A close-up of numbers

Description automatically generated

**Report for <Assessment\_TYPE>**

**Prepared for:** <Client\_Full\_Name>

**A close up of a logo

AI-generated content may be incorrect.**

**Date:** <Publish\_Date>

**Disclaimer**

This report is intended solely for the use of the management of **<CLIENT\_NAME>** and is not to be used or relied upon by others for any purpose whatsoever. The findings and recommendations detailed herein provide management with information regarding the conditions, risks, and internal controls at a specific point in time. Future changes in environmental factors and actions by personnel may significantly and adversely impact these risks and controls in ways that this report could not have anticipated.

This report presents the results of the **<Security\_Assessment\_TYPE>** performed for **<CLIENT\_NAME>**. It is designed to provide the reader with an understanding of the assessed security level, identify security deficiencies and areas of weakness, and offer recommendations to correct vulnerabilities and mitigate the associated risks.

All information security systems, which rely on human operators, are inherently vulnerable to some degree. While the team believes it has identified the major risks in the systems analyzed, there can be no assurance that all risks have been identified or that all recommendations are exhaustive or operationally viable. This report reflects risks detected only during the **<Security\_Assessment\_TYPE>** conducted **<ONSITE\_or\_Remotely>**. The scope and approach were agreed upon at the outset, but had additional procedures been performed, other issues might have been identified and reported.

This report contains information about potential risks identified within **<CLIENT\_NAME>**'s environment. It is strongly recommended that special precautions be taken to protect the confidentiality of both this document and the information contained herein.

**Table of Contents**

[**1.** **Assessment Details** 3](#_Toc193729328)

[**2.** **Scope** 4](#_Toc193729329)

[**3.** **Executive Summary** 5](#_Toc193729330)

[**4.** **Technical Summary** 6](#_Toc193729331)

[**5.** **Recommendations** 7](#_Toc193729332)

[**6.** **Key Findings and Detailed Reports:** 8](#_Toc193729333)

[**1.** **<Issue\_name..1>** 8](#_Toc193729334)

[**2.** **<Issue\_name..2>.** 11](#_Toc193729335)

[**3.** **<Issue\_name..3>** 13](#_Toc193729336)

[**4.** **<Issue\_name..4>** 15](#_Toc193729337)

[**5.** **<Issue\_name..5>** 17](#_Toc193729338)

[**6.** **<Issue\_name..6>** 19](#_Toc193729339)

[**7.** **<Issue\_name..7>** 21](#_Toc193729340)

[**7.** **Impact Analysis** 23](#_Toc193729341)

[**8. Prioritized Recommendations and Remediation Plan** 23](#_Toc193729342)

[**Conclusion** 24](#_Toc193729343)

# **Assessment Details**

|  |  |
| --- | --- |
| **Title** | Web/Android/API/Malware Application Security Assessment |
| **Version** | 0.1 |
| **Client Name** | ABC XYZ |
| **Client SPOC** | SPOC NAME |
| **Application Name** | Name of application |
| **Application Type** | Web/Mobile/API |
| **Application Environment** | UAT/Prod |
| **Application URL** | https://example.com/ |
| **Assessment Type** | External/Internal Grey/Black Box |
| **Tester** | Name Security Executive |
| **Reviewer** | Name |
| **Test Start** | 17-03-2024 **Test End** 24-03-2024 |

# **Scope**

This section outlines the specific systems, areas, or components that were assessed during the **<Security\_Assessment\_TYPE>** for **<CLIENT\_NAME>**. The scope table below identifies the core areas of focus for this assessment. Please review the scope carefully to ensure it aligns with the agreed-upon parameters of the engagement.

|  |  |  |
| --- | --- | --- |
| Scope Item | Description | Included/Excluded |
| Web/android/api | URL. | Included |

# **Note:** Any items that is not mentioned in the above illustration are excluded, they are outside the scope of this assessment and therefore were not reviewed or evaluated. If required, additional procedures can be defined for future engagements to assess the excluded items.

# **Executive Summary**

This Vulnerability Assessment was carried out on the organization’s **<APPLICATION\_NAME>** uncover potential weaknesses. The review identified **<NO.>** issues, including critical concerns such as **<DEFINE THE MAJOR ISSUE OF THE APPLICATION EX>** unsecured connections and weak password protection.

To protect against potential threats, we recommend prioritizing critical issues immediately and addressing lower risks as part of regular security upgrades. These steps will strengthen the organization’s overall digital security.

**Vulnerability statistics: Host-wise**

\*This below graph represents of critical, high, medium, low & info-risk vulnerabilities in the application.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Application Name** | CRITICAL | HIGH | MEDIUM | LOW |
|  | 0 | 0 | 2 | 5 |

Below is the table of observations for the <Application\_name> assessment conducted during <start-Date> to <End-Date>:

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. | Observation | Risk | Status |
| 1 | Host header | Medium | OPEN |
| 2 | Missing Input Validation | Low | Closed |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **Technical Summary**

OWASP has rated the Top Ten Vulnerabilities found in applications worldwide. The table shows how the application compares with respect to the OWASP Top 10 list.

SIDR Solutions application security tests are modelled along the methodologies specified by the Open Web Applications Security Project (OWASP).

|  |  |  |
| --- | --- | --- |
| **S/N** | **Vulnerabilities (OWASP TOP 10-2021)** | **Test Status** |
| A1 | Broken Access Control | Unsafe |
| A2 | Cryptographic Failures | Unsafe |
| A3 | Injection | Safe |
| A4 | Insecure Design | Safe |
| A5 | Security Misconfiguration | Unsafe |
| A6 | Vulnerable and Outdated Components | Safe |
| A7 | Identification and Authentication Failures | Unsafe |
| A8 | Software and Data Integrity Failures | Safe |
| A9 | Security Logging and Monitoring Failures | Safe |
| A10 | Server-Side Request Forgery | Safe |

# **Key Findings and Detailed Reports:**

|  |  |
| --- | --- |
| **W1. <Observation\_name>** | **Severity** |
| **Observation:** | |
| Details | |
| **CVSS Score: <version>** | |
| <score> : cvss\_vector | |
| **Exploitability:** | |
| Details | |
| **Impact:** | |
| Details | |
| **Recommendation:** | |
| Details | |

|  |  |
| --- | --- |
| **W1. WordPress Default Login Page Exposure** | **Low** |
| **Observation:** | |
| It is observed during the assessment that, the default login page for WordPress is typically located at /wp-login.php or /wp-admin/. This exposure can be a security risk as it provides a known entry point for attackers to attempt brute force attacks or other malicious activities aimed at gaining unauthorized access to the WordPress admin panel. | |
| **CVSS Score: V3** | |
| 2.7 : AV:N/AC:L/Au:N/C:P/I:P/A:P | |
| **Exploitability:** | |
| Difficult | |
| **Affected URL** | |
| <https://example.com/> | |
| **Impact:** | |
| 1.Brute Force Attacks: Attackers can easily target the default login page with automated tools to guess usernames and passwords, increasing the risk of unauthorized access.2.Account Compromise: Successful brute force attacks can lead to compromised administrator accounts, allowing attackers to modify site content, install malicious plugins, or steal sensitive data.3.Reputation Damage: A compromised WordPress site can lead to loss of user trust, potential data breaches, and damage to the organization’s reputation, along with possible legal implications | |
| **Recommendation:** | |
| It is recommended to implement the following controls: 1.Change the Default Login URL: Use plugins or custom code to change the default login URL to a less predictable one, making it harder for attackers to find.2.Implement Rate Limiting: Use security plugins or server configurations to limit the number of login attempts from a single IP address, reducing the effectiveness of brute force attacks. | |

**Steps to Reproduce/POC:**

* 1. In the below screenshots WordPress login page publicly available.

A screenshot of a computer

AI-generated content may be incorrect.

**7.** **Impact Analysis**

The vulnerabilities identified indicate a moderate security risk, with two **medium** severity issues and five **Low** severity issues. The most critical concerns relate to Cross-Origin Resource Sharing (CORS) misconfigurations and PHP version disclosure, which could expose the application to unauthorized data access and potential exploitation of known vulnerabilities. The remaining issues, while categorized as Low, contribute to a weak security posture, making the application susceptible to phishing, clickjacking, session hijacking, and brute force attacks.

* **Data Exposure & Exploitation Risk:** Improper CORS configurations may allow untrusted origins to access sensitive resources. PHP version disclosure could assist attackers in targeting known vulnerabilities.
* **Weak Security Headers:** Lack of CSP, HSTS, and X-Frame-Options headers increases susceptibility to clickjacking, MITM attacks, and unauthorized content execution.
* **Session Hijacking & CSRF Risks:** Missing SameSite cookie attributes could allow unauthorized cross-site request forgery (CSRF) attacks.
* **Brute Force Exposure:** The publicly accessible WordPress login page increases the risk of credential stuffing and brute force attacks.

# **8. Prioritized Recommendations and Remediation Plan**

**Immediate Actions:**

The most critical risks involve **CORS misconfigurations** and **PHP version disclosure**. Allowing arbitrary origins can expose sensitive data, so access should be restricted to trusted domains. PHP version disclosure makes the application vulnerable to targeted attacks, requiring immediate removal of version details from HTTP response headers. Addressing these risks will reduce potential exploitation.

**Medium-Term Actions:**

Security hardening measures include enforcing **HTTP Strict Transport Security (HSTS)** to prevent downgrade attacks and implementing a **secure Content Security Policy (CSP)** to mitigate XSS risks. Adding **X-Frame-Options headers** will defend against clickjacking, while **SameSite cookie attributes** will reduce CSRF risks. Strengthening the **WordPress login page** with rate limiting, MFA, and IP restrictions will help prevent brute force attacks.

**Long-Term Actions:**

Ongoing security efforts should focus on **regular security audits and monitoring** to detect emerging threats. A **Web Application Firewall (WAF)** should be deployed to block malicious traffic, and **real-time security monitoring** should be implemented. Routine **software updates, access control policies, and security awareness training** will help maintain a strong security posture.

# **Conclusion**

The assessment identified Seven vulnerabilities with varying levels of severity. Addressing these findings will significantly enhance the application’s security, protect user data, and ensure compliance with industry standards. Implementing the recommended remediation actions will establish a robust security posture for the organization.

**Prepared by:** SIDR Solutions

**Note:** This contains sensitive information. Please restrict access to authorized personnel only.

**Test Cases:**

|  |  |  |
| --- | --- | --- |
| **1. Information Gathering** | |  |
| **ID** | **Test Name** | **Status** |
| 1.1 | Conduct Search Engine Discovery Reconnaissance for Information Leakage | Not started |
| 1.2 | Fingerprint Web Server | Not started |
| 1.3 | Review Webserver Metafiles for Information Leakage | Not started |
| 1.4 | Enumerate Applications on Webserver | Not started |
| 1.5 | Review Webpage Content for Information Leakage | Not started |
| 1.6 | Identify Application Entry Points | Not started |
| 1.7 | Map Execution Paths Through Application | Not started |
| 1.8 | Fingerprint Web Application Framework | Not started |
| 1.9 | Fingerprint Web Application | Not started |
| 1.1 | Map Application Architecture | Not started |
|  |  |  |
| **2. Configuration and Deploy Management Testing** | |  |
| 2.1 | Test Network Infrastructure Configuration | Not started |
| 2.2 | Test Application Platform Configuration | Not started |
| 2.3 | Test File Extensions Handling for Sensitive Information | Not started |
| 2.4 | Review Old Backup and Unreferenced Files for Sensitive Information | Not started |
| 2.5 | Enumerate Infrastructure and Application Admin Interfaces | Not started |
| 2.6 | Test HTTP Methods | Not started |
| 2.7 | Test HTTP Strict Transport Security | Not started |
| 2.8 | Test RIA Cross Domain Policy | Not started |
| 2.9 | Test File Permission | Not started |
| 2.1 | Test for Subdomain Takeover | Not started |
| 2.11 | Test Cloud Storage | Not started |
| 2.12 | Testing for Content Security Policy | Not started |
| 2.13 | Test Path Confusion | Not started |
|  |  |  |
| **3. Identity Management Testing** | |  |
| 3.1 | Test Role Definitions | Not started |
| 3.2 | Test User Registration Process | Not started |
| 3.3 | Test Account Provisioning Process | Not started |
| 3.4 | Testing for Account Enumeration and Guessable User Account | Not started |
| 3.5 | Testing for Weak or Unenforced Username Policy | Not started |
|  |  |  |
| **4. Authentication Testing** | |  |
| 4.1 | Testing for Credentials Transported over an Encrypted Channel | Not started |
| 4.2 | Testing for Default Credentials | Not started |
| 4.3 | Testing for Weak Lock Out Mechanism | Not started |
| 4.4 | Testing for Bypassing Authentication Schema | Not started |
| 4.5 | Testing for Vulnerable Remember Password | Not started |
| 4.6 | Testing for Browser Cache Weaknesses | Not started |
| 4.7 | Testing for Weak Password Policy | Not started |
| 4.8 | Testing for Weak Security Question Answer | Not started |
| 4.9 | Testing for Weak Password Change or Reset Functionalities | Not started |
| 4.1 | Testing for Weaker Authentication in Alternative Channel | Not started |
| 4.11 | Testing Multi-Factor Authentication (MFA) | Not started |
|  |  |  |
| **5. Authorization Testing** | |  |
| 5.1 | Testing Directory Traversal File Include | Not started |
| 5.2 | Testing for Bypassing Authorization Schema | Not started |
| 5.3 | Testing for Privilege Escalation | Not started |
| 5.4 | Testing for Insecure Direct Object References | Not started |
| 5.5 | Testing for OAuth Weaknesses | Not started |
|  |  |  |
| **6. Session Management Testing** | |  |
| 6.1 | Testing for Session Management Schema | Not started |
| 6.2 | Testing for Cookies Attributes | Not started |
| 6.3 | Testing for Session Fixation | Not started |
| 6.4 | Testing for Exposed Session Variables | Not started |
| 6.5 | Testing for Cross Site Request Forgery | Not started |
| 6.6 | Testing for Logout Functionality | Not started |
| 6.7 | Testing Session Timeout | Not started |
| 6.8 | Testing for Session Puzzling | Not started |
| 6.9 | Testing for Session Hijacking | Not started |
| 6.1 | Testing JSON Web Tokens | Not started |
|  |  |  |
| **7. Data Validation Testing** | |  |
| 7.1 | Testing for Reflected Cross Site Scripting | Not started |
| 7.2 | Testing for Stored Cross Site Scripting | Not started |
| 7.3 | Testing for HTTP Verb Tampering | Not started |
| 7.4 | Testing for HTTP Parameter Pollution | Not started |
| 7.5 | Testing for SQL Injection | Not started |
| 7.6 | Testing for LDAP Injection | Not started |
| 7.7 | Testing for XML Injection | Not started |
| 7.8 | Testing for SSI Injection | Not started |
| 7.9 | Testing for XPath Injection | Not started |
| 7.1 | Testing for IMAP SMTP Injection | Not started |
| 7.11 | Testing for Code Injection | Not started |
| 7.12 | Testing for Command Injection | Not started |
| 7.13 | Testing for Format String Injection | Not started |
| 7.14 | Testing for Incubated Vulnerability | Not started |
| 7.15 | Testing for HTTP Splitting Smuggling | Not started |
| 7.16 | Testing for HTTP Incoming Requests | Not started |
| 7.17 | Testing for Host Header Injection | Not started |
| 7.18 | Testing for Server-side Template Injection | Not started |
| 7.19 | Testing for Server-Side Request Forgery | Not started |
| 7.2 | Testing for Mass Assignment | Not started |
|  |  |  |
| **8. Error Handling** | |  |
| 8.1 | Testing for Improper Error Handling | Not started |
| 8.2 | Testing for Stack Traces | Not started |
|  |  |  |
| **9. Cryptography** | |  |
| 9.1 | Testing for Weak Transport Layer Security | Not started |
| 9.2 | Testing for Padding Oracle | Not started |
| 9.3 | Testing for Sensitive Information Sent via Unencrypted Channels | Not started |
| 9.4 | Testing for Weak Encryption | Not started |
|  |  |  |
| **10. Business logic Testing** | |  |
| 10.1 | Test Business Logic Data Validation | Not started |
| 10.2 | Test Ability to Forge Requests | Not started |
| 10.3 | Test Integrity Checks | Not started |
| 10.4 | Test for Process Timing | Not started |
| 10.5 | Test Number of Times a Function Can Be Used Limits | Not started |
| 10.6 | Testing for the Circumvention of Work Flows | Not started |
| 10.7 | Test Defenses Against Application Misuse | Not started |
| 10.8 | Test Upload of Unexpected File Types | Not started |
| 10.9 | Test Upload of Malicious Files | Not started |
| 10.1 | Test Payment Functionality | Not started |
|  |  |  |
| **11. Client Side Testing** | |  |
| 11.1 | Testing for DOM-Based Cross Site Scripting | Not started |
| 11.2 | Testing for JavaScript Execution | Not started |
| 11.3 | Testing for HTML Injection | Not started |
| 11.4 | Testing for Client-side URL Redirect | Not started |
| 11.5 | Testing for CSS Injection | Not started |
| 11.6 | Testing for Client-side Resource Manipulation | Not started |
| 11.7 | Testing Cross Origin Resource Sharing | Not started |
| 11.8 | Testing for Cross Site Flashing | Not started |
| 11.9 | Testing for Clickjacking | Not started |
| 11.1 | Testing WebSockets | Not started |
| 11.12 | Testing Web Messaging | Not started |
| 11.13 | Testing Browser Storage | Not started |
| 11.14 | Testing for Cross Site Script Inclusion | Not started |
| 11.15 | Testing for Reverse Tabnabbing | Not started |