Fortify Standalone Report Generator

Developer Workbook

akka-cluster-metrics

Table of Contents

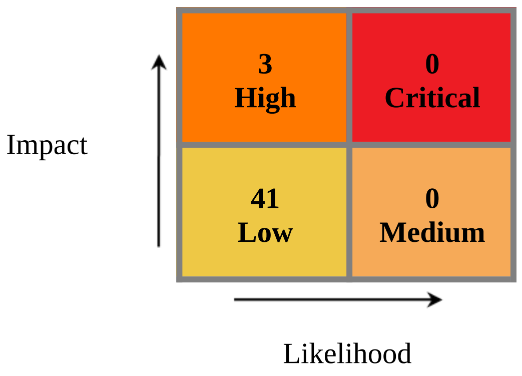
Executive Summary

This workbook is intended to provide all necessary details and information for a developer to understand and remediate the different issues discovered during the akka-cluster-metrics project audit. The information contained in this workbook is targeted at project managers and developers.

This section provides an overview of the issues uncovered during analysis.

Project Name:	akka-cluster-metrics
Project Version:	
SCA:	Results Present
WebInspect:	Results Not Present
WebInspect Agent:	Results Not Present
Other:	Results Not Present

Issues by Priority



Top Ten Critical Categories

This project does not contain any critical issues

Project Description

This section provides an overview of the Fortify scan engines used for this project, as well as the project meta-information.

SCA

Date of Last Analysis:	Jun 16, 2022, 11:20 AM	**Engine Version:**	21.1.1.0009
Host Name:	Jacks-Work-MBP.local	**Certification:**	VALID
Number of Files:	10	**Lines of Code:**	730

Rulepack Name	Rulepack Version
Fortify Secure Coding Rules, Extended, Java	2022.1.0.0007
Fortify Secure Coding Rules, Core, Scala	2022.1.0.0007
Fortify Secure Coding Rules, Extended, JSP	2022.1.0.0007
Fortify Secure Coding Rules, Core, Android	2022.1.0.0007
Fortify Secure Coding Rules, Extended, Content	2022.1.0.0007
Fortify Secure Coding Rules, Extended, Configuration	2022.1.0.0007
Fortify Secure Coding Rules, Core, Annotations	2022.1.0.0007
Fortify Secure Coding Rules, Community, Cloud	2022.1.0.0007
Fortify Secure Coding Rules, Core, Universal	2022.1.0.0007
Fortify Secure Coding Rules, Core, Java	2022.1.0.0007
Fortify Secure Coding Rules, Community, Universal	2022.1.0.0007

Issue Breakdown by Fortify Categories

The following table depicts a summary of all issues grouped vertically by Fortify Category. For each category, the total number of issues is shown by Fortify Priority Order, including information about the number of audited issues.

Category	Fortify Priority (audited/total)				Total Issues
	Critical	High	Medium	Low	
Code Correctness: Constructor Invokes Overridable Function	0	0	0	0 / 35	0 / 35
Code Correctness: Erroneous String Compare	0	0	0	0 / 4	0 / 4
Insecure Randomness	0	0 / 3	0	0	0 / 3
Poor Error Handling: Overly Broad Catch	0	0	0	0 / 2	0 / 2

Results Outline

Code Correctness: Constructor Invokes Overridable Function (35 issues)

Abstract

A constructor of the class calls a function that can be overridden.

Explanation

When a constructor calls an overridable function, it may allow an attacker to access the `this` reference prior to the object being fully initialized, which can in turn lead to a vulnerability. **Example 1:** The following calls a method that can be overridden.

```
...
class User {
  private String username;
  private boolean valid;
  public User(String username, String password){
    this.username = username;
    this.valid = validateUser(username, password);
  }
  public boolean validateUser(String username, String password){
    //validate user is real and can authenticate
    ...
  }
  public final boolean isValid(){
    return valid;
  }
}
```

Since the function `validateUser` and the class are not `final`, it means that they can be overridden, and then initializing a variable to the subclass that overrides this function would allow bypassing of the `validateUser` functionality. For example:

```
...
class Attacker extends User{
  public Attacker(String username, String password){
    super(username, password);
  }
  public boolean validateUser(String username, String password){
    return true;
  }
}
...
class MainClass{
  public static void main(String[] args){
    User hacker = new Attacker("Evil", "Hacker");
    if (hacker.isValid()){
      System.out.println("Attack successful!");
    }else{
      System.out.println("Attack failed");
    }
  }
}
```

The code in `Example 1` prints "Attack successful!", since the `Attacker` class overrides the `validateUser()` function that is called from the constructor of the superclass `User`, and Java will first look in the subclass for functions called from the constructor.

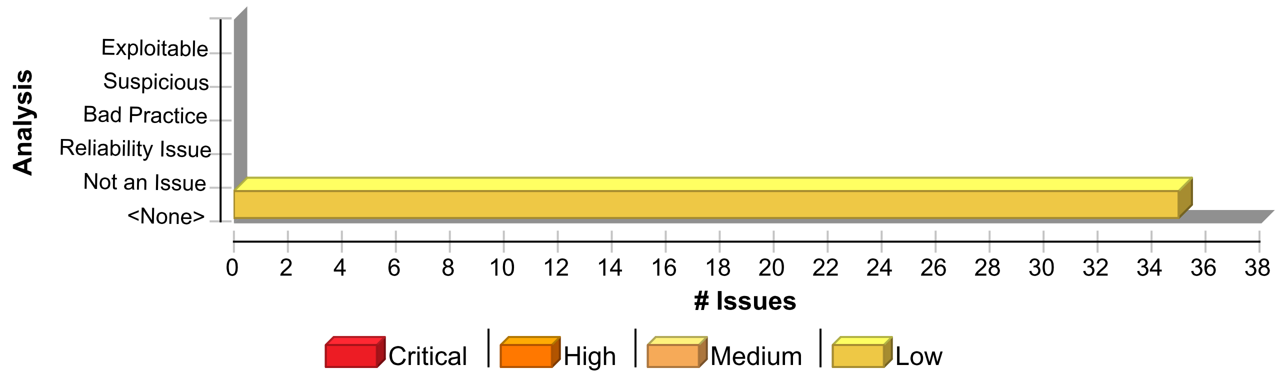
Recommendation

Constructors should not call functions that can be overridden, either by specifying them as `final`, or specifying the class as `final`. Alternatively if this code is only ever needed in the constructor, the `private` access specifier can be used, or the logic could be placed directly into the constructor of the superclass. **Example 2:** The following makes the class `final` to prevent the function from being overridden elsewhere.

```
  ...
  final class User {
    private String username;
    private boolean valid;
    public User(String username, String password){
      this.username = username;
      this.valid = validateUser(username, password);
    }
    private boolean validateUser(String username, String password){
      //validate user is real and can authenticate
      ...
    }
    public final boolean isValid(){
      return valid;
    }
  }
```

This example specifies the class as `final`, so that it cannot be subclassed, and changes the `validateUser()` function to `private`, since it is not needed elsewhere in this application. This is programming defensively, since at a later date it may be decided that the `User` class needs to be subclassed, which would result in this vulnerability reappearing if the `validateUser()` function was not set to `private`.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Constructor Invokes Overridable Function	35	0	0	35
Total	**35**	**0**	**0**	**35**

Code Correctness: Constructor Invokes Overridable Function	**Low**
Package: akka.cluster.metrics	
MetricsCollector.scala, line 207 (Code Correctness: Constructor Invokes Overridable Function)	**Low**

Issue Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

MetricsCollector.scala, line 207 (Code Correctness: Constructor Invokes Overridable Function) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metrics
Enclosing Method: SigarMetricsCollector()
File: MetricsCollector.scala:207
Taint Flags:

```
/**
* Verify at the end of construction that Sigar is operational.
*/
metrics()

// Construction complete.

```

ClusterMetricsCollector.scala, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cluster
Enclosing Method: ClusterMetricsCollector()
File: ClusterMetricsCollector.scala:165
Taint Flags:

```
/**
* Start periodic gossip to random nodes in cluster
*/
val gossipTask =
scheduler.scheduleWithFixedDelay(
PeriodicTasksInitialDelay max CollectorGossipInterval,
CollectorGossipInterval,
```

ClusterMetricsCollector.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsCollector.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: cluster
Enclosing Method: ClusterMetricsCollector()
File: ClusterMetricsCollector.scala:175
Taint Flags:

```
172 /**
173 * Start periodic metrics collection
174 */
175 val sampleTask = scheduler.scheduleWithFixedDelay(
176 PeriodicTasksInitialDelay max CollectorSampleInterval,
177 CollectorSampleInterval,
178 self,
```

ClusterMetricsCollector.scala, line 58 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metrics
Enclosing Method: ClusterMetricsSupervisor()
File: ClusterMetricsCollector.scala:58
Taint Flags:

```
55 import context._
56 import metrics.settings._
57
58 override val supervisorStrategy = metrics.strategy
59
60 var collectorInstance = 0
61
```

ClusterMetricsRouting.scala, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$5
Enclosing Method: AdaptiveLoadBalancingPool()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: ClusterMetricsRouting.scala:150
Taint Flags:

```
147 * on remaining capacity as indicated by the node metrics
148 * @param nr initial number of routees in the pool
149 */
150 def this(metricsSelector: MetricsSelector, nr: Int) = this(nrOfInstances = nr)
151
152 override def resizer: Option[Resizer] = None
153
```

ClusterMetricsRouting.scala, line 139 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$4
Enclosing Method: AdaptiveLoadBalancingPool()
File: ClusterMetricsRouting.scala:139
Taint Flags:

```
136 extends Pool {
137
138 def this(config: Config, dynamicAccess: DynamicAccess) =
139 this(
140 nrOfInstances = ClusterRouterSettingsBase.getMaxTotalNrOfInstances(config),
141 metricsSelector = MetricsSelector.fromConfig(config, dynamicAccess),
142 usePoolDispatcher = config.hasPath("pool-dispatcher"))
```

ClusterMetricsRouting.scala, line 222 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$3
Enclosing Method: AdaptiveLoadBalancingGroup()
File: ClusterMetricsRouting.scala:222
Taint Flags:

```
219 extends Group {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 222 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
220
221 def this(config: Config, dynamicAccess: DynamicAccess) =
222 this(
223 metricsSelector = MetricsSelector.fromConfig(config, dynamicAccess),
224 paths = immutableSeq(config.getStringList("routees.paths")))
225
```

ClusterMetricsRouting.scala, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$4
Enclosing Method: AdaptiveLoadBalancingPool()
File: ClusterMetricsRouting.scala:150
Taint Flags:

```
147 * on remaining capacity as indicated by the node metrics
148 * @param nr initial number of routees in the pool
149 */
150 def this(metricsSelector: MetricsSelector, nr: Int) = this(nrOfInstances = nr)
151
152 override def resizer: Option[Resizer] = None
153
```

ClusterMetricsExtension.scala, line 50 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: ClusterMetricsExtension()
File: ClusterMetricsExtension.scala:50
Taint Flags:

```
47 *
48 * Supervision strategy.
49 */
50 private[metrics] val strategy = system.dynamicAccess
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsExtension.scala, line 50 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
.createInstanceFor[SupervisorStrategy](
SupervisorStrategyProvider,
immutable.Seq(classOf[Config] -> SupervisorStrategyConfiguration))
```

ClusterMetricsExtension.scala, line 53 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: ClusterMetricsExtension()
File: ClusterMetricsExtension.scala:53
Taint Flags:

```
private[metrics] val strategy = system.dynamicAccess
.createInstanceFor[SupervisorStrategy](
SupervisorStrategyProvider,
immutable.Seq(classOf[Config] -> SupervisorStrategyConfiguration))
.getOrElse {
val log: LoggingAdapter = Logging(system, classOf[ClusterMetricsExtension])
log.error(s"Configured strategy provider ${SupervisorStrategyProvider} failed to load, using default ${classOf[
```

ClusterMetricsExtension.scala, line 65 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: ClusterMetricsExtension()
File: ClusterMetricsExtension.scala:65
Taint Flags:

```
* Supervisor actor.
* Accepts subtypes of [[CollectionControlMessage]]s to manage metrics collection at runtime.
*/
val supervisor = system.systemActorOf(
Props(classOf[ClusterMetricsSupervisor]).withDispatcher(MetricsDispatcher).withDeploy(Deploy.local),
SupervisorName)

```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsExtension.scala, line 65 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: ClusterMetricsExtension()
File: ClusterMetricsExtension.scala:65
Taint Flags:

```
62 * Supervisor actor.
63 * Accepts subtypes of [[CollectionControlMessage]]s to manage metrics collection at runtime.
64 */
65 val supervisor = system.systemActorOf(
66 Props(classOf[ClusterMetricsSupervisor]).withDispatcher(MetricsDispatcher).withDeploy(Deploy.local),
67 SupervisorName)
68
```

ClusterMetricsRouting.scala, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$3
Enclosing Method: AdaptiveLoadBalancingPool()
File: ClusterMetricsRouting.scala:150
Taint Flags:

```
147 * on remaining capacity as indicated by the node metrics
148 * @param nr initial number of routees in the pool
149 */
150 def this(metricsSelector: MetricsSelector, nr: Int) = this(nrOfInstances = nr)
151
152 override def resizer: Option[Resizer] = None
153
```

ClusterMetricsRouting.scala, line 139 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 139 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: $default$3
Enclosing Method: AdaptiveLoadBalancingPool()
File: ClusterMetricsRouting.scala:139
Taint Flags:

```
136 extends Pool {
137
138 def this(config: Config, dynamicAccess: DynamicAccess) =
139 this(
140 nrOfInstances = ClusterRouterSettingsBase.getMaxTotalNrOfInstances(config),
141 metricsSelector = MetricsSelector.fromConfig(config, dynamicAccess),
142 usePoolDispatcher = config.hasPath("pool-dispatcher"))
```

ClusterMetricsCollector.scala, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metrics
Enclosing Method: ClusterMetricsCollector()
File: ClusterMetricsCollector.scala:165
Taint Flags:

```
162 /**
163 * Start periodic gossip to random nodes in cluster
164 */
165 val gossipTask =
166 scheduler.scheduleWithFixedDelay(
167 PeriodicTasksInitialDelay max CollectorGossipInterval,
168 CollectorGossipInterval,
```

ClusterMetricsCollector.scala, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metrics
Enclosing Method: ClusterMetricsCollector()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsCollector.scala, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: ClusterMetricsCollector.scala:165
Taint Flags:

```
162 /**
163 * Start periodic gossip to random nodes in cluster
164 */
165 val gossipTask =
166 scheduler.scheduleWithFixedDelay(
167 PeriodicTasksInitialDelay max CollectorGossipInterval,
168 CollectorGossipInterval,
```

ClusterMetricsCollector.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metrics
Enclosing Method: ClusterMetricsCollector()
File: ClusterMetricsCollector.scala:175
Taint Flags:

```
172 /**
173 * Start periodic metrics collection
174 */
175 val sampleTask = scheduler.scheduleWithFixedDelay(
176 PeriodicTasksInitialDelay max CollectorSampleInterval,
177 CollectorSampleInterval,
178 self,
```

ClusterMetricsCollector.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metrics
Enclosing Method: ClusterMetricsCollector()
File: ClusterMetricsCollector.scala:175
Taint Flags:

```
172 /**
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsCollector.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
173 * Start periodic metrics collection
174 */
175 val sampleTask = scheduler.scheduleWithFixedDelay(
176 PeriodicTasksInitialDelay max CollectorSampleInterval,
177 CollectorSampleInterval,
178 self,
```

ClusterMetricsRouting.scala, line 234 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$3
Enclosing Method: AdaptiveLoadBalancingGroup()
File: ClusterMetricsRouting.scala:234
Taint Flags:

```
231 * sent with [[akka.actor.ActorSelection]] to these paths
232 */
233 def this(metricsSelector: MetricsSelector, routeesPaths: java.lang.Iterable[String]) =
234 this(paths = immutableSeq(routeesPaths))
235
236 override def paths(system: ActorSystem): immutable.Iterable[String] = this.paths
237
```

ClusterMetricsRouting.scala, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$1
Enclosing Method: AdaptiveLoadBalancingPool()
File: ClusterMetricsRouting.scala:150
Taint Flags:

```
147 * on remaining capacity as indicated by the node metrics
148 * @param nr initial number of routees in the pool
149 */
150 def this(metricsSelector: MetricsSelector, nr: Int) = this(nrOfInstances = nr)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 150 (Code Correctness: Constructor Invokes Overridable Function) — Low

```

override def resizer: Option[Resizer] = None

```

ClusterMetricsSettings.scala, line 23 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:23
Taint Flags:

```
private val cc = config.getConfig("akka.cluster.metrics")

// Extension.
val MetricsDispatcher: String = cc.getString("dispatcher")
val PeriodicTasksInitialDelay: FiniteDuration = cc.getMillisDuration("periodic-tasks-initial-delay")
val NativeLibraryExtractFolder: String = cc.getString("native-library-extract-folder")

```

ClusterMetricsSettings.scala, line 24 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:24
Taint Flags:

```

// Extension.
val MetricsDispatcher: String = cc.getString("dispatcher")
val PeriodicTasksInitialDelay: FiniteDuration = cc.getMillisDuration("periodic-tasks-initial-delay")
val NativeLibraryExtractFolder: String = cc.getString("native-library-extract-folder")

// Supervisor.
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsSettings.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:25
Taint Flags:

```
22  // Extension.
23  val MetricsDispatcher: String = cc.getString("dispatcher")
24  val PeriodicTasksInitialDelay: FiniteDuration = cc.getMillisDuration("periodic-tasks-initial-delay")
25  val NativeLibraryExtractFolder: String = cc.getString("native-library-extract-folder")
26
27  // Supervisor.
28  val SupervisorName: String = cc.getString("supervisor.name")
```

ClusterMetricsSettings.scala, line 28 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:28
Taint Flags:

```
25  val NativeLibraryExtractFolder: String = cc.getString("native-library-extract-folder")
26
27  // Supervisor.
28  val SupervisorName: String = cc.getString("supervisor.name")
29  val SupervisorStrategyProvider: String = cc.getString("supervisor.strategy.provider")
30  val SupervisorStrategyConfiguration: Config = cc.getConfig("supervisor.strategy.configuration")
31
```

ClusterMetricsSettings.scala, line 29 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsSettings.scala, line 29 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:29
Taint Flags:

```
26
27 // Supervisor.
28 val SupervisorName: String = cc.getString("supervisor.name")
29 val SupervisorStrategyProvider: String = cc.getString("supervisor.strategy.provider")
30 val SupervisorStrategyConfiguration: Config = cc.getConfig("supervisor.strategy.configuration")
31
32 // Collector.
```

ClusterMetricsSettings.scala, line 30 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:30
Taint Flags:

```
27 // Supervisor.
28 val SupervisorName: String = cc.getString("supervisor.name")
29 val SupervisorStrategyProvider: String = cc.getString("supervisor.strategy.provider")
30 val SupervisorStrategyConfiguration: Config = cc.getConfig("supervisor.strategy.configuration")
31
32 // Collector.
33 val CollectorEnabled: Boolean = cc.getBoolean("collector.enabled")
```

ClusterMetricsSettings.scala, line 33 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsSettings.scala, line 33 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: ClusterMetricsSettings.scala:33
Taint Flags:

```
30  val SupervisorStrategyConfiguration: Config = cc.getConfig("supervisor.strategy.configuration")
31
32  // Collector.
33  val CollectorEnabled: Boolean = cc.getBoolean("collector.enabled")
34  val CollectorProvider: String = cc.getString("collector.provider")
35  val CollectorFallback: Boolean = cc.getBoolean("collector.fallback")
36  val CollectorSampleInterval: FiniteDuration = {
```

ClusterMetricsSettings.scala, line 34 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:34
Taint Flags:

```
31
32  // Collector.
33  val CollectorEnabled: Boolean = cc.getBoolean("collector.enabled")
34  val CollectorProvider: String = cc.getString("collector.provider")
35  val CollectorFallback: Boolean = cc.getBoolean("collector.fallback")
36  val CollectorSampleInterval: FiniteDuration = {
37  cc.getMillisDuration("collector.sample-interval")
```

ClusterMetricsSettings.scala, line 35 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:35
Taint Flags:

```
32  // Collector.
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsSettings.scala, line 35 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
33 val CollectorEnabled: Boolean = cc.getBoolean("collector.enabled")
34 val CollectorProvider: String = cc.getString("collector.provider")
35 val CollectorFallback: Boolean = cc.getBoolean("collector.fallback")
36 val CollectorSampleInterval: FiniteDuration = {
37 cc.getMillisDuration("collector.sample-interval")
38 }.requiring(_ > Duration.Zero, "collector.sample-interval must be > 0")
```

ClusterMetricsSettings.scala, line 36 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:36
Taint Flags:

```
33 val CollectorEnabled: Boolean = cc.getBoolean("collector.enabled")
34 val CollectorProvider: String = cc.getString("collector.provider")
35 val CollectorFallback: Boolean = cc.getBoolean("collector.fallback")
36 val CollectorSampleInterval: FiniteDuration = {
37 cc.getMillisDuration("collector.sample-interval")
38 }.requiring(_ > Duration.Zero, "collector.sample-interval must be > 0")
39 val CollectorGossipInterval: FiniteDuration = {
```

ClusterMetricsSettings.scala, line 39 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:39
Taint Flags:

```
36 val CollectorSampleInterval: FiniteDuration = {
37 cc.getMillisDuration("collector.sample-interval")
38 }.requiring(_ > Duration.Zero, "collector.sample-interval must be > 0")
39 val CollectorGossipInterval: FiniteDuration = {
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.cluster.metrics

ClusterMetricsSettings.scala, line 39 (Code Correctness: Constructor Invokes Overridable Function)	Low

```
cc.getMillisDuration("collector.gossip-interval")
}.requiring(_ > Duration.Zero, "collector.gossip-interval must be > 0")
val CollectorMovingAverageHalfLife: FiniteDuration = {
```

ClusterMetricsSettings.scala, line 42 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cc
Enclosing Method: ClusterMetricsSettings()
File: ClusterMetricsSettings.scala:42
Taint Flags:

```
val CollectorGossipInterval: FiniteDuration = {
cc.getMillisDuration("collector.gossip-interval")
}.requiring(_ > Duration.Zero, "collector.gossip-interval must be > 0")
val CollectorMovingAverageHalfLife: FiniteDuration = {
cc.getMillisDuration("collector.moving-average-half-life")
}.requiring(_ > Duration.Zero, "collector.moving-average-half-life must be > 0")

```

ClusterMetricsRouting.scala, line 234 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: $default$1
Enclosing Method: AdaptiveLoadBalancingGroup()
File: ClusterMetricsRouting.scala:234
Taint Flags:

```
* sent with [[akka.actor.ActorSelection]] to these paths
*/
def this(metricsSelector: MetricsSelector, routeesPaths: java.lang.Iterable[String]) =
this(paths = immutableSeq(routeesPaths))

override def paths(system: ActorSystem): immutable.Iterable[String] = this.paths

```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.cluster.metrics

ClusterMetricsStrategy.scala, line 17 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: metricsDecider
Enclosing Method: ClusterMetricsStrategy()
File: ClusterMetricsStrategy.scala:17
Taint Flags:

```
14  * A configurable [[akka.actor.OneForOneStrategy]] with restart-on-throwable decider.
15  */
16 class ClusterMetricsStrategy(config: Config)
17  extends OneForOneStrategy(
18  maxNrOfRetries = config.getInt("maxNrOfRetries"),
19  withinTimeRange = config.getMillisDuration("withinTimeRange"),
20  loggingEnabled = config.getBoolean("loggingEnabled"))(ClusterMetricsStrategy.metricsDecider)
```

ClusterMetricsRouting.scala, line 307 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: factor
Enclosing Method: CpuMetricsSelector()
File: ClusterMetricsRouting.scala:307
Taint Flags:

```
304  // TODO read factor from reference.conf
305  /** How much extra weight to give to the stolen time. */
306  val factor = 0.3
307  require(0.0 <= factor, s"factor must be non negative: ${factor}")
308
309  override def capacity(nodeMetrics: Set[NodeMetrics]): Map[Address, Double] = {
310  nodeMetrics.collect {
```

Code Correctness: Erroneous String Compare (4 issues)

Abstract

Strings should be compared with the `equals()` method, not `==` or `!=`.

Explanation

This program uses `==` or `!=` to compare two strings for equality, which compares two objects for equality, not their values. Chances are good that the two references will never be equal. **Example 1:** The following branch will never be taken.

```
  if (args[0] == STRING_CONSTANT) {
      logger.info("miracle");
  }
```

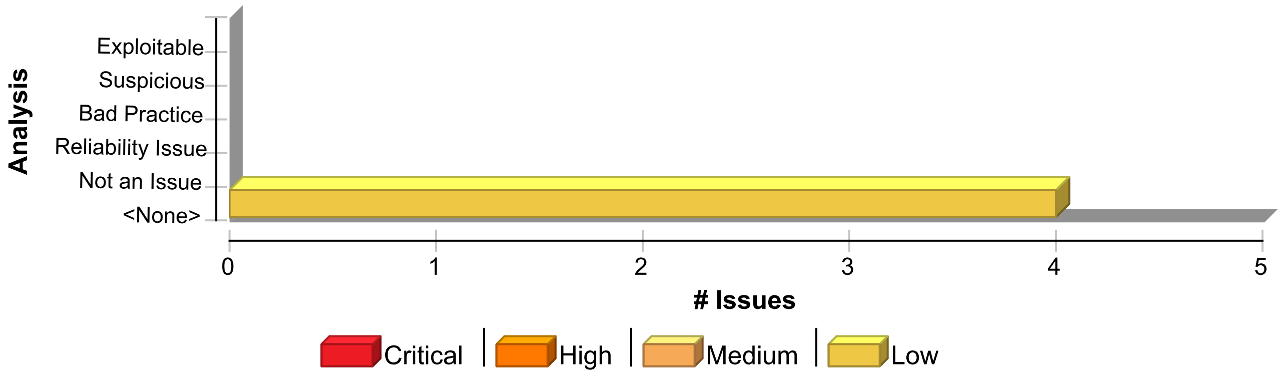
The `==` and `!=` operators will only behave as expected when they are used to compare strings contained in objects that are equal. The most common way for this to occur is for the strings to be interned, whereby the strings are added to a pool of objects maintained by the `String` class. Once a string is interned, all uses of that string will use the same object and equality operators will behave as expected. All string literals and string-valued constants are interned automatically. Other strings can be interned manually be calling `String.intern()`, which will return a canonical instance of the current string, creating one if necessary.

Recommendation

Use `equals()` to compare strings. **Example 2:** The code in `Example 1` could be rewritten in the following way:

```
  if (STRING_CONSTANT.equals(args[0])) {
      logger.info("could happen");
  }
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Erroneous String Compare	4	0	0	4
Total	**4**	**0**	**0**	**4**

Code Correctness: Erroneous String Compare | Low

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 399 (Code Correctness: Erroneous String Compare) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: fromConfig()
File: ClusterMetricsRouting.scala:399
Taint Flags:

```
396
397 object MetricsSelector {
398  def fromConfig(config: Config, dynamicAccess: DynamicAccess) =
399  config.getString("metrics-selector") match {
400  case "mix" => MixMetricsSelector
401  case "heap" => HeapMetricsSelector
402  case "cpu" => CpuMetricsSelector
```

ClusterMetricsRouting.scala, line 399 (Code Correctness: Erroneous String Compare) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: fromConfig()
File: ClusterMetricsRouting.scala:399
Taint Flags:

```
396
397 object MetricsSelector {
398  def fromConfig(config: Config, dynamicAccess: DynamicAccess) =
399  config.getString("metrics-selector") match {
400  case "mix" => MixMetricsSelector
401  case "heap" => HeapMetricsSelector
402  case "cpu" => CpuMetricsSelector
```

ClusterMetricsRouting.scala, line 399 (Code Correctness: Erroneous String Compare) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Erroneous String Compare — Low

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 399 (Code Correctness: Erroneous String Compare) — Low

Sink: Operation
Enclosing Method: fromConfig()
File: ClusterMetricsRouting.scala:399
Taint Flags:

```
396
397 object MetricsSelector {
398  def fromConfig(config: Config, dynamicAccess: DynamicAccess) =
399  config.getString("metrics-selector") match {
400  case "mix" => MixMetricsSelector
401  case "heap" => HeapMetricsSelector
402  case "cpu" => CpuMetricsSelector
```

ClusterMetricsRouting.scala, line 399 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: fromConfig()
File: ClusterMetricsRouting.scala:399
Taint Flags:

```
396
397 object MetricsSelector {
398  def fromConfig(config: Config, dynamicAccess: DynamicAccess) =
399  config.getString("metrics-selector") match {
400  case "mix" => MixMetricsSelector
401  case "heap" => HeapMetricsSelector
402  case "cpu" => CpuMetricsSelector
```

Insecure Randomness (3 issues)

Abstract

Standard pseudorandom number generators cannot withstand cryptographic attacks.

Explanation

Insecure randomness errors occur when a function that can produce predictable values is used as a source of randomness in a security-sensitive context. Computers are deterministic machines, and as such are unable to produce true randomness. Pseudorandom Number Generators (PRNGs) approximate randomness algorithmically, starting with a seed from which subsequent values are calculated. There are two types of PRNGs: statistical and cryptographic. Statistical PRNGs provide useful statistical properties, but their output is highly predictable and form an easy to reproduce numeric stream that is unsuitable for use in cases where security depends on generated values being unpredictable. Cryptographic PRNGs address this problem by generating output that is more difficult to predict. For a value to be cryptographically secure, it must be impossible or highly improbable for an attacker to distinguish between the generated random value and a truly random value. In general, if a PRNG algorithm is not advertised as being cryptographically secure, then it is probably a statistical PRNG and should not be used in security-sensitive contexts, where its use can lead to serious vulnerabilities such as easy-to-guess temporary passwords, predictable cryptographic keys, session hijacking, and DNS spoofing. **Example:** The following code uses a statistical PRNG to create a URL for a receipt that remains active for some period of time after a purchase.

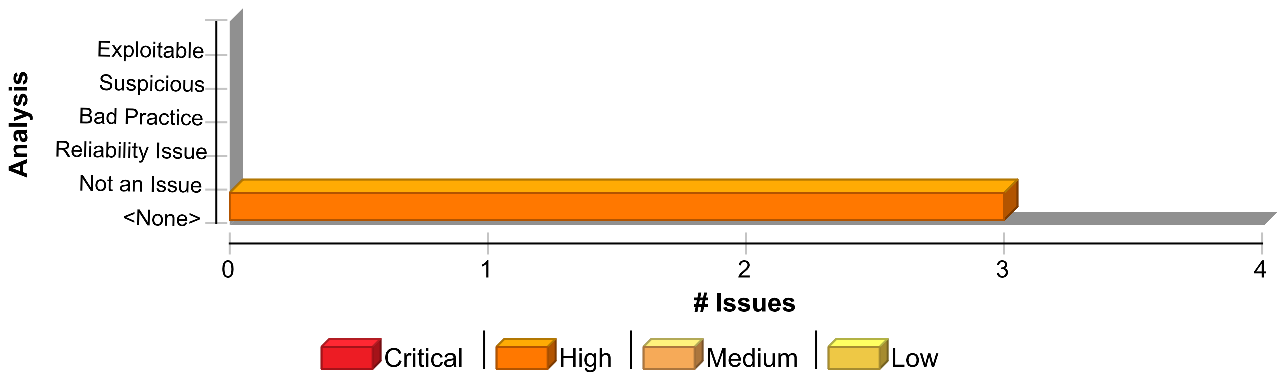
```
String GenerateReceiptURL(String baseUrl) {
    Random ranGen = new Random();
    ranGen.setSeed((new Date()).getTime());
    return (baseUrl + ranGen.nextInt(400000000) + ".html");
}
```

This code uses the `Random.nextInt()` function to generate "unique" identifiers for the receipt pages it generates. Since `Random.nextInt()` is a statistical PRNG, it is easy for an attacker to guess the strings it generates. Although the underlying design of the receipt system is also faulty, it would be more secure if it used a random number generator that did not produce predictable receipt identifiers, such as a cryptographic PRNG.

Recommendation

When unpredictability is critical, as is the case with most security-sensitive uses of randomness, use a cryptographic PRNG. Regardless of the PRNG you choose, always use a value with sufficient entropy to seed the algorithm. (Do not use values such as the current time because it offers only negligible entropy.) The Java language provides a cryptographic PRNG in `java.security.SecureRandom`. As is the case with other algorithm-based classes in `java.security`, `SecureRandom` provides an implementation-independent wrapper around a particular set of algorithms. When you request an instance of a `SecureRandom` object using `SecureRandom.getInstance()`, you can request a specific implementation of the algorithm. If the algorithm is available, then it is given as a `SecureRandom` object. If it is unavailable or if you do not specify a particular implementation, then you are given a `SecureRandom` implementation selected by the system. Sun provides a single `SecureRandom` implementation with the Java distribution named `SHA1PRNG`, which Sun describes as computing: "The SHA-1 hash over a true-random seed value concatenated with a 64-bit counter which is incremented by 1 for each operation. From the 160-bit SHA-1 output, only 64 bits are used [1]." However, the specifics of the Sun implementation of the `SHA1PRNG` algorithm are poorly documented, and it is unclear what sources of entropy the implementation uses and therefore what amount of true randomness exists in its output. Although there is speculation on the Web about the Sun implementation, there is no evidence to contradict the claim that the algorithm is cryptographically strong and can be used safely in security-sensitive contexts.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Insecure Randomness	3	0	0	3
Total	**3**	**0**	**0**	**3**

Insecure Randomness — High

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 86 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: select()
File: ClusterMetricsRouting.scala:86
Taint Flags:

```
83 updateWeightedRoutees() match {
84 case Some(weighted) =>
85 if (weighted.isEmpty) NoRoutee
86 else weighted(ThreadLocalRandom.current.nextInt(weighted.total) + 1)
87 case None =>
88 routees(ThreadLocalRandom.current.nextInt(routees.size))
89 }
```

ClusterMetricsRouting.scala, line 88 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: select()
File: ClusterMetricsRouting.scala:88
Taint Flags:

Insecure Randomness — High

Package: akka.cluster.metrics

ClusterMetricsRouting.scala, line 88 (Insecure Randomness) — High

```
if (weighted.isEmpty) NoRoutee
else weighted(ThreadLocalRandom.current.nextInt(weighted.total) + 1)
case None =>
routees(ThreadLocalRandom.current.nextInt(routees.size))
}

}
```

ClusterMetricsCollector.scala, line 272 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: selectRandomNode()
File: ClusterMetricsCollector.scala:272
Taint Flags:

```
context.actorSelection(self.path.toStringWithAddress(address)) ! envelope

def selectRandomNode(addresses: immutable.IndexedSeq[Address]): Option[Address] =
if (addresses.isEmpty) None else Some(addresses(ThreadLocalRandom.current.nextInt(addresses.size)))

/**
* Publishes to the event stream.
```

Poor Error Handling: Overly Broad Catch (2 issues)

Abstract

The catch block handles a broad swath of exceptions, potentially trapping dissimilar issues or problems that should not be dealt with at this point in the program.

Explanation

Multiple catch blocks can get repetitive, but "condensing" catch blocks by catching a high-level class such as `Exception` can obscure exceptions that deserve special treatment or that should not be caught at this point in the program. Catching an overly broad exception essentially defeats the purpose of Java's typed exceptions, and can become particularly dangerous if the program grows and begins to throw new types of exceptions. The new exception types will not receive any attention. **Example:** The following code excerpt handles three types of exceptions in an identical fashion.

```
  try {
    doExchange();
  }
  catch (IOException e) {
    logger.error("doExchange failed", e);
  }
  catch (InvocationTargetException e) {
    logger.error("doExchange failed", e);
  }
  catch (SQLException e) {
    logger.error("doExchange failed", e);
  }
```

At first blush, it may seem preferable to deal with these exceptions in a single catch block, as follows:

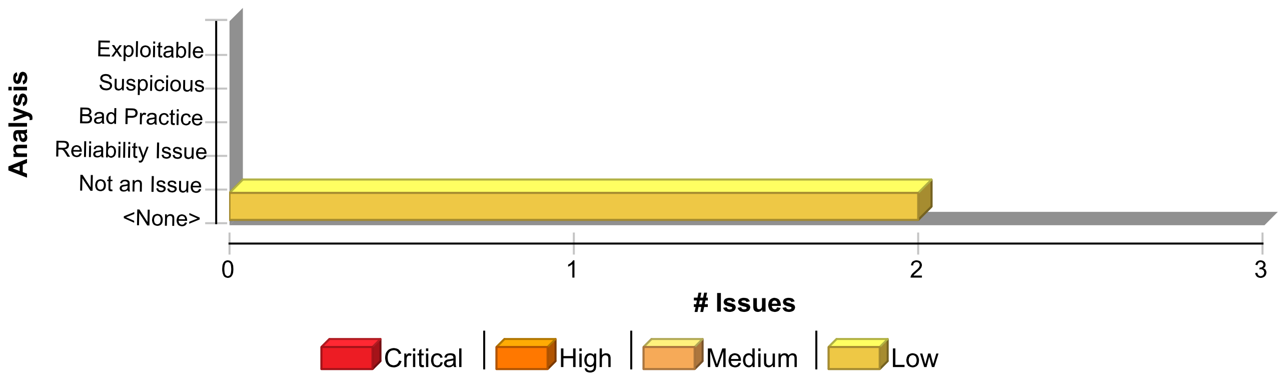
```
  try {
    doExchange();
  }
  catch (Exception e) {
    logger.error("doExchange failed", e);
  }
```

However, if `doExchange()` is modified to throw a new type of exception that should be handled in some different kind of way, the broad catch block will prevent the compiler from pointing out the situation. Further, the new catch block will now also handle exceptions derived from `RuntimeException` such as `ClassCastException`, and `NullPointerException`, which is not the programmer's intent.

Recommendation

Do not catch broad exception classes such as `Exception`, `Throwable`, `Error`, or `RuntimeException` except at the very top level of the program or thread.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Error Handling: Overly Broad Catch	2	0	0	2
Total	**2**	**0**	**0**	**2**

Poor Error Handling: Overly Broad Catch — Low

Package: akka.cluster.metrics

Provision.scala, line 44 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Sink Details

Sink: CatchBlock
Enclosing Method: isNativeLoaded()
File: Provision.scala:44
Taint Flags:

```
41 SigarProvider.close(sigar)
42 true
43 } catch {
44 case _: Throwable => false
45 }
46
47 /** Create sigar and verify it works. */
```

Provision.scala, line 107 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Sink Details

Sink: CatchBlock
Enclosing Method: apply()
File: Provision.scala:107
Taint Flags:

Poor Error Handling: Overly Broad Catch	Low

Package: akka.cluster.metrics

Provision.scala, line 107 (Poor Error Handling: Overly Broad Catch)	Low

```
 try Success(r)
 catch {
 // catching all, for example java.lang.LinkageError that are not caught by `NonFatal` in `Try`
 case e: Throwable => Failure(e)
 }
}

```