Security Scan Report

Executive Summary

This report summarizes a security scan of the GitHub repository: https://github.com/digininja/DVWA.git. We found 74 potential security issues. Breakdown: 6 Critical, 68 High, 0 Medium, 0 Low, 0 Informational issues.

Overall Risk Assessment

Risk Score: 81.6/100 (Critical Risk)

What We Found

Critical Severity Issues (6)

File	Line	Issue	Context
vulnerabilities/view_help.php	20	Use of eval() - Potential code injection risk	eval('?>' . file_get_contents(DVWA_WEB_PAGE_TO_R...
vulnerabilities/view_help.php	22	Use of eval() - Potential code injection risk	eval('?>' . file_get_contents(DVWA_WEB_PAGE_TO_R...
dvwa/js/dvwaPage.js	7	Use of eval() - Potential code injection risk	//eval("page" + id + " = window.open(URL, '" + id ...
vulnerabilities/javascript/source/high_unobfuscated.js	83	Use of eval() - Potential code injection risk	var crypto = eval("require('crypto')");
vulnerabilities/javascript/source/high_unobfuscated.js	84	Use of eval() - Potential code injection risk	var Buffer = eval("require('buffer').Buffer");
vulnerabilities/javascript/source/high.js	1	Use of eval() - Potential code injection risk	var a=['fromCharCode','toString','replace','BeJ','...

High Severity Issues (68)

File	Line	Issue	Context
login.php	39	Potential hardcoded credential	$query = "SELECT * FROM `users` WHERE user='$user...
dvwa/includes/dvwaPage.inc.php	108	Unfiltered POST parameter - Injection risk	if(array_key_exists ("Login", $_POST) && $_POST['...

security.php	13	Unfiltered POST parameter - Injection risk	if(isset($_POST['seclev_submit'])) {
external/recaptcha/recaptchalib.php	41	Potential hardcoded credential	<br /> <div class='g-recaptcha' data-theme='dark' ...
vulnerabilities/brute/source/impossible.php	3	Unfiltered POST parameter - Injection risk	if(isset($_POST['Login']) && isset ($_POST['u...
vulnerabilities/brute/source/low.php	12	Potential hardcoded credential	$query = "SELECT * FROM `users` WHERE user = '$us...
vulnerabilities/brute/source/high.php	19	Potential hardcoded credential	$query = "SELECT * FROM `users` WHERE user = '$us...
vulnerabilities/sqli/test.php	4	Potential hardcoded credential	$password = "password";
vulnerabilities/brute/source/medium.php	14	Potential hardcoded credential	$query = "SELECT * FROM `users` WHERE user = '$us...
vulnerabilities/cryptography/source/ecb_attack.php	17	Potential hardcoded credential	$key = "ik ben een aardbei";
vulnerabilities/cryptography/source/token_library_impossible.php	40	Potential hardcoded credential	$token = "userid:2";
vulnerabilities/cryptography/source/xor_theory.php	19	Potential hardcoded credential	$key = "wachtwoord";
vulnerabilities/cryptography/source/token_library_high.php	37	Potential hardcoded credential	$token = "userid:2";
vulnerabilities/cryptography/source/low.php	16	Potential hardcoded credential	$key = "wachtwoord";
vulnerabilities/cryptography/source/low.php	29	Unfiltered POST parameter - Injection risk	$message = $_POST['message'];
vulnerabilities/cryptography/source/low.php	30	Unfiltered POST parameter - Injection risk	if (array_key_exists ('direction', $_POST) && $_PO...
vulnerabilities/cryptography/source/low.php	39	Unfiltered POST parameter - Injection risk	$password = $_POST['password'];
vulnerabilities/open_redirect/source/medium.php	3	Unfiltered GET parameter - Injection risk	if (array_key_exists ("redirect", $_GET) && $_GET[...
vulnerabilities/open_redirect/source/medium.php	4	Unfiltered GET parameter - Injection risk	if (preg_match ("/http:\/\/|https:\/\//i", $_GET['...
vulnerabilities/open_redirect/source/medium.php	11	Unfiltered GET parameter - Injection risk	header ("location: " . $_GET['redirect']);
vulnerabilities/open_redirect/source/high.php	3	Unfiltered GET parameter - Injection risk	if (array_key_exists ("redirect", $_GET) && $_GET[...
vulnerabilities/open_redirect/source/high.php	4	Unfiltered GET parameter - Injection risk	if (strpos($_GET['redirect'], "info.php") !== fals...
vulnerabilities/open_redirect/source/high.php	5	Unfiltered GET parameter - Injection risk	header ("location: " . $_GET['redirect']);

vulnerabilities/open_redirect/source/info.php	17	Unfiltered GET parameter - Injection risk	if (array_key_exists ("id", $_GET) && is_numeric($...
vulnerabilities/open_redirect/source/info.php	18	Unfiltered GET parameter - Injection risk	switch (intval ($_GET['id'])) {
vulnerabilities/open_redirect/source/low.php	3	Unfiltered GET parameter - Injection risk	if (array_key_exists ("redirect", $_GET) && $_GET[...
vulnerabilities/open_redirect/source/low.php	4	Unfiltered GET parameter - Injection risk	header ("location: " . $_GET['redirect']);
vulnerabilities/open_redirect/source/impossible.php	5	Unfiltered GET parameter - Injection risk	if (array_key_exists ("redirect", $_GET) && is_num...
vulnerabilities/open_redirect/source/impossible.php	6	Unfiltered GET parameter - Injection risk	switch (intval ($_GET['redirect'])) {
vulnerabilities/authbypass/authbypass.js	42	Direct innerHTML manipulation - XSS risk	cell0.innerHTML = user['user_id'] + '<input type="...
vulnerabilities/authbypass/authbypass.js	45	Direct innerHTML manipulation - XSS risk	cell1.innerHTML = '<input type="text" id="first_na...
vulnerabilities/authbypass/authbypass.js	47	Direct innerHTML manipulation - XSS risk	cell2.innerHTML = '<input type="text" id="surname_...
vulnerabilities/authbypass/authbypass.js	49	Direct innerHTML manipulation - XSS risk	cell3.innerHTML = '<input type="button" value="Upd...
vulnerabilities/cryptography/source/medium.php	10	Potential hardcoded credential	$key = "ik ben een aardbei";
vulnerabilities/cryptography/source/medium.php	21	Unfiltered POST parameter - Injection risk	$token = $_POST['token'];
vulnerabilities/captcha/source/medium.php	14	Unfiltered POST parameter - Injection risk	$_POST['g-recaptcha-response']
vulnerabilities/captcha/source/medium.php	68	Potential hardcoded credential	$insert = "UPDATE `users` SET password = '$pass_ne...
vulnerabilities/captcha/source/high.php	14	Unfiltered POST parameter - Injection risk	$_POST['g-recaptcha-response']
vulnerabilities/captcha/source/high.php	30	Potential hardcoded credential	$insert = "UPDATE `users` SET password = '$pass_ne...
vulnerabilities/captcha/source/low.php	14	Unfiltered POST parameter - Injection risk	$_POST['g-recaptcha-response']
vulnerabilities/captcha/source/low.php	60	Potential hardcoded credential	$insert = "UPDATE `users` SET password = '$pass_ne...
vulnerabilities/captcha/source/impossible.php	29	Unfiltered POST parameter - Injection risk	$_POST['g-recaptcha-response']
vulnerabilities/csrf/source/medium.php	18	Potential hardcoded credential	$insert = "UPDATE `users` SET password = '$pass_ne...

vulnerabilities/csrf/test_credentials.php	21	Potential hardcoded credential	$query = "SELECT * FROM `users` WHERE user='$user...
vulnerabilities/csrf/source/high.php	43	Potential hardcoded credential	$insert = "UPDATE `users` SET password = '" . $pas...
vulnerabilities/xss_d/source/medium.php	5	Unfiltered GET parameter - Injection risk	$default = $_GET['default'];
vulnerabilities/xss_d/source/high.php	7	Unfiltered GET parameter - Injection risk	switch ($_GET['default']) {
vulnerabilities/csp/source/high.js	9	Direct innerHTML manipulation - XSS risk	document.getElementById("answer").innerHTML = obj[...
vulnerabilities/csp/source/jsonp.php	5	Unfiltered GET parameter - Injection risk	$callback = $_GET['callback'];
vulnerabilities/csrf/source/low.php	16	Potential hardcoded credential	$insert = "UPDATE `users` SET password = '$pass_ne...
vulnerabilities/csp/source/medium.php	14	Unfiltered POST parameter - Injection risk	if (isset ($_POST['include'])) {
vulnerabilities/csp/source/medium.php	16	Unfiltered POST parameter - Injection risk	" . $_POST['include'] . "
vulnerabilities/csp/source/low.php	13	Unfiltered POST parameter - Injection risk	if (isset ($_POST['include'])) {
vulnerabilities/csp/source/low.php	15	Unfiltered POST parameter - Injection risk	<script src='" . $_POST['include'] . "'></script>
vulnerabilities/csp/source/impossible.php	9	Unfiltered POST parameter - Injection risk	if (isset ($_POST['include'])) {
vulnerabilities/csp/source/impossible.php	11	Unfiltered POST parameter - Injection risk	" . $_POST['include'] . "
vulnerabilities/csp/source/high.php	8	Unfiltered POST parameter - Injection risk	if (isset ($_POST['include'])) {
vulnerabilities/csp/source/high.php	10	Unfiltered POST parameter - Injection risk	" . $_POST['include'] . "
vulnerabilities/csp/source/impossible.js	9	Direct innerHTML manipulation - XSS risk	document.getElementById("answer").innerHTML = obj[...
vulnerabilities/api/src/LoginController.php	89	Unfiltered POST parameter - Injection risk	switch ($_POST['grant_type']) {
vulnerabilities/api/src/LoginController.php	93	Unfiltered POST parameter - Injection risk	$username = $_POST['username'];
vulnerabilities/api/src/LoginController.php	94	Unfiltered POST parameter - Injection risk	$password = $_POST['password'];
vulnerabilities/api/src/LoginController.php	110	Unfiltered POST parameter - Injection risk	$refresh_token = $_POST['refresh_token'];
vulnerabilities/api/src/Login.php	10	Potential hardcoded credential	private const ACCESS_TOKEN_SECRET = "12345";
vulnerabilities/api/src/Login.php	12	Potential hardcoded credential	private const REFRESH_TOKEN_SECRET = "98765";
vulnerabilities/api/src/Token.php	10	Potential hardcoded credential	private const ENCRYPTION_KEY = "Paintbrush";

vulnerabilities/javascript/index.php	37	Unfiltered POST parameter - Injection risk	$phrase = $_POST['phrase'];
vulnerabilities/javascript/index.php	38	Unfiltered POST parameter - Injection risk	$token = $_POST['token'];

What You Can Do

• Avoid using risky functions like 'eval()' that can run harmful code.
• Keep passwords and secrets out of your code—use secure storage instead.
• Run this scan regularly to catch issues early.
• Ask your team to follow safe coding habits.

Note: This tool is very thorough, but it's always a good idea to have an expert double-check important systems.