Author

CAST

******

**OWASP 2013 TOP 10**

**Summary Report**

Application Name –

Version –

CAST AIP -

|  |
| --- |
|  |
|  |

Monday, xx July 2012

My Application Name

Version Number

My CAST Version

# Table of Content

Table of Content

1. Introduction

1.1. Application Characteristics

2. Security Violation Overview

2.1. OWASP -2013 Top 10 vulnerabilities

2.2. OWASP -2013 A1 - Injection

2.3. OWASP -2013 A2 – Broken Authentication & Session Management

2.4. OWASP -2013 A3 – Cross-Site Scripting

2.5. OWASP -2013 A4 – Insecure Direct Object References

2.6. OWASP -2013 A5 – Security Misconfiguration

2.7. OWASP -2013 A6 – Sensitive Data Exposure

2.8. OWASP -2013 A7 – Missing Function Level Access Control

2.9. OWASP -2013 A8 – Cross Site Request Forgery

2.10. OWASP -2013 A9 – Using Components with Known Vulnerabilities

2.11. OWASP -2013 A10 – Unvalidated Redirects & Forwards

3. Appendix

3.1. About CAST Software Intelligence

3.2. About CAST Security

# Introduction

This assessment is an effort to determine the security health of the application and identify some of the root causes of current Security concerns, as well as any risks of future degradation. This assessment uses the CAST Application Intelligence Platform (AIP) to automatically scan the implementation of these applications to review the architecture, design, and code against OWASP standards.

CAST AIP adapts the quality rules from best-in-class industry standards (OWASP, CWE, CISQ). With its unique ability to perform dataflow and system-level analysis (From Presentation layer to Database layer), CAST provides the most accurate security findings, reducing a lot of false positives.

## Application Characteristics

This assessment is focused solely on the technical implementation of the said application (user interface to database), with no investigation of the functionality.

|  |  |
| --- | --- |
| Name | Value |
| kLoC | 504 |
| Files | 6,586 |
| Classes | 593 |
| SQL Art. | 0 |
| Tables | 119 |

*Fig 1: Application Technology characteristics Table 1: Application characteristics*

# Security Violation Overview

This section provides a summary of the most severe security vulnerability identified in the structural quality analysis and measurement by CAST AIP against the OWASP 2013 standard. Details about OWASP Security Standard can be found [here](https://www.owasp.org/index.php/Top_10_2013-Top_10).

## OWASP -2013 Top 10 vulnerabilities

The [OWASP Top 10](https://www.owasp.org/index.php/Top_10_2013-Top_10) focuses on identifying the most serious web application security risks for a broad array of organizations.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| OWASP-2013 | Exploitability | Weakness Prevalence | Weakness Detectability | Technical Impact |
| A1 - Injection | Easy | Common | Average | Severe |
| A2 - Broken Authentication | Average | Widespread | Average | Severe |
| A3 - Sensitive Data Exposure | Average | Very Widespread | Easy | Moderate |
| A4 - XML External Entities (XXE) | Easy | Common | Easy | Moderate |
| A5 - Broken Access Control | East | Common | Easy | Moderate |
| A6 - Security Misconfiguration | Difficult | Uncommon | Average | Severe |
| A7 - Cross-Site Scripting (XSS) | Easy | Common | Average | Moderate |
| A8 - Insecure Deserialization | Average | Common | Easy | Moderate |
| A9 - Using Components with Known Vulnerabilities | Average | Widespread | Difficult | Moderate |
| A10 - Insufficient Logging & Monitoring | Average | Widespread | Difficult | Moderate |

List of OWASP -2013 rules that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| OWASP-2013 | Total | Added | Removed |
| A1 | 0 | 0 | 0 |
| A2 | 0 | 0 | 0 |
| A3 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 |
| … | 0 | 0 | 0 |

*Table 2: OWASP 2013 Top 10 Rules*

## OWASP -2013 A1 - Injection

This category of rules primarily deals with issues such as - Injection flaws, such as SQL, OS, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker’s hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

List of A1-Injection rules that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 3: A1 - Injection vulnerabilities*

## OWASP -2013 A2 – Broken Authentication & Session Management

Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users’ identities.

List of A2-Broken Authentication & Session Managementrules that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 4: A2- Broken Authentication & Session Management vulnerabilities*

## OWASP -2013 A3 – Cross-Site Scripting

XSS flaws occur whenever an application takes untrusted data and sends it to a web browser without proper validation or escaping. XSS allows attackers to execute scripts in the victim’s browser which can hijack user sessions, deface web sites, or redirect the user to malicious sites.

List of A3-Cross-site Scripting rules that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 5: A3- Cross-site scripting vulnerabilities*

## OWASP -2013 A4 – Insecure Direct Object References

A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other protection, attackers can manipulate these references to access unauthorized data.

List of A4- Insecure Direct Object References rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 6: A4- Insecure Direct Object References vulnerabilities*

## OWASP -2013 A5 – Security Misconfiguration

Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. Secure settings should be defined, implemented, and maintained, as defaults are often insecure. Additionally, software should be kept up to date.

List of A5 – Security Misconfiguration rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 7: A5- Security Misconfiguration vulnerabilities*

## OWASP -2013 A6 – Sensitive Data Exposure

Many web applications do not properly protect sensitive data, such as credit cards, tax IDs, and authentication credentials. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data deserves extra protection such as encryption at rest or in transit, as well as special precautions when exchanged with the browser.

List of A6 – Sensitive Data Exposure rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 8: A6 – Sensitive Data Exposure vulnerabilities*

## OWASP -2013 A7 – Missing Function Level Access Control

Most web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to forge requests in order to access functionality without proper authorization.

List of A7 – Missing Function Level Access Control rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 9: A7 – Missing Function Level Access Control vulnerabilities*

## OWASP -2013 A8 – Cross Site Request Forgery

A CSRF attack forces a logged-on victim’s browser to send a forged HTTP request, including the victim’s session cookie and any other automatically included authentication information, to a vulnerable web application. This allows the attacker to force the victim’s browser to generate requests the vulnerable application thinks are legitimate requests from the victim.

List of A8 – Cross site Request Forgery rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 10: A8 – Cross Site Request Forgery vulnerabilities*

## OWASP -2013 A9 – Using Components with Known Vulnerabilities

Components, such as libraries, frameworks, and other software modules, almost always run with full privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts.

List of A9 – Using Components with known vulnerabilities rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 11: A9 – Using Components with Known Vulnerabilities*

## OWASP -2013 A10 – Unvalidated Redirects & Forwards

Web applications frequently redirect and forward users to other pages and websites and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages

List of A10 – Using Unvalidated Redirects & Forwards rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| CAST Rules | Total | Added | Removed |
| Rule 1 | 0 | 0 | 0 |
| Rule 2 | 0 | 0 | 0 |
| Rule 3 | 0 | 0 | 0 |
| Rule 4 | 0 | 0 | 0 |
| Rule 5 | 0 | 0 | 0 |

*Table 11: A10 – Unvalidated Redirects & Forward vulnerabilities*

# Appendix

## About CAST Software Intelligence

Software Intelligence creates understanding into software architecture, end to end transaction flows, data access patterns and more, helping teams work confidently and faster. Hundreds of companies rely on CAST Software Intelligence to improve end-user satisfaction and time-to-market, prevent business disruption and reduce cost, enabling them to move past today’s obstacles and to tackle the next wave of innovation.

[Click here](https://www.castsoftware.com/software-intelligence) for more information about CAST Software Intelligence.

## About CAST Security

Cyber risk and application security require a proactive and intelligence-driven approach. CAST Software Intelligence shifts insight into security strategy blind spots before development starts. With its unique ability to do dataflow and system-level analysis, CAST provides the most accurate security findings, reducing a lot of false positives. CAST Security rules are adapted from best-in-class industry standards – CISQ, CWE, and OWASP.

To find out more about CAST Security, [click here](https://www.castsoftware.com/use-cases/application-security).