

# XSS Attack & Defense Lab: Identifying, Exploiting, and Mitigating Cross-Site Scripting Vulnerabilities

**Presented By**  
Lamiya  
Laboni Akter Mitu

**Presented TO**  
MD. Nessar Rahman Porag

# Objectives

- To understand the fundamentals of Cross-Site Scripting (XSS) vulnerabilities.
- To identify XSS weaknesses in a vulnerable web application.
- To practically exploit Stored, Reflected, and DOM-Based XSS using DVWA.
- To analyze HTTP requests and responses through Burp Suite.
- To apply effective security mechanisms to mitigate XSS attacks.

# Lab Environment & Tools

**Attacker Machine:** Kali Linux

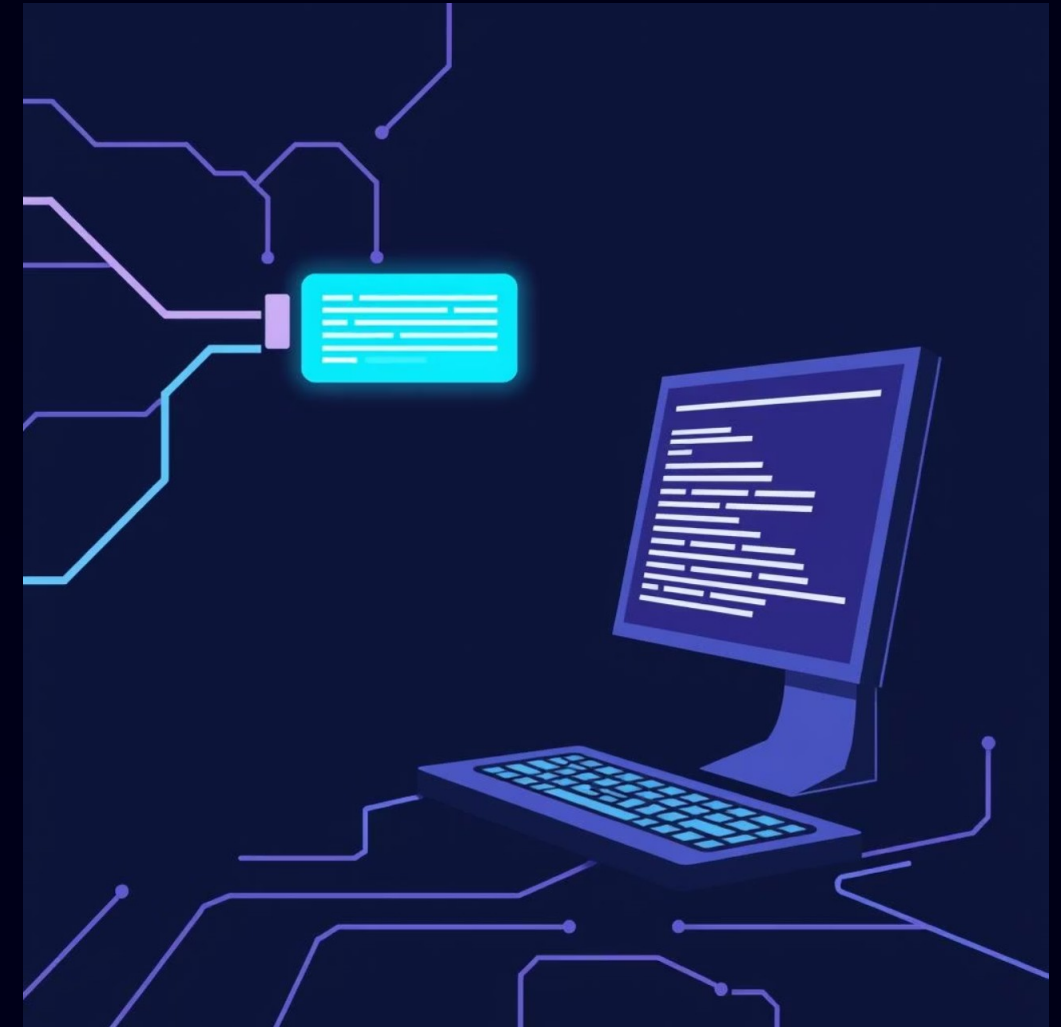
**Target Server:** Metasploitable2

**Vulnerable Application:** Damn Vulnerable Web Application (DVWA)

**Tools Used:** Firefox Browser, Burp Suite, Apache Server, ModSecurity WAF

# What is Cross-Site Scripting (XSS)?

- Cross-Site Scripting (XSS) is a common web application security vulnerability
- It allows attackers to inject malicious JavaScript code into a web application
- The injected script executes in the victim's browser
- This can lead to cookie theft, session hijacking, and unauthorized actions



# The Attack Surface: Three Primary Types

Understanding the delivery method is key to identifying where your application is most vulnerable.



## Reflected XSS

- Script is reflected in server response
- Executes when victim clicks a malicious link



## Stored (Persistent) XSS

- Malicious script is stored in the database
- Executes automatically when page loads
- Affects multiple users



## DOM-Based XSS

- Occurs entirely on the client side
- Vulnerability exists in JavaScript handling of user input
- Server-side processing is not involved

# DVWA Overview

Damn Vulnerable Web Application (DVWA) is a purposely insecure web application. It is designed for learning and practicing web security attacks. The security level was set to Low to demonstrate XSS vulnerabilities

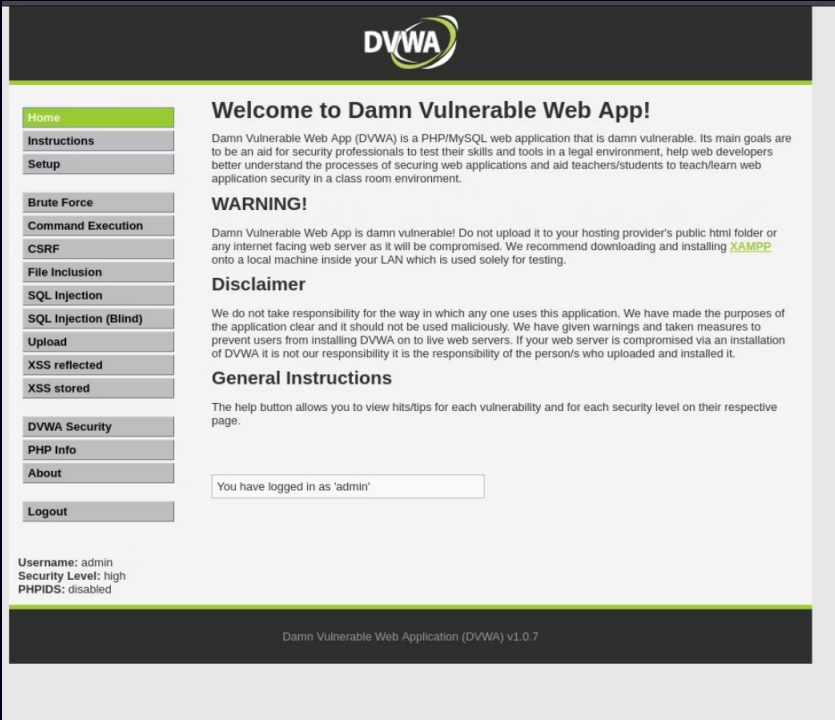


Fig -1 : Home page



Fig-2 : Security page

# XSS Vulnerability Identification

- Application input fields were tested using JavaScript payloads.
- User input was reflected without proper validation or sanitization
- Multiple XSS vulnerabilities were successfully identified

# Stored XSS Exploitation

- A malicious JavaScript payload was submitted through an input form
- The payload was stored in the application database
- The script executed automatically whenever the page was loaded

Kali NetHunter Exploit-DB Google Hacking DB

**DVWA**

Home  
Instructions  
Setup  
Brute Force  
Command Execution  
CSRF  
File Inclusion  
SQL Injection  
SQL Injection (Blind)  
Upload  
XSS reflected  
**XSS stored**  
DVWA Security  
PHP Info  
About  
Logout

### Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

Name: test  
Message: This is a test comment.

Name: test 1  
Message: &lt;script&gt;alert(&quot;hi&quot;)&lt;/script&gt;

Name: test 2  
Message: &lt;script&gt;alert(&quot;hacked&quot;)&lt;/script&gt;

Name: test 3  
Message: &lt;script src=1 href=1 onerror=&quot;javascript:alert(1)&quot;

#### More info

<http://hacker.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

Username: admin  
Security Level: high  
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7



[Home](#)[Instructions](#)[Setup](#)[Brute Force](#)[Command Execution](#)[CSRF](#)[File Inclusion](#)[SQL Injection](#)[SQL Injection \(Blind\)](#)[Upload](#)[XSS reflected](#)[XSS stored](#)[DVWA Security](#)[PHP Info](#)[About](#)[Logout](#)

## Vulnerability: Stored Cross Site Scripting (XSS)

Name \*

Message \*

Name: test  
Message: This is a test comment.

Name: test 1  
Message: <script>alert("hi")</script>

Name: test 2  
Message: <script>alert("hacked")</script>

Name: test 3  
Message: <script src=1 href=1 onerror="javascript:alert(1)">

Name: test 4  
Message: <script src="data:text/plainx2Cjavascript:alert(1)">

### More info

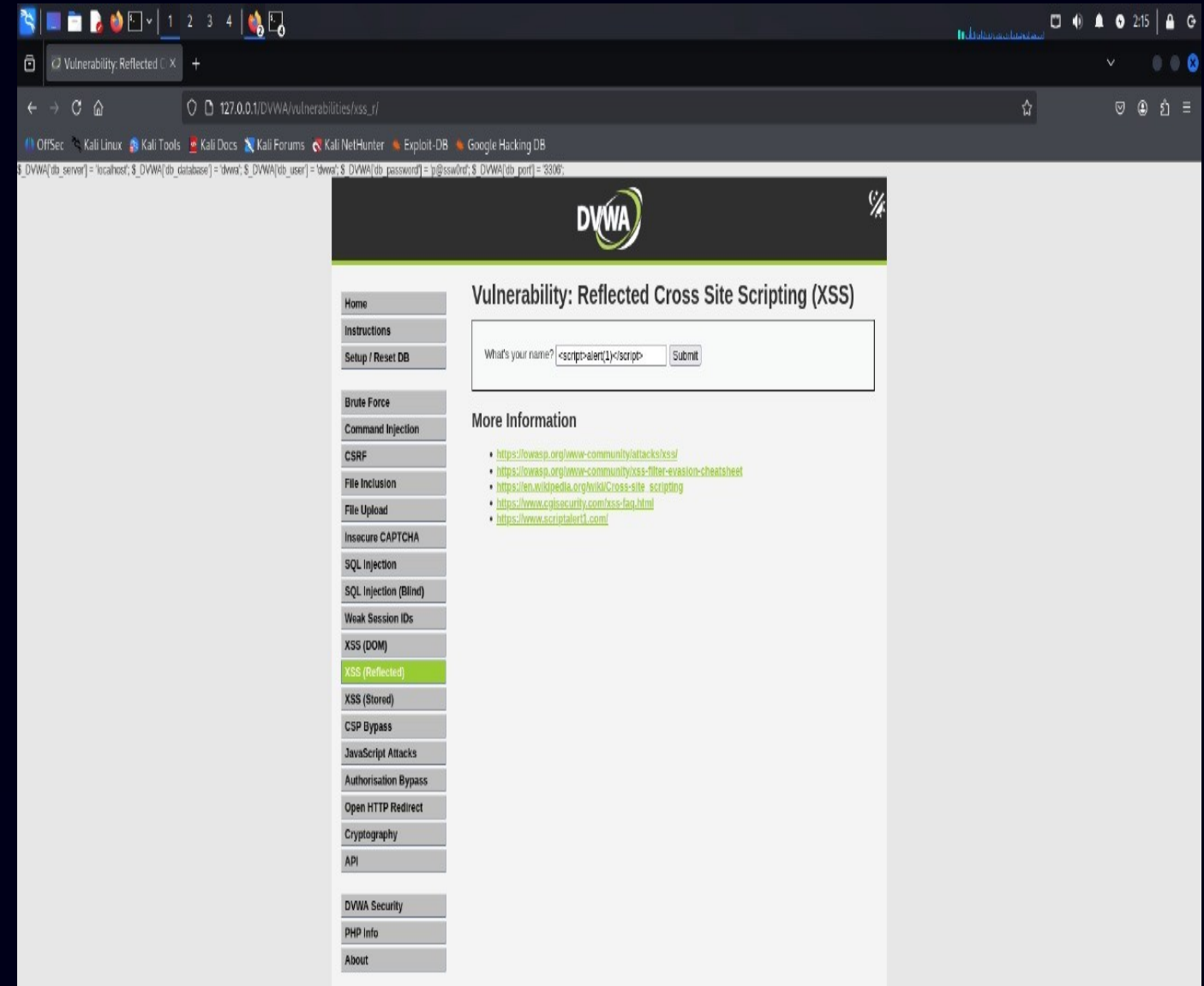
<http://ha.ckers.org/xss.html>[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)<http://www.cgisecurity.com/xss-faq.html>

Username: admin  
Security Level: high  
PHPIDS: disabled

[View Source](#) [View Help](#)

# Reflected XSS Exploitation

- A crafted payload was sent through an HTTP request
- The server reflected the input in its response
- JavaScript executed immediately in the browser



System tray icons: network, volume, battery, clock (2:15), and security indicators.

Browser tabs: Vulnerability: Reflected

Address bar: 127.0.0.1/DVWA/vulnerabilities/xss\_r/?name=<script>alert(1)<%2Fscript>#

Navigation icons: back, forward, home, search, star, and social media links.

Quick links: OffSec, Kali Linux, Kali Tools, Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, Google Hacking DB.

Page content: \$ \_DVWA[db\_server] = 'localhost'; \$ \_DVWA[db\_database] = 'dwwa'; \$ \_DVWA[db\_user] = 'dwwa'; \$ \_DVWA[db\_password] = 'p@ssw0rd'; \$ \_DVWA[db\_port] = '3306';

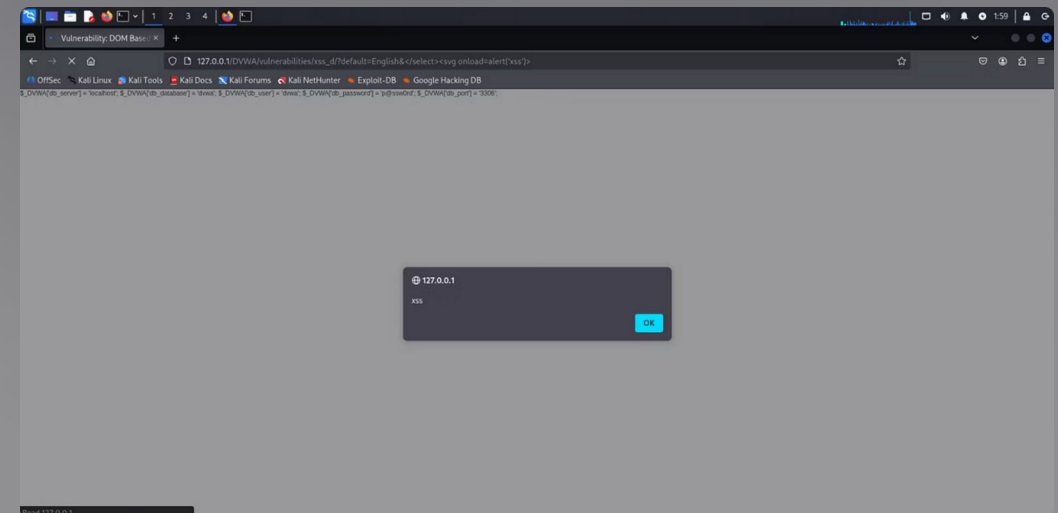
127.0.0.1

1

OK

# DOM-Based XSS Exploitation

- Payloads were injected through URL parameters
- Client-side JavaScript processed the malicious input
- Script execution occurred without server involvement



## Request & Response Analysis Using Burp Suite

- Burp Suite Proxy was enabled to intercept HTTP traffic
- Requests containing malicious payloads were captured
- Server responses confirmed payload reflection or storage

## Importance of Burp Suite

- Provides visibility into HTTP request and response flow
- Helps trace how malicious payloads travel through the application
- Acts as strong evidence of XSS vulnerabilities

# XSS Stored

Dashboard

Target

Proxy

Intruder

Repeater

Collaborator

Sequencer

Decoder

Comparer

Logger

Organizer

Extensions

Learn

Intercept

HTTP history

WebSockets history

Match and replace

Proxy settings

Filter settings: Hiding CSS and image content; hiding specific extensions

Filter on

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start respons...
1	https://www.google.com	GET	/warmup.html			200	1366	HTML	html	Warmup Page		✓	172.217.24.68	__Secure-STRP=AD...	10:08:27 22 D...	8080	158
2	https://www.google.com	GET	/warmup.html			200	1366	HTML	html	Warmup Page		✓	172.217.24.68	__Secure-STRP=AD...	10:08:50 22 D...	8080	157
3	http://192.168.0.104	GET	/dvwa/security.php			200	4452	HTML	php	Damn Vulnerable Web Ap...			192.168.0.104		10:40:15 22 D...	8080	18
4	http://192.168.0.104	GET	/dvwa/vulnerabilities/xss_s/			200	5935	HTML		Damn Vulnerable Web Ap...			192.168.0.104		10:40:17 22 D...	8080	18
5	http://192.168.0.104	POST	/dvwa/vulnerabilities/xss_s/	✓		200	6031	HTML		Damn Vulnerable Web Ap...			192.168.0.104		10:41:55 22 D...	8080	13

Inspector

Request attributes

2

Protocol

HTTP/1

HTTP/2

Name

Value

Method

POST

Path

/dvwa/vulnerabilities/...

Request body parameters

3

Name

Value

txtName

test 6

mtxMessage

<script>alert(2)</scri...

btnSign

Sign Guestbook

Request cookies

2

Name

Value

security

low

PHPSESSID

daf649d40e808ed7b...

Request headers

13

Name

Value

Host

192.168.0.104

User-Agent

Mozilla/5.0 (X11; Linu...

Accept

text/html,application...

Accept-Language

en-US,en;q=0.5

Accept-Encoding

gzip, deflate, br

Request

Pretty

Raw

Hex

1

POST /dvwa/vulnerabilities/xss\_s/ HTTP/1.1

2

Host: 192.168.0.104

3

User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:140.0) Gecko/20100101 Firefox/140.0

4

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

5

Accept-Language: en-US,en;q=0.5

6

Accept-Encoding: gzip, deflate, br

7

Content-Type: application/x-www-form-urlencoded

8

Content-Length: 94

9

Origin: http://192.168.0.104

10

Connection: keep-alive

11

Referer: http://192.168.0.104/dvwa/vulnerabilities/xss\_s/

12

Cookie: security=low; PHPSESSID=daf649d40e808ed7b73f07b9ed96c5d8

13

Upgrade-Insecure-Requests: 1

14

Priority: u=0, i

15

txtName=test+6&mtxMessage=%3Cscript%3Ealert%28%29%3C%2Fscript%3E%0D%0A&btnSign=

16

Sign+Guestbook

Response

Pretty

Raw

Hex

Render

1

HTTP/1.1 200 OK

2

Date: Mon, 22 Dec 2025 16:41:55 GMT

3

Server: Apache/2.2.8 (Ubuntu) DAV/2

4

X-Powered-By: PHP/5.2.4-2ubuntu5.10

5

Pragma: no-cache

6

Cache-Control: no-cache, must-revalidate

7

Expires: Tue, 23 Jun 2009 12:00:00 GMT

8

Content-Length: 5683

9

Keep-Alive: timeout=15, max=100

10

Connection: Keep-Alive

11

Content-Type: text/html; charset=utf-8

12

13

14

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

15

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

16

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

17

18

<head>

19

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

20

21

<title>

22

Damn Vulnerable Web App (DVWA) v1.0.7 :: Vulnerability: Stored Cross Site Scripting

23

(XSS)

24

</title>

25

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

26

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

27

<script type="text/javascript" src="../../dvwa/js/dvwaPage.js">

Event log (1)

All issues

Memory: 156.2MB of 2.81GB

Disabled



# XSS Reflected

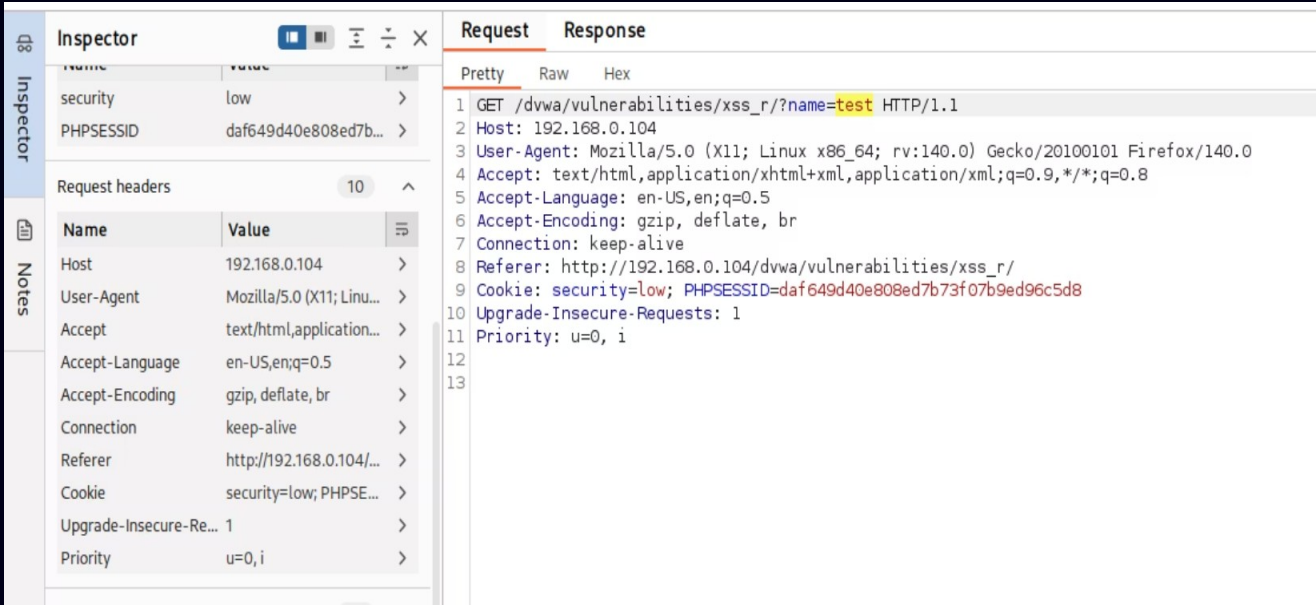


Fig-1: Request

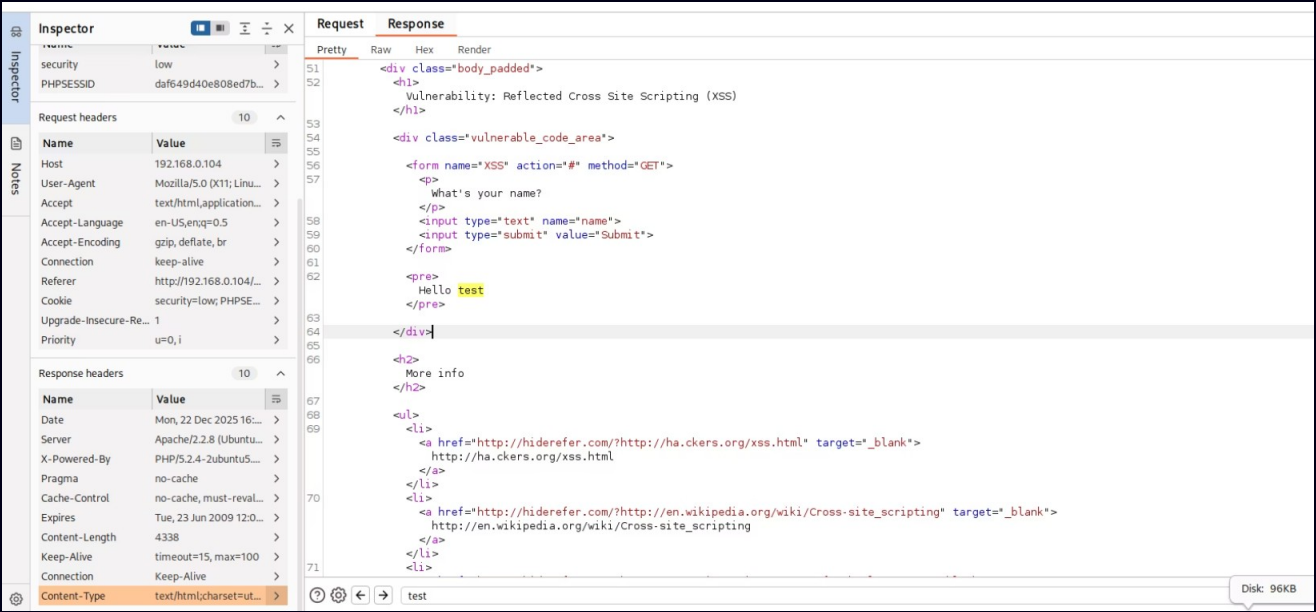


Fig-2: Respons

# XSS Dom

127.0.0.1/DVWA/vulnerabilities/xss\_d/?default=<script>ale

OffSec Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB

\$\_DVWA[db\_server]='localhost'; \$\_DVWA[db\_database]='dvwa'; \$\_DVWA[db\_user]='dvwa'; \$\_DVWA[db\_password]='p@ssw0rd'; \$\_DVWA[db\_port]='3306';

Home Instructions Setup / Reset DB

Please choose a language

Brute Force Command Injection CSRF

Weak Session IDs XSS (DOM) XSS (Reflected) XSS (Stored) CSP Bypass JavaScript Attacks Authorisation Bypass Open HTTP Redirect Cryptography API

127.0.0.1

XSS

OK

Read 127.0.0.1

Burp Suite Community Edition v2025.7.4 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer

Intercept HTTP history WebSockets history Match and replace Proxy settings

Filter settings: Hiding CSS, image and general binary content

| # | Host                               | Method | URL               | Params | Edited | Status code | Length | MIME type | Extension | Title | Notes |
|---|------------------------------------|--------|-------------------|--------|--------|-------------|--------|-----------|-----------|-------|-------|
| 2 | https://contile.services.mozill... | GET    | /v1/tiles         |        |        | 204         | 181    |           |           |       |       |
| 7 | https://contile.services.mozill... | GET    | /v1/tiles         |        |        | 204         | 181    |           |           |       |       |
| 6 | http://detectportal.firefox.c...   | GET    | /success.txt?ip=4 |        |        | 200         | 216    | text      | txt       |       |       |
| 5 | http://detectportal.firefox.c...   | GET    | /success.txt?ip=6 |        |        | 200         | 216    | text      | txt       |       |       |
| 3 | https://spocs.getpocket.com        | POST   | /spocs            |        |        | 200         | 1395   | JSON      |           |       |       |

Request

1 POST /spocs HTTP/1.1

2 Host: spocs.getpocket.com

3 User-Agent: Mozilla/5.0 (X11; Linux x86\_64; rv:128.0) Gecko/20100101 Firefox/128.0

4 Accept: \*/\*

5 Accept-Language: en-US,en;q=0.5

6 Accept-Encoding: gzip, deflate, br

7 Content-Type: application/json

8 Content-Length: 197

9 Priority: u=4

10 Te: trailers

11 Connection: keep-alive

12

13 {

14 "pocket\_id":"{7fa93ab1-dd1e-4458-af85-05b5d239e6bc}",

15 "version":2,

16 "consumer\_key":"40249-e88c401e1b1f2242d9e441c4",

17 "placements":[

18 {

19 "name":"sponsored-top-sites",

20 "ad\_types":[

21 3120

22 ],

23 "zone\_ids":[

24 280143

25 ]

26 }

27 }

28 }

29 }

30 }

31 }

32 }

33 }

34 }

35 }

36 }

37 }

38 }

39 }

40 }

41 }

42 }

43 }

44 }

45 }

46 }

47 }

48 }

49 }

50 }

51 }

52 }

53 }

54 }

55 }

56 }

57 }

58 }

59 }

60 }

61 }

62 }

63 }

64 }

65 }

66 }

67 }

68 }

69 }

70 }

71 }

72 }

73 }

74 }

75 }

76 }

77 }

78 }

79 }

80 }

81 }

82 }

83 }

84 }

85 }

86 }

87 }

88 }

89 }

90 }

91 }

92 }

93 }

94 }

95 }

96 }

97 }

98 }

99 }

100 }

101 }

102 }

103 }

104 }

105 }

106 }

107 }

108 }

109 }

110 }

111 }

112 }

113 }

114 }

115 }

116 }

117 }

118 }

119 }

120 }

121 }

122 }

123 }

124 }

125 }

126 }

127 }

128 }

129 }

130 }

131 }

132 }

133 }

134 }

135 }

136 }

137 }

138 }

139 }

140 }

141 }

142 }

143 }

144 }

145 }

146 }

147 }

148 }

149 }

150 }

151 }

152 }

153 }

154 }

155 }

156 }

157 }

158 }

159 }

160 }

161 }

162 }

163 }

164 }

165 }

166 }

167 }

168 }

169 }

170 }

171 }

172 }

173 }

174 }

175 }

176 }

177 }

178 }

179 }

180 }

181 }

182 }

183 }

184 }

185 }

186 }

187 }

188 }

189 }

190 }

191 }

192 }

193 }

194 }

195 }

196 }

197 }

198 }

199 }

200 }

201 }

202 }

203 }

204 }

205 }

206 }

207 }

208 }

209 }

210 }

211 }

212 }

213 }

214 }

215 }

216 }

217 }

218 }

219 }

220 }

221 }

222 }

223 }

224 }

225 }

226 }

227 }

228 }

229 }

230 }

231 }

232 }

233 }

234 }

235 }

236 }

237 }

238 }

239 }

240 }

241 }

242 }

243 }

244 }

245 }

246 }

247 }

248 }

249 }

250 }

251 }

252 }

253 }

254 }

255 }

256 }

257 }

258 }

259 }

260 }

261 }

262 }

263 }

264 }

265 }

266 }

267 }

268 }

269 }

270 }

271 }

272 }

273 }

274 }

275 }

276 }

277 }

278 }

279 }

280 }

281 }

282 }

283 }

284 }

285 }

286 }

287 }

288 }

289 }

290 }

291 }

292 }

293 }

294 }

295 }

296 }

297 }

298 }

299 }

300 }

301 }

302 }

303 }

304 }

305 }

306 }

307 }

308 }

309 }

310 }

311 }

312 }

313 }

314 }

315 }

316 }

317 }

318 }

319 }

320 }

321 }

322 }

323 }

324 }

325 }

326 }

327 }

328 }

329 }

330 }

331 }

332 }

333 }

334 }

335 }

336 }

337 }

338 }

339 }

340 }

341 }

342 }

343 }

344 }

345 }

346 }

347 }

348 }

349 }

350 }

351 }

352 }

353 }

354 }

355 }

356 }

357 }

358 }

359 }

360 }

361 }

362 }

363 }

364 }

365 }

366 }

367 }

368 }

369 }

370 }

371 }

372 }

373 }

374 }

375 }

376 }

377 }

378 }

379 }

380 }

381 }

382 }

383 }

384 }

385 }

386 }

387 }

388 }

389 }

390 }

391 }

392 }

393 }

394 }

395 }

396 }

397 }

398 }

399 }

400 }

401 }

402 }

403 }

404 }

405 }

406 }

407 }

408 }

409 }

410 }

411 }

412 }

413 }

414 }

415 }

416 }

417 }

418 }

419 }

420 }

421 }

422 }

423 }

424 }

425 }

426 }

427 }

428 }

429 }

430 }

431 }

432 }

433 }

434 }

435 }

436 }

437 }

438 }

439 }

440 }

441 }

442 }

443 }

444 }

445 }

446 }

447 }

448 }

449 }

450 }

451 }

452 }

453 }

454 }

455 }

456 }

457 }

458 }

459 }

460 }

461 }

462 }

463 }

464 }

465 }

466 }

467 }

468 }

469 }

470 }

471 }

472 }

473 }

474 }

475 }

476 }

477 }

478 }

479 }

480 }

481 }

482 }

483 }

484 }

485 }

486 }

487 }

488 }

489 }

490 }

491 }

492 }

493 }

494 }

495 }

496 }

497 }

498 }

499 }

500 }

501 }

502 }

503 }

504 }

505 }

506 }

507 }

508 }

509 }

510 }

511 }

512 }

513 }

514 }

515 }

516 }

517 }

518 }

519 }

520 }

521 }

522 }

523 }

524 }

525 }

526 }

527 }

528 }

529 }

530 }

531 }

532 }

533 }

534 }

535 }

536 }

537 }

538 }

539 }

540 }

541 }

542 }

543 }

544 }

545 }

546 }

547 }

548 }

549 }

550 }

551 }

552 }

553 }

554 }

555 }

556 }

557 }

558 }

559 }

560 }

561 }

562 }

563 }

564 }

565 }

566 }

567 }

568 }

569 }

570 }

571 }

572 }

573 }

574 }

575 }

576 }

577 }

578 }

579 }

580 }

581 }

582 }

583 }

584 }

585 }

586 }

587 }

588 }

589 }

590 }

591 }

592 }

593 }

594 }

595 }

596 }

597 }

598 }

599 }

600 }

601 }

602 }

603 }

604 }

605 }

606 }

607 }

608 }

609 }

610 }

611 }

612 }

613 }

614 }

615 }

616 }

617 }

618 }

619 }

620 }

621 }

622 }

623 }

624 }

625 }

626 }

627 }

628 }

629 }

630 }

631 }

632 }

633 }

634 }

635 }

636 }

637 }

638 }

639 }

640 }

641 }

642 }

643 }

644 }

645 }

646 }

647 }

648 }

649 }

650 }

651 }

652 }

653 }

654 }

655 }

656 }

657 }

658 }

659 }

660 }

661 }

662 }

663 }

664 }

665 }

666 }

667 }

668 }

669 }

670 }

671 }

672 }

673 }

674 }

675 }

676 }

677 }

678 }

679 }

680 }

681 }

682 }

683 }

684 }

685 }

686 }

687 }

688 }

689 }

690 }

691 }

692 }

693 }

694 }

695 }

696 }

697 }

698 }

699 }

700 }

701 }

702 }

703 }

704 }

705 }

706 }

707 }

708 }

709 }

710 }

711 }

712 }

713 }

714 }

715 }

716 }

717 }

718 }

719 }

720 }

721 }

722 }

723 }

724 }

725 }

726 }

727 }

728 }

729 }

730 }

731 }

732 }

733 }

734 }

735 }

736 }

737 }

738 }

739 }

740 }

741 }

742 }

743 }

744 }

745 }

746 }

747 }

748 }

749 }

750 }

751 }

752 }

753 }

754 }

755 }

756 }

757 }

758 }

759 }

760 }

761 }

762 }

763 }

764 }

765 }

766 }

767 }

768 }

769 }

770 }

771 }

772 }

773 }

774 }

775 }

776 }

777 }

778 }

779 }

780 }

781 }

782 }

783 }

784 }

785 }

786 }

787 }

788 }

789 }

790 }

791 }

792 }

793 }

794 }

795 }

796 }

797 }

798 }

799 }

800 }

801 }

802 }

803 }

804 }

805 }

806 }

807 }

808 }

809 }

810 }

811 }

812 }

813 }

814 }

815 }

816 }

817 }

818 }

819 }

820 }

821 }

822 }

823 }

824 }

825 }

826 }

827 }

828 }

829 }

830 }

831 }

832 }

833 }

834 }

835 }

836 }

837 }

838 }

839 }

840 }

841 }

842 }

843 }

844 }

845 }

846 }

847 }

848 }

849 }

850 }

851 }

852 }

853 }

854 }

855 }

856 }

857 }

858 }

859 }

860 }

861 }

862 }

863 }

864 }

865 }

866 }

867 }

868 }

869 }

870 }

871 }

872 }

873 }

874 }

875 }

876 }

877 }

878 }

879 }

880 }

881 }

882 }

883 }

884 }

885 }

886 }

887 }

888 }

889 }

890 }

891 }

892 }

893 }

894 }

895 }

896 }

897 }

898 }

899 }

900 }

901 }

902 }

903 }

904 }

905 }

906 }

907 }

908 }

909 }

910 }

911 }

912 }

913 }

914 }

915 }

916 }

917 }

918 }

919 }

920 }

921 }

922 }

923 }

924 }

925 }

926 }

927 }

928 }

929 }

930 }

931 }

932 }

933 }

934 }

935 }

936 }

937 }

938 }

939 }

940 }

941 }

942 }

943 }

944 }

945 }

946 }

947 }

948 }

949 }

950 }

951 }

952 }

953 }

954 }

955 }

956 }

957 }

958 }

959 }

960 }

961 }

962 }

963 }

964 }

965 }

966 }

967 }

968 }

969 }

970 }

971 }

972 }

973 }

974 }

975 }

976 }

977 }

978 }

979 }

980 }

981 }

982 }

983 }

984 }

985 }

986 }

987 }

988 }

989 }

990 }

991 }

992 }

993 }

994 }

995 }

996 }

997 }

998 }

999 }

1000 }

1001 }

1002 }

1003 }

1004 }

1005 }

1006 }

1007 }

1008 }

1009 }

1010 }

1011 }

1012 }

1013 }

1014 }

1015 }

1016 }

1017 }

1018 }

1019 }

1020 }

1021 }

1022 }

1023 }

1024 }

1025 }

1026 }

1027 }

1028 }

1029 }

1030 }

1031 }

1032 }

1033 }

1034 }

1035 }

1036 }

1037 }

1038 }

1039 }

1040 }

1041 }

1042 }

1043 }

1044 }

1045 }

1046 }

1047 }

1048 }

1049 }

1050 }

1051 }

1052 }

1053 }

1054 }

1055 }

1056 }

1057 }

1058 }

1059 }

1060 }

1061 }

1062 }

1063 }

1064 }

1065 }

1066 }

1067 }

1068 }

1069 }

1070 }

1071 }

1072 }

1073 }

1074 }

1075 }

1076 }

1077 }

1078 }

1079 }

1080 }

1081 }

1082 }

1083 }

1084 }

1085 }

1086 }

1087 }

1088 }

1089 }

1090 }

1091 }

1092 }

1093 }

1094 }

1095 }

1096 }

1097 }

1098 }

1099 }

1100 }

1101 }

1102 }

1103 }

1104 }

1105 }

1106 }

1107 }

1108 }

1109 }

1110 }

1111 }

1112 }

1113 }

1114 }

1115 }

1116 }

1117 }

1118 }

1119 }

1120 }

1121 }

1122 }

1123 }

1124 }

1125 }

1126 }

1127 }

1128 }

1129 }

1130 }

1131 }

1132 }

1133 }

1134 }

1135 }

1136 }

1137 }

1138 }

1139 }

1140 }

1141 }

1142 }

1143 }

1144 }

1145 }

1146 }

1147 }

1148 }

1149 }

1150 }

1151 }

1152 }

1153 }

1154 }

1155 }

1156 }

1157 }

1158 }

1159 }

1160 }

1161 }

1162 }

1163 }

1164 }

1165 }

1166 }

1167 }

1168 }

1169 }

1170 }

1171 }

1172 }

1173 }

1174 }

1175 }

1176 }

1177 }

1178 }

1179 }

1180 }

1181 }

1182 }

1183 }

1184 }

1185 }

1186 }

1187 }

1188 }

1189 }

1190 }

1191 }

1192 }

1193 }

1194 }

1195 }

1196 }

1197 }

1198 }

1199 }

1200 }

1201 }

1202 }

1203 }

1204 }

1205 }

1206 }

1207 }

1208 }

1209 }

1210 }

1211 }

1212 }

1213 }

1214 }

1215 }

1216 }

1217 }

1218 }

1219 }

1220 }

1221 }

1222 }

1223 }

1224 }

1225 }

1226 }

1227 }

1228 }

1229 }

1230 }

1231 }

1232 }

1233 }

1234 }

1235 }

1236 }

1237 }

1238 }

1239 }

1240 }

1241 }

1242 }

1243 }

1244 }

1245 }

1246 }

1247 }

1248 }

1249 }

1250 }

1251 }

1252 }

1253 }

1254 }

1255 }

1256 }

1257 }

1258 }

1259 }

1260 }

1261 }

1262 }

1263 }

1264 }

1265 }

1266 }

1267 }

1268 }

1269 }

1270 }

1271 }

1272 }

1273 }

1274 }

1275 }

1276 }

1277 }

1278 }

1279 }

1280 }

1281 }

1282 }

1283 }

1284 }

1285 }

1286 }

1287 }

1288 }

1289 }

1290 }

1291 }

1292 }

1293 }

1294 }

1295 }

1296 }

1297 }

1298 }

1299 }

1300 }

1301 }

1302 }

1303 }

1304 }

1305 }

1306 }

1307 }

1308 }

1309 }

1310 }

1311 }

1312 }

1313 }

1314 }

1315 }

1316 }

1317 }

1318 }

1319 }

1320 }

1321 }

1322 }

1323 }

1324 }

1325 }

1326 }

1327 }

1328 }

1329 }

1330 }

1331 }

1332 }

1333 }

1334 }

1335 }

1336 }

1337 }

1338 }

1339 }

1340 }

1341 }

1342 }

1343 }

1344 }

1345 }

1346 }

1347 }

1348 }

1349 }

1350 }

1351 }

1352 }

1353 }

1354 }

1355 }

1356 }

1357 }

1358 }

1359 }

1360 }

1361 }

1362 }

1363 }

1364 }

1365 }

1366 }

1367 }

1368 }

1369 }

1370 }

1371 }

1372 }

1373 }

1374 }

1375 }

1376 }

1377 }</




# Defense Phase Overview

- Defense mechanisms were applied after successful exploitation
- The goal was to prevent JavaScript execution
- Multiple security layers were implemented

# XSS ATTACKS



# Stored xss



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 Attacks

Authorisation Bypass

Open HTTP Redirect

Cryptography

API

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: test 1  
Message: &lt;script&gt;alert(1)&lt;/script&gt;



### 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](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <https://www.cgisecurity.com/xss-faq.html>
- <https://www.scriptalert1.com/>

Username: admin  
Security Level: Security Level: impossible  
Locale: en

View Source

# Reflected XSS



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 Attacks

Authorisation Bypass

Open HTTP Redirect

Cryptography

API

DVWA Security

PHP Info

About

Logout

## Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hello `<script>alert(1)</script>`

### 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](https://en.wikipedia.org/wiki/Cross-site_scripting)
- <https://www.cgisecurity.com/xss-faq.html>
- <https://www.scriptalert1.com/>

Username: admin



Security Level: Security Level: impossible

Locale: en

SQLi DB: mysql

Damn Vulnerable Web Application (DVWA)

# DOM-Based XSS



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 Attacks

Authorisation Bypass

Open HTTP Redirect

Cryptography

API

DVWA Security

PHP Info

About

Logout

## Vulnerability: DOM Based Cross Site Scripting (XSS)

Please choose a language:  

English ▼ Select

### More Information

- <https://owasp.org/www-community/attacks/xss/>
- [https://owasp.org/www-community/attacks/DOM\\_Based\\_XSS](https://owasp.org/www-community/attacks/DOM_Based_XSS)
- <https://www.acunetix.com/blog/articles/dom-xss-explained/>

View Source

View Help

Username: admin  
Security Level: Security Level: impossible  
Locale: en  
SQLi DB: mysql

Damn Vulnerable Web Application (DVWA)

Made with GAMMA

# Defense & Mitigation Strategies

## Input Validation

User inputs were validated before processing. Suspicious script tags were filtered. Reduced the risk of malicious injection.



## Output Encoding

Special characters were converted into HTML entities. Scripts were displayed as plain text. Browser execution was successfully prevented



## ModSecurity Web Application Firewall (WAF)

ModSecurity was integrated with the Apache web server. OWASP Core Rule Set (CRS) was enabled. Incoming requests were inspected for attack patterns

# ModSecurity Configuration & Log Analysis

- ModSecurity engine was enabled successfully
- Suspicious XSS payloads were detected
- OWASP CRS rules were triggered
- Audit logs were generated for analysis



# Result Analysis



## Stored XSS

- No alert box was executed
- Payload was automatically HTML-encoded
- Stored XSS vulnerability was mitigated



## Reflected XSS

- JavaScript execution was blocked
- Payload appeared as harmless text
- WAF successfully detected malicious patterns



## DOM-Based XSS

- No JavaScript execution occurred
- Web page loaded normally
- Client-side protection proved effective

# Overall Defense Evaluation

The overall defense implementation in this project proved to be effective in mitigating Cross-Site Scripting (XSS) vulnerabilities. After enabling input validation, output encoding, and deploying ModSecurity Web Application Firewall with the OWASP Core Rule Set, all previously exploited XSS payloads failed to execute. The browser treated injected scripts as harmless text, preventing unauthorized JavaScript execution and protecting user sessions.

Additionally, ModSecurity actively inspected incoming HTTP requests and successfully detected suspicious patterns related to XSS attacks. Relevant security rules were triggered, and detailed audit logs were generated, confirming that the Web Application Firewall was functioning as intended. This layered defense approach significantly reduced the attack surface of the application and demonstrated how combining secure coding practices with automated security controls can effectively protect web applications from XSS attacks.



# Conclusion

This project provided a comprehensive understanding of Cross-Site Scripting (XSS) vulnerabilities by combining both theoretical knowledge and practical experimentation. Through the use of Damn Vulnerable Web Application (DVWA), different types of XSS attacks—Stored, Reflected, and DOM-Based—were successfully identified and exploited in a controlled lab environment. These experiments demonstrated how improper input handling and lack of output sanitization can allow malicious JavaScript to execute in a user's browser, leading to serious security risks such as session hijacking and data theft.

Furthermore, the project emphasized the importance of defensive mechanisms in securing web applications. By implementing input validation, output encoding, and deploying ModSecurity with the OWASP Core Rule Set, the effectiveness of layered security defenses was clearly observed. After enabling these protections, malicious scripts failed to execute, and suspicious activities were detected and logged by the Web Application Firewall. Overall, this project highlights the critical need for secure coding practices and proactive security measures to mitigate XSS vulnerabilities and strengthen the security of modern web applications.