Author

CAST



**OWASP 2017 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 2

1. Introduction 3

1.1. Application Characteristics 3

1.2. Application Health (Security) 4

2. Security Violation Overview 5

2.1. OWASP -2017 Top 10 violations 5

2.2. OWASP -2017 A1 - Injection 6

2.3. OWASP -2017 A3 – Sensitive Data Exposure 6

2.4. OWASP -2017 A6 – Security Misconfiguration 7

2.5. OWASP -2017 A7 – Cross-Site Scripting (XSS) 7

2.6. OWASP -2017 A8 – Insecure Deserialization 8

2.7. OWASP -2017 A9 – Using Components with known Vulnerabilities 8

3. Appendix 9

3.1. About CAST Software Intelligence 9

3.2. How CAST AIP Works 9

# 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 applies over 1200 engineering checks based on standards and measurements developed by the Software Engineering Institute (SEI), International Standards Organization (ISO), Consortium for IT Software Quality (CISQ), the Institute of Electrical and Electronics Engineers (IEEE), Department of Homeland Security (DHS), US Computer Emergency Response Team (CERT), the National Institute of Standards and Technology (NIST), MITRE, Open Web Application Security Project (OWASP) and the technology provider industry. The resulting analysis identifies specific flaws in the software and aggregates this information into metrics to objectively quantify the structural quality of the application.

## 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 provide a summary of the most severe scurity vulnerability identified in the structural quality analysis and mesurement by CAST AIP against the OWASP 2017 standard. Details about OWASP Security Standard can be found at - <https://www.owasp.org/index.php/Top_10-2017_Top_10>

## OWASP -2017 Top 10 violations

List of OWASP -2017 rules that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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: OWASP 2017 Top 10 Rules*

## OWASP -2017 A1 - Injection

This category of rules primarily deals with issues such as - Injection flaws, such as SQL, NoSQL, 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 violations that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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: A1- Injection violations*

## OWASP -2017 A3 – Sensitive Data Exposure

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

List of A3-Injection violations that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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- Sensitive Data Exposure violations*

## OWASP -2017 A6 – Security Misconfiguration

Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.

List of A6-Security Misconfiguration violations that had any findings in this application.

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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: A6- Security Misconfiguration violations*

## OWASP -2017 A7 – Cross-Site Scripting (XSS)

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. 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 A7 – Cross-Site Scripting rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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: A7- Cross-Site Scripting violations*

## OWASP -2017 A8 – Insecure Deserialization

Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

List of A8 – Insecure Deserialization rules that had any findings in this application

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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: A7- Insecure Deserialization violations*

## OWASP -2017 A9 – Using Components with known Vulnerabilities

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

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

|  |  |  |  |
| --- | --- | --- | --- |
| Rules | Total Violations | Added Violations | Removed Violations |
| 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: A9 – Using Components with known vulnerabilities violations*

# 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.

For more information on CAST Software Intelligence, visit - <https://www.castsoftware.com/software-intelligence>

## How CAST AIP Works

CAST connects into all major SCM systems or can take source code in whatever format it is maintained in the organization. Source code is then processed and stored in the CAST Knowledge Base as metadata, which forms the basis for the analysis and information provided by CAST AIP. CAST looks at the entire application—including legacy components, packaged app customizations, and all modern distributed technology environments. Data from third party code analyzers can be integrated into the CAST Knowledge Base and displayed in AIP dashboards.

