Fortify Security Report

May 27, 2015

igen194

Executive Summary

Issues Overview

On May 27, 2015, a source code review was performed over the fortify-priority-inconsistency-demo code base. 4 files, 22 LOC (Executable) were scanned and reviewed for defects that could lead to potential security vulnerabilities. A total of 6 reviewed findings were uncovered during the analysis.

Issues by Fortify Priority Order	
Low	5
High	1

Recommendations and Conclusions

The Issues Category section provides Fortify recommendations for addressing issues at a generic level. The recommendations for specific fixes can be extrapolated from those generic recommendations by the development group.

Project Summary

Code Base Summary

Code location: C:/Users/IGEN194/workspace_eclipse_kepler/fortify-priority-inconsistency-demo
Number of Files: 4
Lines of Code: 22
Build Label: <No Build Label>

Scan Information

Scan time: 00:19
SCA Engine version: 6.21.0005
Machine Name: INFO-DS039E51F
Username running scan: igen194

Results Certification

Results Certification Valid

Details:

Results Signature:

SCA Analysis Results has Valid signature

Rules Signature:

There were no custom rules used in this scan

Attack Surface

Attack Surface:
Command Line Arguments:
demo.fortify.priority.fortify_priority_inconsistency_demo.App.main

System Information:
null.null.null

Filter Set Summary

Current Enabled Filter Set:
Security Auditor View

Filter Set Details:

Folder Filters:
If [fortify priority order] contains critical Then set folder to Critical
If [fortify priority order] contains high Then set folder to High
If [fortify priority order] contains medium Then set folder to Medium
If [fortify priority order] contains low Then set folder to Low
Visibility Filters:

Audit Guide Summary

File System Inputs

Hide issues involving file system inputs.
Depending on your system, inputs from files may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from the file system if it is trusted.
Enable if you trust file system inputs.

Filters:
If taint contains file_system Then hide issue
If taint contains constantfile Then hide issue
If taint contains stream Then hide issue
If category is file access race condition Then hide issueTaint from Command-Line Arguments

Hide issues involving taint from command-line arguments.
Depending on your system, inputs from command-line arguments may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from command-line arguments if they are trusted.
Enable if you trust command-line arguments.

Filters:
If taint contains args Then hide issueProperty File Inputs

Hide inputs from properties files.
Depending on your system, inputs from properties files may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from properties files if they are trusted.
Enable if you trust inputs from properties files.

Filters:
If taint contains property Then hide issueEnvironment Variable Inputs

Hide issues involving environment variable inputs.
Depending on your system, inputs from environment variables may or may not come from trusted users. AuditGuide can hide issues that are based on data coming from environment variables if they are trusted.
Enable if you trust environment variable inputs.

Filters:
If taint contains environment Then hide issue

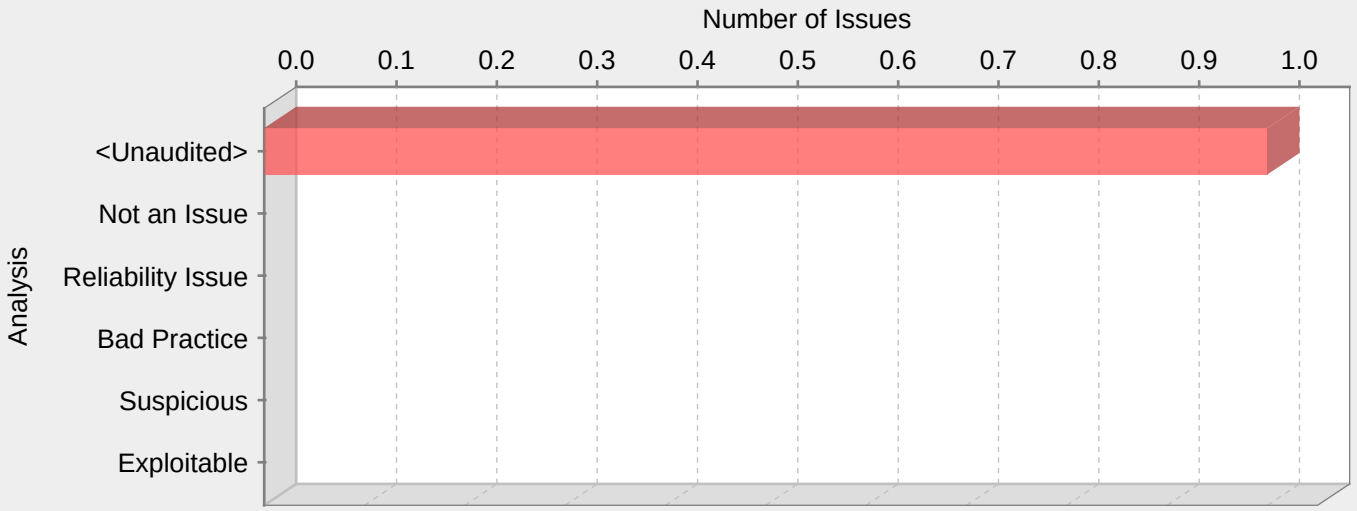
Results Outline

Overall number of results

The scan found 6 issues.

Vulnerability Examples by Category

Category: XML External Entity Injection (1 Issues)



Abstract:

Using XML parsers configured to not prevent nor limit external entities resolution can expose the parser to an XML External Entities attack

Explanation:

XML External Entities attacks benefit from an XML feature to build documents dynamically at the time of processing. An XML entity allows inclusion of data dynamically from a given resource. External entities allow an XML document to include data from an external URI. Unless configured to do otherwise, external entities force the XML parser to access the resource specified by the URI, e.g., a file on the local machine or on a remote system. This behavior exposes the application to XML External Entity (XXE) attacks, which can be used to perform denial of service of the local system, gain unauthorized access to files on the local machine, scan remote machines, and perform denial of service of remote systems.

The following XML document shows an example of an XXE attack.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///dev/random" >]><foo>&xxe;</foo>
```

This example could crash the server (on a UNIX system), if the XML parser attempts to substitute the entity with the contents of the /dev/random file.

Recommendations:

An XML parser should be configured securely so that it does not allow external entities as part of an incoming XML document.

To avoid XXE injections the following properties should be set for an XML factory, parser or reader:

```
factory.setFeature("http://xml.org/sax/features/external-general-entities", false);
factory.setFeature("http://xml.org/sax/features/external-parameter-entities", false);
```

If inline DOCTYPE declaration is not needed, it can be completely disabled with the following property:

```
factory.setFeature("http://apache.org/xml/features/disallow-doctype-decl", true);
```

DefaultXmlValidator.java, line 54 (XML External Entity Injection)

Fortify Priority:	High	**Folder**	High
Kingdom:	Input Validation and Representation		
Abstract:	XML parser configured in DefaultXmlValidator.java:54 does not prevent nor limit external entities resolution. This can expose the parser to an XML External Entities attack.		

Sink:	DefaultXmlValidator.java:54 parser.parse(...) : XML document parsed allowing external entity resolution()

```
52
53                      final ByteArrayInputStream inputStream = new
ByteArrayInputStream(xmlString.getBytes(Charset.forName("UTF-8")));
54                      final Document document = parser.parse(inputStream);
55                      inputStream.reset();
56                      inputStream.close();
```

Issue Count by Category

Issues by Category	
Build Misconfiguration: External Maven Dependency Repository	1
J2EE Bad Practices: Leftover Debug Code	1
Missing XML Validation	1
Poor Logging Practice: Use of a System Output Stream	1
XML Entity Expansion Injection	1
XML External Entity Injection	1

Issue Breakdown by Analysis

Issues by Analysis