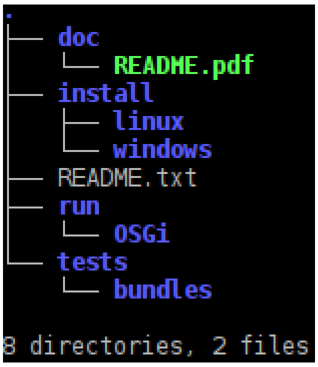
OSGi RFP-176 Malicious Bundle Test Framework - Readme

Table of Content

1 Test framework installation

1.1 Framework content



```
.
├── doc
│   └── README.pdf
├── install
│   ├── linux
│   └── windows
├── README.txt
├── run
│   └── OSGi
└── tests
    └── bundles

8 directories, 2 files
```

- **Root directory** : contains README.txt file, and the following directories.
- **./doc** : contains the README.pdf file (this file).
- **./install** : contains installation scripts for Linux and Windows platforms.
    - **./install/linux** : contains installation bash scripts of Felix OSGi framework for Linux platform.
    - **./install/windows** : contains installation bash scripts of Felix OSGi framework for Windows platform.
    - **./run** : contains execution scripts of security tests.
        - **./run/OSGi** : contains execution scripts of OSGi security tests for Linux and Windows.
- **./tests** : contains all OSGi security tests, pre-compiled malicious bundles and the server binaries for tests which need it.

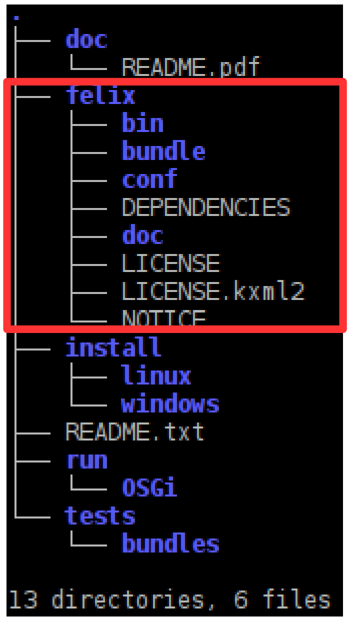
After framework installation and/or execution, some temporary directories will be created :

- ./logs : contains log files of bundles executed by the framework.
- ./felix : contains Felix OSGi framework system files.
- ./tmp : contains temporary files generated by Felix OSGi framework.

1.2 OSGi tests installation

OSGi tests framework execution needs an installed Java virtual machine on the platform (Java SE 1.5 or higher). This framework is compatible Linux and Windows.

The first step is the installation of the framework. Some scripts, located in **./install** directory, need to be executed. Depending on the platform, the correct directory must be selected (windows or linux). Scripts of these directories will download and install Apache Felix OSGi framework in the root directory.



```
.
├── doc
│   └── README.pdf
├── felix
│   ├── bin
│   ├── bundle
│   ├── conf
│   ├── DEPENDENCIES
│   ├── doc
│   ├── LICENSE
│   ├── LICENSE.kxml2
│   └── NOTICE
├── install
│   ├── linux
│   └── windows
├── README.txt
├── run
│   └── OSGi
└── tests
    └── bundles

13 directories, 6 files
```

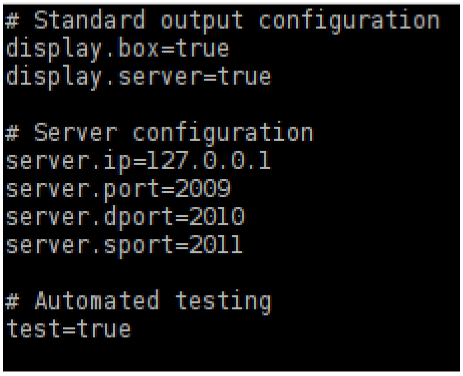
2 Test framework execution

In this part, we will detail how to use the test framework.

2.1 Configuration file and remote server

Located in the **./tests/bundles/server** directory, the configuration file **com.sogetiht.otb.properties.cfg** contains display and server parameters. Two display options are available : writing in the framework console (i.e. the terminal where the test is launched) or writing on a distant machine (i.e. the server).

In this last case, it is necessary to launch the server before executing the test. First at all, a short configuration of the server may be required :



```
# Standard output configuration
display.box=true
display.server=true

# Server configuration
server.ip=127.0.0.1
server.port=2009
server.dport=2010
server.sport=2011

# Automated testing
test=true
```

If **display.server** parameter is set at true, the server will be required to execute the tests. If the server is not launched, tests will not be executed.

To execute the server, it is essential to open another terminal, where the **runServer.sh** script is executed :

$ ./run/OSGi/linux/runServer.sh

NB : some tests require server use in all cases, like all file manipulation tests*.* If **display.server** parameter is not set at true, these tests will not be executed.

2.2 Tests execution

To execute a test, the script **runAutomatedTests.sh** contained in the **./run/linux** directory need to be launch. There are two possibilities to use this script :

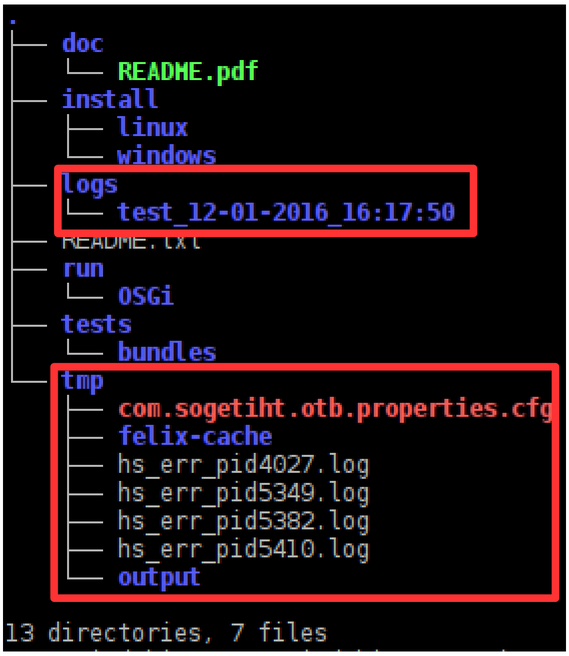
- Without option, the script will execute all the bundles contained in the **./tests/bundles** directory.

- With « -b » option, we can specify which bundles will be executed (in the example below, the launched bundle is the SYS-ATT-173).

Examples : *$ ./run/OSGi/linux/runAutomatedTests.sh*

$ ./run/OSGi/linux/runAutomatedTests.sh -b 173

As said ealier in this document, framework execution carry temporary directories creation : **./tmp** and **./logs**.



```
.
├── doc
│   └── README.pdf
├── install
│   ├── linux
│   └── windows
├── logs
│   └── test_12-01-2016_16:17:50
├── README.txt
├── run
│   └── OSGi
├── tests
│   └── bundles
└── tmp
    ├── com.sogetiht.otb.properties.cfg
    ├── felix-cache
    ├── hs_err_pid4027.log
    ├── hs_err_pid5349.log
    ├── hs_err_pid5382.log
    ├── hs_err_pid5410.log
    └── output

13 directories, 7 files
```

3 OSGi Attack cases and countermeasures

Reader will find here below a detailed list of potential cyber-attacks that are achievable on OSGi platforms, through different attack targets:

- **OS** attacks, which are performed via OSGi bundles and target the OS.
- **OSGi Framework** attacks, which are performed via OSGi bundles and target the framework.
- **OSGi Bundle** attacks, which are performed via malicious bundles, previously injected with an attack. For this part, credentials are considered as compromised in order to pass this step.

3.1 OS attacks

OS attacks group contains 9 attacks, which are divided in three types: **file/data leakage**, **code execution** and **file manipulation**. As the name suggests it, these attacks target the OS. The aim here is to hijack critical system data, manipulate system files or execute native code.

3.1.1 Files & Data leakage

SYS-ATT-100: Data leakage	
Description	These system properties can be used in order to perform attacks which exploit known vulnerabilities of a specific version of an installed software/component. This attack need a launched server.
Impact	System properties drop, thanks to System linked major methods (System.getProperties, System.getEnv) and clipboard elements/ keylogger clipboard, we are able to collect some informations about system properties.
Recommendations	Set a sandbox system for bundle execution, in order to limit access to files and/or properties of the system.

SYS-ATT-110: Log files hijacking	
Description	System log pertinent files hijacking, and sending of these files on the network to malicious distant server. This attack need a launched server.
Impact	Log files can contain some informations about system components/libraries/softwares (type, used version…). These informations can permit to perform attacks.
Recommendations	Set a sandbox system for bundle execution, in order to limit access to files and/or properties of the system.

SYS-ATT-115: Password files hijacking	
Description	Hijack of “shadow” password file, which contains hashs of passwords. This attack need a launched server.
Impact	If “passwd” file is also hijacked, it is possible to retrieve passwords by using a password cracking software tool.
Recommendations	Implement java.io.FilePermission and write permission rules.

SYS-ATT-120: Tree diagram route	
Description	Search of pertinent files (“passwd”, “shadow”) by executing a tree diagram route from distant server, followed by a data. This attack need a launched server.

Impact	This attack permits to hijack different types of sensitive data: passwords, logs…
Recommendations	Implement java.io.FilePermission and write permission rules.

3.1.2 Code execution

SYS-ATT-135: Code execution (JNA)	
Description	Use of JNA (Java Native Access) related libraries to execute native code (programs based on other languages like C, C++, assembler...) from JAVA JVM. This bundle executes a C library from JVM, which contains printf function, and prints a string without using System.out.println() JAVA API function.
Impact	This library permits to execute malicious C code from JVM, and perform several types of attacks.
Recommendations	Check Bundle.NativeCode header usage in the bundle. Potentially remotable at OSGi level via bundle permissions and native code header. Forbid native code in bundles by default (at manifest level).

SYS-ATT-136: Code execution (JNI)	
Description	Use of JNI (Java Native Interface) related libraries which permit to execute native applications (programs based on other languages like C, C++, assembler...) from JAVA JVM. This bundle executes a C program from JVM, which tries to write in a fordidden memory area (RAM). The system stops the JVM (Segmentation Fault).
Impact	This library permits to execute malicious C code from JVM, and perform several types of attacks.
Recommendations	Check Bundle.NativeCode header usage in the bundle. Potentially remotable at OSGi level via bundle permissions and native code header. Forbid native code in bundles by default (at manifest level).

3.1.3 Filesystem manipulation

SYS-ATT-140: Symbolic links creation	
Description	Creation of symbolic links pointing on critical files or folders (root, binaries, libraries).
Impact	An attacker can hijack some useful information, like credentials or installed software/component versions, and use these informations to perform an attack.
Recommendations	Filesystem permissions management (or runtime permissions): Setup a quota system.

SYS-ATT-145: System configuration manipulation	
Description	System configuration files manipulation.
Impact	Some of these files contain critical informations (network, boot, credentials, …) which can be exploited by an attacker to take control of the system.
Recommendations	Filesystem permissions management: Implement java.io.FilePermission and write permission rules.

SYS-ATT-146: System configuration manipulation (redirect std ouput)	
Description	System configuration file manipulation by redirecting system.out.println output in a file.
Impact	Some of these files contain critical informations (network, boot, credentials, …) which can be exploited by an attacker to take control of the system.

Recommendations	Filesystem permissions management: Implement java.io.FilePermission and write permission rules.

3.2 OSGi Framework attacks

OSGi attacks group is composed of 14 attacks, separated in four types: **data/information extraction badly formatted input data or files**, **Denial-of-Service** and **framework services manipulation**. Attacks of this group are more offensive than OS attacks group, as a large part of these attacks belongs to Denial-of-Service attack type. The goal here is to put the system out of service by Denial-of-Service attacks and manipulation of framework services.

3.2.1 Data/Information extraction

SYS-ATT-150: Data hijacking	
Description	Data hijacking by event use (FrameworkEvent, BundleEvent, ServiceEvent): event subscription, authorized values and typing dictionnaries hijacking.
Impact	An attacker can hijack some useful information like framework used packages and their versions.
Recommendations	Execution context privilege management: Setup a file sandboxing system at the OS level.

SYS-ATT-155: Bundle hijacking	
Description	Platform installed bundles drop (with BundleContext.bundles or BundleContext.getBundles).
Impact	It is possible to study content and code of dropped bundles. These bundles can be an entry point for an attack. This attack need a server.
Recommendations	Execution context privilege management: Setup a file sandboxing system at the OS level.

3.2.2 Denial-of-Service attacks

3.2.2.1 DoS – Malicious Bundles or Packages execution

SYS-ATT-172: DoS - Resource exhaustion: Decompression bomb	
Description	Malicious bundle creation as a small sized JAR file, but whose decompression by decompression tools such as Jar Tool provokes the creation of a huge bundle in terms of size. This provokes resource exhaustion.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Need to ensure OSGi Framework will reserve minimum RAM space to let system survive even if bundle used all application RAM. Setup a quota system and a runtime conflict resolution and monitoring system.

SYS-ATT-173: DoS - Resource exhaustion: infinite & complex operations	
Description	Resources exhaustion by infinite complex calculations (key research in an exhaustive manner, non homogeneous systems resolution, infinite loop with object creation at each loop).
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.

Recommendations	Resource Monitoring: Setup a quota system and a runtime conflict resolution and monitoring system.

3.2.2.2 DoS – Malicious service operations

SYS-ATT-174: DoS - Resource Exhaustion: too many services…	
Description	Declaration of an important number of services.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a quota system and a runtime conflict resolution and monitoring system.

SYS-ATT-175: Infinite loop: mutually dependant services	
Description	Resource exhaustion by mutually dependant service subscriptions (ex: in a bilateral scenario, bundle A subscribes at bundle B services and conversely : bundle A state change provokes event sending, processed by bundle B which change its state, which provoke a event sending... etc, as an infinite loop plan).
Impact	The system may be not accessible when the infinite loop occurs. A restart may be necessary to access again to the system.
Recommendations	Resource Monitoring: Setup a quota system and a runtime conflict resolution and monitoring system.

SYS-ATT-176: DoS - Resource Exhaustion: StackOverFlowError exception	
Description	StackOverFlowError exception provocation in a third party bundle by mutually dependant service subscriptions. Third party bundle service calls the malicious service which, instead of executes itself normally, calls the third party bundle service.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system.

SYS-ATT-177: Infinite loop: service self subscription	
Description	Resources exhaustion by service self subscription. A state change of this service produces a new state change, and so on, as an infinite loop plan.
Impact	The system may be not accessible when the infinite loop occurs. A restart may be necessary to access again to the system.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system.

SYS-ATT-178: DoS – Deadlock	
Description	Block framework execution with an infinite loop in the Bundle Activator (bundle start() or stop() methods).
Impact	The system may be not accessible when the deadlock occurs. A restart may be necessary to access again to the system.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system.

SYS-ATT-179: DoS – Deadlock	

Description	Block framework execution with a Thread hanging in the Bundle Activator (bundle start() or stop() methods).
Impact	The system may be not accessible when the deadlock occurs. A restart may be necessary to access again to the system.
Recommendations	Execution context privilege management

3.2.2.3 Framework services manipulation

SYS-ATT-170: Bundle injection	
Description	Bundle injection in the framework. This bundle hijacks a new bundle, then installs it and starts it in the framework.
Impact	The system may be compromised depending on the new bundle aim: malicious code, DoS, resource exhaustion, data hijack…
Recommendations	Execution context privilege management: Setup a quota system.

SYS-ATT-171: Services subscription propagation	
Description	Generation of several bundles which process, in a successive and interlinked manner, the subscription of services which depend themselves of underlying services, in a recursive manner. If A, B, ..., N-1, N are bundles, they subscribe respectively to services b (of bundle B), c, ..., n. This bundle uses pointlessly system resources.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a quota system and a runtime conflict resolution and monitoring system.

SYS-ATT-206: Dos - Deadlock - Infinite loop in service calls	
Description	Deadlock exploitation: creation of a bundle which lists a multitude of classical services (LogService) implemented in a certain manner which provokes a deadlock situation of the calling service. Infinite loops in furnished APIs provoke deadlock situations.
Impact	The system may be not accessible when the infinite loop occurs. A restart may be necessary to access again to the system.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system.

SYS-ATT-261: DoS - Framework recursive loading from a bundle	
Description	Framework inclusion in a bundle (embedded framework) and bundle launching in this new framework (infinite loop). This bundle creates a new instance of the framework (embedded framework) and launches itself in this new framework. This recursive execution of the framework and the bundle monopolize system resources (CPU and RAM).
Impact	The functioning of the system can be strongly altered. A restart may be necessary to restore the system. Indeed, without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Execution context privilege management + Resource monitoring: Setup a quota system.

3.3 OSGi bundle attacks

OSGi bundle attacks group, which is composed of 17 attacks, contains offensive attacks like OSGi Framework attacks group. Splited in two types (**Denial-of-Service** and **data injection**, these attacks intend to put the system out of service by Denial-of-Service attacks or data injection.

3.3.1 Denial-of-Service

SYS-ATT-185: DoS – Infinite loop (thread)	
Description	DoS Attack - Resource overload (thread) Resources exhaustion by infinite loop implementation in a thread.
Impact	The system may be not accessible when the deadlock occurs. A restart may be necessary to access again to the system.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. The OSGi framework should run all event handler methods in a separate thread, and then wait on that thread with a finite time out. If the time out elapses then the bundle should be rejected (stopped, uninstalled, etc.). Event handlers are not defined only by the activator. Any class of the bundle can register a handler (service listener, bundle listener, etc.). CPU usage limits must be enforced by the JVM from a Feature.

SYS-ATT-186: DoS – Infinite loop (bundle method)	
Description	DoS Attack - Resource overload (bundle method) Resources exhaustion by infinite loop implementation in a method of the bundle.
Impact	The system may be not accessible when the deadlock occurs. A restart may be necessary to access again to the system.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. The OSGi framework should run all event handler methods in a separate thread, and then wait on that thread with a finite time out. If the time out elapses then the bundle should be rejected (stopped, uninstalled, etc.). Event handlers are not defined only by the activator. Any class of the bundle can register a handler (service listener, bundle listener, etc.). CPU usage limits must be enforced by the JVM from a Feature.

SYS-ATT-187: DoS - Resource overload (too many bundles)	
Description	Resources exhaustion by launching of an important number of bundles on the framework.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a quota system.

SYS-ATT-190: DoS - Resource exhaustion - Zombie data	
Description	Creation of bulky temporary files in neutral public area (and common for all bundles of the framework) which is never cleaned.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Filesystem permissions management + Resource Monitoring: Setup a quota system.

	Setup a file sandboxing system at the OS level. Set bundle created files in a given directory and delete this directory when the bundle is uninstalled. The OSGi framework should remove all bundle data when it is uninstalled.

SYS-ATT-195: DoS - Resource exhaustion (GarbageCollector)

Description	Deny of service obtained by excessive use of GarbageCollector (System.gc). Unlike C/C++, memory used by bundles is managed by the JVM (more particularly the Garbage Collector). This bundle will drain system resources by allocating memory and by calling Garbage Collector in order to free it.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system.

SYS-ATT-199: DoS – Allocation (RAM Exhaustion – Important allocation in Activator Class)

Description	Important tab allocation in Activator class attributes. This bundle installation allocates all JVM RAM and, consequently, the framework raises OutOfMemoryError exception and finishes its execution.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Setup a quota system.

SYS-ATT-200: DoS – Allocation (object cascade)

Description	Bulky but empty object cascade declaration (which forces useless use of Garbage Collector), Loading and allocation of useless resources from the network (or generation of resources in RAM memory). RAM system is monopolized by these bulky allocations. Launching new processes is not possible anymore (error: "can't fork").
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Setup a quota system.

SYS-ATT-202: DoS – Allocation (Multiple objects)

Description	Creation of a multitude of objects (RAM). JVM memory is monopolized (OutOfMemoryError). That affects other working bundle behaviors.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Setup a quota system.

SYS-ATT-203: DoS – Memory allocation through native code execution

Description	Native code execution: allocation of the entire RAM. This bundle executes from the JVM a C

	program, which reserves memory in RAM by using malloc() function.
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Setup a quota system.

SYS-ATT-205: DoS - Deadlock - Semaphore/Mutex lock

Description	Deadlock exploitation: shared pertinent resources monopolization (internal local storage (files), RAM storage (objects)). A "lock" is made on a shared file. A bundle which wants to use it will infinitely wait.
Impact	The system may be not accessible when the deadlock occurs. A restart may be necessary to access again to the system.
Recommendations	Filesystem permissions management + Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Implement java.io.FilePermission and write permission rules.

SYS-ATT-210: DoS – Race condition

Description	Race condition exploitation: periodic modification (short periods) of pertinent shared data. File modification by a malicious bundle during a legitimate bundle accesses to this file.
Impact	The behavior of the bundle and, by extension, of the system can be unpredictable (unavailability, security flaw…).
Recommendations	Filesystem permissions management: Implement java.io.FilePermission and write permission rules.

SYS-ATT-220: DoS – Allocation (thread multiplication)

Description	Simultaneous generation and execution of a thread multitude.
Impact	This attack causes resource exhaustion, and without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Setup a quota system.

SYS-ATT-221: Resource exhaustion - Sleeping bundle

Description	Memory resources exhaustion with sleeping multithreaded bundle (generation of a huge number of sleeping threads).
Impact	Without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Resource Monitoring: Setup a runtime conflict resolution and monitoring system. Setup a quota system.

SYS-ATT-225: DoS - Bundle states modification (Bundle Context)

Description	Bundle state modification by Bundle Context use (start()/stop()/uninstall() methods). Some of enabled bundles are stopped and uninstalled.
Impact	The functioning of the system can be strongly altered. A restart may be necessary to restore the

	system.
Recommendations	Execution context privilege management: Setup a file sandboxing system at the OS level.

SYS-ATT-226: DoS - Bundle state alteration (calling start() in stop() method)	
Description	Service deny obtained by call of start() method in the bundle stop() method.
Impact	The functioning of the system can be strongly altered. A restart may be necessary to restore the system.
Recommendations	Execution context privilege management: Setup a file sandboxing system at the OS level.

SYS-ATT-261: DoS - Framework recursive loading	
Description	Framework inclusion in a bundle (embedded framework) and bundle launching in this new framework (infinite loop). This bundle creates a new instance of the framework (embedded framework) and launches itself in this new framework. This recursive execution of the framework and the bundle monopolize system resources (CPU and RAM).
Impact	The functioning of the system can be strongly altered. A restart may be necessary to restore the system. Indeed, without resources (RAM, CPU or disk), the system may be not accessible, which is problematic.
Recommendations	Execution context privilege management + Resource monitoring: Setup a quota system.

3.3.2 Data injection

SYS-ATT-230: Random inputs to services	
Description	This bundle injects random data in services. If service API did not anticipate this data type, the bundle which is implementing the service can crash. This bundle can permit to identity APIs vulnerabilities.
Impact	The functioning of the system can be strongly altered. A restart may be necessary to restore the system.
Recommendations	Secure coding, input validation / error handling.

4 Authors

For any suggestion or support, please contact :

Julien Helmer
Cyber-security Technical leader
julien.helmer@sogeti.com

Romain Siche
Cyber-security engineer
romain.siche@sogeti.com

Thibault Mainand
Cyber-security apprentice
thibault.mainand@sogeti.com