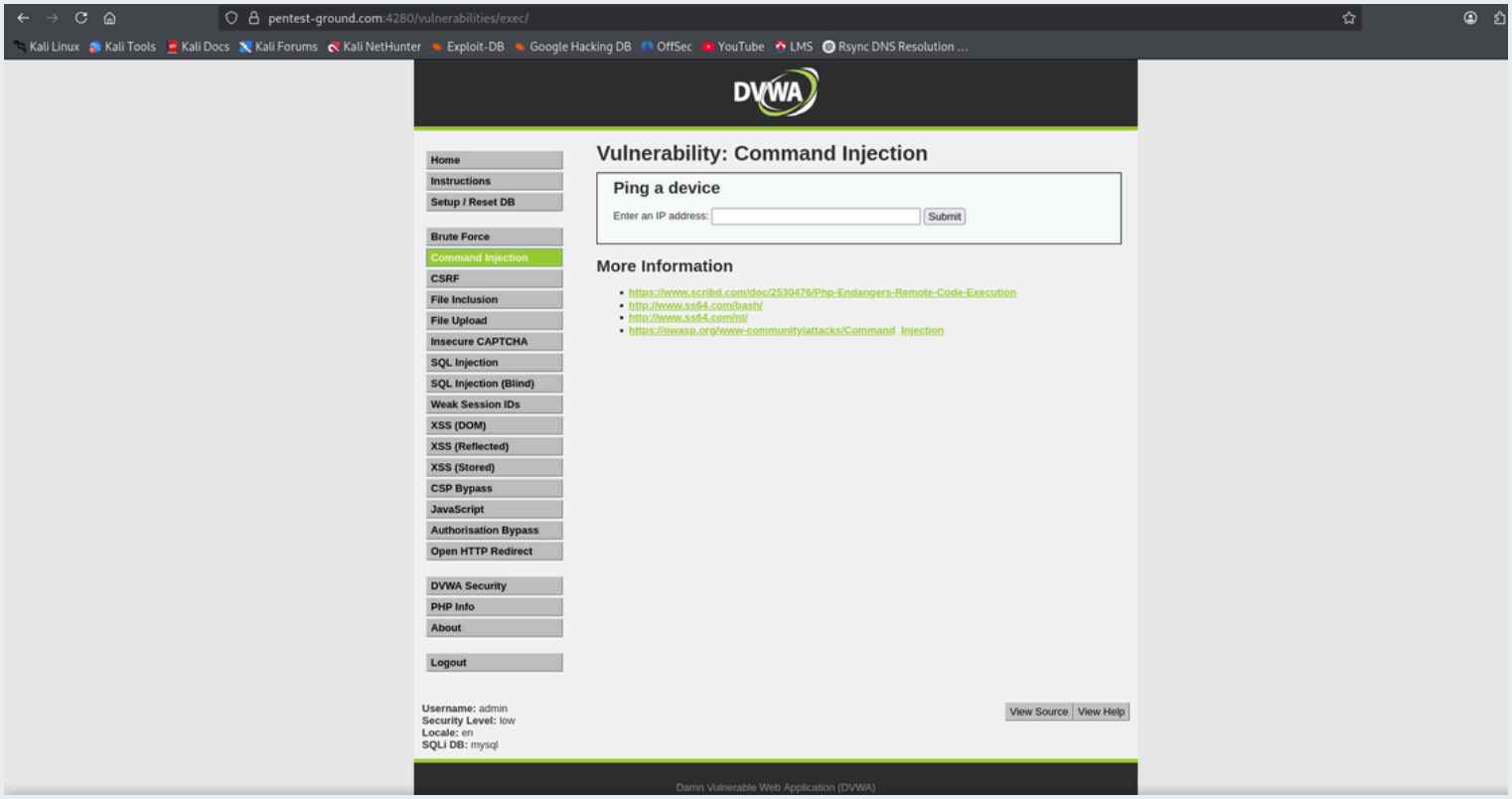
Critical — An attacker could execute arbitrary commands on the server, potentially leading to full system compromise, data theft, data destruction, or service disruption.

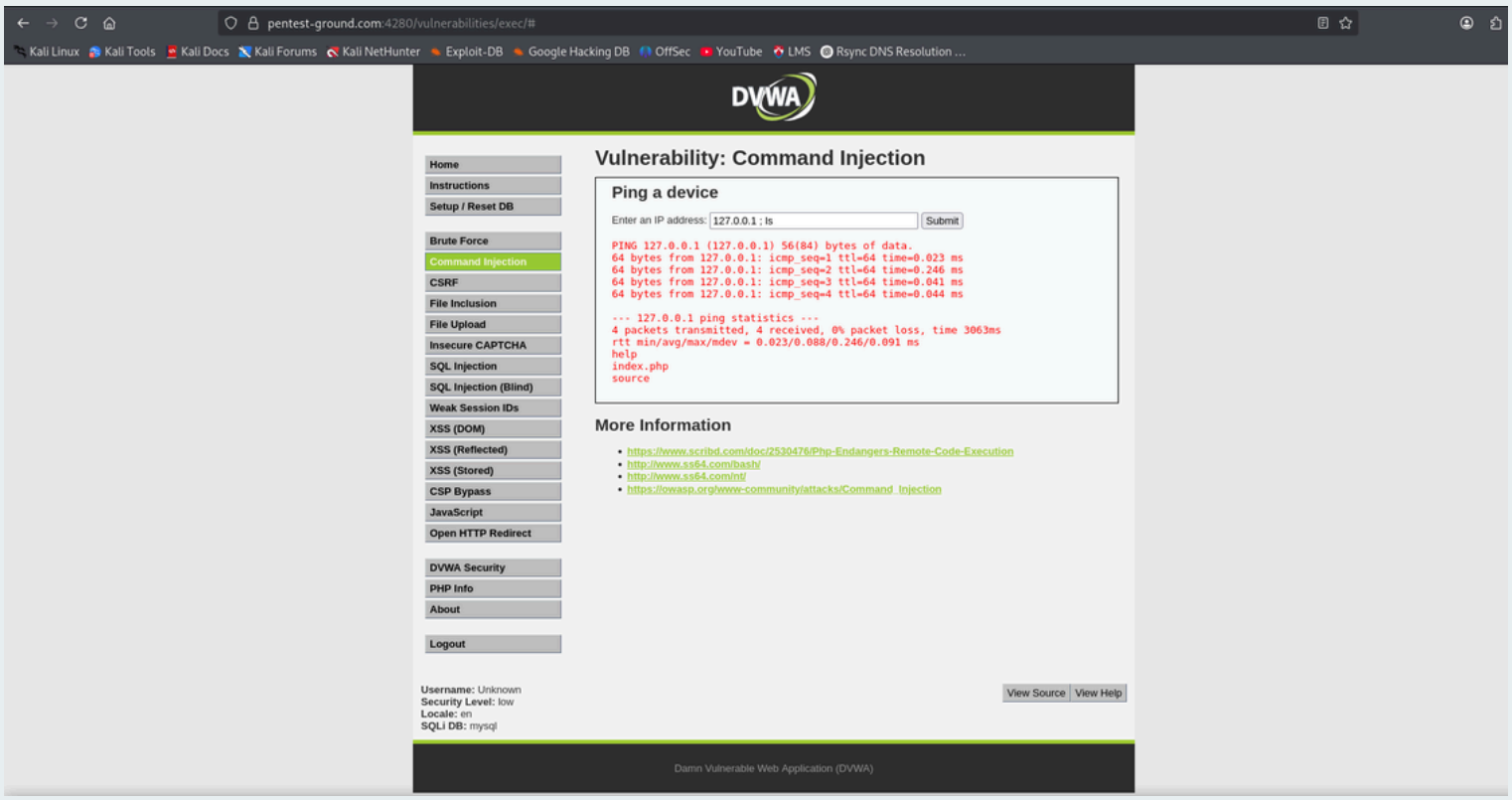
System

Damn Vulnerable Web Application (DVWA) — Command Injection Module

Proof of concept:

User input was processed by the underlying operating system and resulted in output that was not part of the intended application functionality, demonstrating that user input is directly executed at the system level.


pentest-ground.com:4280/vulnerabilities/exec/
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: Command Injection
Ping a device
Enter an IP address:
Submit
More Information
https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/
https://owasp.org/www-community/attacks/Command_Injection
Username: admin
Security Level: low
Locale: en
SQLi DB: mysql
View Source
View Help
Damn Vulnerable Web Application (DVWA)


pentest-ground.com:4280/vulnerabilities/exec/#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: Command Injection
Ping a device
Enter an IP address: 127.0.0.1 ; ls
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=0.023 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.246 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.041 ms
64 bytes from 127.0.0.1: icmp_seq=4 ttl=64 time=0.044 ms
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3063ms
rtt min/avg/max/mdev = 0.023/0.088/0.246/0.091 ms
help
index.php
source
More Information
https://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/
https://owasp.org/www-community/attacks/Command_Injection
Username: Unknown
Security Level: low
Locale: en
SQLi DB: mysql
View Source
View Help
Damn Vulnerable Web Application (DVWA)

Recommendation:

- Never pass user input directly to system commands.
- Use safe APIs that do not invoke shell execution.
- Validate and strictly filter all user input.
- Run system commands with minimal privileges or avoid them entirely.

5) Brute Force Attack Possible (High)

Description

The login functionality does not implement any protection mechanisms against repeated authentication attempts. The application allows unlimited login attempts without account lockout, rate limiting, CAPTCHA, or delay mechanisms.

Impact

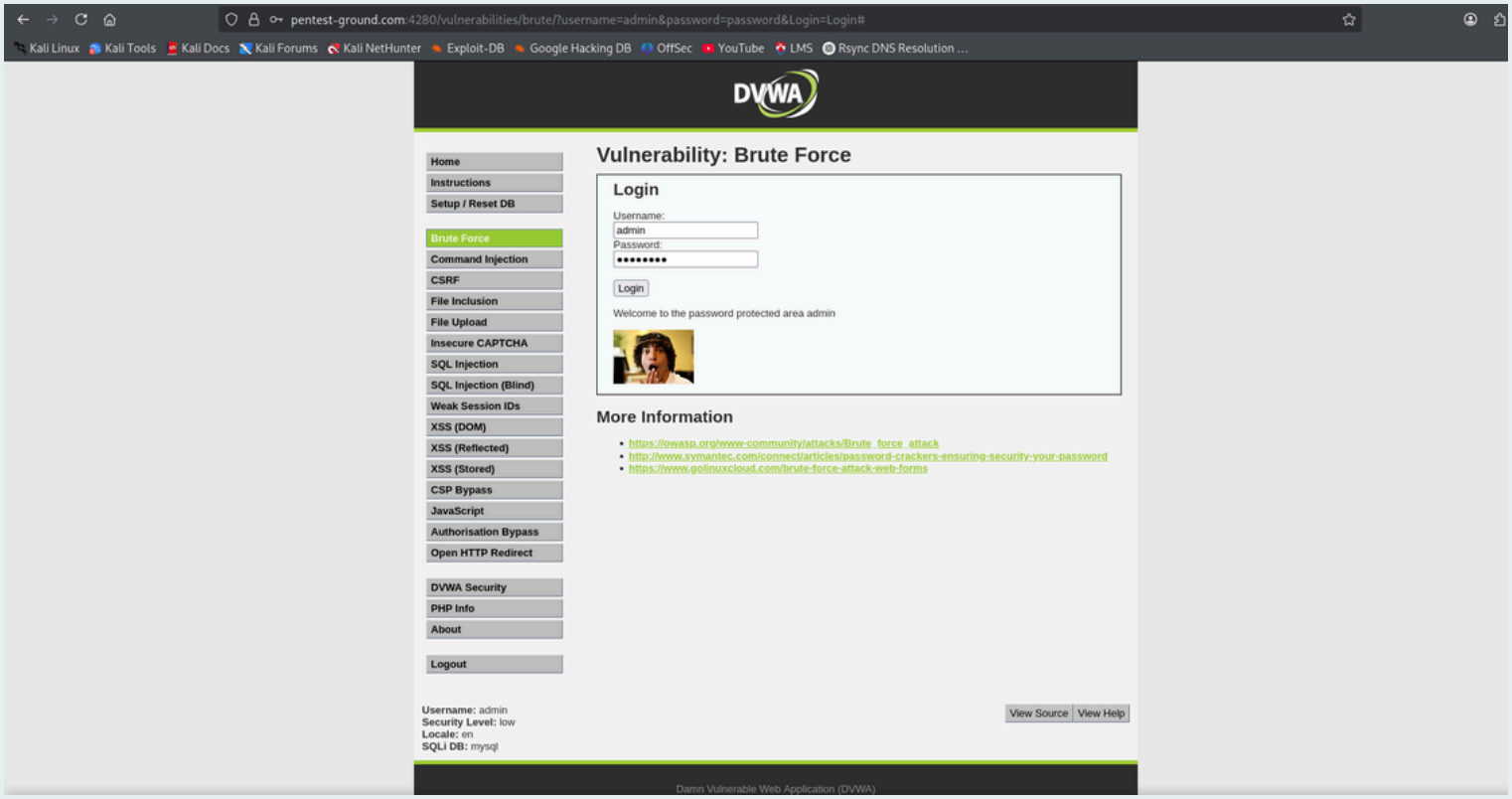
High — An attacker could perform automated password guessing attacks against user accounts, potentially leading to account compromise, unauthorized access, and data exposure.

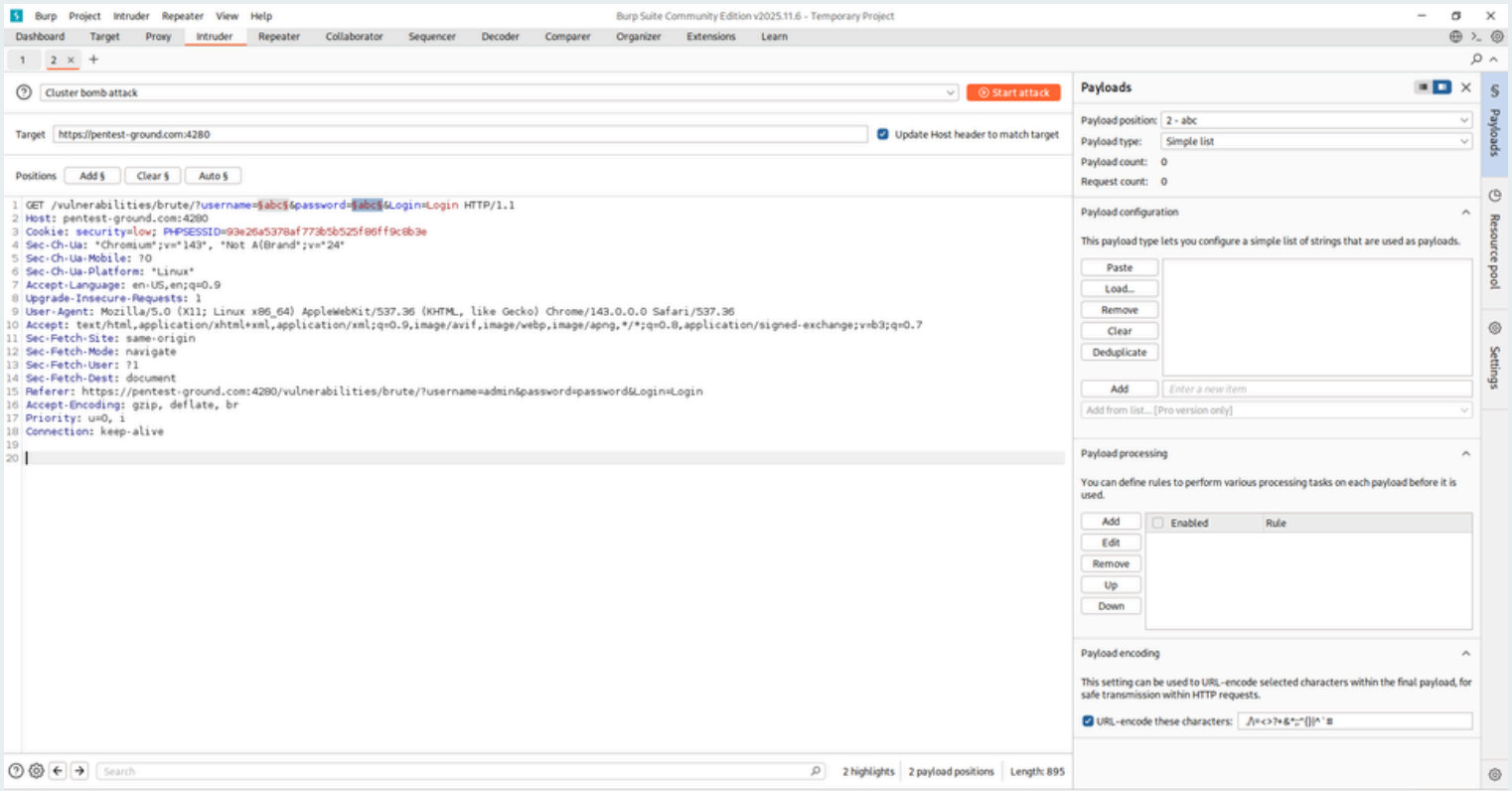
System

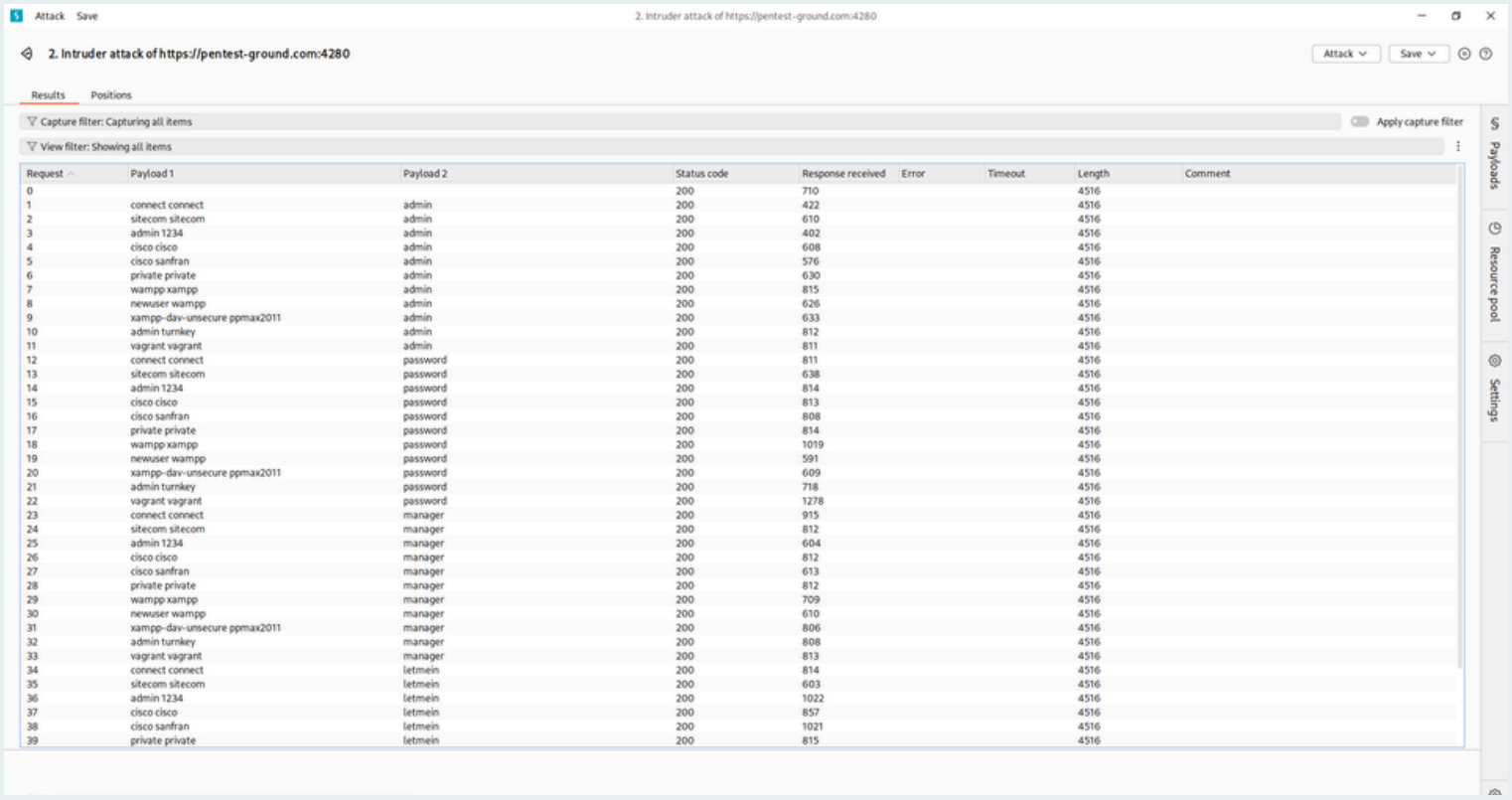
Damn Vulnerable Web Application (DVWA) — Brute Force Module

Proof of concept:

The login form allowed repeated authentication attempts without triggering any blocking, throttling, CAPTCHA challenge, or warning message, indicating the absence of brute-force protection.


pentest-ground.com:4280/vulnerabilities/brute/?username=admin&password=password&Login=Login#
Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec YouTube LMS Rsync DNS Resolution ...
DVWA
Home
Instructions
Setup / Reset DB
Brute Force
Command Injection
CSRF
File Inclusion
File Upload
Insecure CAPTCHA
SQL Injection
SQL Injection (Blind)
Weak Session IDs
XSS (DOM)
XSS (Reflected)
XSS (Stored)
CSP Bypass
JavaScript
Authorisation Bypass
Open HTTP Redirect
DVWA Security
PHP Info
About
Logout
Vulnerability: Brute Force
Login
Username:
admin
Password:
Login
Welcome to the password protected area admin
More Information
https://owasp.org/www-community/attacks/Brute_force_attack
http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password
https://www.golinuxcloud.com/brute-force-attack-web-forms
Username: admin
Security Level: low
Locale: en
SQLi DB: mysql
View Source
View Help
Damn Vulnerable Web Application (DVWA)


Burp Project Intruder Repeater View Help
Burp Suite Community Edition v2025.11.6 - Temporary Project
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Organizer Extensions Learn
Cluster bomb attack
Start attack
Target https://pentest-ground.com:4280
Update Host header to match target
Positions Add § Clear § Auto §
1 GET /vulnerabilities/brute/?username=§abc§&password=§abc§&Login=Login HTTP/1.1
2 Host: pentest-ground.com:4280
3 Cookie: security=low; PHPSESSID=93e26a5378af773b5b525f86ff9c6b3e
4 Sec-Ch-Ua: "Chromium";v="143", "Not A(Brand";v="24"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Accept-Language: en-US,en;q=0.9
8 Upgrade-Insecure-Requests: 1
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/143.0.0.0 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: navigate
13 Sec-Fetch-User: ?1
14 Sec-Fetch-Dest: document
15 Referer: https://pentest-ground.com:4280/vulnerabilities/brute/?username=admin&password=password&Login=Login
16 Accept-Encoding: gzip, deflate, br
17 Priority: u=0, i
18 Connection: keep-alive
19
20
Search
2 highlights 2 payload positions Length: 895
Payloads
Payload position: 2 - abc
Payload type: Simple list
Payload count: 0
Request count: 0
Payload configuration
This payload type lets you configure a simple list of strings that are used as payloads.
Paste
Load...
Remove
Clear
Deduplicate
Add
Enter a new item
Add from list... [Pro version only]
Payload processing
You can define rules to perform various processing tasks on each payload before it is used.
Add
Edit
Remove
Up
Down
Enabled
Rule
Payload encoding
This setting can be used to URL-encode selected characters within the final payload, for safe transmission within HTTP requests.
URL-encode these characters: ./\=<>?+&*;:"{}|^`#
Payloads
Resource pool
Settings



Request	Payload 1	Payload 2	Status code	Response received	Error	Timeout	Length	Comment
0			200	710			4516	
1	connect connect	admin	200	422			4516	
2	sitecom sitecom	admin	200	610			4516	
3	admin 1234	admin	200	402			4516	
4	cisco cisco	admin	200	608			4516	
5	cisco sanfran	admin	200	576			4516	
6	private private	admin	200	630			4516	
7	wampp xampp	admin	200	815			4516	
8	newuser wampp	admin	200	626			4516	
9	xampp-dav-unsecure ppmax2011	admin	200	633			4516	
10	admin turnkey	admin	200	812			4516	
11	vagrant vagrant	admin	200	811			4516	
12	connect connect	password	200	811			4516	
13	sitecom sitecom	password	200	638			4516	
14	admin 1234	password	200	814			4516	
15	cisco cisco	password	200	813			4516	
16	cisco sanfran	password	200	808			4516	
17	private private	password	200	814			4516	
18	wampp xampp	password	200	1019			4516	
19	newuser wampp	password	200	591			4516	
20	xampp-dav-unsecure ppmax2011	password	200	609			4516	
21	admin turnkey	password	200	718			4516	
22	vagrant vagrant	password	200	1278			4516	
23	connect connect	manager	200	915			4516	
24	sitecom sitecom	manager	200	812			4516	
25	admin 1234	manager	200	604			4516	
26	cisco cisco	manager	200	812			4516	
27	cisco sanfran	manager	200	613			4516	
28	private private	manager	200	812			4516	
29	wampp xampp	manager	200	709			4516	
30	newuser wampp	manager	200	610			4516	
31	xampp-dav-unsecure ppmax2011	manager	200	806			4516	
32	admin turnkey	manager	200	808			4516	
33	vagrant vagrant	manager	200	813			4516	
34	connect connect	letmein	200	814			4516	
35	sitecom sitecom	letmein	200	603			4516	
36	admin 1234	letmein	200	1022			4516	
37	cisco cisco	letmein	200	857			4516	
38	cisco sanfran	letmein	200	1021			4516	
39	private private	letmein	200	815			4516	

Recommendation:

- Implement account lockout after a defined number of failed attempts.
- Add rate limiting on authentication requests.
- Introduce CAPTCHA after multiple failed login attempts.
- Log and monitor failed authentication attempts for anomaly detection.

Conclusion

This assessment highlights the importance of secure design and development practices in web applications. The identified vulnerabilities demonstrate how small configuration or coding gaps can lead to significant security risks.

By addressing the issues identified in this report and following the recommended remediation steps, organizations can reduce their exposure to cyber threats, protect user data, and improve trust in their digital services.

Regular security assessments should be considered an essential part of any application’s lifecycle.

Security Weakness

Weak Input Validation

The application does not properly validate or sanitize user input across multiple input points. This allows malicious or unexpected input to be processed by the system, leading to vulnerabilities such as SQL Injection, Cross-Site Scripting, and Command Injection.

This weakness increases the risk of data exposure, system compromise, and client-side attacks.

Weak Output Encoding

User-supplied data is rendered back to users without proper output encoding. This enables client-side script execution and increases the risk of Cross-Site Scripting attacks, which could impact end users and damage trust in the application.

Unrestricted Login Attempts

The login functionality allows unlimited authentication attempts without rate limiting, account lockout, or challenge mechanisms. This significantly increases the risk of brute-force attacks and unauthorized account access.

Lack of Secure Authentication Controls

The application relies solely on username and password authentication and does not implement additional security controls such as multi-factor authentication or adaptive authentication checks. This makes user accounts more vulnerable to compromise if credentials are weak or leaked.

Unsafe Server-Side Command Handling

The application processes user input directly within system-level commands without sufficient filtering or restriction. This exposes the system to risks that could result in unauthorized system access, data loss, or service disruption.