RFP 176 - OSGi Testcases - Malicious bundles framework

Draft

12 Pages

Abstract

Present document is a proposal to extend existing OSGi test cases framework with a set of malicious bundles aimed at assessing robustness of OSGi platforms against known Java/OSGi security vulnerabilities. The goal is also to provide a set of known OSGi vulnerabilities assessment specifications and a common security test framework to help OSGi partners implement platforms with a minimal level of acceptable robustness against commonly known OSGi cyber-security threats.

0 Document Information

0.1 Table of Contents

0.2 Terminology and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in 6.1.

```
Source code is shown in this typeface.
```

0.3 Revision History

The last named individual in this history is currently responsible for this document.

Revision	Date	Comments
Initial, Rev1.0	07-24-2015	RFP Creation Julien HELMER, Sogeti High-Tech, julien.helmer@sogeti.com
Rev2.0	07-28-2015	Typo corrections, requirements clarification, added mention about known OSGi vulnerabilities list and malicious test cases specifications, mentioned a testsuite execution model.
Rev2.1	07-28-2015	Removed statements related to commitments on security recommendations
Rev3.0	07-30-2015	Reformatting to better fit in OSGi expected format for RFP.

1 Introduction

A state-of-the-art list of known OSGi vulnerabilities and associated OSGi test framework were created in past Open the Box project ([3].) within SOGETI HIGH-TECH to assess robustness of several OSGi embedded platforms from various manufacturers. This framework is based on malicious bundles specifications and test suite that were developed specifically to assess robustness of OSGi platforms against commonly known OSGi vulnerabilities.. Detailed list and description of the malicious bundles developed until end of 2014 is presented in [4].

The vulnerabilities covered were estimated from state-of-the-art studies, especially work published by INRIA in past years ([7]., [8]., [9]., [10].).

The test framework also includes some basic scripts so as to execute every compiled malicious bundle test case as automatic and simple as possible on OSGi target platform..

Following a discussion with Orange Labs, we decided to propose this RFP so as to share this security test framework with OSGi IoT expert group (IoT EG). The goal is to continue the development of this security test framework in collaboration with OSGi IoT EG, and build from this framework a tool that could help OSGi members build more robust platforms. The framework can be splitted as below:

- The state-of-the-art list of known OSGi vulnerabilities (currently up to Q4”2014, will be updated when need be)
  - This document provides a view on known OSGi vulnerabilities and associated attack vectors. This can be used as guidelines by OSGi members when considering evaluation of overall platform robustness level against cyber-security threats.
- Malicious bundle test cases technical specifications
  - This document can be used as guidelines by OSGi members when considering specific security test case(s) development for internal assessment of their OSGi platform implementation.
- Malicious bundles test suite
  - This test suite can either be integrated within the standard OSGi test suite shared amoung all OSGi partners, or can be provided as a service from OSGi Alliance “test center” to the OSGi partners (possibly better model to cope with confidentiality concerns of OSGi partners, also allow limited audience for malicious bundle source code access and consequently reduce the risk of malicious test bundles dissemination)

2 Application Domain

This proposal is focused on OSGi test case framework.

2.1 Terminology + Abbreviations

TC(s) = Test Case(s)

RSA = Remote Service Admin

JAAS = Java Authentication and Authorization Service

2.2 OSGi test cases & Security

OSGi test case s are currently oriented on functional testing of OSGi framework API. Main goal of currently available OSGi tests cases is to ensure implementation will be inline with OSGi specification so that compatibility between framework/services will be guaranteed given a specific version of OSGi specification.
Few OSGi test cases cover security considerations (ex: JMX framework secure TC, JNDI secure TC, RSA secure TC), following the approach of functional testing with security enabled (permission manager effective). Hence, “secure” test case s are not focusing on security vulnerabilities but aimed to ensure a service will still operate correctly under restricted permissions, or will handle permissions as expected by OSGi framework specification.

2.3 OSGi framework considerations

According to OSGi framework documents available online for main framework (Felix, Equinox, Knopflerfish), security configuration (permissions definition) is mainly left to developer responsibility. In most cases, documents also clearly highlight the fact that developers will most of the time never have to consider security restriction as by default all permissions are given to any services once security is enabled.

Implementing secure OSGi framework or bundles come more from developers specific needs rather than commonly shared best practices provided to developers community.

In addition, some OSGi framework provider could consider security or hardening as an added value subject to specific commercial conditions (pricing, level of service).

3 Problem Description

Current OSGi test case s do not provide any malicious security test case that is aimed at vulnerabilities assessment within OSGi framework/bundle implementations. By consequence, most developers have no practical mean to easily assess robustness level of their OSGi framework or bundles implementation, except a few of them who are already skilled in Java/OSGi security. This potentially means that many unsecured OSGi platforms will reach public market with a risk of security or privacy breaches being exposed.
The goal of our proposal is to bring into OSGi community a framework that could help developers in getting a first estimate of their implementation robustness. As presented in the introduction, this framework is constituted of:

- A state-of-the-art list of known OSGi vulnerabilities
- Malicious bundle test cases technical specifications
- Malicious bundles test suite

Malicious bundles are aimed at implementing attacks exploiting commonly known OSGi vulnerabilities. When malicious bundle is executed on developer's target, results will allow to determine if the implemented platform is vulnerable or robust against each attacks. List of currently available malicious bundles will be described in next section.

Because security is a major concern in IoT business, OSGi community has to build secure solutions based on a first level of commonly shared vulnerabilities assessment . Then of course, depending the expertise level of each OSGi solution provider, security level can be extended and promoted as a commercial differenciator. At least a minimum acceptable security level shall be a common requirement for any OSGi product, and proposed security framework can help reaching this common milestone among the OSGi partners.

4 Use Cases

Find here below the description of a set of security attacks. They are grouped by topics:

- Network communication
- OS
- OSGi core framework
- other topics

Those test cases were developed as malicious bundles and tested under OSGi R4 specification, within:

- Felix 4.2.1
- Knopflerfish 5.3.3

Compiled using Java SE 1.5 or greater.

4.1 Malicious test cases targeting network communication

This type of attacks targets a malicious usage of the network interface on OSGi platform. The groups of malicious test cases covered in past Open the Box project [4]. are listed below:

- Network discovery, ports & services identification
- Network resource exhaustion
- Pharming, network configuration manipulation (ex: DNS config)

  Pharming[p] is a cyber attack intended to redirect a website's traffic to another, fake site

  Pharming definition on Wikipedia, link : https://en.wikipedia.org/wiki/Pharming

4.2 Malicious test cases targeting OS

This type of security attacks targets a malicious interaction on JVM or Native Operating System

The groups of malicious test cases that are implemented in past Open the Box project [4]. are listed below:

- Data leakage
- Log files recovery and sharing
- Password files access permission
- Filesystem discovery
- Execution flow manipulation
- Symbolic links manipulation
- System configuration files manipulation

4.3 Malicious test cases targeting OSGi framework

This type of security attacks targets a malicious interaction with OSGi core framework. The groups of malicious test cases that were implemented in past Open the Box project [4]. are listed below:

- Theft of data
- Theft of bundles
- Theft of framework
- Data injection (ex: Manifest.mf)

- Denial-of-Service (ex: resource exhaustion, infinite loop, dead lock, nested service subscription)
- Bundles injection

4.4 Malicious test cases within bundles

This type of security attacks targets malicious operations possible from within an OSGi bundle.

Currently covered malicious test cases groups for this use-case are listed below:

- Denial-of-service (infinite loop, resource overload, resource exhaustion, allocation, dead lock, race condition, bundle state manipulation, nested service/framework invocation)
- Access to protected package
- Malicious data forging/injection
- Load malicious fragment class
- Use of obsoleted / unsafe classes
- Temporary files extraction
- Compiled Java classes reverse-engineering
- JVM crashing

4.5 Potential future developments

Find here below a list of interesting topics foreseen for OSGi malicious test cases developments:

- Bundles signature manipulation

  Ensure digital signature can not be re-built without previous access or knowledge of original legitimate signing key (certificate chain-of-trust robustness). Test cases also need to assess that signature tampering effectively leads to bundle loading failure. To be mentioned here that INRIA Sfelix team already published in 2007 some considerations with regard to secure deployment of bundles including digital signature framework ([6].)

- Bundle code injection (Java ByteCode manipulation/injection)
- Authentication mechanisms robustness

  This topic has to be further studied, either in scope of OSGi service(s) providing user authentication features (ex: JAAS) or in case components based access control (CBAC) is used.

- Bundles fingerprinting

  Identify available bundles on a platform and associated versions, then to compare this current bundles list+version to commonly known vulnerable bundles list+version.

– OSGi UPnP service: NAT manipulation

Because UPnP specification states security is optional, security concerns related to manipulation of NAT rules through UPnP are rarely addressed in product implementation. In case of OSGi bundle, this could result in malicious bundles using UPnP to open backdoor on WAN gateways for external attackers to remotely access end-users LAN or machines.

– OSGi USB service: robustness against BadUSB-like attacks

Ensure USB service/driver implements support only for expected type of USB devices (support for others shall be removed or disabled), also with minimum set of USB operations and permissions so to restrict USB usage to only few usecases supported by the platform.

5 Requirements

- REQ-00: Test framework MUST support all existing OSGi execution environments since OSGi R4.
- REQ-01: Test framework MUST integrate in all existing OSGi test cases framework since OSGi R4.
- REQ-02: Test framework MUST be appliable on any OSGi framework (be it open-source or not).
- REQ-03: Every malicious bundle test case MUST provide description of the vulnerability tested.
- REQ-04: Every malicious bundle test case MUST provide clear vulnerability assessment result (PASS/FAIL). In case of FAIL, a longer description of the possible failure causes SHOULD be provided.
- REQ-05: It MAY be required to develop vulnerability scoring system like CVSS to sort test results by severity (see [5]. for more details)
- REQ-06: A tracking spread sheet for all malicious TC with associated status (TC Identifier, description, availability, maturity, severity of the vulnerability) MUST be maintained inside OSGi Alliance
- REQ-07: Along with malicious bundles, shell scripts aimed at automating malicious bundles execution on targeted platform SHOULD also be provided.
- REQ-08: List of supported Java/OSGi known vulnerabilities MUST be periodically updated on regular basis, so to keep align on state-of-the-art practices for this domain

6 Document Support

6.1 References

[1]. Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, RFC2119, March 1997.

[2]. Software Requirements & Specifications. Michael Jackson. ISBN 0-201-87712-0

[3]. Open the box project websites: http://openthebox.org/, Minalogic project description (link)

[4]. Open the Box deliverable SP4.3.5b - Exigences techniques Framework de Pentest, v0.31 (2014)

[5]. CVSS Specification : https://www.first.org/cvss/specification-document

[6]. INRIA SecureFelix project: http://sfelix.gforge.inria.fr

[7]. Parrend, P., Frénot, F. : Java Components Vulnerabilities - An Experimental Classification Targeted at the OSGi Platform. INRIA - Rapport de recherche n 6231 (June 2007)

[8]. Parrend, P., Frénot, F. : More Vulnerabilities in the Java/OSGi Platform : A Focus on Bundle Interactions. INRIA - Rapport de recherche n 6649 (September 2008)

[9]. Parrend, P., Frénot, F. : Security benchmarks of OSGi plateform : toward Hardened OSGi. JohnWilay & Sons, Inc. (April 2009)

[10]. Goichon, F., Salagnac, G., Frénot, S. : Exploiting Java Code Interactions. INRIA - Rapport de recherche n 0419 (December 2011)

6.2 Authors Address

Name	HELMER Julien
Company	SOGETI HIGH-TECH
Address	95 CHEMIN DE L'ETOILE – 38330 MONTBONNOT-SAINT-MARTIN – FRANCE
Voice	-
e-mail	julien.helmer@sogeti.com

Name	BOTTARO André
Company	Orange
Address	28 Chemin du Vieux Chêne, 38243 Meylan, France
Voice	+33 4 76 76 41 03
e-mail	andre.bottaro@orange.com

End of Document