Fortify Standalone Report Generator

Developer Workbook

akka-remote

Table of Contents

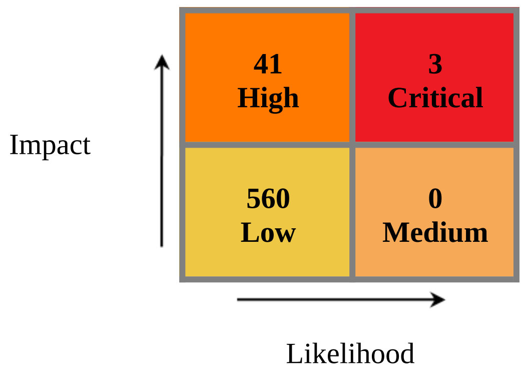
Executive Summary

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-remote project audit. The information contained in this workbook is targeted at project managers and developers.

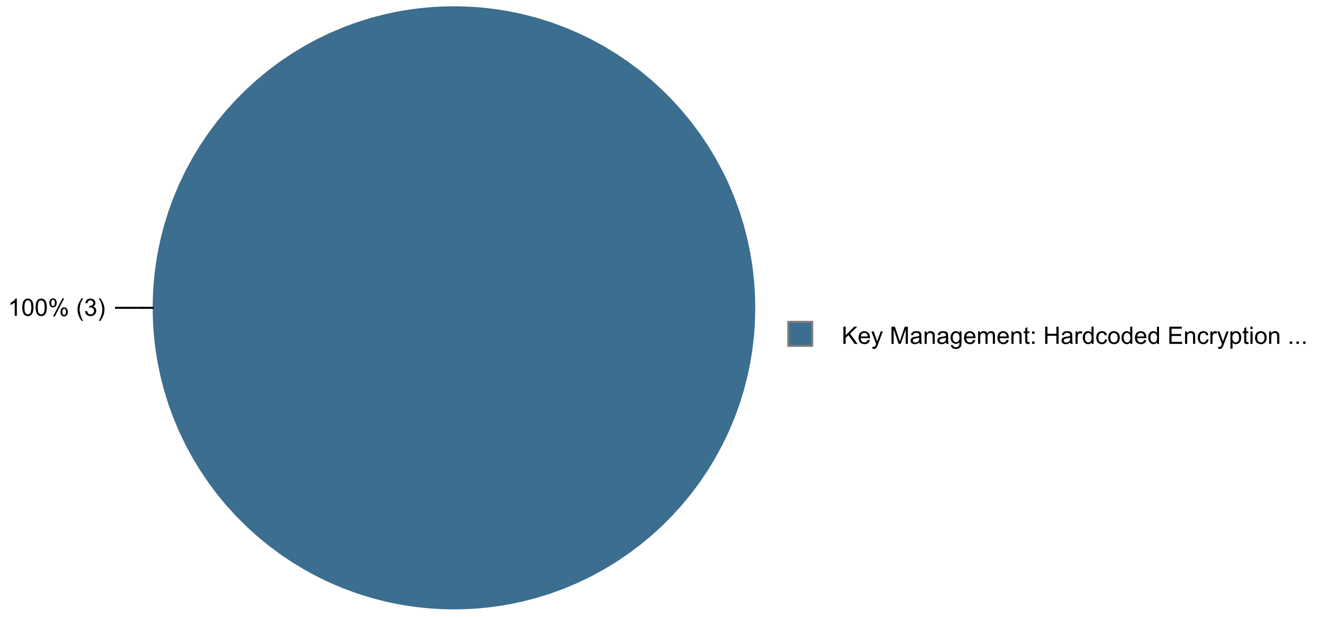
This section provides an overview of the issues uncovered during analysis.

Project Name:	akka-remote
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



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, 11:43 AM	**Engine Version:**	21.1.1.0009
Host Name:	Jacks-Work-MBP.local	**Certification:**	VALID
Number of Files:	197	**Lines of Code:**	17,212

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: Byte Array to String Conversion	0	0	0	0 / 1	0 / 1
Code Correctness: Constructor Invokes Overridable Function	0	0	0	0 / 286	0 / 286
Code Correctness: Erroneous String Compare	0	0	0	0 / 27	0 / 27
Code Correctness: Non-Static Inner Class Implements Serializable	0	0	0	0 / 133	0 / 133
Command Injection	0	0	0	0 / 10	0 / 10
Dead Code: Expression is Always false	0	0	0	0 / 27	0 / 27
Dead Code: Expression is Always true	0	0	0	0 / 6	0 / 6
Denial of Service	0	0	0	0 / 10	0 / 10
Insecure Randomness	0	0 / 19	0	0	0 / 19
J2EE Bad Practices: Sockets	0	0	0	0 / 7	0 / 7
J2EE Bad Practices: Threads	0	0	0	0 / 23	0 / 23
Key Management: Hardcoded Encryption Key	0 / 3	0	0	0	0 / 3
Missing Check against Null	0	0	0	0 / 13	0 / 13
Null Dereference	0	0 / 2	0	0	0 / 2
Object Model Violation: Just one of equals() and hashCode() Defined	0	0	0	0 / 4	0 / 4
Often Misused: Authentication	0	0 / 19	0	0	0 / 19
Poor Error Handling: Empty Catch Block	0	0	0	0 / 1	0 / 1
Poor Error Handling: Overly Broad Catch	0	0	0	0 / 3	0 / 3
Poor Style: Value Never Read	0	0	0	0 / 4	0 / 4
System Information Leak	0	0	0	0 / 3	0 / 3
Unchecked Return Value	0	0	0	0 / 2	0 / 2
Unreleased Resource: Synchronization	0	0 / 1	0	0	0 / 1

Results Outline

Code Correctness: Byte Array to String Conversion (1 issue)

Abstract

Converting a byte array into a `String` may lead to data loss.

Explanation

When data from a byte array is converted into a `String`, it is unspecified what will happen to any data that is outside of the applicable character set. This can lead to data being lost, or a decrease in the level of security when binary data is needed to ensure proper security measures are followed. **Example 1:** The following code converts data into a String in order to create a hash.

```
  ...
  FileInputStream fis = new FileInputStream(myFile);
  byte[] byteArr = byte[BUFSIZE];
  ...
  int count = fis.read(byteArr);
  ...
  String fileString = new String(byteArr);
  String fileSHA256Hex = DigestUtils.sha256Hex(fileString);
  // use fileSHA256Hex to validate file
  ...
```

Assuming the size of the file is less than `BUFSIZE`, this works fine as long as the information in `myFile` is encoded the same as the default character set, however if it's using a different encoding, or is a binary file, it will lose information. This in turn will cause the resulting SHA hash to be less reliable, and could mean it's far easier to cause collisions, especially if any data outside of the default character set is represented by the same value, such as a question mark.

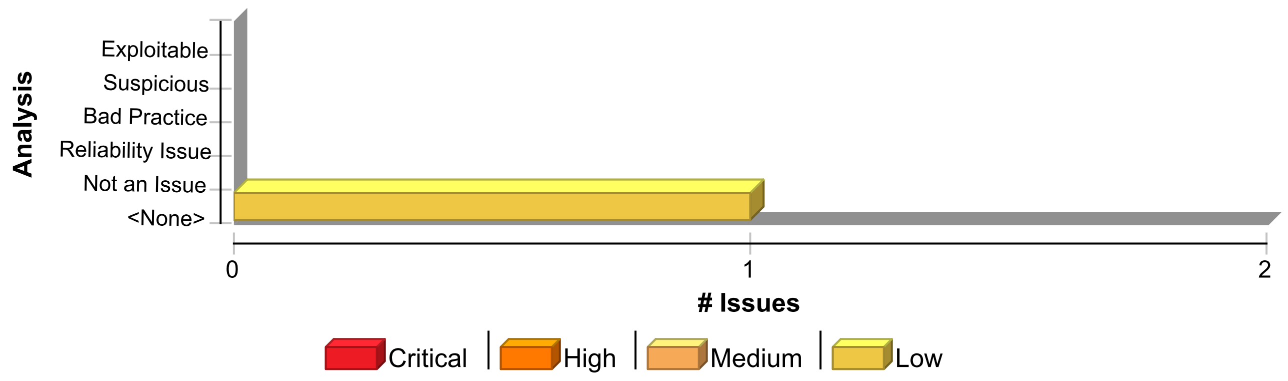
Recommendation

Generally speaking, a byte array potentially containing noncharacter data should never be converted into a `String` object as it may break functionality, but in some cases this can cause much larger security concerns. In a lot of cases there is no need to actually convert a byte array into a String, but if there is a specific reason to be able to create a `String` object from binary data, it must first be encoded in a way such that it will fit into the default character set. **Example 2:** The following uses a different variant of the API in `Example 1` to prevent any validation problems.

```
  ...
  FileInputStream fis = new FileInputStream(myFile);
  byte[] byteArr = byte[BUFSIZE];
  ...
  int count = fis.read(byteArr);
  ...
  byte[] fileSHA256 = DigestUtils.sha256(byteArr);
  // use fileSHA256 to validate file, comparing hash byte-by-byte.
  ...
```

In this case, it is straightforward to rectify, since this API has overloaded variants including one that accepts a byte array, and this could be simplified even further by using another overloaded variant of `DigestUtils.sha256()` that accepts a `FileInputStream` object as its argument. Other scenarios may need careful consideration as to whether it's possible that the byte array could contain data outside of the character set, and further refactoring may be required.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Byte Array to String Conversion	1	0	0	1
Total	**1**	**0**	**0**	**1**

Code Correctness: Byte Array to String Conversion — Low

Package: akka.remote.artery.compress

test/scala/akka/remote/artery/compress/CompressionIntegrationSpec.scala, line 432 (Code Correctness: Byte Array to String Conversion) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Semantic)

Sink Details

Sink: String()
Enclosing Method: fromBinary()
File: test/scala/akka/remote/artery/compress/CompressionIntegrationSpec.scala:432
Taint Flags:

```
429
430  override def fromBinary(bytes: Array[Byte], manifest: String): AnyRef = {
431  manifest match {
432  case TestMessageManifest => TestMessage(new String(bytes))
433  case unknown => throw new Exception("Unknown manifest: " + unknown)
434  }
435  }
```

Code Correctness: Constructor Invokes Overridable Function (286 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.

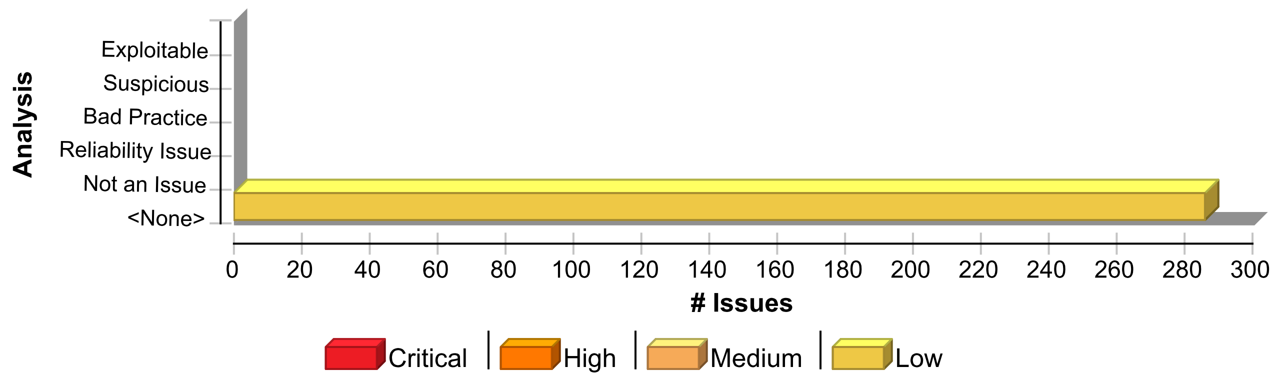
Recommendation

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	286	0	0	286
Total	**286**	**0**	**0**	**286**

Code Correctness: Constructor Invokes Overridable Function	**Low**
Package: akka.remote	
test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function)	**Low**

Issue Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: protocol
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57 val protocol =
58 if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59 else "akka.tcp"
60 val conf = ConfigFactory.parseString(s"""
61 akka {
62 actor.deployment {
63 /blub {
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: protocol
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57 val protocol =
58 if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59 else "akka.tcp"
60 val conf = ConfigFactory.parseString(s"""
61 akka {
62 actor.deployment {
63 /blub {
```

main/scala/akka/remote/Endpoint.scala, line 307 (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.remote

main/scala/akka/remote/Endpoint.scala, line 307 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: akka$remote$ReliableDeliverySupervisor$$createWriter
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:307
Taint Flags:

```
304 SeqNo(tmp)
305 }
306
307 var writer: ActorRef = createWriter()
308 var uid: Option[Int] = handleOrActive.map { _.handshakeInfo.uid }
309 var bailoutAt: Option[Deadline] = None
310 var maxSilenceTimer: Option[Cancellable] = None
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 56 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sysName
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:56
Taint Flags:

```
53
54 val port = system.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress.port.get
55 val sysName = system.name
56 val masterSystemName = "Master" + sysName
57 val protocol =
58 if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59 else "akka.tcp"
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sysName
Enclosing Method: RemoteRouterSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57  val protocol =
58  if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59  else "akka.tcp"
60  val conf = ConfigFactory.parseString(s"""
61  akka {
62  actor.deployment {
63  /blub {
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sysName
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57  val protocol =
58  if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59  else "akka.tcp"
60  val conf = ConfigFactory.parseString(s"""
61  akka {
62  actor.deployment {
63  /blub {
```

main/scala/akka/remote/Remoting.scala, line 509 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: pruneInterval
Enclosing Method: EndpointManager()
File: main/scala/akka/remote/Remoting.scala:509
Taint Flags:

```
506
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 509 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
507 val pruneInterval: FiniteDuration = (settings.RetryGateClosedFor * 2).max(1.second).min(10.seconds)
508
509 val pruneTimerCancellable: Cancellable =
510 context.system.scheduler.scheduleWithFixedDelay(pruneInterval, pruneInterval, self, Prune)
511
512 var pendingReadHandoffs = Map[ActorRef, AkkaProtocolHandle]()
```

main/scala/akka/remote/Remoting.scala, line 509 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: pruneInterval
Enclosing Method: EndpointManager()
File: main/scala/akka/remote/Remoting.scala:509
Taint Flags:

```
506
507 val pruneInterval: FiniteDuration = (settings.RetryGateClosedFor * 2).max(1.second).min(10.seconds)
508
509 val pruneTimerCancellable: Cancellable =
510 context.system.scheduler.scheduleWithFixedDelay(pruneInterval, pruneInterval, self, Prune)
511
512 var pendingReadHandoffs = Map[ActorRef, AkkaProtocolHandle]()
```

main/scala/akka/remote/RemoteWatcher.scala, line 113 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: artery
Enclosing Method: RemoteWatcher()
File: main/scala/akka/remote/RemoteWatcher.scala:113
Taint Flags:

```
110 val artery = remoteProvider.remoteSettings.Artery.Enabled
111
112 val (heartBeatMsg, selfHeartbeatRspMsg) =
113 if (artery) (ArteryHeartbeat, ArteryHeartbeatRsp(AddressUidExtension(context.system).longAddressUid))
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/RemoteWatcher.scala, line 113 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
114 else {
115 // For classic remoting the 'int' part is sufficient
116 @nowarn("msg=deprecated")
```

main/scala/akka/remote/RemoteDaemon.scala, line 79 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: allowListEnabled
Enclosing Method: RemoteSystemDaemon()
File: main/scala/akka/remote/RemoteDaemon.scala:79
Taint Flags:

```
76 private val allowListEnabled = system.settings.config.getBoolean("akka.remote.deployment.enable-allow-list")
77 private val remoteDeploymentAllowList: immutable.Set[String] = {
78 import akka.util.ccompat.JavaConverters._
79 if (allowListEnabled)
80 system.settings.config.getStringList("akka.remote.deployment.allowed-actor-classes").asScala.toSet
81 else Set.empty
82 }
```

main/scala/akka/remote/RemoteSettings.scala, line 190 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: configToMap
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:190
Taint Flags:

```
187 }
188
189 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
190 val Adapters: Map[String, String] = configToMap(getConfig("akka.remote.classic.adapters"))
191
192 private def transportNames: immutable.Seq[String] =
193 immutableSeq(getStringList("akka.remote.classic.enabled-transports"))
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote

test/scala/akka/remote/TypedActorRemoteDeploySpec.scala, line 41 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: TypedActorRemoteDeploySpec()
File: test/scala/akka/remote/TypedActorRemoteDeploySpec.scala:41
Taint Flags:

```
38
39 }
40
41 class TypedActorRemoteDeploySpec extends AkkaSpec(conf) {
42  val remoteName = "remote-sys"
43  val remoteSystem = ActorSystem(remoteName, conf)
44  val remoteAddress = RARP(remoteSystem).provider.getDefaultAddress
```

test/scala/akka/remote/TypedActorRemoteDeploySpec.scala, line 43 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: TypedActorRemoteDeploySpec()
File: test/scala/akka/remote/TypedActorRemoteDeploySpec.scala:43
Taint Flags:

```
40
41 class TypedActorRemoteDeploySpec extends AkkaSpec(conf) {
42  val remoteName = "remote-sys"
43  val remoteSystem = ActorSystem(remoteName, conf)
44  val remoteAddress = RARP(remoteSystem).provider.getDefaultAddress
45
46  @nowarn
```

main/scala/akka/remote/Endpoint.scala, line 659 (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.remote

main/scala/akka/remote/Endpoint.scala, line 659 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: provider
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:659
Taint Flags:

```
656 }
657
658 val provider = RARP(extendedSystem).provider
659 val msgDispatch = new DefaultMessageDispatcher(extendedSystem, provider, markLog)
660
661 val inbound = handle.isDefined
662 var stopReason: DisassociateInfo = AssociationHandle.Unknown
```

test/scala/akka/remote/TypedActorRemoteDeploySpec.scala, line 44 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSystem
Enclosing Method: TypedActorRemoteDeploySpec()
File: test/scala/akka/remote/TypedActorRemoteDeploySpec.scala:44
Taint Flags:

```
41 class TypedActorRemoteDeploySpec extends AkkaSpec(conf) {
42  val remoteName = "remote-sys"
43  val remoteSystem = ActorSystem(remoteName, conf)
44  val remoteAddress = RARP(remoteSystem).provider.getDefaultAddress
45
46  @nowarn
47  def verify[T](f: RemoteNameService => Future[T], expected: T) = {
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteRouterSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57  val protocol =
58  if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59  else "akka.tcp"
60  val conf = ConfigFactory.parseString(s"""
61  akka {
62  actor.deployment {
63  /blub {
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57  val protocol =
58  if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59  else "akka.tcp"
60  val conf = ConfigFactory.parseString(s"""
61  akka {
62  actor.deployment {
63  /blub {
```

main/scala/akka/remote/Remoting.scala, line 499 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: EndpointManager()
File: main/scala/akka/remote/Remoting.scala:499
Taint Flags:

```
496  val extendedSystem = context.system.asInstanceOf[ExtendedActorSystem]
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 499 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
497  val endpointId: Iterator[Int] = Iterator.from(0)
498
499  val eventPublisher = new EventPublisher(context.system, log, settings.RemoteLifecycleEventsLogLevel)
500
501  // Mapping between addresses and endpoint actors. If passive connections are turned off, incoming connections
502  // will be not part of this map!
```

main/scala/akka/remote/Remoting.scala, line 507 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: EndpointManager()
File: main/scala/akka/remote/Remoting.scala:507
Taint Flags:

```
504  // Mapping between transports and the local addresses they listen to
505  var transportMapping: Map[Address, AkkaProtocolTransport] = Map()
506
507  val pruneInterval: FiniteDuration = (settings.RetryGateClosedFor * 2).max(1.second).min(10.seconds)
508
509  val pruneTimerCancellable: Cancellable =
510  context.system.scheduler.scheduleWithFixedDelay(pruneInterval, pruneInterval, self, Prune)
```

main/scala/akka/remote/Endpoint.scala, line 299 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: reset
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:299
Taint Flags:

```
296  bailoutAt = None
297  }
298
299  reset()
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 299 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
300
301  def nextSeq(): SeqNo = {
302  val tmp = seqCounter
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 94 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:94
Taint Flags:

```
91  }
92  }
93  }""").withFallback(system.settings.config)
94  val masterSystem = ActorSystem(masterSystemName, conf)
95
96  override def afterTermination(): Unit = {
97  shutdown(masterSystem)
```

main/scala/akka/remote/Remoting.scala, line 161 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: log
Enclosing Method: Remoting()
File: main/scala/akka/remote/Remoting.scala:161
Taint Flags:

```
158  Remoting.localAddressForRemote(transportMapping, remote)
159
160  val log: LoggingAdapter = Logging(system.eventStream, classOf[Remoting])
161  val eventPublisher = new EventPublisher(system, log, RemoteLifecycleEventsLogLevel)
162
163  private def notifyError(msg: String, cause: Throwable): Unit =
164  eventPublisher.notifyListeners(RemotingErrorEvent(new RemoteTransportException(msg, cause)))
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 317 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: uid
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:317
Taint Flags:

```
314 // it serves a separator.
315 // If we already have an inbound handle then UID is initially confirmed.
316 // (This actor is never restarted)
317 var uidConfirmed: Boolean = uid.isDefined && (uid != refuseUid)
318
319 if (uid.isDefined && (uid == refuseUid))
320 throw new HopelessAssociation(
```

main/scala/akka/remote/Endpoint.scala, line 317 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: uid
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:317
Taint Flags:

```
314 // it serves a separator.
315 // If we already have an inbound handle then UID is initially confirmed.
316 // (This actor is never restarted)
317 var uidConfirmed: Boolean = uid.isDefined && (uid != refuseUid)
318
319 if (uid.isDefined && (uid == refuseUid))
320 throw new HopelessAssociation(
```

main/scala/akka/remote/Endpoint.scala, line 319 (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.remote

main/scala/akka/remote/Endpoint.scala, line 319 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: uid
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:319
Taint Flags:

```
316 // (This actor is never restarted)
317 var uidConfirmed: Boolean = uid.isDefined && (uid != refuseUid)
318
319 if (uid.isDefined && (uid == refuseUid))
320 throw new HopelessAssociation(
321 localAddress,
322 remoteAddress,
```

main/scala/akka/remote/Endpoint.scala, line 319 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: uid
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:319
Taint Flags:

```
316 // (This actor is never restarted)
317 var uidConfirmed: Boolean = uid.isDefined && (uid != refuseUid)
318
319 if (uid.isDefined && (uid == refuseUid))
320 throw new HopelessAssociation(
321 localAddress,
322 remoteAddress,
```

main/scala/akka/remote/Endpoint.scala, line 320 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: uid
Enclosing Method: ReliableDeliverySupervisor()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 320 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/Endpoint.scala:320
Taint Flags:

```
317 var uidConfirmed: Boolean = uid.isDefined && (uid != refuseUid)
318
319 if (uid.isDefined && (uid == refuseUid))
320 throw new HopelessAssociation(
321 localAddress,
322 remoteAddress,
323 uid,
```

main/scala/akka/remote/Endpoint.scala, line 324 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: uid
Enclosing Method: ReliableDeliverySupervisor()
File: main/scala/akka/remote/Endpoint.scala:324
Taint Flags:

```
321 localAddress,
322 remoteAddress,
323 uid,
324 new IllegalStateException(
325 s"The remote system [$remoteAddress] has a UID [${uid.get}] that has been quarantined. Association aborted."))
326
327 override def postStop(): Unit = {
```

main/scala/akka/remote/Endpoint.scala, line 641 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: extendedSystem
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:641
Taint Flags:

```
638
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 641 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
639 private val markLog = Logging.withMarker(this)
640 val extendedSystem: ExtendedActorSystem = context.system.asInstanceOf[ExtendedActorSystem]
641 val remoteMetrics = RemoteMetricsExtension(extendedSystem)
642 val backoffDispatcher = context.system.dispatchers.lookup("akka.remote.classic.backoff-remote-dispatcher")
643
644 var reader: Option[ActorRef] = None
```

main/scala/akka/remote/Endpoint.scala, line 658 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: extendedSystem
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:658
Taint Flags:

```
655 case NonFatal(e) => publishAndThrow(e, Logging.ErrorLevel)
656 }
657
658 val provider = RARP(extendedSystem).provider
659 val msgDispatch = new DefaultMessageDispatcher(extendedSystem, provider, markLog)
660
661 val inbound = handle.isDefined
```

main/scala/akka/remote/Endpoint.scala, line 659 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: extendedSystem
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:659
Taint Flags:

```
656 }
657
658 val provider = RARP(extendedSystem).provider
659 val msgDispatch = new DefaultMessageDispatcher(extendedSystem, provider, markLog)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 659 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
660
661  val inbound = handle.isDefined
662  var stopReason: DisassociateInfo = AssociationHandle.Unknown
```

test/scala/akka/remote/RemoteDeployerSpec.scala, line 36 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: deployerConf
Enclosing Method: RemoteDeployerSpec()
File: test/scala/akka/remote/RemoteDeployerSpec.scala:36
Taint Flags:

```
33
34 }
35
36 class RemoteDeployerSpec extends AkkaSpec(RemoteDeployerSpec.deployerConf) {
37
38  "A RemoteDeployer" must {
39
```

main/scala/akka/remote/Endpoint.scala, line 751 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: akka$remote$EndpointWriter$$MaxWriteCount
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:751
Taint Flags:

```
748  }
749
750  var writeCount = 0
751  var maxWriteCount = MaxWriteCount
752  var adaptiveBackoffNanos = 1000000L // 1 ms
753  var fullBackoff = false
754
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 649 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: newAckDeadline
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:649
Taint Flags:

```
646  val readerId = Iterator.from(0)
647
648  def newAckDeadline: Deadline = Deadline.now + settings.SysMsgAckTimeout
649  var ackDeadline: Deadline = newAckDeadline
650
651  var lastAck: Option[Ack] = None
652
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 171 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: createDeployer
Enclosing Method: RemoteActorRefProvider()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:171
Taint Flags:

```
168  !remoteSettings.UseUnsafeRemoteFeaturesWithoutCluster &&
169  remoteSettings.WarnUnsafeWatchWithoutCluster
170
171  override val deployer: Deployer = createDeployer
172
173  /**
174  * Factory method to make it possible to override deployer in subclass
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 188 (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.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 188 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: local
Enclosing Method: RemoteActorRefProvider()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:188
Taint Flags:

```
185  Some(deadLettersPath => new RemoteDeadLetterActorRef(this, deadLettersPath, eventStream)))
186
187  @volatile
188  private var _log = local.log
189  def log: LoggingAdapter = _log
190
191  override def rootPath: ActorPath = local.rootPath
```

main/scala/akka/remote/RemoteSettings.scala, line 181 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: transportNames
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:181
Taint Flags:

```
178  WatchFailureDetectorConfig.getMillisDuration("expected-response-after")
179  }.requiring(_ > Duration.Zero, "watch-failure-detector.expected-response-after > 0")
180
181  val Transports: immutable.Seq[(String, immutable.Seq[String], Config)] = transportNames.map { name =>
182  val transportConfig = transportConfigFor(name)
183  (
184  transportConfig.getString("transport-class"),
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 179 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: deployer
Enclosing Method: RemoteActorRefProvider()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 179 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/RemoteActorRefProvider.scala:179
Taint Flags:

```
176 */
177 protected def createDeployer: RemoteDeployer = new RemoteDeployer(settings, dynamicAccess)
178
179 private[akka] val local = new LocalActorRefProvider(
180 systemName,
181 settings,
182 eventStream,
```

main/scala/akka/remote/Endpoint.scala, line 661 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: handle
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:661
Taint Flags:

```
658 val provider = RARP(extendedSystem).provider
659 val msgDispatch = new DefaultMessageDispatcher(extendedSystem, provider, markLog)
660
661 val inbound = handle.isDefined
662 var stopReason: DisassociateInfo = AssociationHandle.Unknown
663
664 // Use an internal buffer instead of Stash for efficiency
```

test/scala/akka/remote/RemoteFeaturesSpec.scala, line 58 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: common
Enclosing Method: RemoteFeaturesSpec()
File: test/scala/akka/remote/RemoteFeaturesSpec.scala:58
Taint Flags:

```
55
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

test/scala/akka/remote/RemoteFeaturesSpec.scala, line 58 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
protected final val useUnsafe: Boolean = provider.remoteSettings.UseUnsafeRemoteFeaturesWithoutCluster

protected val remoteSystem1 = newRemoteSystem(name = Some("RS1"), extraConfig = Some(common(useUnsafe)))

@nowarn("msg=deprecated")
private def mute(): Unit = {
```

main/scala/akka/remote/RemoteWatcher.scala, line 110 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteProvider
Enclosing Method: RemoteWatcher()
File: main/scala/akka/remote/RemoteWatcher.scala:110
Taint Flags:

```
def scheduler = context.system.scheduler

val remoteProvider: RemoteActorRefProvider = RARP(context.system).provider
val artery = remoteProvider.remoteSettings.Artery.Enabled

val (heartBeatMsg, selfHeartbeatRspMsg) =
if (artery) (ArteryHeartbeat, ArteryHeartbeatRsp(AddressUidExtension(context.system).longAddressUid))
```

main/scala/akka/remote/RemoteWatcher.scala, line 135 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: scheduler
Enclosing Method: RemoteWatcher()
File: main/scala/akka/remote/RemoteWatcher.scala:135
Taint Flags:

```
var unreachable: Set[Address] = Set.empty
var addressUids: Map[Address, Long] = Map.empty

val heartbeatTask = scheduler.scheduleWithFixedDelay(heartbeatInterval, heartbeatInterval, self, HeartbeatTick)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/RemoteWatcher.scala, line 135 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
val failureDetectorReaperTask =
scheduler.scheduleWithFixedDelay(unreachableReaperInterval, unreachableReaperInterval, self, ReapUnreachableTick)

```

main/scala/akka/remote/RemoteWatcher.scala, line 136 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: scheduler
Enclosing Method: RemoteWatcher()
File: main/scala/akka/remote/RemoteWatcher.scala:136
Taint Flags:

```
var addressUids: Map[Address, Long] = Map.empty

val heartbeatTask = scheduler.scheduleWithFixedDelay(heartbeatInterval, heartbeatInterval, self, HeartbeatTick)
val failureDetectorReaperTask =
scheduler.scheduleWithFixedDelay(unreachableReaperInterval, unreachableReaperInterval, self, ReapUnreachableTick)

override def postStop(): Unit = {
```

main/scala/akka/remote/PhiAccrualFailureDetector.scala, line 122 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: firstHeartbeat
Enclosing Method: PhiAccrualFailureDetector()
File: main/scala/akka/remote/PhiAccrualFailureDetector.scala:122
Taint Flags:

```
*/
private case class State(history: HeartbeatHistory, timestamp: Option[Long])

private val state = new AtomicReference[State](State(history = firstHeartbeat, timestamp = None))

override def isAvailable: Boolean = isAvailable(clock())

```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/AddressUidExtension.scala, line 37 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: arteryEnabled
Enclosing Method: AddressUidExtension()
File: main/scala/akka/remote/AddressUidExtension.scala:37
Taint Flags:

```
34  private def arteryEnabled = system.provider.asInstanceOf[RemoteActorRefProvider].remoteSettings.Artery.Enabled
35
36  val longAddressUid: Long =
37  if (arteryEnabled) system.uid
38  // with the old remoting we need to make toInt.toLong return the same number
39  // to keep wire compatibility
40  else system.uid.toInt.toLong
```

test/scala/akka/remote/TypedActorRemoteDeploySpec.scala, line 43 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteName
Enclosing Method: TypedActorRemoteDeploySpec()
File: test/scala/akka/remote/TypedActorRemoteDeploySpec.scala:43
Taint Flags:

```
40
41 class TypedActorRemoteDeploySpec extends AkkaSpec(conf) {
42  val remoteName = "remote-sys"
43  val remoteSystem = ActorSystem(remoteName, conf)
44  val remoteAddress = RARP(remoteSystem).provider.getDefaultAddress
45
46  @nowarn
```

main/scala/akka/remote/Endpoint.scala, line 659 (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.remote

main/scala/akka/remote/Endpoint.scala, line 659 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: markLog
Enclosing Method: EndpointWriter()
File: main/scala/akka/remote/Endpoint.scala:659
Taint Flags:

```
656 }
657
658 val provider = RARP(extendedSystem).provider
659 val msgDispatch = new DefaultMessageDispatcher(extendedSystem, provider, markLog)
660
661 val inbound = handle.isDefined
662 var stopReason: DisassociateInfo = AssociationHandle.Unknown
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: masterSystemName
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/RemoteRouterSpec.scala:60
Taint Flags:

```
57 val protocol =
58 if (RARP(system).provider.remoteSettings.Artery.Enabled) "akka"
59 else "akka.tcp"
60 val conf = ConfigFactory.parseString(s"""
61 akka {
62 actor.deployment {
63 /blub {
```

test/scala/akka/remote/RemoteRouterSpec.scala, line 94 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: masterSystemName
Enclosing Method: RemoteRouterSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

test/scala/akka/remote/RemoteRouterSpec.scala, line 94 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/RemoteRouterSpec.scala:94
Taint Flags:

```
91  }
92  }
93  }""").withFallback(system.settings.config)
94  val masterSystem = ActorSystem(masterSystemName, conf)
95
96  override def afterTermination(): Unit = {
97  shutdown(masterSystem)
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 164 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSettings
Enclosing Method: RemoteActorRefProvider()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:164
Taint Flags:

```
161
162  val remoteSettings: RemoteSettings = new RemoteSettings(settings.config)
163
164  private[akka] final val hasClusterOrUseUnsafe = settings.HasCluster || remoteSettings.UseUnsafeRemoteFeaturesWithoutCluster
165
166  private val warnOnUnsafeRemote =
167  !settings.HasCluster &&
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 168 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSettings
Enclosing Method: RemoteActorRefProvider()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:168
Taint Flags:

```
165
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 168 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
166 private val warnOnUnsafeRemote =
167 !settings.HasCluster &&
168 !remoteSettings.UseUnsafeRemoteFeaturesWithoutCluster &&
169 remoteSettings.WarnUnsafeWatchWithoutCluster
170
171 override val deployer: Deployer = createDeployer
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 169 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSettings
Enclosing Method: RemoteActorRefProvider()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:169
Taint Flags:

```
166 private val warnOnUnsafeRemote =
167 !settings.HasCluster &&
168 !remoteSettings.UseUnsafeRemoteFeaturesWithoutCluster &&
169 remoteSettings.WarnUnsafeWatchWithoutCluster
170
171 override val deployer: Deployer = createDeployer
172
```

main/scala/akka/remote/RemoteSettings.scala, line 170 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: WatchFailureDetectorConfig
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:170
Taint Flags:

```
167 val WarnUnsafeWatchWithoutCluster: Boolean = getBoolean("akka.remote.warn-unsafe-watch-outside-cluster")
168
169 val WatchFailureDetectorConfig: Config = getConfig("akka.remote.watch-failure-detector")
170 val WatchFailureDetectorImplementationClass: String = WatchFailureDetectorConfig.getString("implementation-class")
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/RemoteSettings.scala, line 170 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
171 val WatchHeartBeatInterval: FiniteDuration = {
172 WatchFailureDetectorConfig.getMillisDuration("heartbeat-interval")
173 }.requiring(_ > Duration.Zero, "watch-failure-detector.heartbeat-interval must be > 0")
```

main/scala/akka/remote/RemoteSettings.scala, line 171 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: WatchFailureDetectorConfig
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:171
Taint Flags:

```
168
169 val WatchFailureDetectorConfig: Config = getConfig("akka.remote.watch-failure-detector")
170 val WatchFailureDetectorImplementationClass: String = WatchFailureDetectorConfig.getString("implementation-class")
171 val WatchHeartBeatInterval: FiniteDuration = {
172 WatchFailureDetectorConfig.getMillisDuration("heartbeat-interval")
173 }.requiring(_ > Duration.Zero, "watch-failure-detector.heartbeat-interval must be > 0")
174 val WatchUnreachableReaperInterval: FiniteDuration = {
```

main/scala/akka/remote/RemoteSettings.scala, line 174 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: WatchFailureDetectorConfig
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:174
Taint Flags:

```
171 val WatchHeartBeatInterval: FiniteDuration = {
172 WatchFailureDetectorConfig.getMillisDuration("heartbeat-interval")
173 }.requiring(_ > Duration.Zero, "watch-failure-detector.heartbeat-interval must be > 0")
174 val WatchUnreachableReaperInterval: FiniteDuration = {
175 WatchFailureDetectorConfig.getMillisDuration("unreachable-nodes-reaper-interval")
176 }.requiring(_ > Duration.Zero, "watch-failure-detector.unreachable-nodes-reaper-interval must be > 0")
177 val WatchHeartbeatExpectedResponseAfter: FiniteDuration = {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote

main/scala/akka/remote/RemoteSettings.scala, line 177 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: WatchFailureDetectorConfig
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:177
Taint Flags:

```
174 val WatchUnreachableReaperInterval: FiniteDuration = {
175 WatchFailureDetectorConfig.getMillisDuration("unreachable-nodes-reaper-interval")
176 }.requiring(_ > Duration.Zero, "watch-failure-detector.unreachable-nodes-reaper-interval must be > 0")
177 val WatchHeartbeatExpectedResponseAfter: FiniteDuration = {
178 WatchFailureDetectorConfig.getMillisDuration("expected-response-after")
179 }.requiring(_ > Duration.Zero, "watch-failure-detector.expected-response-after > 0")
180
```

Package: akka.remote.artery

main/scala/akka/remote/artery/Control.scala, line 119 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: InboundControlJunction()
File: main/scala/akka/remote/artery/Control.scala:119
Taint Flags:

```
116
117 val in: Inlet[InboundEnvelope] = Inlet("InboundControlJunction.in")
118 val out: Outlet[InboundEnvelope] = Outlet("InboundControlJunction.out")
119 override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
120
121 override def createLogicAndMaterializedValue(inheritedAttributes: Attributes) = {
122 val logic = new GraphStageLogic(shape) with InHandler with OutHandler with ControlMessageSubject {
```

main/scala/akka/remote/artery/Codecs.scala, line 719 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Codecs.scala, line 719 (Code Correctness: Constructor Invokes Overridable Function) — Low

Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: DuplicateHandshakeReq()
File: main/scala/akka/remote/artery/Codecs.scala:719
Taint Flags:

```
716
717  val in: Inlet[InboundEnvelope] = Inlet("Artery.DuplicateHandshakeReq.in")
718  val out: Outlet[InboundEnvelope] = Outlet("Artery.DuplicateHandshakeReq.out")
719  val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
720
721  override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
722  new GraphStageLogic(shape) with InHandler with OutHandler {
```

main/scala/akka/remote/artery/Association.scala, line 154 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: advancedSettings
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:154
Taint Flags:

```
151
152  override def settings = transport.settings
153  private def advancedSettings = transport.settings.Advanced
154  private val deathWatchNotificationFlushEnabled = advancedSettings.DeathWatchNotificationFlushTimeout > Duration.Zero &&
transport.provider.settings.HasCluster
155
156  private val restartCounter =
157  new RestartCounter(advancedSettings.OutboundMaxRestarts, advancedSettings.OutboundRestartTimeout)
```

main/scala/akka/remote/artery/Association.scala, line 157 (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.remote.artery

main/scala/akka/remote/artery/Association.scala, line 157 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: advancedSettings
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:157
Taint Flags:

```
154  private val deathWatchNotificationFlushEnabled = advancedSettings.DeathWatchNotificationFlushTimeout > Duration.Zero &&
transport.provider.settings.HasCluster
155
156  private val restartCounter =
157  new RestartCounter(advancedSettings.OutboundMaxRestarts, advancedSettings.OutboundRestartTimeout)
158
159  // We start with the raw wrapped queue and then it is replaced with the materialized value of
160  // the `SendQueue` after materialization. Using same underlying queue. This makes it possible to
```

main/scala/akka/remote/artery/Association.scala, line 157 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: advancedSettings
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:157
Taint Flags:

```
154  private val deathWatchNotificationFlushEnabled = advancedSettings.DeathWatchNotificationFlushTimeout > Duration.Zero &&
transport.provider.settings.HasCluster
155
156  private val restartCounter =
157  new RestartCounter(advancedSettings.OutboundMaxRestarts, advancedSettings.OutboundRestartTimeout)
158
159  // We start with the raw wrapped queue and then it is replaced with the materialized value of
160  // the `SendQueue` after materialization. Using same underlying queue. This makes it possible to
```

main/scala/akka/remote/artery/Association.scala, line 171 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: advancedSettings

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 171 (Code Correctness: Constructor Invokes Overridable Function) — Low

Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:171
Taint Flags:

```
168  new ManyToOneConcurrentArrayQueue[OutboundEnvelope](capacity)
169  }
170
171  private val outboundLanes = advancedSettings.OutboundLanes
172  private val controlQueueSize = advancedSettings.OutboundControlQueueSize
173  private val queueSize = advancedSettings.OutboundMessageQueueSize
174  private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize
```

main/scala/akka/remote/artery/Association.scala, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: advancedSettings
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:172
Taint Flags:

```
169  }
170
171  private val outboundLanes = advancedSettings.OutboundLanes
172  private val controlQueueSize = advancedSettings.OutboundControlQueueSize
173  private val queueSize = advancedSettings.OutboundMessageQueueSize
174  private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize
175
```

main/scala/akka/remote/artery/Association.scala, line 173 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: advancedSettings
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:173
Taint Flags:

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 173 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
170
171 private val outboundLanes = advancedSettings.OutboundLanes
172 private val controlQueueSize = advancedSettings.OutboundControlQueueSize
173 private val queueSize = advancedSettings.OutboundMessageQueueSize
174 private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize
175
176 private[this] val queues: Array[SendQueue.ProducerApi[OutboundEnvelope]] = new Array(2 + outboundLanes)
```

main/scala/akka/remote/artery/Association.scala, line 174 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: advancedSettings
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:174
Taint Flags:

```
171 private val outboundLanes = advancedSettings.OutboundLanes
172 private val controlQueueSize = advancedSettings.OutboundControlQueueSize
173 private val queueSize = advancedSettings.OutboundMessageQueueSize
174 private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize
175
176 private[this] val queues: Array[SendQueue.ProducerApi[OutboundEnvelope]] = new Array(2 + outboundLanes)
177 queues(ControlQueueIndex) = QueueWrapperImpl(createQueue(controlQueueSize, ControlQueueIndex)) // control stream
```

main/scala/akka/remote/artery/Codecs.scala, line 638 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: Deserializer()
File: main/scala/akka/remote/artery/Codecs.scala:638
Taint Flags:

```
635
636 val in: Inlet[InboundEnvelope] = Inlet("Artery.Deserializer.in")
637 val out: Outlet[InboundEnvelope] = Outlet("Artery.Deserializer.out")
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Codecs.scala, line 638 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)

override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
new GraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 93 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: masterSystem
Enclosing Method: RemoteDeploymentSpec()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:93
Taint Flags:

```
""""

val masterSystem = newRemoteSystem(name = Some("Master" + system.name), extraConfig = Some(conf))
val masterPort = address(masterSystem).port.get

"Remoting" must {

```

main/scala/akka/remote/artery/Association.scala, line 180 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: createQueue
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:180
Taint Flags:

```
queues(ControlQueueIndex) = QueueWrapperImpl(createQueue(controlQueueSize, ControlQueueIndex)) // control stream
queues(LargeQueueIndex) =
if (transport.largeMessageChannelEnabled) // large messages stream
QueueWrapperImpl(createQueue(largeQueueSize, LargeQueueIndex))
else
DisabledQueueWrapper
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 180 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
183
```

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/artery/RemoteRouterSpec.scala:84
Taint Flags:

```
81  }
82  }""").withFallback(system.settings.config)
83
84  val masterSystem = ActorSystem("Master" + sysName, conf)
85
86  override def afterTermination(): Unit = {
87  shutdown(masterSystem)
```

test/scala/akka/remote/artery/HandshakeRetrySpec.scala, line 23 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: commonConfig
Enclosing Method: HandshakeRetrySpec()
File: test/scala/akka/remote/artery/HandshakeRetrySpec.scala:23
Taint Flags:

```
20
21 }
22
23 class HandshakeRetrySpec extends ArteryMultiNodeSpec(HandshakeRetrySpec.commonConfig) with ImplicitSender {
24
25  val portB = freePort()
26
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 49 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sysName
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/artery/RemoteRouterSpec.scala:49
Taint Flags:

```
46
47  val port = RARP(system).provider.getDefaultAddress.port.get
48  val sysName = system.name
49  val conf = ConfigFactory.parseString(s"""
50  akka {
51  actor.deployment {
52  /blub {
```

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 49 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sysName
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/artery/RemoteRouterSpec.scala:49
Taint Flags:

```
46
47  val port = RARP(system).provider.getDefaultAddress.port.get
48  val sysName = system.name
49  val conf = ConfigFactory.parseString(s"""
50  akka {
51  actor.deployment {
52  /blub {
```

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 84 (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.remote.artery

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: sysName
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/artery/RemoteRouterSpec.scala:84
Taint Flags:

```
81 }
82 }""").withFallback(system.settings.config)
83
84 val masterSystem = ActorSystem("Master" + sysName, conf)
85
86 override def afterTermination(): Unit = {
87 shutdown(masterSystem)
```

test/scala/akka/remote/artery/RemoteWatcherSpec.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSystem
Enclosing Method: RemoteWatcherSpec()
File: test/scala/akka/remote/artery/RemoteWatcherSpec.scala:84
Taint Flags:

```
81 override def expectedTestDuration = 2.minutes
82
83 val remoteSystem = newRemoteSystem(name = Some("RemoteSystem"))
84 val remoteAddress = address(remoteSystem)
85 def remoteAddressUid = AddressUidExtension(remoteSystem).longAddressUid
86
87 override def afterTermination(): Unit = {
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: SystemMessageDelivery()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:84
Taint Flags:

```

val in: Inlet[OutboundEnvelope] = Inlet("SystemMessageDelivery.in")
val out: Outlet[OutboundEnvelope] = Outlet("SystemMessageDelivery.out")
override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)

override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
new TimerGraphStageLogic(shape) with InHandler with OutHandler with ControlMessageObserver with StageLogging {
```

main/scala/akka/remote/artery/RemoteInstrument.scala, line 184 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: instruments
Enclosing Method: RemoteInstruments()
File: main/scala/akka/remote/artery/RemoteInstrument.scala:184
Taint Flags:

```
// keep the remote instruments sorted by identifier to speed up deserialization
private val instruments: Vector[RemoteInstrument] = _instruments.sortBy(_.identifier)
// does any of the instruments want serialization timing?
private val serializationTimingEnabled = instruments.exists(_.serializationTimingEnabled)

def serialize(outboundEnvelope: OptionVal[OutboundEnvelope], buffer: ByteBuffer): Unit = {
if (instruments.nonEmpty && outboundEnvelope.isDefined) {
```

main/scala/akka/remote/artery/Codecs.scala, line 787 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: DuplicateFlush()
File: main/scala/akka/remote/artery/Codecs.scala:787
Taint Flags:

```

```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Codecs.scala, line 787 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
 val in: Inlet[InboundEnvelope] = Inlet("Artery.DuplicateFlush.in")
 val out: Outlet[InboundEnvelope] = Outlet("Artery.DuplicateFlush.out")
 val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)

 override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
 new GraphStageLogic(shape) with InHandler with OutHandler {
```

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 116 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: config
Enclosing Method: SystemMessageDeliverySpec()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:116
Taint Flags:

```
 }
}

class SystemMessageDeliverySpec extends AbstractSystemMessageDeliverySpec(SystemMessageDeliverySpec.config) {
 import SystemMessageDeliverySpec._

 "System messages" must {
```

main/scala/akka/remote/artery/TestStage.scala, line 116 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: OutboundTestStage()
File: main/scala/akka/remote/artery/TestStage.scala:116
Taint Flags:

```
 extends GraphStage[FlowShape[OutboundEnvelope, OutboundEnvelope]] {
 val in: Inlet[OutboundEnvelope] = Inlet("OutboundTestStage.in")
 val out: Outlet[OutboundEnvelope] = Outlet("OutboundTestStage.out")
 override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/TestStage.scala, line 116 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
117
118  override def createLogic(inheritedAttributes: Attributes) =
119  new TimerGraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/HandshakeFailureSpec.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: commonConfig
Enclosing Method: HandshakeFailureSpec()
File: test/scala/akka/remote/artery/HandshakeFailureSpec.scala:25
Taint Flags:

```
22
23 }
24
25 class HandshakeFailureSpec extends ArteryMultiNodeSpec(HandshakeFailureSpec.commonConfig) with ImplicitSender {
26
27  val portB = freePort()
28
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 333 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: SystemMessageAcker()
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:333
Taint Flags:

```
330
331  val in: Inlet[InboundEnvelope] = Inlet("SystemMessageAcker.in")
332  val out: Outlet[InboundEnvelope] = Outlet("SystemMessageAcker.out")
333  override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
334
335  override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
336  new GraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Codecs.scala, line 719 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: DuplicateHandshakeReq()
File: main/scala/akka/remote/artery/Codecs.scala:719
Taint Flags:

```

val in: Inlet[InboundEnvelope] = Inlet("Artery.DuplicateHandshakeReq.in")
val out: Outlet[InboundEnvelope] = Outlet("Artery.DuplicateHandshakeReq.out")
val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)

override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
new GraphStageLogic(shape) with InHandler with OutHandler {
```

test/scala/akka/remote/artery/SerializationErrorSpec.scala, line 29 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: SerializationErrorSpec()
File: test/scala/akka/remote/artery/SerializationErrorSpec.scala:29
Taint Flags:

```
"akka.serialization.ByteArraySerializer" = -4
}
"""))
systemB.actorOf(TestActors.echoActorProps, "echo")
val addressB = address(systemB)
val rootB = RootActorPath(addressB)

```

test/scala/akka/remote/artery/SerializationErrorSpec.scala, line 30 (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.remote.artery

test/scala/akka/remote/artery/SerializationErrorSpec.scala, line 30 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: SerializationErrorSpec()
File: test/scala/akka/remote/artery/SerializationErrorSpec.scala:30
Taint Flags:

```
27 }
28 """))
29 systemB.actorOf(TestActors.echoActorProps, "echo")
30 val addressB = address(systemB)
31 val rootB = RootActorPath(addressB)
32
33 "Serialization error" must {
```

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 92 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: RemoteDeploymentSpec()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:92
Taint Flags:

```
89 akka.remote.artery.advanced.outbound-lanes = 3
90 """
91
92 val masterSystem = newRemoteSystem(name = Some("Master" + system.name), extraConfig = Some(conf))
93 val masterPort = address(masterSystem).port.get
94
95 "Remoting" must {
```

main/scala/akka/remote/artery/MessageDispatcher.scala, line 27 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: log
Enclosing Method: MessageDispatcher()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/MessageDispatcher.scala, line 27 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/MessageDispatcher.scala:27
Taint Flags:

```
24
25 private val remoteDaemon = provider.remoteDaemon
26 private val log = Logging.withMarker(system, getClass.getName)
27 private val debugLogEnabled: Boolean = log.isDebugEnabled
28
29 def dispatch(inboundEnvelope: InboundEnvelope): Unit = {
30 import Logging.messageClassName
```

main/scala/akka/remote/artery/InboundQuarantineCheck.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: InboundQuarantineCheck()
File: main/scala/akka/remote/artery/InboundQuarantineCheck.scala:25
Taint Flags:

```
22 extends GraphStage[FlowShape[InboundEnvelope, InboundEnvelope]] {
23 val in: Inlet[InboundEnvelope] = Inlet("InboundQuarantineCheck.in")
24 val out: Outlet[InboundEnvelope] = Outlet("InboundQuarantineCheck.out")
25 override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
26
27 override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
28 new GraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

main/scala/akka/remote/artery/Association.scala, line 180 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: largeQueueSize
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:180
Taint Flags:

```
177 queues(ControlQueueIndex) = QueueWrapperImpl(createQueue(controlQueueSize, ControlQueueIndex)) // control stream
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 180 (Code Correctness: Constructor Invokes Overridable Function)	Low

```
queues(LargeQueueIndex) =
if (transport.largeMessageChannelEnabled) // large messages stream
QueueWrapperImpl(createQueue(largeQueueSize, LargeQueueIndex))
else
DisabledQueueWrapper

```

main/scala/akka/remote/artery/Codecs.scala, line 69 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: Encoder()
File: main/scala/akka/remote/artery/Codecs.scala:69
Taint Flags:

```

val in: Inlet[OutboundEnvelope] = Inlet("Artery.Encoder.in")
val out: Outlet[EnvelopeBuffer] = Outlet("Artery.Encoder.out")
val shape: FlowShape[OutboundEnvelope, EnvelopeBuffer] = FlowShape(in, out)

override def createLogicAndMaterializedValue(
inheritedAttributes: Attributes): (GraphStageLogic, OutboundCompressionAccess) = {
```

test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala, line 47 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: config
Enclosing Method: RemoteDeathWatchSpec()
File: test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala:47
Taint Flags:

```
}

class RemoteDeathWatchSpec
extends ArteryMultiNodeSpec(RemoteDeathWatchSpec.config)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala, line 47 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
with ImplicitSender
with DefaultTimeout
with DeathWatchSpec {
```

main/scala/akka/remote/artery/Association.scala, line 177 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: controlQueueSize
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:177
Taint Flags:

```
private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize

private[this] val queues: Array[SendQueue.ProducerApi[OutboundEnvelope]] = new Array(2 + outboundLanes)
queues(ControlQueueIndex) = QueueWrapperImpl(createQueue(controlQueueSize, ControlQueueIndex)) // control stream
queues(LargeQueueIndex) =
if (transport.largeMessageChannelEnabled) // large messages stream
QueueWrapperImpl(createQueue(largeQueueSize, LargeQueueIndex))
```

main/scala/akka/remote/artery/Codecs.scala, line 368 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: Decoder()
File: main/scala/akka/remote/artery/Codecs.scala:368
Taint Flags:

```
import Decoder.Tick
val in: Inlet[EnvelopeBuffer] = Inlet("Artery.Decoder.in")
val out: Outlet[InboundEnvelope] = Outlet("Artery.Decoder.out")
val shape: FlowShape[EnvelopeBuffer, InboundEnvelope] = FlowShape(in, out)

def createLogicAndMaterializedValue(inheritedAttributes: Attributes): (GraphStageLogic, InboundCompressionAccess) = {
val logic = new TimerGraphStageLogic(shape)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/SerializationErrorSpec.scala, line 31 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: addressB
Enclosing Method: SerializationErrorSpec()
File: test/scala/akka/remote/artery/SerializationErrorSpec.scala:31
Taint Flags:

```
28 """))
29 systemB.actorOf(TestActors.echoActorProps, "echo")
30 val addressB = address(systemB)
31 val rootB = RootActorPath(addressB)
32
33 "Serialization error" must {
34
```

main/scala/akka/remote/artery/Control.scala, line 119 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: InboundControlJunction()
File: main/scala/akka/remote/artery/Control.scala:119
Taint Flags:

```
116
117 val in: Inlet[InboundEnvelope] = Inlet("InboundControlJunction.in")
118 val out: Outlet[InboundEnvelope] = Outlet("InboundControlJunction.out")
119 override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
120
121 override def createLogicAndMaterializedValue(inheritedAttributes: Attributes) = {
122 val logic = new GraphStageLogic(shape) with InHandler with OutHandler with ControlMessageSubject {
```

main/scala/akka/remote/artery/Control.scala, line 194 (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.remote.artery

main/scala/akka/remote/artery/Control.scala, line 194 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: out
Enclosing Method: OutboundControlJunction()
File: main/scala/akka/remote/artery/Control.scala:194
Taint Flags:

```
191 import OutboundControlJunction._
192 val in: Inlet[OutboundEnvelope] = Inlet("OutboundControlJunction.in")
193 val out: Outlet[OutboundEnvelope] = Outlet("OutboundControlJunction.out")
194 override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)
195
196 override def createLogicAndMaterializedValue(inheritedAttributes: Attributes) = {
197
```

test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala, line 21 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: otherPort
Enclosing Method: RemoteDeathWatchSpec()
File: test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala:21
Taint Flags:

```
18 object RemoteDeathWatchSpec {
19 val otherPort = ArteryMultiNodeSpec.freePort(ConfigFactory.load())
20
21 val config = ConfigFactory.parseString(s"""
22 akka {
23 actor {
24 provider = remote
```

main/scala/akka/remote/artery/SendQueue.scala, line 49 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: SendQueue()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/SendQueue.scala, line 49 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/SendQueue.scala:49
Taint Flags:

```
46 import SendQueue._
47
48 val out: Outlet[T] = Outlet("SendQueue.out")
49 override val shape: SourceShape[T] = SourceShape(out)
50
51 override def createLogicAndMaterializedValue(inheritedAttributes: Attributes): (GraphStageLogic, QueueValue[T]) = {
52 @volatile var needWakeup = false
```

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 50 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: safe
Enclosing Method: SystemMessageDeliverySpec()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:50
Taint Flags:

```
47 akka.stream.materializer.debug.fuzzing-mode = on
48 """).withFallback(ArterySpecSupport.defaultConfig)
49
50 val config =
51 ConfigFactory.parseString("akka.remote.use-unsafe-remote-features-outside-cluster = on").withFallback(safe)
52 }
53
```

test/scala/akka/remote/artery/HandshakeDenySpec.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: commonConfig
Enclosing Method: HandshakeDenySpec()
File: test/scala/akka/remote/artery/HandshakeDenySpec.scala:25
Taint Flags:

```
22
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/HandshakeDenySpec.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
23 }
24
25 class HandshakeDenySpec extends ArteryMultiNodeSpec(HandshakeDenySpec.commonConfig) with ImplicitSender {
26
27  var systemB = newRemoteSystem(name = Some("systemB"))
28
```

test/scala/akka/remote/artery/UntrustedSpec.scala, line 69 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: config
Enclosing Method: UntrustedSpec()
File: test/scala/akka/remote/artery/UntrustedSpec.scala:69
Taint Flags:

```
66
67 }
68
69 class UntrustedSpec extends ArteryMultiNodeSpec(UntrustedSpec.config) with ImplicitSender {
70
71  import UntrustedSpec._
72
```

main/scala/akka/remote/artery/Handshake.scala, line 226 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: InboundHandshake()
File: main/scala/akka/remote/artery/Handshake.scala:226
Taint Flags:

```
223  extends GraphStage[FlowShape[InboundEnvelope, InboundEnvelope]] {
224  val in: Inlet[InboundEnvelope] = Inlet("InboundHandshake.in")
225  val out: Outlet[InboundEnvelope] = Outlet("InboundHandshake.out")
226  override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Handshake.scala, line 226 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
227
228  override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
229  new TimerGraphStageLogic(shape) with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/LateConnectSpec.scala, line 26 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: config
Enclosing Method: LateConnectSpec()
File: test/scala/akka/remote/artery/LateConnectSpec.scala:26
Taint Flags:

```
23
24 }
25
26 class LateConnectSpec extends ArteryMultiNodeSpec(LateConnectSpec.config) with ImplicitSender {
27
28  val portB = freePort()
29  lazy val systemB =
```

test/scala/akka/remote/artery/RemoteMessageSerializationSpec.scala, line 31 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSystem
Enclosing Method: RemoteMessageSerializationSpec()
File: test/scala/akka/remote/artery/RemoteMessageSerializationSpec.scala:31
Taint Flags:

```
28  val maxPayloadBytes = RARP(system).provider.remoteSettings.Artery.Advanced.MaximumFrameSize
29
30  val remoteSystem = newRemoteSystem()
31  val remotePort = port(remoteSystem)
32
33  "Remote message serialization" should {
34
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Handshake.scala, line 65 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: OutboundHandshake()
File: main/scala/akka/remote/artery/Handshake.scala:65
Taint Flags:

```
62
63 val in: Inlet[OutboundEnvelope] = Inlet("OutboundHandshake.in")
64 val out: Outlet[OutboundEnvelope] = Outlet("OutboundHandshake.out")
65 override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)
66
67 override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
68 new TimerGraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/RemoteWatcherSpec.scala, line 92 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteAddressUid
Enclosing Method: RemoteWatcherSpec()
File: test/scala/akka/remote/artery/RemoteWatcherSpec.scala:92
Taint Flags:

```
89 super.afterTermination()
90 }
91
92 val heartbeatRspB = ArteryHeartbeatRsp(remoteAddressUid)
93
94 def createRemoteActor(props: Props, name: String): InternalActorRef = {
95 remoteSystem.actorOf(props, name)
```

test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala, line 55 (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.remote.artery

test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala, line 55 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: otherPort
Enclosing Method: RemoteDeathWatchSpec()
File: test/scala/akka/remote/artery/RemoteDeathWatchSpec.scala:55
Taint Flags:

```
52
53  system.eventStream.publish(TestEvent.Mute(EventFilter[io.aeron.exceptions.RegistrationException]()))
54
55  val other = newRemoteSystem(name = Some("other"), extraConfig = Some(s"akka.remote.artery.canonical.port=$otherPort"))
56
57  override def expectedTestDuration: FiniteDuration = 120.seconds
58
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 333 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: SystemMessageAcker()
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:333
Taint Flags:

```
330
331  val in: Inlet[InboundEnvelope] = Inlet("SystemMessageAcker.in")
332  val out: Outlet[InboundEnvelope] = Outlet("SystemMessageAcker.out")
333  override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
334
335  override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
336  new GraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

main/scala/akka/remote/artery/RemoteInstrument.scala, line 178 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: create
Enclosing Method: RemoteInstruments()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/RemoteInstrument.scala, line 178 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/RemoteInstrument.scala:178
Taint Flags:

```
 _instruments: Vector[RemoteInstrument]) {
 import RemoteInstruments._

 def this(system: ExtendedActorSystem, log: LoggingAdapter) = this(system, log, RemoteInstruments.create(system, log))
 def this(system: ExtendedActorSystem) = this(system, Logging.getLogger(system, classOf[RemoteInstruments]))

 // keep the remote instruments sorted by identifier to speed up deserialization
```

test/scala/akka/remote/artery/RemoteDeployerSpec.scala, line 33 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: deployerConf
Enclosing Method: RemoteDeployerSpec()
File: test/scala/akka/remote/artery/RemoteDeployerSpec.scala:33
Taint Flags:

```

}

class RemoteDeployerSpec extends AkkaSpec(RemoteDeployerSpec.deployerConf) {

 "A RemoteDeployer" must {

```

main/scala/akka/remote/artery/Association.scala, line 176 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: outboundLanes
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:176
Taint Flags:

```
 private val queueSize = advancedSettings.OutboundMessageQueueSize
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 176 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
174 private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize
175
176 private[this] val queues: Array[SendQueue.ProducerApi[OutboundEnvelope]] = new Array(2 + outboundLanes)
177 queues(ControlQueueIndex) = QueueWrapperImpl(createQueue(controlQueueSize, ControlQueueIndex)) // control stream
178 queues(LargeQueueIndex) =
179 if (transport.largeMessageChannelEnabled) // large messages stream
```

main/scala/akka/remote/artery/Association.scala, line 184 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: outboundLanes
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:184
Taint Flags:

```
181 else
182 DisabledQueueWrapper
183
184 (0 until outboundLanes).foreach { i =>
185 queues(OrdinaryQueueIndex + i) = QueueWrapperImpl(createQueue(queueSize, OrdinaryQueueIndex + i)) // ordinary messages stream
186 }
187 @volatile private[this] var queuesVisibility = false
```

main/scala/akka/remote/artery/Association.scala, line 177 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: createQueue
Enclosing Method: Association()
File: main/scala/akka/remote/artery/Association.scala:177
Taint Flags:

```
174 private val largeQueueSize = advancedSettings.OutboundLargeMessageQueueSize
175
176 private[this] val queues: Array[SendQueue.ProducerApi[OutboundEnvelope]] = new Array(2 + outboundLanes)
177 queues(ControlQueueIndex) = QueueWrapperImpl(createQueue(controlQueueSize, ControlQueueIndex)) // control stream
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 177 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
178 queues(LargeQueueIndex) =
179 if (transport.largeMessageChannelEnabled) // large messages stream
180 QueueWrapperImpl(createQueue(largeQueueSize, LargeQueueIndex))
```

main/scala/akka/remote/artery/Codecs.scala, line 638 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: Deserializer()
File: main/scala/akka/remote/artery/Codecs.scala:638
Taint Flags:

```
635
636 val in: Inlet[InboundEnvelope] = Inlet("Artery.Deserializer.in")
637 val out: Outlet[InboundEnvelope] = Outlet("Artery.Deserializer.out")
638 val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
639
640 override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
641 new GraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 81 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteDeploymentSpec()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:81
Taint Flags:

```
78 import RemoteDeploymentSpec._
79
80 val port = RARP(system).provider.getDefaultAddress.port.get
81 val conf =
82 s"""
83 akka.actor.deployment {
84 /blub.remote = "akka://${system.name}@localhost:$port"
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 81 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteDeploymentSpec()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:81
Taint Flags:

```
78 import RemoteDeploymentSpec._
79
80 val port = RARP(system).provider.getDefaultAddress.port.get
81 val conf =
82 s"""
83 akka.actor.deployment {
84 /blub.remote = "akka://${system.name}@localhost:$port"
```

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 81 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteDeploymentSpec()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:81
Taint Flags:

```
78 import RemoteDeploymentSpec._
79
80 val port = RARP(system).provider.getDefaultAddress.port.get
81 val conf =
82 s"""
83 akka.actor.deployment {
84 /blub.remote = "akka://${system.name}@localhost:$port"
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 40 (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.remote.artery

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 40 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: charset
Enclosing Method: TestInstrument()
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:40
Taint Flags:

```
37 import akka.remote.artery.MetadataCarryingSpy._
38
39 private val charset = Charset.forName("UTF-8")
40 private val encoder = charset.newEncoder()
41 private val decoder = charset.newDecoder()
42
43 override val identifier: Byte = 1
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 41 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: charset
Enclosing Method: TestInstrument()
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:41
Taint Flags:

```
38
39 private val charset = Charset.forName("UTF-8")
40 private val encoder = charset.newEncoder()
41 private val decoder = charset.newDecoder()
42
43 override val identifier: Byte = 1
44
```

main/scala/akka/remote/artery/Codecs.scala, line 787 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: DuplicateFlush()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Codecs.scala, line 787 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/Codecs.scala:787
Taint Flags:

```
784
785  val in: Inlet[InboundEnvelope] = Inlet("Artery.DuplicateFlush.in")
786  val out: Outlet[InboundEnvelope] = Outlet("Artery.DuplicateFlush.out")
787  val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
788
789  override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
790  new GraphStageLogic(shape) with InHandler with OutHandler {
```

main/scala/akka/remote/artery/Codecs.scala, line 368 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: Decoder()
File: main/scala/akka/remote/artery/Codecs.scala:368
Taint Flags:

```
365  import Decoder.Tick
366  val in: Inlet[EnvelopeBuffer] = Inlet("Artery.Decoder.in")
367  val out: Outlet[InboundEnvelope] = Outlet("Artery.Decoder.out")
368  val shape: FlowShape[EnvelopeBuffer, InboundEnvelope] = FlowShape(in, out)
369
370  def createLogicAndMaterializedValue(inheritedAttributes: Attributes): (GraphStageLogic, InboundCompressionAccess) = {
371  val logic = new TimerGraphStageLogic(shape)
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: SystemMessageDelivery()
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:84
Taint Flags:

```
81
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 84 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
82 val in: Inlet[OutboundEnvelope] = Inlet("SystemMessageDelivery.in")
83 val out: Outlet[OutboundEnvelope] = Outlet("SystemMessageDelivery.out")
84 override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)
85
86 override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
87 new TimerGraphStageLogic(shape) with InHandler with OutHandler with ControlMessageObserver with StageLogging {
```

main/scala/akka/remote/artery/TestStage.scala, line 149 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: InboundTestStage()
File: main/scala/akka/remote/artery/TestStage.scala:149
Taint Flags:

```
146 extends GraphStage[FlowShape[InboundEnvelope, InboundEnvelope]] {
147 val in: Inlet[InboundEnvelope] = Inlet("InboundTestStage.in")
148 val out: Outlet[InboundEnvelope] = Outlet("InboundTestStage.out")
149 override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
150
151 override def createLogic(inheritedAttributes: Attributes) =
152 new TimerGraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

main/scala/akka/remote/artery/InboundQuarantineCheck.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: InboundQuarantineCheck()
File: main/scala/akka/remote/artery/InboundQuarantineCheck.scala:25
Taint Flags:

```
22 extends GraphStage[FlowShape[InboundEnvelope, InboundEnvelope]] {
23 val in: Inlet[InboundEnvelope] = Inlet("InboundQuarantineCheck.in")
24 val out: Outlet[InboundEnvelope] = Outlet("InboundQuarantineCheck.out")
25 override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/InboundQuarantineCheck.scala, line 25 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
new GraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 49 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/artery/RemoteRouterSpec.scala:49
Taint Flags:

```
val port = RARP(system).provider.getDefaultAddress.port.get
val sysName = system.name
val conf = ConfigFactory.parseString(s"""
akka {
actor.deployment {
/blub {
```

test/scala/akka/remote/artery/RemoteRouterSpec.scala, line 49 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: port
Enclosing Method: RemoteRouterSpec()
File: test/scala/akka/remote/artery/RemoteRouterSpec.scala:49
Taint Flags:

```
val port = RARP(system).provider.getDefaultAddress.port.get
val sysName = system.name
val conf = ConfigFactory.parseString(s"""
akka {
actor.deployment {
/blub {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Codecs.scala, line 69 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: Encoder()
File: main/scala/akka/remote/artery/Codecs.scala:69
Taint Flags:

```
66
67  val in: Inlet[OutboundEnvelope] = Inlet("Artery.Encoder.in")
68  val out: Outlet[EnvelopeBuffer] = Outlet("Artery.Encoder.out")
69  val shape: FlowShape[OutboundEnvelope, EnvelopeBuffer] = FlowShape(in, out)
70
71  override def createLogicAndMaterializedValue(
72  inheritedAttributes: Attributes): (GraphStageLogic, OutboundCompressionAccess) = {
```

main/scala/akka/remote/artery/RemoteInstrument.scala, line 100 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: LoggingRemoteInstrument()
File: main/scala/akka/remote/artery/RemoteInstrument.scala:100
Taint Flags:

```
97  .transport
98  .asInstanceOf[ArteryTransport]
99  .settings
100  private val logFrameSizeExceeding = settings.LogFrameSizeExceeding.get
101
102  private val log = Logging(system, classOf[LoggingRemoteInstrument])
103
```

main/scala/akka/remote/artery/Handshake.scala, line 65 (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.remote.artery

main/scala/akka/remote/artery/Handshake.scala, line 65 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: out
Enclosing Method: OutboundHandshake()
File: main/scala/akka/remote/artery/Handshake.scala:65
Taint Flags:

```
62
63  val in: Inlet[OutboundEnvelope] = Inlet("OutboundHandshake.in")
64  val out: Outlet[OutboundEnvelope] = Outlet("OutboundHandshake.out")
65  override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)
66
67  override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
68  new TimerGraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

main/scala/akka/remote/artery/TestStage.scala, line 116 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: OutboundTestStage()
File: main/scala/akka/remote/artery/TestStage.scala:116
Taint Flags:

```
113  extends GraphStage[FlowShape[OutboundEnvelope, OutboundEnvelope]] {
114  val in: Inlet[OutboundEnvelope] = Inlet("OutboundTestStage.in")
115  val out: Outlet[OutboundEnvelope] = Outlet("OutboundTestStage.out")
116  override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)
117
118  override def createLogic(inheritedAttributes: Attributes) =
119  new TimerGraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

main/scala/akka/remote/artery/Handshake.scala, line 226 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: InboundHandshake()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Handshake.scala, line 226 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/Handshake.scala:226
Taint Flags:

```
extends GraphStage[FlowShape[InboundEnvelope, InboundEnvelope]] {
val in: Inlet[InboundEnvelope] = Inlet("InboundHandshake.in")
val out: Outlet[InboundEnvelope] = Outlet("InboundHandshake.out")
override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)

override def createLogic(inheritedAttributes: Attributes): GraphStageLogic =
new TimerGraphStageLogic(shape) with OutHandler with StageLogging {
```

main/scala/akka/remote/artery/Control.scala, line 194 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: OutboundControlJunction()
File: main/scala/akka/remote/artery/Control.scala:194
Taint Flags:

```
import OutboundControlJunction._
val in: Inlet[OutboundEnvelope] = Inlet("OutboundControlJunction.in")
val out: Outlet[OutboundEnvelope] = Outlet("OutboundControlJunction.out")
override val shape: FlowShape[OutboundEnvelope, OutboundEnvelope] = FlowShape(in, out)

override def createLogicAndMaterializedValue(inheritedAttributes: Attributes) = {

```

main/scala/akka/remote/artery/TestStage.scala, line 149 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: InboundTestStage()
File: main/scala/akka/remote/artery/TestStage.scala:149
Taint Flags:

```
extends GraphStage[FlowShape[InboundEnvelope, InboundEnvelope]] {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/TestStage.scala, line 149 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
147 val in: Inlet[InboundEnvelope] = Inlet("InboundTestStage.in")
148 val out: Outlet[InboundEnvelope] = Outlet("InboundTestStage.out")
149 override val shape: FlowShape[InboundEnvelope, InboundEnvelope] = FlowShape(in, out)
150
151 override def createLogic(inheritedAttributes: Attributes) =
152 new TimerGraphStageLogic(shape) with InHandler with OutHandler with StageLogging {
```

test/scala/akka/remote/artery/RemoteActorRefProviderSpec.scala, line 19 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: RemoteActorRefProviderSpec()
File: test/scala/akka/remote/artery/RemoteActorRefProviderSpec.scala:19
Taint Flags:

```
16 system.actorOf(TestActors.echoActorProps, "echo")
17
18 val systemB = newRemoteSystem()
19 val addressB = address(systemB)
20 systemB.actorOf(TestActors.echoActorProps, "echo")
21
22 "RemoteActorRefProvider" must {
```

test/scala/akka/remote/artery/RemoteActorRefProviderSpec.scala, line 20 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: RemoteActorRefProviderSpec()
File: test/scala/akka/remote/artery/RemoteActorRefProviderSpec.scala:20
Taint Flags:

```
17
18 val systemB = newRemoteSystem()
19 val addressB = address(systemB)
20 systemB.actorOf(TestActors.echoActorProps, "echo")
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteActorRefProviderSpec.scala, line 20 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
21
22 "RemoteActorRefProvider" must {
23
```

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/AeronSink.scala, line 104 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: in
Enclosing Method: AeronSink()
File: main/scala/akka/remote/artery/aeron/AeronSink.scala:104
Taint Flags:

```
101 import TaskRunner._
102
103 val in: Inlet[EnvelopeBuffer] = Inlet("AeronSink")
104 override val shape: SinkShape[EnvelopeBuffer] = SinkShape(in)
105
106 override def createLogicAndMaterializedValue(inheritedAttributes: Attributes): (GraphStageLogic, Future[Done]) = {
107 val completed = Promise[Done]()
```

test/scala/akka/remote/artery/aeron/AeronSinkSpec.scala, line 34 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: driver
Enclosing Method: AeronSinkSpec()
File: test/scala/akka/remote/artery/aeron/AeronSinkSpec.scala:34
Taint Flags:

```
31
32 val aeron = {
33 val ctx = new Aeron.Context
34 ctx.aeronDirectoryName(driver.aeronDirectoryName)
35 Aeron.connect(ctx)
36 }
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.aeron

test/scala/akka/remote/artery/aeron/AeronSinkSpec.scala, line 34 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
37
```

main/scala/akka/remote/artery/aeron/AeronSource.scala, line 96 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: out
Enclosing Method: AeronSource()
File: main/scala/akka/remote/artery/aeron/AeronSource.scala:96
Taint Flags:

```
93 import TaskRunner._
94
95 val out: Outlet[EnvelopeBuffer] = Outlet("AeronSource")
96 override val shape: SourceShape[EnvelopeBuffer] = SourceShape(out)
97
98 override def createLogicAndMaterializedValue(inheritedAttributes: Attributes) = {
99 val logic = new GraphStageLogic(shape) with OutHandler with AeronLifecycle with StageLogging {
```

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 125 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: createIdleStrategy
Enclosing Method: TaskRunner()
File: main/scala/akka/remote/artery/aeron/TaskRunner.scala:125
Taint Flags:

```
122 private[this] val tasks = new ArrayBag[Task]
123 private[this] val shutdown = Promise[Done]()
124
125 private val idleStrategy = createIdleStrategy(idleCpuLevel)
126 private var reset = false
127
128 def start(): Unit = {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.aeron

test/scala/akka/remote/artery/aeron/AeronSinkSpec.scala, line 40 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: idleCpuLevel
Enclosing Method: AeronSinkSpec()
File: test/scala/akka/remote/artery/aeron/AeronSinkSpec.scala:40
Taint Flags:

```
37
38 val idleCpuLevel = 5
39 val taskRunner = {
40 val r = new TaskRunner(system.asInstanceOf[ExtendedActorSystem], idleCpuLevel)
41 r.start()
42 r
43 }
```

main/scala/akka/remote/artery/aeron/AeronSink.scala, line 42 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: TimerCheckPeriod
Enclosing Method: AeronSink()
File: main/scala/akka/remote/artery/aeron/AeronSink.scala:42
Taint Flags:

```
39 final class PublicationClosedException(msg: String) extends RuntimeException(msg) with NoStackTrace
40
41 private val TimerCheckPeriod = 1 << 13 // 8192
42 private val TimerCheckMask = TimerCheckPeriod - 1
43
44 private final class OfferTask(
45 pub: Publication,
```

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 29 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 29 (Code Correctness: Constructor Invokes Overridable Function) — Low

Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: adjustedMax
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:29
Taint Flags:

```
26 private[remote] final class TopHeavyHitters[T >: Null](val max: Int)(implicit classTag: ClassTag[T]) { self =>
27
28  private val adjustedMax = if (max == 0) 1 else max // need at least one
29  require(
30  (adjustedMax & (adjustedMax - 1)) == 0,
31  "Maximum numbers of heavy hitters should be in form of 2^k for any natural k")
32
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 29 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: adjustedMax
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:29
Taint Flags:

```
26 private[remote] final class TopHeavyHitters[T >: Null](val max: Int)(implicit classTag: ClassTag[T]) { self =>
27
28  private val adjustedMax = if (max == 0) 1 else max // need at least one
29  require(
30  (adjustedMax & (adjustedMax - 1)) == 0,
31  "Maximum numbers of heavy hitters should be in form of 2^k for any natural k")
32
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 33 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 33 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink: FunctionCall: adjustedMax
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:33
Taint Flags:

```
30 (adjustedMax & (adjustedMax - 1)) == 0,
31 "Maximum numbers of heavy hitters should be in form of 2^k for any natural k")
32
33 val capacity = adjustedMax * 2
34 val mask = capacity - 1
35
36 import TopHeavyHitters._
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 50 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: adjustedMax
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:50
Taint Flags:

```
47 private[this] val weights: Array[Long] = new Array(capacity)
48
49 // Heap structure containing indices to slots in the hashmap
50 private[this] val heap: Array[Int] = Array.fill(adjustedMax)(-1)
51
52 /*
53 * Invariants (apart from heap and hashmap invariants):
```

test/scala/akka/remote/artery/compress/CompressionIntegrationSpec.scala, line 43 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: commonConfig
Enclosing Method: CompressionIntegrationSpec()
File: test/scala/akka/remote/artery/compress/CompressionIntegrationSpec.scala:43
Taint Flags:

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.compress

test/scala/akka/remote/artery/compress/CompressionIntegrationSpec.scala, line 43 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
}

class CompressionIntegrationSpec
 extends ArteryMultiNodeSpec(CompressionIntegrationSpec.commonConfig)
 with ImplicitSender {

 val systemB = newRemoteSystem(name = Some("systemB"))
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 34 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: capacity
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:34
Taint Flags:

```
 "Maximum numbers of heavy hitters should be in form of 2^k for any natural k")

 val capacity = adjustedMax * 2
 val mask = capacity - 1

 import TopHeavyHitters._

```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 40 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: capacity
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:40
Taint Flags:

```

 // Contains the hash value for each entry in the hashmap. Used for quicker lookups (equality check can be avoided
 // if hashes don't match)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 40 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
private[this] val hashes: Array[Int] = new Array(capacity)
// Actual stored elements in the hashmap
private[this] val items: Array[T] = Array.ofDim[T](capacity)
// Index of stored element in the associated heap
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 42 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: capacity
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:42
Taint Flags:

```
// if hashes don't match)
private[this] val hashes: Array[Int] = new Array(capacity)
// Actual stored elements in the hashmap
private[this] val items: Array[T] = Array.ofDim[T](capacity)
// Index of stored element in the associated heap
private[this] val heapIndex: Array[Int] = Array.fill(capacity)(-1)
// Weights associated with an entry in the hashmap. Used to maintain the heap property and give easy access to low
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 44 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: capacity
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:44
Taint Flags:

```
// Actual stored elements in the hashmap
private[this] val items: Array[T] = Array.ofDim[T](capacity)
// Index of stored element in the associated heap
private[this] val heapIndex: Array[Int] = Array.fill(capacity)(-1)
// Weights associated with an entry in the hashmap. Used to maintain the heap property and give easy access to low
// weight entries
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 44 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
47 private[this] val weights: Array[Long] = new Array(capacity)
```

main/scala/akka/remote/artery/compress/TopHeavyHitters.scala, line 47 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: capacity
Enclosing Method: TopHeavyHitters()
File: main/scala/akka/remote/artery/compress/TopHeavyHitters.scala:47
Taint Flags:

```
44 private[this] val heapIndex: Array[Int] = Array.fill(capacity)(-1)
45 // Weights associated with an entry in the hashmap. Used to maintain the heap property and give easy access to low
46 // weight entries
47 private[this] val weights: Array[Long] = new Array(capacity)
48
49 // Heap structure containing indices to slots in the hashmap
50 private[this] val heap: Array[Int] = Array.fill(adjustedMax)(-1)
```

test/scala/akka/remote/artery/compress/HandshakeShouldDropCompressionTableSpec.scala, line 39 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: commonConfig
Enclosing Method: HandshakeShouldDropCompressionTableSpec()
File: test/scala/akka/remote/artery/compress/HandshakeShouldDropCompressionTableSpec.scala:39
Taint Flags:

```
36 }
37
38 class HandshakeShouldDropCompressionTableSpec
39 extends ArteryMultiNodeSpec(HandshakeShouldDropCompressionTableSpec.commonConfig)
40 with ImplicitSender
41 with BeforeAndAfter {
42
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 78 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: config
Enclosing Method: TlsTcpSpec()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:78
Taint Flags:

```
75 }
76
77 abstract class TlsTcpSpec(config: Config)
78  extends ArteryMultiNodeSpec(config.withFallback(TlsTcpSpec.config))
79  with ImplicitSender
80  with Matchers {
81
```

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 244 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: addressB
Enclosing Method: TlsTcpWithActorSystemSetupSpec()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:244
Taint Flags:

```
241
242  val systemB = newRemoteSystem(name = Some("systemB"), setup = Some(ActorSystemSetup(sslProviderSetup)))
243  val addressB = address(systemB)
244  val rootB = RootActorPath(addressB)
245
246  "Artery with TLS/TCP with SSLEngineProvider defined via Setup" must {
247  "use the right SSLEngineProvider" in {
```

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 52 (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.remote.artery.tcp

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 52 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: sslEngineConfig
Enclosing Method: ConfigSSLEngineProvider()
File: main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala:52
Taint Flags:

```
49 val SSLKeyStorePassword: String = config.getString("key-store-password")
50 val SSLKeyPassword: String = config.getString("key-password")
51 val SSLTrustStorePassword: String = config.getString("trust-store-password")
52 val SSLEnabledAlgorithms: Set[String] = sslEngineConfig.SSLEnabledAlgorithms
53 val SSLProtocol: String = sslEngineConfig.SSLProtocol
54 val SSLRandomNumberGenerator: String = sslEngineConfig.SSLRandomNumberGenerator
55 val SSLRequireMutualAuthentication: Boolean = sslEngineConfig.SSLRequireMutualAuthentication
```

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 53 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslEngineConfig
Enclosing Method: ConfigSSLEngineProvider()
File: main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala:53
Taint Flags:

```
50 val SSLKeyPassword: String = config.getString("key-password")
51 val SSLTrustStorePassword: String = config.getString("trust-store-password")
52 val SSLEnabledAlgorithms: Set[String] = sslEngineConfig.SSLEnabledAlgorithms
53 val SSLProtocol: String = sslEngineConfig.SSLProtocol
54 val SSLRandomNumberGenerator: String = sslEngineConfig.SSLRandomNumberGenerator
55 val SSLRequireMutualAuthentication: Boolean = sslEngineConfig.SSLRequireMutualAuthentication
56 val HostnameVerification: Boolean = sslEngineConfig.HostnameVerification
```

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 54 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslEngineConfig
Enclosing Method: ConfigSSLEngineProvider()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 54 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala:54
Taint Flags:

```
51  val SSLTrustStorePassword: String = config.getString("trust-store-password")
52  val SSLEnabledAlgorithms: Set[String] = sslEngineConfig.SSLEnabledAlgorithms
53  val SSLProtocol: String = sslEngineConfig.SSLProtocol
54  val SSLRandomNumberGenerator: String = sslEngineConfig.SSLRandomNumberGenerator
55  val SSLRequireMutualAuthentication: Boolean = sslEngineConfig.SSLRequireMutualAuthentication
56  val HostnameVerification: Boolean = sslEngineConfig.HostnameVerification
57
```

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 55 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslEngineConfig
Enclosing Method: ConfigSSLEngineProvider()
File: main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala:55
Taint Flags:

```
52  val SSLEnabledAlgorithms: Set[String] = sslEngineConfig.SSLEnabledAlgorithms
53  val SSLProtocol: String = sslEngineConfig.SSLProtocol
54  val SSLRandomNumberGenerator: String = sslEngineConfig.SSLRandomNumberGenerator
55  val SSLRequireMutualAuthentication: Boolean = sslEngineConfig.SSLRequireMutualAuthentication
56  val HostnameVerification: Boolean = sslEngineConfig.HostnameVerification
57
58  private lazy val sslContext: SSLContext = {
```

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 56 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslEngineConfig
Enclosing Method: ConfigSSLEngineProvider()
File: main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala:56
Taint Flags:

```
53  val SSLProtocol: String = sslEngineConfig.SSLProtocol
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp

main/scala/akka/remote/artery/tcp/ConfigSSLEngineProvider.scala, line 56 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
val SSLRandomNumberGenerator: String = sslEngineConfig.SSLRandomNumberGenerator
val SSLRequireMutualAuthentication: Boolean = sslEngineConfig.SSLRequireMutualAuthentication
val HostnameVerification: Boolean = sslEngineConfig.HostnameVerification

private lazy val sslContext: SSLContext = {
// log hostname verification warning once
```

main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala, line 86 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: firstConnectionFlow
Enclosing Method: ArteryTcpTransport()
File: main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala:86
Taint Flags:

```
// may change when inbound streams are restarted
@volatile private var serverBinding: Option[ServerBinding] = None
private val firstConnectionFlow = Promise[Flow[ByteString, ByteString, NotUsed]]()
@volatile private var inboundConnectionFlow: Future[Flow[ByteString, ByteString, NotUsed]] =
firstConnectionFlow.future

private val sslEngineProvider: OptionVal[SSLEngineProvider] =
```

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 243 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: TlsTcpWithActorSystemSetupSpec()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:243
Taint Flags:

```
})

val systemB = newRemoteSystem(name = Some("systemB"), setup = Some(ActorSystemSetup(sslProviderSetup)))
val addressB = address(systemB)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 243 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
val rootB = RootActorPath(addressB)

"Artery with TLS/TCP with SSLEngineProvider defined via Setup" must {
```

test/scala/akka/remote/artery/tcp/TcpFramingSpec.scala, line 31 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: rndSeed
Enclosing Method: TcpFramingSpec()
File: test/scala/akka/remote/artery/tcp/TcpFramingSpec.scala:31
Taint Flags:

```
(1 to numberOfFrames).foldLeft(ByteString.empty)((acc, _) => acc ++ encodeFrameHeader(payload5.size) ++ payload5)

private val rndSeed = System.currentTimeMillis()
private val rnd = new Random(rndSeed)

private def rechunk(bytes: ByteString): Iterator[ByteString] = {
var remaining = bytes
```

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 242 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslProviderSetup
Enclosing Method: TlsTcpWithActorSystemSetupSpec()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:242
Taint Flags:

```
delegate.verifyServerSession(hostname, session)
})

val systemB = newRemoteSystem(name = Some("systemB"), setup = Some(ActorSystemSetup(sslProviderSetup)))
val addressB = address(systemB)
val rootB = RootActorPath(addressB)

```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 183 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: temporaryDirectory
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:183
Taint Flags:

```
180 """
181
182 val temporaryDirectory: Path = Files.createTempDirectory("akka-remote-rotating-keys-spec")
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 184 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: temporaryDirectory
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:184
Taint Flags:

```
181
182 val temporaryDirectory: Path = Files.createTempDirectory("akka-remote-rotating-keys-spec")
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 185 (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.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 185 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: temporaryDirectory
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:185
Taint Flags:

```
182 val temporaryDirectory: Path = Files.createTempDirectory("akka-remote-rotating-keys-spec")
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
188 akka.remote.artery.ssl.rotating-keys-engine {
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: temporaryDirectory
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:186
Taint Flags:

```
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
188 akka.remote.artery.ssl.rotating-keys-engine {
189 key-file = ${temporaryDirectory.toFile.getAbsolutePath}/tls.key
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: temporaryDirectory
Enclosing Method: RotatingKeysSSLEngineProviderSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:186
Taint Flags:

```
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
188 akka.remote.artery.ssl.rotating-keys-engine {
189 key-file = ${temporaryDirectory.toFile.getAbsolutePath}/tls.key
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: temporaryDirectory
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:186
Taint Flags:

```
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
188 akka.remote.artery.ssl.rotating-keys-engine {
189 key-file = ${temporaryDirectory.toFile.getAbsolutePath}/tls.key
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: baseConfig
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
170 """
171
172 val resourcesConfig: String = baseConfig +
173 s"""
174 akka.remote.artery.ssl.rotating-keys-engine {
175 key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: baseConfig
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:186
Taint Flags:

```
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
188 akka.remote.artery.ssl.rotating-keys-engine {
189 key-file = ${temporaryDirectory.toFile.getAbsolutePath}/tls.key
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 300 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslProviderSetup
Enclosing Method: RemoteSystem()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:300
Taint Flags:

```
297 sys => new ProbedSSLEngineProvider(sys, sslContextRef, sslProviderServerProbe, sslProviderClientProbe))
298
299 val actorSystem =
300 newRemoteSystem(Some(configString), Some(name), Some(ActorSystemSetup(sslProviderSetup)))
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 300 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
301 val remoteAddress = address(actorSystem)
302 val rootActorPath = RootActorPath(remoteAddress)
303
```

main/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProvider.scala, line 66 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: sslEngineConfig
Enclosing Method: RotatingKeysSSLEngineProvider()
File: main/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProvider.scala:66
Taint Flags:

```
63 import sslEngineConfig._
64
65 // build a PRNG (created once, reused on every instance of SSLContext
66 private val rng: SecureRandom = SecureRandomFactory.createSecureRandom(SSLRandomNumberGenerator, log)
67
68 // handle caching
69 @volatile private var cachedContext: Option[CachedContext] = None
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: arteryNode001Id
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
170 """
171
172 val resourcesConfig: String = baseConfig +
173 s"""
174 akka.remote.artery.ssl.rotating-keys-engine {
175 key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: arteryNode001Id
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
170 """
171
172 val resourcesConfig: String = baseConfig +
173 s"""
174 akka.remote.artery.ssl.rotating-keys-engine {
175 key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 302 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteAddress
Enclosing Method: RemoteSystem()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:302
Taint Flags:

```
299 val actorSystem =
300 newRemoteSystem(Some(configString), Some(name), Some(ActorSystemSetup(sslProviderSetup)))
301 val remoteAddress = address(actorSystem)
302 val rootActorPath = RootActorPath(remoteAddress)
303
304 }
305
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (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.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: cacheTtlInSeconds
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
170 """
171
172 val resourcesConfig: String = baseConfig +
173 s"""
174 akka.remote.artery.ssl.rotating-keys-engine {
175 key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cacheTtlInSeconds
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:186
Taint Flags:

```
183 val keyLocation = new File(temporaryDirectory.toFile, "tls.key")
184 val certLocation = new File(temporaryDirectory.toFile, "tls.crt")
185 val cacertLocation = new File(temporaryDirectory.toFile, "ca.crt")
186 val tempFileConfig: String = baseConfig +
187 s"""
188 akka.remote.artery.ssl.rotating-keys-engine {
189 key-file = ${temporaryDirectory.toFile.getAbsolutePath}/tls.key
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 301 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: actorSystem
Enclosing Method: RemoteSystem()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 301 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:301
Taint Flags:

```
298
299 val actorSystem =
300 newRemoteSystem(Some(configString), Some(name), Some(ActorSystemSetup(sslProviderSetup)))
301 val remoteAddress = address(actorSystem)
302 val rootActorPath = RootActorPath(remoteAddress)
303
304 }
```

Package: akka.remote.classic

test/scala/akka/remote/classic/RemoteDeploymentAllowListSpec.scala, line 133 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: RemoteDeploymentAllowListSpec()
File: test/scala/akka/remote/classic/RemoteDeploymentAllowListSpec.scala:133
Taint Flags:

```
130 }
131 //#allow-list-config
132 """).withFallback(system.settings.config).resolve()
133 val remoteSystem = ActorSystem("remote-sys", conf)
134
135 override def atStartup() = {
136 muteSystem(system)
```

test/scala/akka/remote/classic/RemoteDeploymentAllowListSpec.scala, line 109 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cfg
Enclosing Method: RemoteDeploymentAllowListSpec()
File: test/scala/akka/remote/classic/RemoteDeploymentAllowListSpec.scala:109
Taint Flags:

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemoteDeploymentAllowListSpec.scala, line 109 (Code Correctness: Constructor Invokes Overridable Function) — Low

```

@nowarn("msg=deprecated")
class RemoteDeploymentAllowListSpec
 extends AkkaSpec(RemoteDeploymentAllowListSpec.cfg)
 with ImplicitSender
 with DefaultTimeout {

```

test/scala/akka/remote/classic/RemotingSpec.scala, line 161 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSystem
Enclosing Method: RemotingSpec()
File: test/scala/akka/remote/classic/RemotingSpec.scala:161
Taint Flags:

```
 sys.asInstanceOf[ExtendedActorSystem].provider.asInstanceOf[RemoteActorRefProvider].deployer.deploy(d)
 }

 val remote = remoteSystem.actorOf(Props[Echo2](), "echo")

 val here = RARP(system).provider.resolveActorRef("akka.test://remote-sys@localhost:12346/user/echo")

```

test/scala/akka/remote/classic/RemotingSpec.scala, line 149 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: conf
Enclosing Method: RemotingSpec()
File: test/scala/akka/remote/classic/RemotingSpec.scala:149
Taint Flags:

```
 maximum-payload-bytes = 48000 bytes
 }
 """).withFallback(system.settings.config).resolve()
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 149 (Code Correctness: Constructor Invokes Overridable Function)	Low

```
149 val remoteSystem = ActorSystem("remote-sys", conf)
150
151 for ((name, proto) <- Seq("/gonk" -> "tcp", "/roghtaar" -> "ssl.tcp"))
152 deploy(system, Deploy(name, scope = RemoteScope(getOtherAddress(remoteSystem, proto))))
```

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 95 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSystem
Enclosing Method: RemoteWatcherSpec()
File: test/scala/akka/remote/classic/RemoteWatcherSpec.scala:95
Taint Flags:

```
92 override def expectedTestDuration = 2.minutes
93
94 val remoteSystem = ActorSystem("RemoteSystem", system.settings.config)
95 val remoteAddress = remoteSystem.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress
96 @nowarn
97 def remoteAddressUid = AddressUidExtension(remoteSystem).addressUid
98
```

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 99 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteSystem
Enclosing Method: RemoteWatcherSpec()
File: test/scala/akka/remote/classic/RemoteWatcherSpec.scala:99
Taint Flags:

```
96 @nowarn
97 def remoteAddressUid = AddressUidExtension(remoteSystem).addressUid
98
99 Seq(system, remoteSystem).foreach(
100 muteDeadLetters(
101 akka.remote.transport.AssociationHandle.Disassociated.getClass,
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 99 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
102  akka.remote.transport.ActorTransportAdapter.DisassociateUnderlying.getClass)(_))
```

test/scala/akka/remote/classic/ActorsLeakSpec.scala, line 72 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: config
Enclosing Method: ActorsLeakSpec()
File: test/scala/akka/remote/classic/ActorsLeakSpec.scala:72
Taint Flags:

```
69
70 }
71
72 class ActorsLeakSpec extends AkkaSpec(ActorsLeakSpec.config) with ImplicitSender {
73  import ActorsLeakSpec._
74
75  "Remoting" must {
```

test/scala/akka/remote/classic/RemotingSpec.scala, line 138 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: cfg
Enclosing Method: RemotingSpec()
File: test/scala/akka/remote/classic/RemotingSpec.scala:138
Taint Flags:

```
135 }
136
137 @nowarn
138 class RemotingSpec extends AkkaSpec(RemotingSpec.cfg) with ImplicitSender with DefaultTimeout {
139
140  import RemotingSpec._
141
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 108 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: remoteAddressUid
Enclosing Method: RemoteWatcherSpec()
File: test/scala/akka/remote/classic/RemoteWatcherSpec.scala:108
Taint Flags:

```
105  shutdown(remoteSystem)
106  }
107
108  val heartbeatRspB = HeartbeatRsp(remoteAddressUid)
109
110  def createRemoteActor(props: Props, name: String): InternalActorRef = {
111  remoteSystem.actorOf(props, name)
```

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 78 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: configA
Enclosing Method: ThrottlerTransportAdapterSpec()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:78
Taint Flags:

```
75 }
76
77 @nowarn("msg=deprecated")
78 class ThrottlerTransportAdapterSpec extends AkkaSpec(configA) with ImplicitSender with DefaultTimeout {
79
80  val systemB = ActorSystem("systemB", system.settings.config)
81  val remote = systemB.actorOf(Props[Echo](), "echo")
```

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 107 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 107 (Code Correctness: Constructor Invokes Overridable Function) — Low

Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: baseConfig
Enclosing Method: SystemMessageDeliveryStressTest()
File: test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala:107
Taint Flags:

```
104
105 @nowarn("msg=deprecated")
106 abstract class SystemMessageDeliveryStressTest(msg: String, cfg: String)
107  extends AkkaSpec(ConfigFactory.parseString(cfg).withFallback(SystemMessageDeliveryStressTest.baseConfig))
108  with ImplicitSender
109  with DefaultTimeout {
110  import SystemMessageDeliveryStressTest._
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 105 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: addressB
Enclosing Method: AkkaProtocolStressTest()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:105
Taint Flags:

```
102  }), "echo")
103
104  val addressB = systemB.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress
105  val rootB = RootActorPath(addressB)
106  val here = {
107  system.actorSelection(rootB / "user" / "echo") ! Identify(None)
108  expectMsgType[ActorIdentity].ref.get
```

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink: FunctionCall: codec
Enclosing Method: AkkaProtocolSpec()
File: test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala:85
Taint Flags:

```
82
83 val testMsg =
84 WireFormats.SerializedMessage.newBuilder().setSerializerId(0).setMessage(PByteString.copyFromUtf8("foo")).build
85 val testEnvelope = codec.constructMessage(localAkkaAddress, testActor, testMsg, OptionVal.None)
86 val testMsgPdu: ByteString = codec.constructPayload(testEnvelope)
87
88 def testHeartbeat = InboundPayload(codec.constructHeartbeat)
```

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 86 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: codec
Enclosing Method: AkkaProtocolSpec()
File: test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala:86
Taint Flags:

```
83 val testMsg =
84 WireFormats.SerializedMessage.newBuilder().setSerializerId(0).setMessage(PByteString.copyFromUtf8("foo")).build
85 val testEnvelope = codec.constructMessage(localAkkaAddress, testActor, testMsg, OptionVal.None)
86 val testMsgPdu: ByteString = codec.constructPayload(testEnvelope)
87
88 def testHeartbeat = InboundPayload(codec.constructHeartbeat)
89 def testPayload = InboundPayload(testMsgPdu)
```

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: testMsg
Enclosing Method: AkkaProtocolSpec()
File: test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala:85
Taint Flags:

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
82
83  val testMsg =
84  WireFormats.SerializedMessage.newBuilder().setSerializerId(0).setMessage(PByteString.copyFromUtf8("foo")).build
85  val testEnvelope = codec.constructMessage(localAkkaAddress, testActor, testMsg, OptionVal.None)
86  val testMsgPdu: ByteString = codec.constructPayload(testEnvelope)
87
88  def testHeartbeat = InboundPayload(codec.constructHeartbeat)
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 81 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: ThrottlerTransportAdapterSpec()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:81
Taint Flags:

```
78 class ThrottlerTransportAdapterSpec extends AkkaSpec(configA) with ImplicitSender with DefaultTimeout {
79
80  val systemB = ActorSystem("systemB", system.settings.config)
81  val remote = systemB.actorOf(Props[Echo](), "echo")
82
83  val rootB = RootActorPath(systemB.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress)
84  val here = {
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 83 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: ThrottlerTransportAdapterSpec()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:83
Taint Flags:

```
80  val systemB = ActorSystem("systemB", system.settings.config)
81  val remote = systemB.actorOf(Props[Echo](), "echo")
82
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 83 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
val rootB = RootActorPath(systemB.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress)
val here = {
system.actorSelection(rootB / "user" / "echo") ! Identify(None)
expectMsgType[ActorIdentity].ref.get
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 51 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: PingPacketSize
Enclosing Method: ThrottlerTransportAdapterSpec()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:51
Taint Flags:

```
val PingPacketSize = 148
val MessageCount = 30
val BytesPerSecond = 500
val TotalTime: Long = (MessageCount * PingPacketSize) / BytesPerSecond

class ThrottlingTester(remote: ActorRef, controller: ActorRef) extends Actor {
var messageCount = MessageCount
```

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: localAkkaAddress
Enclosing Method: AkkaProtocolSpec()
File: test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala:85
Taint Flags:

```

val testMsg =
WireFormats.SerializedMessage.newBuilder().setSerializerId(0).setMessage(PByteString.copyFromUtf8("foo")).build
val testEnvelope = codec.constructMessage(localAkkaAddress, testActor, testMsg, OptionVal.None)
val testMsgPdu: ByteString = codec.constructPayload(testEnvelope)

```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
def testHeartbeat = InboundPayload(codec.constructHeartbeat)
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 107 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: rootB
Enclosing Method: AkkaProtocolStressTest()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:107
Taint Flags:

```
104 val addressB = systemB.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress
105 val rootB = RootActorPath(addressB)
106 val here = {
107 system.actorSelection(rootB / "user" / "echo") ! Identify(None)
108 expectMsgType[ActorIdentity].ref.get
109 }
110
```

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 26 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: msgCount
Enclosing Method: SystemMessageDeliveryStressTest()
File: test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala:26
Taint Flags:

```
23 val burstSize = 100
24 val burstDelay = 500.millis
25
26 val baseConfig: Config = ConfigFactory.parseString(s"""
27 akka {
28 #loglevel = DEBUG
29 remote.artery.enabled = false
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 51 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: MessageCount
Enclosing Method: ThrottlerTransportAdapterSpec()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:51
Taint Flags:

```
48 val PingPacketSize = 148
49 val MessageCount = 30
50 val BytesPerSecond = 500
51 val TotalTime: Long = (MessageCount * PingPacketSize) / BytesPerSecond
52
53 class ThrottlingTester(remote: ActorRef, controller: ActorRef) extends Actor {
54 var messageCount = MessageCount
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 85 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: rootB
Enclosing Method: ThrottlerTransportAdapterSpec()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:85
Taint Flags:

```
82
83 val rootB = RootActorPath(systemB.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress)
84 val here = {
85 system.actorSelection(rootB / "user" / "echo") ! Identify(None)
86 expectMsgType[ActorIdentity].ref.get
87 }
88
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 98 (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.remote.classic.transport

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 98 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: AkkaProtocolStressTest()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:98
Taint Flags:

```
95 class AkkaProtocolStressTest extends AkkaSpec(configA) with ImplicitSender with DefaultTimeout {
96
97  val systemB = ActorSystem("systemB", system.settings.config)
98  val remote = systemB.actorOf(Props(new Actor {
99  def receive = {
100  case seq: Int => sender() ! seq
101  }
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 104 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: systemB
Enclosing Method: AkkaProtocolStressTest()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:104
Taint Flags:

```
101  }
102  }), "echo")
103
104  val addressB = systemB.asInstanceOf[ExtendedActorSystem].provider.getDefaultAddress
105  val rootB = RootActorPath(addressB)
106  val here = {
107  system.actorSelection(rootB / "user" / "echo") ! Identify(None)
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 51 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: BytesPerSecond
Enclosing Method: ThrottlerTransportAdapterSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 51 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:51
Taint Flags:

```
48 val PingPacketSize = 148
49 val MessageCount = 30
50 val BytesPerSecond = 500
51 val TotalTime: Long = (MessageCount * PingPacketSize) / BytesPerSecond
52
53 class ThrottlingTester(remote: ActorRef, controller: ActorRef) extends Actor {
54 var messageCount = MessageCount
```

test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala, line 86 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: testEnvelope
Enclosing Method: AkkaProtocolSpec()
File: test/scala/akka/remote/classic/transport/AkkaProtocolSpec.scala:86
Taint Flags:

```
83 val testMsg =
84 WireFormats.SerializedMessage.newBuilder().setSerializerId(0).setMessage(PByteString.copyFromUtf8("foo")).build
85 val testEnvelope = codec.constructMessage(localAkkaAddress, testActor, testMsg, OptionVal.None)
86 val testMsgPdu: ByteString = codec.constructPayload(testEnvelope)
87
88 def testHeartbeat = InboundPayload(codec.constructHeartbeat)
89 def testPayload = InboundPayload(testMsgPdu)
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 95 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: configA
Enclosing Method: AkkaProtocolStressTest()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:95
Taint Flags:

```
92
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 95 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
93 }
94
95 class AkkaProtocolStressTest extends AkkaSpec(configA) with ImplicitSender with DefaultTimeout {
96
97  val systemB = ActorSystem("systemB", system.settings.config)
98  val remote = systemB.actorOf(Props(new Actor {
```

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: NotUsedManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ActorRefManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ActorIdentityManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
private val StatusReplyErrorExceptionManifest = "SE"
private val StatusReplyAckManifest = "SA"

private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: RemoteRouterConfigManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
private val StatusReplyErrorExceptionManifest = "SE"
private val StatusReplyAckManifest = "SA"

private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

```
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/SystemMessageSerializationSpec.scala, line 20 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: serializationTestOverrides
Enclosing Method: SystemMessageSerializationSpec()
File: test/scala/akka/remote/serialization/SystemMessageSerializationSpec.scala:20
Taint Flags:

```
17 """
18 """
19
20 val testConfig = ConfigFactory.parseString(serializationTestOverrides).withFallback(AkkaSpec.testConf)
21
22 class TestException(msg: String) extends RuntimeException(msg) with JavaSerializable {
23 override def equals(other: Any): Boolean = other match {
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: StatusReplyErrorExceptionManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: BroadcastPoolManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: OptionalManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (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.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: RemoteScopeManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: RemoteWatcherHBRespManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ActorInitializationExceptionManifest
Enclosing Method: MiscMessageSerializer()

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ConfigManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: StatusReplySuccessManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

```
private val StatusReplyAckManifest = "SA"

private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: StatusReplyErrorMessageManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
private val StatusReplyErrorExceptionManifest = "SE"
private val StatusReplyAckManifest = "SA"

private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: StatusSuccessManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
private val StatusReplyErrorExceptionManifest = "SE"
private val StatusReplyAckManifest = "SA"

private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala, line 24 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: serializationTestOverrides
Enclosing Method: PrimitivesSerializationSpec()
File: test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala:24
Taint Flags:

```
21 object PrimitivesSerializationSpec {
22  val serializationTestOverrides = ""
23
24  val testConfig = ConfigFactory.parseString(serializationTestOverrides).withFallback(AkkaSpec.testConf)
25 }
26
27 @deprecated("Moved to akka.serialization.* in akka-actor", "2.6.0")
```

main/scala/akka/remote/serialization/PrimitiveSerializers.scala, line 45 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: delegate
Enclosing Method: StringSerializer()
File: main/scala/akka/remote/serialization/PrimitiveSerializers.scala:45
Taint Flags:

```
42  private val delegate = new akka.serialization.StringSerializer(system)
43
44  override def includeManifest: Boolean = delegate.includeManifest
45  override val identifier: Int = delegate.identifier
46  override def toBinary(o: AnyRef, buf: ByteBuffer): Unit = delegate.toBinary(o, buf)
47  override def fromBinary(buf: ByteBuffer, manifest: String): AnyRef = delegate.fromBinary(buf, manifest)
48  override def toBinary(o: AnyRef): Array[Byte] = delegate.toBinary(o)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: PoisonPillManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/PrimitiveSerializers.scala, line 58 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: delegate
Enclosing Method: ByteStringSerializer()
File: main/scala/akka/remote/serialization/PrimitiveSerializers.scala:58
Taint Flags:

```
55 private val delegate = new akka.serialization.ByteStringSerializer(system)
56
57 override def includeManifest: Boolean = delegate.includeManifest
58 override val identifier: Int = delegate.identifier
59 override def toBinary(o: AnyRef, buf: ByteBuffer): Unit = delegate.toBinary(o, buf)
60 override def fromBinary(buf: ByteBuffer, manifest: String): AnyRef = delegate.fromBinary(buf, manifest)
61 override def toBinary(o: AnyRef): Array[Byte] = delegate.toBinary(o)
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (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.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: DoneManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/PrimitiveSerializers.scala, line 32 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: delegate
Enclosing Method: IntSerializer()
File: main/scala/akka/remote/serialization/PrimitiveSerializers.scala:32
Taint Flags:

```
29 private val delegate = new akka.serialization.IntSerializer(system)
30
31 override def includeManifest: Boolean = delegate.includeManifest
32 override val identifier: Int = delegate.identifier
33 override def toBinary(o: AnyRef, buf: ByteBuffer): Unit = delegate.toBinary(o, buf)
34 override def fromBinary(buf: ByteBuffer, manifest: String): AnyRef = delegate.fromBinary(buf, manifest)
35 override def toBinary(o: AnyRef): Array[Byte] = delegate.toBinary(o)
```

main/scala/akka/remote/serialization/PrimitiveSerializers.scala, line 19 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: delegate
Enclosing Method: LongSerializer()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/PrimitiveSerializers.scala, line 19 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/serialization/PrimitiveSerializers.scala:19
Taint Flags:

```
16  private val delegate = new akka.serialization.LongSerializer(system)
17
18  override def includeManifest: Boolean = delegate.includeManifest
19  override val identifier: Int = delegate.identifier
20  override def toBinary(o: AnyRef, buf: ByteBuffer): Unit = delegate.toBinary(o, buf)
21  override def fromBinary(buf: ByteBuffer, manifest: String): AnyRef = delegate.fromBinary(buf, manifest)
22  override def toBinary(o: AnyRef): Array[Byte] = delegate.toBinary(o)
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: RandomPoolManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: DefaultResizerManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 65 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: actorSystemSettings
Enclosing Method: AllowJavaSerializationOffSpec()
File: test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala:65
Taint Flags:

```
62 }
63
64 class AllowJavaSerializationOffSpec
65  extends AkkaSpec(ActorSystem("AllowJavaSerializationOffSpec", AllowJavaSerializationOffSpec.actorSystemSettings)) {
66
67  import AllowJavaSerializationOffSpec._
68
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: StatusFailureManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala, line 28 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: testConfig
Enclosing Method: PrimitivesSerializationSpec()
File: test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala:28
Taint Flags:

```
25 }
26
27 @deprecated("Moved to akka.serialization.* in akka-actor", "2.6.0")
28 class PrimitivesSerializationSpec extends AkkaSpec(PrimitivesSerializationSpec.testConfig) {
29
30  val buffer = {
31  val b = ByteBuffer.allocate(4096)
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: TailChoppingPoolManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

Code Correctness: Constructor Invokes Overridable Function | Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 75 (Code Correctness: Constructor Invokes Overridable Function) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: testConfig
Enclosing Method: MiscMessageSerializerSpec()
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:75
Taint Flags:

```
72
73 }
74
75 class MiscMessageSerializerSpec extends AkkaSpec(MiscMessageSerializerSpec.testConfig) {
76  import MiscMessageSerializerSpec._
77
78  val ref = system.actorOf(Props.empty, "hello")
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: AddressManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 60 (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.remote.serialization

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 60 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: noJavaSerializationSystem
Enclosing Method: AllowJavaSerializationOffSpec()
File: test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala:60
Taint Flags:

```
57  }
58  }
59  """.stripMargin))
60  val noJavaSerializer = new DisabledJavaSerializer(noJavaSerializationSystem.asInstanceOf[ExtendedActorSystem])
61
62 }
63
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ScatterGatherPoolManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 37 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: serializationTestOverrides
Enclosing Method: MiscMessageSerializerSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 37 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:37
Taint Flags:

```
34  }
35  """
36
37  val testConfig = ConfigFactory.parseString(serializationTestOverrides).withFallback(AkkaSpec.testConf)
38
39  class TestException(msg: String, cause: Throwable) extends RuntimeException(msg, cause) {
40  def this(msg: String) = this(msg, null)
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: OptionManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: BalancingPoolManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: KillManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: IdentifyManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: RemoteWatcherHBManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
private val StatusReplyErrorExceptionManifest = "SE"
private val StatusReplyAckManifest = "SA"

private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 98 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: addedJavaSerializationSettings
Enclosing Method: AllowJavaSerializationOffSpec()
File: test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala:98
Taint Flags:

```
val dontAllowJavaSystem =
ActorSystem(
"addedJavaSerializationSystem",
ActorSystemSetup(addedJavaSerializationProgramaticallyButDisabledSettings, addedJavaSerializationSettings))

private def verifySerialization(sys: ActorSystem, obj: AnyRef): Unit = {
val serialization = SerializationExtension(sys)
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ThrowableManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 47 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: serializationSettings
Enclosing Method: AllowJavaSerializationOffSpec()
File: test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala:47
Taint Flags:

```
44 }
45 """)),
46 None)
47 val actorSystemSettings = ActorSystemSetup(bootstrapSettings, serializationSettings)
48
49 val noJavaSerializationSystem = ActorSystem(
50 "AllowJavaSerializationOffSpec" + "NoJavaSerialization",
```

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 47 (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.remote.serialization

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 47 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: bootstrapSettings
Enclosing Method: AllowJavaSerializationOffSpec()
File: test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala:47
Taint Flags:

```
44 }
45 """)),
46 None)
47 val actorSystemSettings = ActorSystemSetup(bootstrapSettings, serializationSettings)
48
49 val noJavaSerializationSystem = ActorSystem(
50 "AllowJavaSerializationOffSpec" + "NoJavaSerialization",
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: UniqueAddressManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 98 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: addedJavaSerializationProgramaticallyButDisabledSettings
Enclosing Method: AllowJavaSerializationOffSpec()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala, line 98 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: test/scala/akka/remote/serialization/AllowJavaSerializationOffSpec.scala:98
Taint Flags:

```
95  val dontAllowJavaSystem =
96  ActorSystem(
97  "addedJavaSerializationSystem",
98  ActorSystemSetup(addedJavaSerializationProgramaticallyButDisabledSettings, addedJavaSerializationSettings))
99
100  private def verifySerialization(sys: ActorSystem, obj: AnyRef): Unit = {
101  val serialization = SerializationExtension(sys)
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: StatusReplyAckManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: RoundRobinPoolManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: FromConfigManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339  IdentifyManifest -> deserializeIdentify,
340  ActorIdentityManifest -> deserializeActorIdentity,
341  StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: LocalScopeManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335  private val StatusReplyErrorExceptionManifest = "SE"
336  private val StatusReplyAckManifest = "SA"
337
338  private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

```
IdentifyManifest -> deserializeIdentify,
ActorIdentityManifest -> deserializeActorIdentity,
StatusSuccessManifest -> deserializeStatusSuccess,
```

main/scala/akka/remote/serialization/MiscMessageSerializer.scala, line 338 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: ThrowableNotSerializableExceptionManifest
Enclosing Method: MiscMessageSerializer()
File: main/scala/akka/remote/serialization/MiscMessageSerializer.scala:338
Taint Flags:

```
335 private val StatusReplyErrorExceptionManifest = "SE"
336 private val StatusReplyAckManifest = "SA"
337
338 private val fromBinaryMap = Map[String, Array[Byte] => AnyRef](
339 IdentifyManifest -> deserializeIdentify,
340 ActorIdentityManifest -> deserializeActorIdentity,
341 StatusSuccessManifest -> deserializeStatusSuccess,
```

Package: akka.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 78 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: FailureInjectorSchemeIdentifier
Enclosing Method: FailureInjectorTransportAdapter()
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:78
Taint Flags:

```
75 @volatile private var upstreamListener: Option[AssociationEventListener] = None
76 private[transport] val addressChaosTable = new ConcurrentHashMap[Address, GremlinMode]()
77
78 override val addedSchemeIdentifier = FailureInjectorSchemeIdentifier
79 protected def maximumOverhead = 0
80
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 78 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
81  override def managementCommand(cmd: Any): Future[Boolean] = cmd match {
```

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 174 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: constructControlMessagePdu
Enclosing Method: AkkaPduProtobufCodec()
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:174
Taint Flags:

```
171  constructControlMessagePdu(WireFormats.CommandType.ASSOCIATE, Some(handshakeInfo))
172  }
173
174  private val DISASSOCIATE = constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE, None)
175  private val DISASSOCIATE_SHUTTING_DOWN =
176  constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE_SHUTTING_DOWN, None)
177  private val DISASSOCIATE_QUARANTINED =
```

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 177 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: constructControlMessagePdu
Enclosing Method: AkkaPduProtobufCodec()
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:177
Taint Flags:

```
174  private val DISASSOCIATE = constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE, None)
175  private val DISASSOCIATE_SHUTTING_DOWN =
176  constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE_SHUTTING_DOWN, None)
177  private val DISASSOCIATE_QUARANTINED =
178  constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE_QUARANTINED, None)
179
180  override def constructDisassociate(info: AssociationHandle.DisassociateInfo): ByteString = info match {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 186 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: constructControlMessagePdu
Enclosing Method: AkkaPduProtobufCodec()
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:186
Taint Flags:

```
183 case AssociationHandle.Quarantined => DISASSOCIATE_QUARANTINED
184 }
185
186 override val constructHeartbeat: ByteString =
187 constructControlMessagePdu(WireFormats.CommandType.HEARTBEAT, None)
188
189 override def decodePdu(raw: ByteString): AkkaPdu = {
```

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: constructControlMessagePdu
Enclosing Method: AkkaPduProtobufCodec()
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:175
Taint Flags:

```
172 }
173
174 private val DISASSOCIATE = constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE, None)
175 private val DISASSOCIATE_SHUTTING_DOWN =
176 constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE_SHUTTING_DOWN, None)
177 private val DISASSOCIATE_QUARANTINED =
178 constructControlMessagePdu(WireFormats.CommandType.DISASSOCIATE_QUARANTINED, None)
```

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 176 (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.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 176 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: readHandlerPromise
Enclosing Method: FailureInjectorHandle()
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:176
Taint Flags:

```
173 @volatile private var upstreamListener: HandleEventListener = null
174
175 override val readHandlerPromise: Promise[HandleEventListener] = Promise()
176 readHandlerPromise.future.foreach { listener =>
177 upstreamListener = listener
178 wrappedHandle.readHandlerPromise.success(this)
179 }
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 44 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: TransportFailureDetectorConfig
Enclosing Method: AkkaProtocolSettings()
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:44
Taint Flags:

```
41 import akka.util.Helpers.ConfigOps
42
43 val TransportFailureDetectorConfig: Config = getConfig("akka.remote.classic.transport-failure-detector")
44 val TransportFailureDetectorImplementationClass: String =
45 TransportFailureDetectorConfig.getString("implementation-class")
46 val TransportHeartBeatInterval: FiniteDuration = {
47 TransportFailureDetectorConfig.getMillisDuration("heartbeat-interval")
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 46 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: TransportFailureDetectorConfig
Enclosing Method: AkkaProtocolSettings()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 46 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:46
Taint Flags:

```
43  val TransportFailureDetectorConfig: Config = getConfig("akka.remote.classic.transport-failure-detector")
44  val TransportFailureDetectorImplementationClass: String =
45  TransportFailureDetectorConfig.getString("implementation-class")
46  val TransportHeartBeatInterval: FiniteDuration = {
47  TransportFailureDetectorConfig.getMillisDuration("heartbeat-interval")
48  }.requiring(_ > Duration.Zero, "transport-failure-detector.heartbeat-interval must be > 0")
49
```

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 34 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: settings
Enclosing Method: TransportAdapters()
File: main/scala/akka/remote/transport/AbstractTransportAdapter.scala:34
Taint Flags:

```
31 class TransportAdapters(system: ExtendedActorSystem) extends Extension {
32  val settings = RARP(system).provider.remoteSettings
33
34  private val adaptersTable: Map[String, TransportAdapterProvider] = for ((name, fqn) <- settings.Adapters) yield {
35  name -> system.dynamicAccess
36  .createInstanceFor[TransportAdapterProvider](fqn, immutable.Seq.empty)
37  .recover({
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 127 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: AkkaOverhead
Enclosing Method: AkkaProtocolTransport()
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:127
Taint Flags:

```
124  statusPromise.future.mapTo[AkkaProtocolHandle]
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 127 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
}

override val maximumOverhead: Int = AkkaProtocolTransport.AkkaOverhead
protected def managerName = s"akkaprotocolmanager.${wrappedTransport.schemeIdentifier}${UniqueId.getAndIncrement}"
protected def managerProps = {
val wt = wrappedTransport
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 395 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: initHandshakeTimer
Enclosing Method: ProtocolStateActor()
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:395
Taint Flags:

```

case d: InboundUnassociated =>
d.wrappedHandle.readHandlerPromise.success(ActorHandleEventListener(self))
initHandshakeTimer()
startWith(WaitHandshake, d)

case _ => throw new IllegalStateException() // won't happen, compiler exhaustiveness check pleaser
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 401 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: initHandshakeTimer
Enclosing Method: ProtocolStateActor()
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:401
Taint Flags:

```
case _ => throw new IllegalStateException() // won't happen, compiler exhaustiveness check pleaser
}

initHandshakeTimer()
```

Code Correctness: Constructor Invokes Overridable Function	Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 401 (Code Correctness: Constructor Invokes Overridable Function)	Low

```

when(Closed) {

```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 114 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: AkkaScheme
Enclosing Method: AkkaProtocolTransport()
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:114
Taint Flags:

```
private val codec: AkkaPduCodec)
extends ActorTransportAdapter(wrappedTransport, system) {

override val addedSchemeIdentifier: String = AkkaScheme

override def managementCommand(cmd: Any): Future[Boolean] = wrappedTransport.managementCommand(cmd)

```

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 165 (Code Correctness: Constructor Invokes Overridable Function)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: Hostname
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:165
Taint Flags:

```
}

val BindHostname: String = getString("bind-hostname") match {
case "" => Hostname
case value => value
}
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 165 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
168
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 181 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: computeWPS
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:181
Taint Flags:

```
178
179 val SslSettings: Option[SSLSettings] = if (EnableSsl) Some(new SSLSettings(config.getConfig("security"))) else None
180
181 val ServerSocketWorkerPoolSize: Int = computeWPS(config.getConfig("server-socket-worker-pool"))
182
183 val ClientSocketWorkerPoolSize: Int = computeWPS(config.getConfig("client-socket-worker-pool"))
184
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 204 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: throwInvalidNettyVersion
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:204
Taint Flags:

```
201 try {
202 val segments: Array[String] = nettyVersion.split("[.-]")
203 if (segments.length < 3 || segments(0).toInt != 3 || segments(1).toInt != 10 || segments(2).toInt < 6)
204 throwInvalidNettyVersion()
205 } catch {
206 case _: NumberFormatException =>
207 throwInvalidNettyVersion()
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 207 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: throwInvalidNettyVersion
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:207
Taint Flags:

```
204 throwInvalidNettyVersion()
205 } catch {
206 case _: NumberFormatException =>
207 throwInvalidNettyVersion()
208 }
209 }
210
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 476 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: serverChannelFactory
Enclosing Method: NettyTransport()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:476
Taint Flags:

```
473 }
474
475 private val inboundBootstrap: Bootstrap = {
476 setupBootstrap(new ServerBootstrap(serverChannelFactory), serverPipelineFactory)
477 }
478
479 private def outboundBootstrap(remoteAddress: Address): ClientBootstrap = {
```

main/scala/akka/remote/transport/netty/NettyTransport.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.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 175 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: PortSelector
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:175
Taint Flags:

```
@deprecated("WARNING: This should only be used by professionals.", "2.4")
@nowarn("msg=deprecated")
val BindPortSelector: Int = getString("bind-port") match {
case "" => PortSelector
case value => value.toInt
}

```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 384 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: createExecutorService
Enclosing Method: NettyTransport()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:384
Taint Flags:

```
uniqueIdCounter.getAndIncrement)

private val clientChannelFactory: ChannelFactory = {
val boss, worker = createExecutorService()
new NioClientSocketChannelFactory(
boss,
1,
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 393 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: createExecutorService
Enclosing Method: NettyTransport()

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 393 (Code Correctness: Constructor Invokes Overridable Function) — Low

File: main/scala/akka/remote/transport/netty/NettyTransport.scala:393
Taint Flags:

```
390 }
391
392 private val serverChannelFactory: ChannelFactory = {
393 val boss, worker = createExecutorService()
394 // This does not create a HashedWheelTimer internally
395 new NioServerSocketChannelFactory(boss, worker, ServerSocketWorkerPoolSize)
396 }
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 122 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: EnableSsl
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:122
Taint Flags:

```
119
120 val EnableSsl: Boolean = getBoolean("enable-ssl")
121
122 val SSLEngineProviderClassName: String = if (EnableSsl) getString("ssl-engine-provider") else ""
123
124 val UseDispatcherForIo: Option[String] = getString("use-dispatcher-for-io") match {
125 case "" | null => None
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 179 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: EnableSsl
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:179
Taint Flags:

```
176 case value => value.toInt
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 179 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
}

val SslSettings: Option[SSLSettings] = if (EnableSsl) Some(new SSLSettings(config.getConfig("security"))) else None

val ServerSocketWorkerPoolSize: Int = computeWPS(config.getConfig("server-socket-worker-pool"))

```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 183 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: computeWPS
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:183
Taint Flags:

```

val ServerSocketWorkerPoolSize: Int = computeWPS(config.getConfig("server-socket-worker-pool"))

val ClientSocketWorkerPoolSize: Int = computeWPS(config.getConfig("client-socket-worker-pool"))

private def computeWPS(config: Config): Int =
ThreadPoolConfig.scaledPoolSize(
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 475 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: setupBootstrap
Enclosing Method: NettyTransport()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:475
Taint Flags:

```
bootstrap
}

private val inboundBootstrap: Bootstrap = {
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 475 (Code Correctness: Constructor Invokes Overridable Function) — Low

```
476 setupBootstrap(new ServerBootstrap(serverChannelFactory), serverPipelineFactory)
477 }
478
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 141 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: optionSize
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:141
Taint Flags:

```
138
139 val WriteBufferLowWaterMark: Option[Int] = optionSize("write-buffer-low-water-mark")
140
141 val SendBufferSize: Option[Int] = optionSize("send-buffer-size")
142
143 val ReceiveBufferSize: Option[Int] = optionSize("receive-buffer-size")
144
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 139 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: optionSize
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:139
Taint Flags:

```
136
137 val WriteBufferHighWaterMark: Option[Int] = optionSize("write-buffer-high-water-mark")
138
139 val WriteBufferLowWaterMark: Option[Int] = optionSize("write-buffer-low-water-mark")
140
141 val SendBufferSize: Option[Int] = optionSize("send-buffer-size")
142
```

Code Correctness: Constructor Invokes Overridable Function — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 143 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: optionSize
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:143
Taint Flags:

```
140
141  val SendBufferSize: Option[Int] = optionSize("send-buffer-size")
142
143  val ReceiveBufferSize: Option[Int] = optionSize("receive-buffer-size")
144
145  val MaxFrameSize: Int = getBytes("maximum-frame-size").toInt
146  .requiring(_ >= 32000, s"Setting 'maximum-frame-size' must be at least 32000 bytes")
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 379 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: uniqueIdCounter
Enclosing Method: NettyTransport()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:379
Taint Flags:

```
376  * The usage of this class is safe in the new remoting, as close() is called after unbind() is finished, and no
377  * outbound connections are initiated in the shutdown phase.
378  */
379  val channelGroup = new DefaultChannelGroup(
380  "akka-netty-transport-driver-channelgroup-" +
381  uniqueIdCounter.getAndIncrement)
382
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 475 (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.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 475 (Code Correctness: Constructor Invokes Overridable Function) — Low

Sink Details

Sink: FunctionCall: serverPipelineFactory
Enclosing Method: NettyTransport()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:475
Taint Flags:

```
472 bootstrap
473 }
474
475 private val inboundBootstrap: Bootstrap = {
476 setupBootstrap(new ServerBootstrap(serverChannelFactory), serverPipelineFactory)
477 }
478
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 137 (Code Correctness: Constructor Invokes Overridable Function) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: FunctionCall: optionSize
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:137
Taint Flags:

```
134
135 val ConnectionTimeout: FiniteDuration = config.getMillisDuration("connection-timeout")
136
137 val WriteBufferHighWaterMark: Option[Int] = optionSize("write-buffer-high-water-mark")
138
139 val WriteBufferLowWaterMark: Option[Int] = optionSize("write-buffer-low-water-mark")
140
```

Code Correctness: Erroneous String Compare (27 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");
  }
```

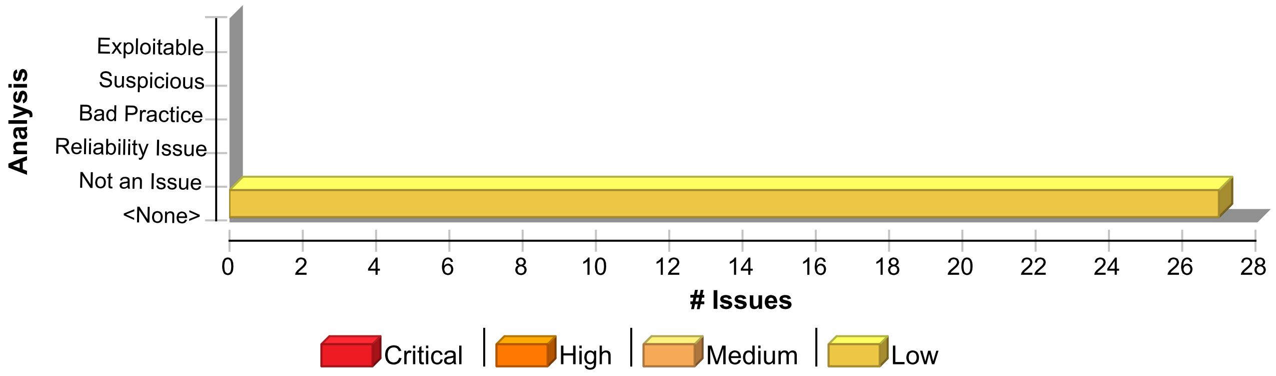
The `==` and `!=` operators will only behave as expected when they are used to compare strings contained in objects that are equal. The most common way for this to occur is for the strings to be interned, whereby the strings are added to a pool of objects maintained by the `String` class. Once a string 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	27	0	0	27
Total	**27**	**0**	**0**	**27**

Code Correctness: Erroneous String Compare — Low

Package: akka.remote

main/scala/akka/remote/RemoteSettings.scala, line 111 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:111
Taint Flags:

```
108 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
109 val LogBufferSizeExceeding: Int = {
110 val key = "akka.remote.classic.log-buffer-size-exceeding"
111 config.getString(key).toLowerCase match {
112 case "off" | "false" => Int.MaxValue
113 case _ => config.getInt(key)
114 }
```

test/scala/akka/remote/TransientSerializationErrorSpec.scala, line 41 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: fromBinary()
File: test/scala/akka/remote/TransientSerializationErrorSpec.scala:41
Taint Flags:

```
38 case _ => Array.emptyByteArray
39 }
40 def fromBinary(bytes: Array[Byte], manifest: String): AnyRef = {
41 manifest match {
42 case "ND" => throw new NotSerializableException() // Not sure this applies here
43 case "IOD" => throw new IllegalArgumentException()
44 case _ => throw new NotSerializableException()
```

main/scala/akka/remote/RemoteSettings.scala, line 145 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Erroneous String Compare | Low

Package: akka.remote

main/scala/akka/remote/RemoteSettings.scala, line 145 (Code Correctness: Erroneous String Compare) | Low

Sink Details

Sink: Operation
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:145
Taint Flags:

```
142 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
143 val QuarantineSilentSystemTimeout: FiniteDuration = {
144 val key = "akka.remote.classic.quarantine-after-silence"
145 config.getString(key).toLowerCase match {
146 case "off" | "false" => Duration.Zero
147 case _ =>
148 config.getMillisDuration(key).requiring(_ > Duration.Zero, "quarantine-after-silence must be > 0")
```

main/scala/akka/remote/RemoteSettings.scala, line 145 (Code Correctness: Erroneous String Compare) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:145
Taint Flags:

```
142 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
143 val QuarantineSilentSystemTimeout: FiniteDuration = {
144 val key = "akka.remote.classic.quarantine-after-silence"
145 config.getString(key).toLowerCase match {
146 case "off" | "false" => Duration.Zero
147 case _ =>
148 config.getMillisDuration(key).requiring(_ > Duration.Zero, "quarantine-after-silence must be > 0")
```

main/scala/akka/remote/RemoteSettings.scala, line 58 (Code Correctness: Erroneous String Compare) | Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: RemoteSettings()

Code Correctness: Erroneous String Compare — Low

Package: akka.remote

main/scala/akka/remote/RemoteSettings.scala, line 58 (Code Correctness: Erroneous String Compare) — Low

File: main/scala/akka/remote/RemoteSettings.scala:58
Taint Flags:

```
55 immutableSeq(getStringList("akka.remote.classic.trusted-selection-paths")).toSet
56
57 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
58 val RemoteLifecycleEventsLogLevel: LogLevel = toRootLowerCase(
59 getString("akka.remote.classic.log-remote-lifecycle-events")) match {
60 case "on" => Logging.DebugLevel
61 case other =>
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 417 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: actorOf()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:417
Taint Flags:

```
414 val elems = path.elements
415 val lookup =
416 if (lookupDeploy)
417 elems.head match {
418 case "user" | "system" => deployer.lookup(elems.drop(1))
419 case "remote" => lookupRemotes(elems)
420 case _ => None
```

test/scala/akka/remote/TransientSerializationErrorSpec.scala, line 41 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: fromBinary()
File: test/scala/akka/remote/TransientSerializationErrorSpec.scala:41
Taint Flags:

```
38 case _ => Array.emptyByteArray
```

Code Correctness: Erroneous String Compare — Low

Package: akka.remote

test/scala/akka/remote/TransientSerializationErrorSpec.scala, line 41 (Code Correctness: Erroneous String Compare) — Low

```
39 }
40 def fromBinary(bytes: Array[Byte], manifest: String): AnyRef = {
41 manifest match {
42 case "ND" => throw new NotSerializableException() // Not sure this applies here
43 case "IOD" => throw new IllegalArgumentException()
44 case _ => throw new NotSerializableException()
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 417 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: actorOf()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:417
Taint Flags:

```
414 val elems = path.elements
415 val lookup =
416 if (lookupDeploy)
417 elems.head match {
418 case "user" | "system" => deployer.lookup(elems.drop(1))
419 case "remote" => lookupRemotes(elems)
420 case _ => None
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 417 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: actorOf()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:417
Taint Flags:

```
414 val elems = path.elements
415 val lookup =
416 if (lookupDeploy)
417 elems.head match {
```

Code Correctness: Erroneous String Compare — Low

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 417 (Code Correctness: Erroneous String Compare) — Low

```
418 case "user" | "system" => deployer.lookup(elems.drop(1))
419 case "remote" => lookupRemotes(elems)
420 case _ => None
```

main/scala/akka/remote/RemoteSettings.scala, line 111 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:111
Taint Flags:

```
108 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
109 val LogBufferSizeExceeding: Int = {
110 val key = "akka.remote.classic.log-buffer-size-exceeding"
111 config.getString(key).toLowerCase match {
112 case "off" | "false" => Int.MaxValue
113 case _ => config.getInt(key)
114 }
```

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 293 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: getHostname()
File: main/scala/akka/remote/artery/ArterySettings.scala:293
Taint Flags:

```
290 final val Debug = false // unlocks additional very verbose debug logging of compression events (to stdout)
291 }
292
293 def getHostname(key: String, config: Config): String = config.getString(key) match {
294 case "<getHostAddress>" => InetAddress.getLocalHost.getHostAddress
295 case "<getHostName>" => InetAddress.getLocalHost.getHostName
```

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 293 (Code Correctness: Erroneous String Compare) — Low

```
296 case other =>
```

main/scala/akka/remote/artery/ArterySettings.scala, line 163 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: ArterySettings$Advanced()
File: main/scala/akka/remote/artery/ArterySettings.scala:163
Taint Flags:

```
160 .getMillisDuration("shutdown-flush-timeout")
161 .requiring(timeout => timeout > Duration.Zero, "shutdown-flush-timeout must be more than zero")
162 val DeathWatchNotificationFlushTimeout: FiniteDuration = {
163 toRootLowerCase(config.getString("death-watch-notification-flush-timeout")) match {
164 case "off" => Duration.Zero
165 case _ =>
166 config
```

main/scala/akka/remote/artery/ArterySettings.scala, line 270 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: ArterySettings$Compression$ActorRefs()
File: main/scala/akka/remote/artery/ArterySettings.scala:270
Taint Flags:

```
267 import config._
268
269 val AdvertisementInterval: FiniteDuration = config.getMillisDuration("advertisement-interval")
270 val Max: Int = toRootLowerCase(getString("max")) match {
271 case "off" => 0
272 case _ => getInt("max")
273 }
```

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 50 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: ArterySettings$Bind()
File: main/scala/akka/remote/artery/ArterySettings.scala:50
Taint Flags:

```
val config: Config = getConfig("bind")
import config._

val Port: Int = getString("port") match {
case "" => Canonical.Port
case _ => getInt("port").requiring(port => 0 to 65535 contains port, "bind.port must be 0 through 65535")
}
```

main/scala/akka/remote/artery/ArterySettings.scala, line 281 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: ArterySettings$Compression$Manifests()
File: main/scala/akka/remote/artery/ArterySettings.scala:281
Taint Flags:

```
import config._

val AdvertisementInterval: FiniteDuration = config.getMillisDuration("advertisement-interval")
val Max: Int = toRootLowerCase(getString("max")) match {
case "off" => 0
case _ => getInt("max")
}
```

main/scala/akka/remote/artery/ArterySettings.scala, line 54 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 54 (Code Correctness: Erroneous String Compare) — Low

Sink Details

Sink: Operation
Enclosing Method: ArterySettings$Bind()
File: main/scala/akka/remote/artery/ArterySettings.scala:54
Taint Flags:

```
case "" => Canonical.Port
case _ => getInt("port").requiring(port => 0 to 65535 contains port, "bind.port must be 0 through 65535")
}
val Hostname: String = getHostname("hostname", config) match {
case "" => Canonical.Hostname
case other => other
}
```

main/scala/akka/remote/artery/ArterySettings.scala, line 245 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: ArterySettings$Advanced$Tcp()
File: main/scala/akka/remote/artery/ArterySettings.scala:245
Taint Flags:

```
.getMillisDuration("connection-timeout")
.requiring(interval => interval > Duration.Zero, "connection-timeout must be more than zero")
val OutboundClientHostname: Option[String] = {
config.getString("outbound-client-hostname") match {
case "" => None
case hostname => Some(hostname)
}
```

main/scala/akka/remote/artery/ArterySettings.scala, line 293 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: getHostname()

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 293 (Code Correctness: Erroneous String Compare) — Low

File: main/scala/akka/remote/artery/ArterySettings.scala:293
Taint Flags:

```
290 final val Debug = false // unlocks additional very verbose debug logging of compression events (to stdout)
291 }
292
293 def getHostname(key: String, config: Config): String = config.getString(key) match {
294 case "<getHostAddress>" => InetAddress.getLocalHost.getHostAddress
295 case "<getHostName>" => InetAddress.getLocalHost.getHostName
296 case other =>
```

Package: akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 87 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: isSupported()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:87
Taint Flags:

```
84 val rootB = RootActorPath(addressB)
85
86 def isSupported: Boolean = {
87 val checked = system.settings.config.getString("akka.remote.artery.ssl.ssl-engine-provider") match {
88 case "akka.remote.artery.tcp.ConfigSSLEngineProvider" =>
89 CipherSuiteSupportCheck.isSupported(system, "akka.remote.artery.ssl.config-ssl-engine")
90 case "akka.remote.artery.tcp.ssl.RotatingKeysSSLEngineProvider" =>
```

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 87 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: isSupported()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:87
Taint Flags:

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 87 (Code Correctness: Erroneous String Compare) — Low

```
val rootB = RootActorPath(addressB)

def isSupported: Boolean = {
val checked = system.settings.config.getString("akka.remote.artery.ssl.ssl-engine-provider") match {
case "akka.remote.artery.tcp.ConfigSSLEngineProvider" =>
CipherSuiteSupportCheck.isSupported(system, "akka.remote.artery.ssl.config-ssl-engine")
case "akka.remote.artery.tcp.ssl.RotatingKeysSSLEngineProvider" =>
```

Package: akka.remote.serialization

test/scala/akka/remote/serialization/SerializationTransportInformationSpec.scala, line 47 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: fromBinary()
File: test/scala/akka/remote/serialization/SerializationTransportInformationSpec.scala:47
Taint Flags:

```
}
def fromBinary(bytes: Array[Byte], manifest: String): AnyRef = {
verifyTransportInfo()
manifest match {
case "A" =>
val parts = new String(bytes, StandardCharsets.UTF_8).split(',')
val fromStr = parts(0)
```

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 159 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:159
Taint Flags:

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 159 (Code Correctness: Erroneous String Compare) — Low

```
case _ => getBoolean("tcp-reuse-addr")
}

val Hostname: String = getString("hostname") match {
case "" => InetAddress.getLocalHost.getHostAddress
case value => value
}
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 164 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:164
Taint Flags:

```
case value => value
}

val BindHostname: String = getString("bind-hostname") match {
case "" => Hostname
case value => value
}
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 124 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:124
Taint Flags:

```

val SSLEngineProviderClassName: String = if (EnableSsl) getString("ssl-engine-provider") else ""

```

Code Correctness: Erroneous String Compare — Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 124 (Code Correctness: Erroneous String Compare) — Low

```
val UseDispatcherForIo: Option[String] = getString("use-dispatcher-for-io") match {
case "" | null => None
case dispatcher => Some(dispatcher)
}
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 174 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:174
Taint Flags:

```

@deprecated("WARNING: This should only be used by professionals.", "2.4")
@nowarn("msg=deprecated")
val BindPortSelector: Int = getString("bind-port") match {
case "" => PortSelector
case value => value.toInt
}
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 154 (Code Correctness: Erroneous String Compare) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:154
Taint Flags:

```

val TcpKeepalive: Boolean = getBoolean("tcp-keepalive")

val TcpReuseAddr: Boolean = getString("tcp-reuse-addr") match {
case "off-for-windows" => !Helpers.isWindows
case _ => getBoolean("tcp-reuse-addr")
```

Code Correctness: Erroneous String Compare	Low

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 154 (Code Correctness: Erroneous String Compare)	Low

```
}
```

Package: main.scala.akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 350 (Code Correctness: Erroneous String Compare)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Operation
Enclosing Method: apply()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:350
Taint Flags:

```

implicit val executionContext: ExecutionContext =
settings.UseDispatcherForIo
.orElse(RARP(system).provider.remoteSettings.Dispatcher match {
case "" => None
case dispatcherName => Some(dispatcherName)
})
```

Code Correctness: Non-Static Inner Class Implements Serializable (133 issues)

Abstract

Inner classes implementing `java.io.Serializable` may cause problems and leak information from the outer class.

Explanation

Serialization of inner classes lead to serialization of the outer class, therefore possibly leaking information or leading to a runtime error if the outer class is not serializable. As well as this, serializing inner classes may cause platform dependencies since the Java compiler creates synthetic fields in order to implement inner classes, but these are implementation dependent, and may vary from compiler to compiler. **Example 1:** The following code allows serialization of an inner class.

```
...
class User implements Serializable {
  private int accessLevel;
  class Registrator implements Serializable {
    ...
  }
}
```

In `Example 1`, when the inner class `Registrator` is serialized, it will also serialize the field `accessLevel` from the outer class `User`.

Recommendation

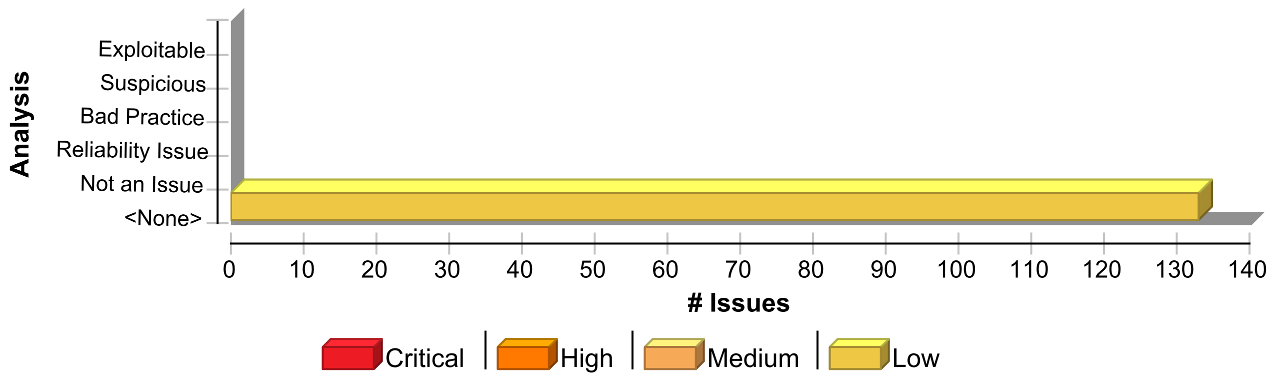
When using inner classes, they should not be serialized, or they should be changed to static-nested classes, since these do not have the drawbacks that non-static inner classes have when serialized. When a nested class is static it inherently has no association with instance variables (including those of the outer class), and would not cause serialization of the outer class. **Example 2:** The following code changes the example in `Example 1`, by stopping the inner class from implementing `java.io.Serializable`.

```
...
class User implements Serializable {
  private int accessLevel;
  class Registrator {
    ...
  }
}
```

Example 2: The following code changes the example in `Example 1`, by making the inner class into a static-nested class.

```
...
class User implements Serializable {
  private int accessLevel;
  static class Registrator implements Serializable {
    ...
  }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Code Correctness: Non-Static Inner Class Implements Serializable	133	0	0	133
Total	**133**	**0**	**0**	**133**

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

test/scala/akka/remote/MessageLoggingSpec.scala, line 45 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MessageLoggingSpec$BadMsg
File: test/scala/akka/remote/MessageLoggingSpec.scala:45
Taint Flags:

```
42 }
43 """.stripMargin)
44
45 case class BadMsg(msg: String) extends CborSerializable {
46 override def toString = throw new RuntimeException("Don't log me")
47
48 }
```

main/scala/akka/remote/Endpoint.scala, line 605 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointWriter$StoppedReading

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 605 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

File: main/scala/akka/remote/Endpoint.scala:605
Taint Flags:

```
private case object FlushAndStopTimeout
case object AckIdleCheckTimer
final case class StopReading(writer: ActorRef, replyTo: ActorRef)
final case class StoppedReading(writer: ActorRef)

final case class Handle(handle: AkkaProtocolHandle) extends NoSerializationVerificationNeeded

```

main/scala/akka/remote/Remoting.scala, line 317 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$ManagementCommand
File: main/scala/akka/remote/Remoting.scala:317
Taint Flags:

```
def seq = seqOpt.get
}
final case class Quarantine(remoteAddress: Address, uid: Option[Int]) extends RemotingCommand
final case class ManagementCommand(cmd: Any) extends RemotingCommand
final case class ManagementCommandAck(status: Boolean)

// Messages internal to EndpointManager
```

main/scala/akka/remote/Endpoint.scala, line 609 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointWriter$OutboundAck
File: main/scala/akka/remote/Endpoint.scala:609
Taint Flags:

```

final case class Handle(handle: AkkaProtocolHandle) extends NoSerializationVerificationNeeded

```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 609 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
609  final case class OutboundAck(ack: Ack)
610
611  // These settings are not configurable because wrong configuration will break the auto-tuning
612  private val SendBufferBatchSize = 5
```

main/scala/akka/remote/RemoteDeploymentWatcher.scala, line 18 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteDeploymentWatcher$WatchRemote
File: main/scala/akka/remote/RemoteDeploymentWatcher.scala:18
Taint Flags:

```
15  * INTERNAL API
16  */
17 private[akka] object RemoteDeploymentWatcher {
18  final case class WatchRemote(actor: ActorRef, supervisor: ActorRef)
19 }
20
21 /**
```

main/scala/akka/remote/Endpoint.scala, line 599 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointWriter$TookOver
File: main/scala/akka/remote/Endpoint.scala:599
Taint Flags:

```
596  * @param handle Handle of the new inbound association.
597  */
598  final case class TakeOver(handle: AkkaProtocolHandle, replyTo: ActorRef) extends NoSerializationVerificationNeeded
599  final case class TookOver(writer: ActorRef, handle: AkkaProtocolHandle) extends NoSerializationVerificationNeeded
600  case object BackoffTimer
601  case object FlushAndStop
602  private case object FlushAndStopTimeout
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 326 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$ListensFailure
File: main/scala/akka/remote/Remoting.scala:326
Taint Flags:

```
323 addressesPromise: Promise[Seq[(AkkaProtocolTransport, Address)]],
324 results: Seq[(AkkaProtocolTransport, Address, Promise[AssociationEventListener])])
325 extends NoSerializationVerificationNeeded
326 final case class ListensFailure(addressesPromise: Promise[Seq[(AkkaProtocolTransport, Address)]], cause: Throwable)
327 extends NoSerializationVerificationNeeded
328
329 // Helper class to store address pairs
```

main/scala/akka/remote/Remoting.scala, line 347 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Quarantined
File: main/scala/akka/remote/Remoting.scala:347
Taint Flags:

```
344 final case class Gated(timeOfRelease: Deadline) extends EndpointPolicy {
345 override def isTombstone: Boolean = true
346 }
347 final case class Quarantined(uid: Int, timeOfRelease: Deadline) extends EndpointPolicy {
348 override def isTombstone: Boolean = true
349 }
350
```

test/scala/akka/remote/AckedDeliverySpec.scala, line 18 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

test/scala/akka/remote/AckedDeliverySpec.scala, line 18 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: AckedDeliverySpec$Sequenced
File: test/scala/akka/remote/AckedDeliverySpec.scala:18
Taint Flags:

```
15 @nowarn("msg=deprecated")
16 object AckedDeliverySpec {
17
18  final case class Sequenced(seq: SeqNo, body: String) extends HasSequenceNumber {
19  override def toString = s"MSG[${seq.rawValue}]"
20  }
21
```

main/scala/akka/remote/Remoting.scala, line 341 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Pass
File: main/scala/akka/remote/Remoting.scala:341
Taint Flags:

```
338  */
339  def isTombstone: Boolean
340  }
341  final case class Pass(endpoint: ActorRef, uid: Option[Int]) extends EndpointPolicy {
342  override def isTombstone: Boolean = false
343  }
344  final case class Gated(timeOfRelease: Deadline) extends EndpointPolicy {
```

main/scala/akka/remote/Endpoint.scala, line 604 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointWriter$StopReading
File: main/scala/akka/remote/Endpoint.scala:604
Taint Flags:

```
601  case object FlushAndStop
602  private case object FlushAndStopTimeout
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 604 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
603  case object AckIdleCheckTimer
604  final case class StopReading(writer: ActorRef, replyTo: ActorRef)
605  final case class StoppedReading(writer: ActorRef)
606
607  final case class Handle(handle: AkkaProtocolHandle) extends NoSerializationVerificationNeeded
```

main/scala/akka/remote/Remoting.scala, line 299 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Listen
File: main/scala/akka/remote/Remoting.scala:299
Taint Flags:

```
296
297  // Messages between Remoting and EndpointManager
298  sealed trait RemotingCommand extends NoSerializationVerificationNeeded
299  final case class Listen(addressesPromise: Promise[Seq[(AkkaProtocolTransport, Address)]]) extends RemotingCommand
300  case object StartupFinished extends RemotingCommand
301  case object ShutdownAndFlush extends RemotingCommand
302  @InternalStableApi
```

main/scala/akka/remote/Endpoint.scala, line 218 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ReliableDeliverySupervisor$GotUid
File: main/scala/akka/remote/Endpoint.scala:218
Taint Flags:

```
215 private[remote] object ReliableDeliverySupervisor {
216  case object Ungate
217  case object AttemptSysMsgRedelivery
218  final case class GotUid(uid: Int, remoteAddres: Address)
219
220  case object IsIdle
221  case object Idle
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 218 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/RemoteActorRefProvider.scala, line 107 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteActorRefProvider$RemoteDeadLetterActorRef
File: main/scala/akka/remote/RemoteActorRefProvider.scala:107
Taint Flags:

```
104 * and handled as dead letters to the original (remote) destination. Without this special case, DeathWatch related
105 * functionality breaks, like the special handling of Watch messages arriving to dead letters.
106 */
107 private class RemoteDeadLetterActorRef(_provider: ActorRefProvider, _path: ActorPath, _eventStream: EventStream)
108 extends DeadLetterActorRef(_provider, _path, _eventStream) {
109 import EndpointManager.Send
110
```

main/scala/akka/remote/RemoteWatcher.scala, line 61 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcher$Stats
File: main/scala/akka/remote/RemoteWatcher.scala:61
Taint Flags:

```
58 lazy val empty: Stats = counts(0, 0)
59 def counts(watching: Int, watchingNodes: Int): Stats = Stats(watching, watchingNodes)(Set.empty, Set.empty)
60 }
61 final case class Stats(watching: Int, watchingNodes: Int)(
62 val watchingRefs: Set[(ActorRef, ActorRef)],
63 val watchingAddresses: Set[Address]) {
64 override def toString: String = {
```

main/scala/akka/remote/Remoting.scala, line 119 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 119 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Remoting$RegisterTransportActor
File: main/scala/akka/remote/Remoting.scala:119
Taint Flags:

```
116  }
117  }
118
119  final case class RegisterTransportActor(props: Props, name: String) extends NoSerializationVerificationNeeded
120
121  private[Remoting] class TransportSupervisor extends Actor with RequiresMessageQueue[UnboundedMessageQueueSemantics] {
122  override def supervisorStrategy = OneForOneStrategy() {
```

main/scala/akka/remote/RemoteWatcher.scala, line 42 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcher$UnwatchRemote
File: main/scala/akka/remote/RemoteWatcher.scala:42
Taint Flags:

```
39  .withDeploy(Deploy.local)
40
41  final case class WatchRemote(watchee: InternalActorRef, watcher: InternalActorRef)
42  final case class UnwatchRemote(watchee: InternalActorRef, watcher: InternalActorRef)
43
44  @SerialVersionUID(1L) case object Heartbeat extends HeartbeatMessage
45  @SerialVersionUID(1L) final case class HeartbeatRsp(addressUid: Int) extends HeartbeatMessage
```

main/scala/akka/remote/Endpoint.scala, line 598 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 598 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: EndpointWriter$TakeOver
File: main/scala/akka/remote/Endpoint.scala:598
Taint Flags:

```
595  * used instead.
596  * @param handle Handle of the new inbound association.
597  */
598  final case class TakeOver(handle: AkkaProtocolHandle, replyTo: ActorRef) extends NoSerializationVerificationNeeded
599  final case class TookOver(writer: ActorRef, handle: AkkaProtocolHandle) extends NoSerializationVerificationNeeded
600  case object BackoffTimer
601  case object FlushAndStop
```

main/scala/akka/remote/RemoteWatcher.scala, line 49 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcher$ArteryHeartbeatRsp
File: main/scala/akka/remote/RemoteWatcher.scala:49
Taint Flags:

```
46
47  // specific pair of messages for artery to allow for protobuf serialization and long uid
48  case object ArteryHeartbeat extends HeartbeatMessage with ArteryMessage
49  final case class ArteryHeartbeatRsp(uid: Long) extends HeartbeatMessage with ArteryMessage
50
51  // sent to self only
52  case object HeartbeatTick
```

main/scala/akka/remote/Endpoint.scala, line 607 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointWriter$Handle
File: main/scala/akka/remote/Endpoint.scala:607
Taint Flags:

```
604  final case class StopReading(writer: ActorRef, replyTo: ActorRef)
605  final case class StoppedReading(writer: ActorRef)
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Endpoint.scala, line 607 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```

final case class Handle(handle: AkkaProtocolHandle) extends NoSerializationVerificationNeeded

final case class OutboundAck(ack: Ack)

```

main/scala/akka/remote/RemoteWatcher.scala, line 54 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcher$ExpectedFirstHeartbeat
File: main/scala/akka/remote/RemoteWatcher.scala:54
Taint Flags:

```
// sent to self only
case object HeartbeatTick
case object ReapUnreachableTick
final case class ExpectedFirstHeartbeat(from: Address)

// test purpose
object Stats {
```

main/scala/akka/remote/Remoting.scala, line 303 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Send
File: main/scala/akka/remote/Remoting.scala:303
Taint Flags:

```
case object StartupFinished extends RemotingCommand
case object ShutdownAndFlush extends RemotingCommand
@InternalStableApi
final case class Send(
message: Any,
senderOption: OptionVal[ActorRef],
recipient: RemoteActorRef,
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 303 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/Remoting.scala, line 318 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$ManagementCommandAck
File: main/scala/akka/remote/Remoting.scala:318
Taint Flags:

```
315 }
316 final case class Quarantine(remoteAddress: Address, uid: Option[Int]) extends RemotingCommand
317 final case class ManagementCommand(cmd: Any) extends RemotingCommand
318 final case class ManagementCommandAck(status: Boolean)
319
320 // Messages internal to EndpointManager
321 case object Prune extends NoSerializationVerificationNeeded
```

main/scala/akka/remote/Remoting.scala, line 316 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Quarantine
File: main/scala/akka/remote/Remoting.scala:316
Taint Flags:

```
313 // acknowledged delivery buffers
314 def seq = seqOpt.get
315 }
316 final case class Quarantine(remoteAddress: Address, uid: Option[Int]) extends RemotingCommand
317 final case class ManagementCommand(cmd: Any) extends RemotingCommand
318 final case class ManagementCommandAck(status: Boolean)
319
```

main/scala/akka/remote/Remoting.scala, line 344 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 344 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Gated
File: main/scala/akka/remote/Remoting.scala:344
Taint Flags:

```
341 final case class Pass(endpoint: ActorRef, uid: Option[Int]) extends EndpointPolicy {
342 override def isTombstone: Boolean = false
343 }
344 final case class Gated(timeOfRelease: Deadline) extends EndpointPolicy {
345 override def isTombstone: Boolean = true
346 }
347 final case class Quarantined(uid: Int, timeOfRelease: Deadline) extends EndpointPolicy {
```

main/scala/akka/remote/Remoting.scala, line 330 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$Link
File: main/scala/akka/remote/Remoting.scala:330
Taint Flags:

```
327 extends NoSerializationVerificationNeeded
328
329 // Helper class to store address pairs
330 final case class Link(localAddress: Address, remoteAddress: Address)
331
332 final case class ResendState(uid: Int, buffer: AckedReceiveBuffer[Message])
333
```

main/scala/akka/remote/Remoting.scala, line 332 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/Remoting.scala, line 332 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: EndpointManager$ResendState
File: main/scala/akka/remote/Remoting.scala:332
Taint Flags:

```
329 // Helper class to store address pairs
330 final case class Link(localAddress: Address, remoteAddress: Address)
331
332 final case class ResendState(uid: Int, buffer: AckedReceiveBuffer[Message])
333
334 sealed trait EndpointPolicy {
335
```

main/scala/akka/remote/RemoteWatcher.scala, line 41 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcher$WatchRemote
File: main/scala/akka/remote/RemoteWatcher.scala:41
Taint Flags:

```
38 .withDispatcher(Dispatchers.InternalDispatcherId)
39 .withDeploy(Deploy.local)
40
41 final case class WatchRemote(watchee: InternalActorRef, watcher: InternalActorRef)
42 final case class UnwatchRemote(watchee: InternalActorRef, watcher: InternalActorRef)
43
44 @SerialVersionUID(1L) case object Heartbeat extends HeartbeatMessage
```

main/scala/akka/remote/PhiAccrualFailureDetector.scala, line 120 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: PhiAccrualFailureDetector$State
File: main/scala/akka/remote/PhiAccrualFailureDetector.scala:120
Taint Flags:

```
117 *
118 * Cannot be final due to https://github.com/scala/bug/issues/4440
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/PhiAccrualFailureDetector.scala, line 120 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
*/
private case class State(history: HeartbeatHistory, timestamp: Option[Long])

private val state = new AtomicReference[State](State(history = firstHeartbeat, timestamp = None))

```

main/scala/akka/remote/Remoting.scala, line 322 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: EndpointManager$ListensResult
File: main/scala/akka/remote/Remoting.scala:322
Taint Flags:

```

// Messages internal to EndpointManager
case object Prune extends NoSerializationVerificationNeeded
final case class ListensResult(
addressesPromise: Promise[Seq[(AkkaProtocolTransport, Address)]],
results: Seq[(AkkaProtocolTransport, Address, Promise[AssociationEventListener])])
extends NoSerializationVerificationNeeded
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 48 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteActorRefProvider$Internals
File: main/scala/akka/remote/RemoteActorRefProvider.scala:48
Taint Flags:

```
@InternalApi
private[akka] object RemoteActorRefProvider {

private final case class Internals(transport: RemoteTransport, remoteDaemon: InternalActorRef)
extends NoSerializationVerificationNeeded

sealed trait TerminatorState
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 48 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/RemoteWatcher.scala, line 45 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcher$HeartbeatRsp
File: main/scala/akka/remote/RemoteWatcher.scala:45
Taint Flags:

```
42  final case class UnwatchRemote(watchee: InternalActorRef, watcher: InternalActorRef)
43
44  @SerialVersionUID(1L) case object Heartbeat extends HeartbeatMessage
45  @SerialVersionUID(1L) final case class HeartbeatRsp(addressUid: Int) extends HeartbeatMessage
46
47  // specific pair of messages for artery to allow for protobuf serialization and long uid
48  case object ArteryHeartbeat extends HeartbeatMessage with ArteryMessage
```

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 106 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AssociationState$UniqueRemoteAddressValue
File: main/scala/akka/remote/artery/ArteryTransport.scala:106
Taint Flags:

```
103  s"Quarantined ${TimeUnit.NANOSECONDS.toSeconds(System.nanoTime() - nanoTime)} seconds ago"
104  }
105
106  private final case class UniqueRemoteAddressValue(
107  uniqueRemoteAddress: Option[UniqueAddress],
108  listeners: List[UniqueAddress => Unit])
109
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteActorSelectionSpec.scala, line 26 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteActorSelectionSpec$ActorCreateReq
File: test/scala/akka/remote/artery/RemoteActorSelectionSpec.scala:26
Taint Flags:

```
23
24 object RemoteActorSelectionSpec {
25  final case class ActorSelReq(s: String) extends JavaSerializable
26  final case class ActorCreateReq(props: Props, name: String) extends JavaSerializable
27
28  class SelectionActor extends Actor with ActorLogging {
29  log.info("Started")
```

test/scala/akka/remote/artery/SendQueueSpec.scala, line 26 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SendQueueSpec$Msg
File: test/scala/akka/remote/artery/SendQueueSpec.scala:26
Taint Flags:

```
23
24  case class ProduceToQueue(from: Int, until: Int, queue: Queue[Msg])
25  case class ProduceToQueueValue(from: Int, until: Int, queue: SendQueue.QueueValue[Msg])
26  case class Msg(fromProducer: String, value: Int)
27
28  def producerProps(producerId: String): Props =
29  Props(new Producer(producerId))
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: SystemMessageDelivery$Nack
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:44
Taint Flags:

```
41 @InternalApi private[remote] object SystemMessageDelivery {
42  final case class SystemMessageEnvelope(message: AnyRef, seqNo: Long, ackReplyTo: UniqueAddress) extends ArteryMessage
43  final case class Ack(seqNo: Long, from: UniqueAddress) extends Reply
44  final case class Nack(seqNo: Long, from: UniqueAddress) extends Reply
45
46  /**
47  * Sent when an incarnation of an Association is quarantined. Consumed by the
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 42 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SystemMessageDelivery$SystemMessageEnvelope
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:42
Taint Flags:

```
39  * INTERNAL API
40  */
41 @InternalApi private[remote] object SystemMessageDelivery {
42  final case class SystemMessageEnvelope(message: AnyRef, seqNo: Long, ackReplyTo: UniqueAddress) extends ArteryMessage
43  final case class Ack(seqNo: Long, from: UniqueAddress) extends Reply
44  final case class Nack(seqNo: Long, from: UniqueAddress) extends Reply
45
```

main/scala/akka/remote/artery/Handshake.scala, line 36 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: OutboundHandshake$HandshakeRsp
File: main/scala/akka/remote/artery/Handshake.scala:36
Taint Flags:

```
33  class HandshakeTimeoutException(msg: String) extends RuntimeException(msg) with NoStackTrace
34
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Handshake.scala, line 36 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
final case class HandshakeReq(from: UniqueAddress, to: Address) extends ControlMessage
final case class HandshakeRsp(from: UniqueAddress) extends Reply

private sealed trait HandshakeState
private case object Start extends HandshakeState
```

test/scala/akka/remote/artery/RemoteFailureSpec.scala, line 16 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteFailureSpec$Ping
File: test/scala/akka/remote/artery/RemoteFailureSpec.scala:16
Taint Flags:

```
import akka.testkit.TestEvent.Mute

object RemoteFailureSpec {
 final case class Ping(s: String) extends CborSerializable
}

class RemoteFailureSpec extends ArteryMultiNodeSpec with ImplicitSender {
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 27 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MetadataCarryingSpy$RemoteReadMetadata
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:27
Taint Flags:

```
 final case class RemoteMessageSent(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
 final case class RemoteMessageReceived(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
 final case class RemoteWriteMetadata(recipient: ActorRef, message: Object, sender: ActorRef)
 final case class RemoteReadMetadata(recipient: ActorRef, message: Object, sender: ActorRef, metadata: String)
}

class MetadataCarryingSpy extends Extension {
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 27 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala, line 17 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: LargeMessagesStreamSpec$Ping
File: test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala:17
Taint Flags:

```
14 import akka.util.ByteString
15
16 object LargeMessagesStreamSpec {
17  case class Ping(payload: ByteString = ByteString.empty) extends JavaSerializable
18  case class Pong(bytesReceived: Long) extends JavaSerializable
19
20  class EchoSize extends Actor {
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 25 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MetadataCarryingSpy$RemoteMessageReceived
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:25
Taint Flags:

```
22  override def createExtension(system: ExtendedActorSystem): MetadataCarryingSpy = new MetadataCarryingSpy
23
24  final case class RemoteMessageSent(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
25  final case class RemoteMessageReceived(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
26  final case class RemoteWriteMetadata(recipient: ActorRef, message: Object, sender: ActorRef)
27  final case class RemoteReadMetadata(recipient: ActorRef, message: Object, sender: ActorRef, metadata: String)
28 }
```

main/scala/akka/remote/artery/RestartCounter.scala, line 17 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/RestartCounter.scala, line 17 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RestartCounter$State
File: main/scala/akka/remote/artery/RestartCounter.scala:17
Taint Flags:

```
14  * INTERNAL API
15  */
16 private[remote] object RestartCounter {
17  final case class State(count: Int, deadline: Deadline)
18 }
19
20 /**
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 26 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MetadataCarryingSpy$RemoteWriteMetadata
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:26
Taint Flags:

```
23
24  final case class RemoteMessageSent(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
25  final case class RemoteMessageReceived(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
26  final case class RemoteWriteMetadata(recipient: ActorRef, message: Object, sender: ActorRef)
27  final case class RemoteReadMetadata(recipient: ActorRef, message: Object, sender: ActorRef, metadata: String)
28 }
29
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 58 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 58 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: SystemMessageDelivery$ClearSystemMessageDelivery
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:58
Taint Flags:

```
55  * The SystemMessageDelivery operator also detects that the incarnation has changed when sending or resending
56  * system messages.
57  */
58  final case class ClearSystemMessageDelivery(incarnation: Int)
59
60  final class GaveUpSystemMessageException(msg: String) extends RuntimeException(msg) with NoStackTrace
61
```

test/scala/akka/remote/artery/RemoteWatcherSpec.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcherSpec$TestRemoteWatcher$AddressTerm
File: test/scala/akka/remote/artery/RemoteWatcherSpec.scala:44
Taint Flags:

```
41  }
42
43  object TestRemoteWatcher {
44  final case class AddressTerm(address: Address) extends JavaSerializable
45  final case class Quarantined(address: Address, uid: Option[Long]) extends JavaSerializable
46  }
47
```

test/scala/akka/remote/artery/RemoteActorSelectionSpec.scala, line 25 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteActorSelectionSpec$ActorSelReq
File: test/scala/akka/remote/artery/RemoteActorSelectionSpec.scala:25
Taint Flags:

```
22 import akka.testkit.JavaSerializable
23
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteActorSelectionSpec.scala, line 25 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
object RemoteActorSelectionSpec {
 final case class ActorSelReq(s: String) extends JavaSerializable
 final case class ActorCreateReq(props: Props, name: String) extends JavaSerializable

 class SelectionActor extends Actor with ActorLogging {
```

main/scala/akka/remote/artery/Association.scala, line 72 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Association$QueueWrapperImpl
File: main/scala/akka/remote/artery/Association.scala:72
Taint Flags:

```
 def queue: Queue[OutboundEnvelope]
 }

 final case class QueueWrapperImpl(queue: Queue[OutboundEnvelope]) extends QueueWrapper {
 override def offer(message: OutboundEnvelope): Boolean = queue.offer(message)

 override def isEnabled: Boolean = true
```

main/scala/akka/remote/artery/Association.scala, line 121 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Association$OutboundStreamMatValues
File: main/scala/akka/remote/artery/Association.scala:121
Taint Flags:

```
 case object OutboundStreamStopIdleSignal extends RuntimeException("") with StopSignal with NoStackTrace
 case object OutboundStreamStopQuarantinedSignal extends RuntimeException("") with StopSignal with NoStackTrace

 final case class OutboundStreamMatValues(
 streamKillSwitch: OptionVal[SharedKillSwitch],
 completed: Future[Done],
 stopping: OptionVal[StopSignal])
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 121 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

test/scala/akka/remote/artery/RemoteWatcherSpec.scala, line 45 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcherSpec$TestRemoteWatcher$Quarantined
File: test/scala/akka/remote/artery/RemoteWatcherSpec.scala:45
Taint Flags:

```

object TestRemoteWatcher {
final case class AddressTerm(address: Address) extends JavaSerializable
final case class Quarantined(address: Address, uid: Option[Long]) extends JavaSerializable
}

class TestRemoteWatcher(heartbeatExpectedResponseAfter: FiniteDuration)
```

main/scala/akka/remote/artery/Handshake.scala, line 35 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: OutboundHandshake$HandshakeReq
File: main/scala/akka/remote/artery/Handshake.scala:35
Taint Flags:

```
*/
class HandshakeTimeoutException(msg: String) extends RuntimeException(msg) with NoStackTrace

final case class HandshakeReq(from: UniqueAddress, to: Address) extends ControlMessage
final case class HandshakeRsp(from: UniqueAddress) extends Reply

private sealed trait HandshakeState
```

test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala, line 18 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala, line 18 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: LargeMessagesStreamSpec$Pong
File: test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala:18
Taint Flags:

```
15
16 object LargeMessagesStreamSpec {
17  case class Ping(payload: ByteString = ByteString.empty) extends JavaSerializable
18  case class Pong(bytesReceived: Long) extends JavaSerializable
19
20  class EchoSize extends Actor {
21  def receive = {
```

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 38 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SystemMessageDeliverySpec$TestSysMsg
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:38
Taint Flags:

```
35
36 object SystemMessageDeliverySpec {
37
38  case class TestSysMsg(s: String) extends SystemMessageDelivery.AckedDeliveryMessage
39
40  val safe = ConfigFactory.parseString(s"""
41  akka.loglevel = INFO
```

test/scala/akka/remote/artery/UntrustedSpec.scala, line 31 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/UntrustedSpec.scala, line 31 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: UntrustedSpec$StopChild
File: test/scala/akka/remote/artery/UntrustedSpec.scala:31
Taint Flags:

```
28
29 object UntrustedSpec {
30  final case class IdentifyReq(path: String) extends CborSerializable
31  final case class StopChild(name: String) extends CborSerializable
32
33  class Receptionist(testActor: ActorRef) extends Actor {
34  context.actorOf(Props(classOf[Child], testActor), "child1")
```

test/scala/akka/remote/artery/SendQueueSpec.scala, line 25 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SendQueueSpec$ProduceToQueueValue
File: test/scala/akka/remote/artery/SendQueueSpec.scala:25
Taint Flags:

```
22 object SendQueueSpec {
23
24  case class ProduceToQueue(from: Int, until: Int, queue: Queue[Msg])
25  case class ProduceToQueueValue(from: Int, until: Int, queue: SendQueue.QueueValue[Msg])
26  case class Msg(fromProducer: String, value: Int)
27
28  def producerProps(producerId: String): Props =
```

main/scala/akka/remote/artery/ArteryTransport.scala, line 101 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AssociationState$QuarantinedTimestamp
File: main/scala/akka/remote/artery/ArteryTransport.scala:101
Taint Flags:

```
98  quarantined = ImmutableLongMap.empty[QuarantinedTimestamp],
99  new AtomicReference(UniqueRemoteAddressValue(None, Nil)))
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 101 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```

final case class QuarantinedTimestamp(nanoTime: Long) {
override def toString: String =
s"Quarantined ${TimeUnit.NANOSECONDS.toSeconds(System.nanoTime() - nanoTime)} seconds ago"
}
```

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 43 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SystemMessageDelivery$Ack
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:43
Taint Flags:

```
*/
@InternalApi private[remote] object SystemMessageDelivery {
final case class SystemMessageEnvelope(message: AnyRef, seqNo: Long, ackReplyTo: UniqueAddress) extends ArteryMessage
final case class Ack(seqNo: Long, from: UniqueAddress) extends Reply
final case class Nack(seqNo: Long, from: UniqueAddress) extends Reply

/**
```

test/scala/akka/remote/artery/SendQueueSpec.scala, line 24 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SendQueueSpec$ProduceToQueue
File: test/scala/akka/remote/artery/SendQueueSpec.scala:24
Taint Flags:

```

object SendQueueSpec {

case class ProduceToQueue(from: Int, until: Int, queue: Queue[Msg])
case class ProduceToQueueValue(from: Int, until: Int, queue: SendQueue.QueueValue[Msg])
case class Msg(fromProducer: String, value: Int)

```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/SendQueueSpec.scala, line 24 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/artery/SystemMessageDelivery.scala, line 60 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SystemMessageDelivery$GaveUpSystemMessageException
File: main/scala/akka/remote/artery/SystemMessageDelivery.scala:60
Taint Flags:

```
57 */
58 final case class ClearSystemMessageDelivery(incarnation: Int)
59
60 final class GaveUpSystemMessageException(msg: String) extends RuntimeException(msg) with NoStackTrace
61
62 private case object ResendTick
63
```

main/scala/akka/remote/artery/Handshake.scala, line 33 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: OutboundHandshake$HandshakeTimeoutException
File: main/scala/akka/remote/artery/Handshake.scala:33
Taint Flags:

```
30 * Stream is failed with this exception if the handshake is not completed
31 * within the handshake timeout.
32 */
33 class HandshakeTimeoutException(msg: String) extends RuntimeException(msg) with NoStackTrace
34
35 final case class HandshakeReq(from: UniqueAddress, to: Address) extends ControlMessage
36 final case class HandshakeRsp(from: UniqueAddress) extends Reply
```

main/scala/akka/remote/artery/Control.scala, line 104 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Control.scala, line 104 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: InboundControlJunction$Dettach
File: main/scala/akka/remote/artery/Control.scala:104
Taint Flags:

```
101  private[InboundControlJunction] sealed trait CallbackMessage
102  private[InboundControlJunction] final case class Attach(observer: ControlMessageObserver, done: Promise[Done])
103  extends CallbackMessage
104  private[InboundControlJunction] final case class Dettach(observer: ControlMessageObserver) extends CallbackMessage
105 }
106
107 /**
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 24 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MetadataCarryingSpy$RemoteMessageSent
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:24
Taint Flags:

```
21  override def lookup = MetadataCarryingSpy
22  override def createExtension(system: ExtendedActorSystem): MetadataCarryingSpy = new MetadataCarryingSpy
23
24  final case class RemoteMessageSent(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
25  final case class RemoteMessageReceived(recipient: ActorRef, message: Object, sender: ActorRef, size: Int, time: Long)
26  final case class RemoteWriteMetadata(recipient: ActorRef, message: Object, sender: ActorRef)
27  final case class RemoteReadMetadata(recipient: ActorRef, message: Object, sender: ActorRef, metadata: String)
```

test/scala/akka/remote/artery/UntrustedSpec.scala, line 30 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/UntrustedSpec.scala, line 30 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: UntrustedSpec$IdentifyReq
File: test/scala/akka/remote/artery/UntrustedSpec.scala:30
Taint Flags:

```
27 import akka.testkit.TestProbe
28
29 object UntrustedSpec {
30  final case class IdentifyReq(path: String) extends CborSerializable
31  final case class StopChild(name: String) extends CborSerializable
32
33  class Receptionist(testActor: ActorRef) extends Actor {
```

test/scala/akka/remote/artery/MetadataCarryingSpec.scala, line 93 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MetadataCarryingSpec$Ping
File: test/scala/akka/remote/artery/MetadataCarryingSpec.scala:93
Taint Flags:

```
90 }
91
92 object MetadataCarryingSpec {
93  final case class Ping(payload: ByteString = ByteString.empty) extends JavaSerializable
94
95  class ProxyActor(local: ActorRef, remotePath: ActorPath) extends Actor {
96  val remote = context.system.actorSelection(remotePath)
```

main/scala/akka/remote/artery/ArteryTransport.scala, line 956 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ArteryTransport$AeronTerminated
File: main/scala/akka/remote/artery/ArteryTransport.scala:956
Taint Flags:

```
953  // ArterySettings.Version can be lower than this HighestVersion to support rolling upgrades.
954  val HighestVersion: Byte = 0
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 956 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
955
956  class AeronTerminated(e: Throwable) extends RuntimeException(e)
957
958  object ShutdownSignal extends RuntimeException with NoStackTrace
959
```

main/scala/akka/remote/artery/Association.scala, line 97 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Association$LazyQueueWrapper
File: main/scala/akka/remote/artery/Association.scala:97
Taint Flags:

```
94  override def isEnabled: Boolean = false
95  }
96
97  final case class LazyQueueWrapper(queue: Queue[OutboundEnvelope], materialize: () => Unit) extends QueueWrapper {
98  private val onlyOnce = new AtomicBoolean
99
100  def runMaterialize(): Unit = {
```

main/scala/akka/remote/artery/Control.scala, line 102 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: InboundControlJunction$Attach
File: main/scala/akka/remote/artery/Control.scala:102
Taint Flags:

```
99
100  // messages for the stream callback
101  private[InboundControlJunction] sealed trait CallbackMessage
102  private[InboundControlJunction] final case class Attach(observer: ControlMessageObserver, done: Promise[Done])
103  extends CallbackMessage
104  private[InboundControlJunction] final case class Dettach(observer: ControlMessageObserver) extends CallbackMessage
105 }
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Control.scala, line 102 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/artery/Codecs.scala, line 241 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Decoder$RetryResolveRemoteDeployedRecipient
File: main/scala/akka/remote/artery/Codecs.scala:241
Taint Flags:

```
238  * INTERNAL API
239  */
240 private[remote] object Decoder {
241  private final case class RetryResolveRemoteDeployedRecipient(
242  attemptsLeft: Int,
243  recipientPath: String,
244  inboundEnvelope: InboundEnvelope)
```

test/scala/akka/remote/artery/RemoteInstrumentsSpec.scala, line 14 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteInstrumentsSpec$KeyLen
File: test/scala/akka/remote/artery/RemoteInstrumentsSpec.scala:14
Taint Flags:

```
11
12 class RemoteInstrumentsSpec extends AnyWordSpec with Matchers with Checkers {
13
14  case class KeyLen(k: Key, l: Len) {
15  override def toString = s" key = ${k}, len = ${l}"
16  }
17  type Key = Byte
```

main/scala/akka/remote/artery/ArteryTransport.scala, line 963 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 963 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ArteryTransport$InboundStreamMatValues
File: main/scala/akka/remote/artery/ArteryTransport.scala:963
Taint Flags:

```
960  // thrown when the transport is shutting down and something triggers a new association
961  object ShuttingDown extends RuntimeException with NoStackTrace
962
963  final case class InboundStreamMatValues[LifeCycle](lifeCycle: LifeCycle, completed: Future[Done])
964
965  val ControlStreamId = 1
966  val OrdinaryStreamId = 2
```

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/AeronSink.scala, line 39 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AeronSink$PublicationClosedException
File: main/scala/akka/remote/artery/aeron/AeronSink.scala:39
Taint Flags:

```
36
37  final class GaveUpMessageException(msg: String) extends RuntimeException(msg) with NoStackTrace
38
39  final class PublicationClosedException(msg: String) extends RuntimeException(msg) with NoStackTrace
40
41  private val TimerCheckPeriod = 1 << 13 // 8192
42  private val TimerCheckMask = TimerCheckPeriod - 1
```

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 30 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 30 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: TaskRunner$Add
File: main/scala/akka/remote/artery/aeron/TaskRunner.scala:30
Taint Flags:

```
27  type Task = () => Boolean
28  sealed trait Command
29  case object Shutdown extends Command
30  final case class Add(task: Task) extends Command
31  final case class Remove(task: Task) extends Command
32
33  final class CommandQueue extends AbstractNodeQueue[Command]
```

main/scala/akka/remote/artery/aeron/AeronSink.scala, line 37 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AeronSink$GaveUpMessageException
File: main/scala/akka/remote/artery/aeron/AeronSink.scala:37
Taint Flags:

```
34  */
35 private[remote] object AeronSink {
36
37  final class GaveUpMessageException(msg: String) extends RuntimeException(msg) with NoStackTrace
38
39  final class PublicationClosedException(msg: String) extends RuntimeException(msg) with NoStackTrace
40
```

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 33 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: TaskRunner$CommandQueue
File: main/scala/akka/remote/artery/aeron/TaskRunner.scala:33
Taint Flags:

```
30  final case class Add(task: Task) extends Command
31  final case class Remove(task: Task) extends Command
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 33 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```

final class CommandQueue extends AbstractNodeQueue[Command]

/**
* A specialized collection with allocation free add, remove and iterate of
```

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 31 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: TaskRunner$Remove
File: main/scala/akka/remote/artery/aeron/TaskRunner.scala:31
Taint Flags:

```
sealed trait Command
case object Shutdown extends Command
final case class Add(task: Task) extends Command
final case class Remove(task: Task) extends Command

final class CommandQueue extends AbstractNodeQueue[Command]

```

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/InboundCompressions.scala, line 272 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: InboundCompression$Tables
File: main/scala/akka/remote/artery/compress/InboundCompressions.scala:272
Taint Flags:

```
* It starts with containing only a single "disabled" table (versioned as `DecompressionTable.DisabledVersion`),
* and from there on continuously accumulates at most [[keepOldTables]] recently used tables.
*/
final case class Tables[T](
oldTables: List[DecompressionTable[T]],
activeTable: DecompressionTable[T],
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.artery.compress

main/scala/akka/remote/artery/compress/InboundCompressions.scala, line 272 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
275  nextTable: DecompressionTable[T],
```

Package: akka.remote.artery.tcp.ssl

main/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProvider.scala, line 172 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RotatingKeysSSLEngineProvider$ConfiguredContext
File: main/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProvider.scala:172
Taint Flags:

```
169  * INTERNAL API
170  */
171  @InternalApi
172  private case class ConfiguredContext(context: SSLContext, sessionVerifier: SessionVerifier)
173
174 }
175
```

main/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProvider.scala, line 166 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RotatingKeysSSLEngineProvider$CachedContext
File: main/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProvider.scala:166
Taint Flags:

```
163  * INTERNAL API
164  */
165  @InternalApi
166  private case class CachedContext(cached: ConfiguredContext, expires: Deadline)
167
168  /**
169  * INTERNAL API
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/UntrustedSpec.scala, line 33 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: UntrustedSpec$StopChild
File: test/scala/akka/remote/classic/UntrustedSpec.scala:33
Taint Flags:

```

object UntrustedSpec {
 final case class IdentifyReq(path: String) extends JavaSerializable
 final case class StopChild(name: String) extends JavaSerializable

 class Receptionist(testActor: ActorRef) extends Actor {
 context.actorOf(Props(classOf[Child], testActor), "child1")
```

test/scala/akka/remote/classic/UntrustedSpec.scala, line 32 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: UntrustedSpec$IdentifyReq
File: test/scala/akka/remote/classic/UntrustedSpec.scala:32
Taint Flags:

```
import akka.testkit.TestProbe

object UntrustedSpec {
 final case class IdentifyReq(path: String) extends JavaSerializable
 final case class StopChild(name: String) extends JavaSerializable

 class Receptionist(testActor: ActorRef) extends Actor {
```

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: RemoteWatcherSpec$TestRemoteWatcher$AddressTerm
File: test/scala/akka/remote/classic/RemoteWatcherSpec.scala:44
Taint Flags:

```
}

object TestRemoteWatcher {
final case class AddressTerm(address: Address)
final case class Quarantined(address: Address, uid: Option[Long])
}

```

test/scala/akka/remote/classic/RemoteWatcherSpec.scala, line 45 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemoteWatcherSpec$TestRemoteWatcher$Quarantined
File: test/scala/akka/remote/classic/RemoteWatcherSpec.scala:45
Taint Flags:

```

object TestRemoteWatcher {
final case class AddressTerm(address: Address)
final case class Quarantined(address: Address, uid: Option[Long])
}

class TestRemoteWatcher(heartbeatExpectedResponseAfter: FiniteDuration)
```

test/scala/akka/remote/classic/RemotingSpec.scala, line 30 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: RemotingSpec$ActorSelReq
File: test/scala/akka/remote/classic/RemotingSpec.scala:30
Taint Flags:

```

object RemotingSpec {
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 30 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

```

final case class ActorSelReq(s: String)

class Echo1 extends Actor {
var target: ActorRef = context.system.deadLetters
```

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 74 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerTransportAdapterSpec$Lost
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:74
Taint Flags:

```
}
}

final case class Lost(msg: String)
}

@nowarn("msg=deprecated")
```

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 60 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SystemMessageDeliveryStressTest$SystemMessageSequenceVerifier
File: test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala:60
Taint Flags:

```
}
""")

private[akka] class SystemMessageSequenceVerifier(system: ActorSystem, testActor: ActorRef) extends MinimalActorRef {
val provider = RARP(system).provider
val path = provider.tempPath()
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 60 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
63
```

Package: akka.remote.serialization

test/scala/akka/remote/serialization/SystemMessageSerializationSpec.scala, line 22 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SystemMessageSerializationSpec$TestException
File: test/scala/akka/remote/serialization/SystemMessageSerializationSpec.scala:22
Taint Flags:

```
19
20  val testConfig = ConfigFactory.parseString(serializationTestOverrides).withFallback(AkkaSpec.testConf)
21
22  class TestException(msg: String) extends RuntimeException(msg) with JavaSerializable {
23  override def equals(other: Any): Boolean = other match {
24  case e: TestException => e.getMessage == getMessage
25  case _ => false
```

test/scala/akka/remote/serialization/SerializationTransportInformationSpec.scala, line 29 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SerializationTransportInformationSpec$JavaSerTestMessage
File: test/scala/akka/remote/serialization/SerializationTransportInformationSpec.scala:29
Taint Flags:

```
26 object SerializationTransportInformationSpec {
27
28  final case class TestMessage(from: ActorRef, to: ActorRef)
29  final case class JavaSerTestMessage(from: ActorRef, to: ActorRef) extends JavaSerializable
30
31  class TestSerializer(system: ExtendedActorSystem) extends SerializerWithStringManifest {
32  def identifier: Int = 666
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/SerializationTransportInformationSpec.scala, line 28 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: SerializationTransportInformationSpec$TestMessage
File: test/scala/akka/remote/serialization/SerializationTransportInformationSpec.scala:28
Taint Flags:

```
25
26 object SerializationTransportInformationSpec {
27
28  final case class TestMessage(from: ActorRef, to: ActorRef)
29  final case class JavaSerTestMessage(from: ActorRef, to: ActorRef) extends JavaSerializable
30
31  class TestSerializer(system: ExtendedActorSystem) extends SerializerWithStringManifest {
```

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 39 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MiscMessageSerializerSpec$TestException
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:39
Taint Flags:

```
36
37  val testConfig = ConfigFactory.parseString(serializationTestOverrides).withFallback(AkkaSpec.testConf)
38
39  class TestException(msg: String, cause: Throwable) extends RuntimeException(msg, cause) {
40  def this(msg: String) = this(msg, null)
41
42  override def equals(other: Any): Boolean = other match {
```

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 58 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 58 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: MiscMessageSerializerSpec$TestExceptionNoStack
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:58
Taint Flags:

```
55 else getStackTrace.toList
56 }
57
58 class TestExceptionNoStack(msg: String) extends TestException(msg) with NoStackTrace {
59 override def equals(other: Any): Boolean = other match {
60 case e: TestExceptionNoStack =>
61 e.getMessage == getMessage && e.stackTrace == stackTrace
```

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 66 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: MiscMessageSerializerSpec$OtherException
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:66
Taint Flags:

```
63 }
64 }
65
66 class OtherException(msg: String) extends IllegalArgumentException(msg) with JavaSerializable {
67 override def equals(other: Any): Boolean = other match {
68 case e: OtherException => e.getMessage == getMessage
69 case _ => false
```

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 36 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AkkaPduCodec$Disassociate
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:36
Taint Flags:

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 36 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
*/
sealed trait AkkaPdu
final case class Associate(info: HandshakeInfo) extends AkkaPdu
final case class Disassociate(reason: AssociationHandle.DisassociateInfo) extends AkkaPdu
case object Heartbeat extends AkkaPdu
final case class Payload(bytes: ByteString) extends AkkaPdu

```

main/scala/akka/remote/transport/Transport.scala, line 40 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Transport$InboundAssociation
File: main/scala/akka/remote/transport/Transport.scala:40
Taint Flags:

```
* @param association
* The handle for the inbound association.
*/
final case class InboundAssociation(association: AssociationHandle) extends AssociationEvent

/**
* An interface that needs to be implemented by the user of a transport to listen to association events
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 262 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$HandleListenerRegistered
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:262
Taint Flags:

```

final case class Handle(handle: AssociationHandle) extends NoSerializationVerificationNeeded

final case class HandleListenerRegistered(listener: HandleEventListener) extends NoSerializationVerificationNeeded

```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 262 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
sealed trait ProtocolStateData
trait InitialProtocolStateData extends ProtocolStateData
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 268 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$OutboundUnassociated
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:268
Taint Flags:

```
trait InitialProtocolStateData extends ProtocolStateData

// Neither the underlying, nor the provided transport is associated
final case class OutboundUnassociated(
remoteAddress: Address,
statusPromise: Promise[AssociationHandle],
transport: Transport)
```

main/scala/akka/remote/transport/Transport.scala, line 171 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AssociationHandle$InboundPayload
File: main/scala/akka/remote/transport/Transport.scala:171
Taint Flags:

```
* @param payload
* The raw bytes that were sent by the remote endpoint.
*/
final case class InboundPayload(payload: ByteString) extends HandleEvent {
override def toString: String = s"InboundPayload(size = ${payload.length} bytes)"
}

```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/TestTransport.scala, line 288 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: TestTransport$DisassociateAttempt
File: main/scala/akka/remote/transport/TestTransport.scala:288
Taint Flags:

```
285 final case class AssociateAttempt(localAddress: Address, remoteAddress: Address) extends Activity
286 final case class ShutdownAttempt(boundAddress: Address) extends Activity
287 final case class WriteAttempt(sender: Address, recipient: Address, payload: ByteString) extends Activity
288 final case class DisassociateAttempt(requester: Address, remote: Address) extends Activity
289
290 /**
291 * Shared state among [[akka.remote.transport.TestTransport]] instances. Coordinates the transports and the means
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 260 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerManager$Handle
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:260
Taint Flags:

```
257 final case class ListenerAndMode(listener: HandleEventListener, mode: ThrottleMode)
258 extends NoSerializationVerificationNeeded
259
260 final case class Handle(handle: ThrottlerHandle) extends NoSerializationVerificationNeeded
261
262 final case class Listener(listener: HandleEventListener) extends NoSerializationVerificationNeeded
263 }
```

main/scala/akka/remote/transport/TestTransport.scala, line 286 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/TestTransport.scala, line 286 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: TestTransport$ShutdownAttempt
File: main/scala/akka/remote/transport/TestTransport.scala:286
Taint Flags:

```
283
284  final case class ListenAttempt(boundAddress: Address) extends Activity
285  final case class AssociateAttempt(localAddress: Address, remoteAddress: Address) extends Activity
286  final case class ShutdownAttempt(boundAddress: Address) extends Activity
287  final case class WriteAttempt(sender: Address, recipient: Address, payload: ByteString) extends Activity
288  final case class DisassociateAttempt(requester: Address, remote: Address) extends Activity
289
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 438 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottledAssociation$FailWith
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:438
Taint Flags:

```
435  case object Uninitialized extends ThrottlerData
436  final case class ExposedHandle(handle: ThrottlerHandle) extends ThrottlerData
437
438  final case class FailWith(reason: DisassociateInfo)
439 }
440
441 /**
```

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: FailureInjectorTransportAdapter$All
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:44
Taint Flags:

```
41  trait FailureInjectorCommand
42  @SerialVersionUID(1L)
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 44 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
@deprecated("Not implemented", "2.5.22")
final case class All(mode: GremlinMode)
@SerialVersionUID(1L)
final case class One(remoteAddress: Address, mode: GremlinMode)

```

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 155 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ActorTransportAdapter$ListenerRegistered
File: main/scala/akka/remote/transport/AbstractTransportAdapter.scala:155
Taint Flags:

```
object ActorTransportAdapter {
 sealed trait TransportOperation extends NoSerializationVerificationNeeded

 final case class ListenerRegistered(listener: AssociationEventListener) extends TransportOperation
 final case class AssociateUnderlying(remoteAddress: Address, statusPromise: Promise[AssociationHandle])
 extends TransportOperation
 final case class ListenUnderlying(listenAddress: Address, upstreamListener: Future[AssociationEventListener])
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 285 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$AssociatedWaitHandler
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:285
Taint Flags:

```
 extends InitialProtocolStateData

 // Both transports are associated, but the handler for the handle has not yet been provided
 final case class AssociatedWaitHandler(
 handleListener: Future[HandleEventListener],
 wrappedHandle: AssociationHandle,
 queue: immutable.Queue[ByteString])
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 285 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 254 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerManager$AssociateResult
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:254
Taint Flags:

```
251 private[transport] object ThrottlerManager {
252  final case class Checkin(origin: Address, handle: ThrottlerHandle) extends NoSerializationVerificationNeeded
253
254  final case class AssociateResult(handle: AssociationHandle, statusPromise: Promise[AssociationHandle])
255  extends NoSerializationVerificationNeeded
256
257  final case class ListenerAndMode(listener: HandleEventListener, mode: ThrottleMode)
```

main/scala/akka/remote/transport/Transport.scala, line 210 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AssociationHandle$ActorHandleEventListener
File: main/scala/akka/remote/transport/Transport.scala:210
Taint Flags:

```
207  * forward event objects as messages to the provided ActorRef.
208  * @param actor
209  */
210  final case class ActorHandleEventListener(actor: ActorRef) extends HandleEventListener {
211  override def notify(ev: HandleEvent): Unit = actor ! ev
212  }
213 }
```

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 35 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 35 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AkkaPduCodec$Associate
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:35
Taint Flags:

```
32  * Trait that represents decoded Akka PDUs (Protocol Data Units)
33  */
34  sealed trait AkkaPdu
35  final case class Associate(info: HandshakeInfo) extends AkkaPdu
36  final case class Disassociate(reason: AssociationHandle.DisassociateInfo) extends AkkaPdu
37  case object Heartbeat extends AkkaPdu
38  final case class Payload(bytes: ByteString) extends AkkaPdu
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 114 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerTransportAdapter$TokenBucket
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:114
Taint Flags:

```
111  }
112
113  @SerialVersionUID(1L)
114  final case class TokenBucket(capacity: Int, tokensPerSecond: Double, nanoTimeOfLastSend: Long, availableTokens: Int)
115  extends ThrottleMode {
116
117  private def isAvailable(nanoTimeOfSend: Long, tokens: Int): Boolean =
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 69 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 69 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: AkkaProtocolTransport$AssociateUnderlyingRefuseUid
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:69
Taint Flags:

```
66 val AkkaOverhead: Int = 0 //Don't know yet
67 val UniqueId = new java.util.concurrent.atomic.AtomicInteger(0)
68
69 final case class AssociateUnderlyingRefuseUid(
70 remoteAddress: Address,
71 statusPromise: Promise[AssociationHandle],
72 refuseUid: Option[Int])
```

main/scala/akka/remote/transport/Transport.scala, line 29 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Transport$InvalidAssociationException
File: main/scala/akka/remote/transport/Transport.scala:29
Taint Flags:

```
26 * hostname, etc.).
27 */
28 @SerialVersionUID(1L)
29 final case class InvalidAssociationException(msg: String, cause: Throwable = null)
30 extends AkkaException(msg, cause)
31 with NoStackTrace
32
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 257 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerManager$ListenerAndMode
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:257
Taint Flags:

```
254 final case class AssociateResult(handle: AssociationHandle, statusPromise: Promise[AssociationHandle])
255 extends NoSerializationVerificationNeeded
```

Code Correctness: Non-Static Inner Class Implements Serializable	Low

Package: akka.remote.transport

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 257 (Code Correctness: Non-Static Inner Class Implements Serializable)	Low

```
256
257 final case class ListenerAndMode(listener: HandleEventListener, mode: ThrottleMode)
258 extends NoSerializationVerificationNeeded
259
260 final case class Handle(handle: ThrottlerHandle) extends NoSerializationVerificationNeeded
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 291 (Code Correctness: Non-Static Inner Class Implements Serializable)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$ListenerReady
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:291
Taint Flags:

```
288 queue: immutable.Queue[ByteString])
289 extends ProtocolStateData
290
291 final case class ListenerReady(listener: HandleEventListener, wrappedHandle: AssociationHandle)
292 extends ProtocolStateData
293
294 case class TimeoutReason(errorMessage: String)
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 262 (Code Correctness: Non-Static Inner Class Implements Serializable)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerManager$Listener
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:262
Taint Flags:

```
259
260 final case class Handle(handle: ThrottlerHandle) extends NoSerializationVerificationNeeded
261
262 final case class Listener(listener: HandleEventListener) extends NoSerializationVerificationNeeded
263 }
264
265 /**
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 262 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 436 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottledAssociation$ExposedHandle
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:436
Taint Flags:

```
433
434  sealed trait ThrottlerData
435  case object Uninitialized extends ThrottlerData
436  final case class ExposedHandle(handle: ThrottlerHandle) extends ThrottlerData
437
438  final case class FailWith(reason: DisassociateInfo)
439 }
```

main/scala/akka/remote/transport/Transport.scala, line 181 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AssociationHandle$Disassociated
File: main/scala/akka/remote/transport/Transport.scala:181
Taint Flags:

```
178  * @param info
179  * information about the reason of disassociation
180  */
181  final case class Disassociated(info: DisassociateInfo) extends HandleEvent with DeadLetterSuppression
182
183  /**
184  * Supertype of possible disassociation reasons
```

main/scala/akka/remote/transport/TestTransport.scala, line 287 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/TestTransport.scala, line 287 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: TestTransport$WriteAttempt
File: main/scala/akka/remote/transport/TestTransport.scala:287
Taint Flags:

```
284 final case class ListenAttempt(boundAddress: Address) extends Activity
285 final case class AssociateAttempt(localAddress: Address, remoteAddress: Address) extends Activity
286 final case class ShutdownAttempt(boundAddress: Address) extends Activity
287 final case class WriteAttempt(sender: Address, recipient: Address, payload: ByteString) extends Activity
288 final case class DisassociateAttempt(requester: Address, remote: Address) extends Activity
289
290 /**
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 97 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerTransportAdapter$SetThrottle
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:97
Taint Flags:

```
94 }
95
96 @SerialVersionUID(1L)
97 final case class SetThrottle(address: Address, direction: Direction, mode: ThrottleMode)
98
99 @SerialVersionUID(1L)
100 case object SetThrottleAck {
```

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 174 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 174 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: ThrottlerTransportAdapter$ForceDisassociateExplicitly
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:174
Taint Flags:

```
171 * Management Command to force disassociation of an address with an explicit error.
172 */
173 @SerialVersionUID(1L)
174 final case class ForceDisassociateExplicitly(address: Address, reason: DisassociateInfo)
175
176 @SerialVersionUID(1L)
177 case object ForceDisassociateAck {
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 294 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$TimeoutReason
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:294
Taint Flags:

```
291 final case class ListenerReady(listener: HandleEventListener, wrappedHandle: AssociationHandle)
292 extends ProtocolStateData
293
294 case class TimeoutReason(errorMessage: String)
295 case object ForbiddenUidReason
296
297 private[remote] def outboundProps(
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 275 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$OutboundUnderlyingAssociated
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:275
Taint Flags:

```
272 extends InitialProtocolStateData
273
```

Code Correctness: Non-Static Inner Class Implements Serializable	Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 275 (Code Correctness: Non-Static Inner Class Implements Serializable)	Low

```
// The underlying transport is associated, but the handshake of the akka protocol is not yet finished
final case class OutboundUnderlyingAssociated(
statusPromise: Promise[AssociationHandle],
wrappedHandle: AssociationHandle)
extends ProtocolStateData
```

main/scala/akka/remote/transport/TestTransport.scala, line 285 (Code Correctness: Non-Static Inner Class Implements Serializable)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: TestTransport$AssociateAttempt
File: main/scala/akka/remote/transport/TestTransport.scala:285
Taint Flags:

```
sealed trait Activity

final case class ListenAttempt(boundAddress: Address) extends Activity
final case class AssociateAttempt(localAddress: Address, remoteAddress: Address) extends Activity
final case class ShutdownAttempt(boundAddress: Address) extends Activity
final case class WriteAttempt(sender: Address, recipient: Address, payload: ByteString) extends Activity
final case class DisassociateAttempt(requester: Address, remote: Address) extends Activity
```

main/scala/akka/remote/transport/Transport.scala, line 59 (Code Correctness: Non-Static Inner Class Implements Serializable)	Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: Transport$ActorAssociationEventListener
File: main/scala/akka/remote/transport/Transport.scala:59
Taint Flags:

```
* forward event objects as messages to the provided ActorRef.
* @param actor
*/
final case class ActorAssociationEventListener(actor: ActorRef) extends AssociationEventListener {
override def notify(ev: AssociationEvent): Unit = actor ! ev
}

```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/Transport.scala, line 59 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 168 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerTransportAdapter$ForceDisassociate
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:168
Taint Flags:

```
165  * Management Command to force disassociation of an address.
166  */
167  @SerialVersionUID(1L)
168  final case class ForceDisassociate(address: Address)
169
170  /**
171  * Management Command to force disassociation of an address with an explicit error.
```

main/scala/akka/remote/transport/TestTransport.scala, line 284 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: TestTransport$ListenAttempt
File: main/scala/akka/remote/transport/TestTransport.scala:284
Taint Flags:

```
281  */
282  sealed trait Activity
283
284  final case class ListenAttempt(boundAddress: Address) extends Activity
285  final case class AssociateAttempt(localAddress: Address, remoteAddress: Address) extends Activity
286  final case class ShutdownAttempt(boundAddress: Address) extends Activity
287  final case class WriteAttempt(sender: Address, recipient: Address, payload: ByteString) extends Activity
```

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 40 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 40 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AkkaPduCodec$Message
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:40
Taint Flags:

```
37  case object Heartbeat extends AkkaPdu
38  final case class Payload(bytes: ByteString) extends AkkaPdu
39
40  final case class Message(
41  recipient: InternalActorRef,
42  recipientAddress: Address,
43  serializedMessage: SerializedMessage,
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 260 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$Handle
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:260
Taint Flags:

```
257
258  case object HandshakeTimer extends NoSerializationVerificationNeeded
259
260  final case class Handle(handle: AssociationHandle) extends NoSerializationVerificationNeeded
261
262  final case class HandleListenerRegistered(listener: HandleEventListener) extends NoSerializationVerificationNeeded
263
```

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 46 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 46 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Sink: Class: FailureInjectorTransportAdapter$One
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:46
Taint Flags:

```
43  @deprecated("Not implemented", "2.5.22")
44  final case class All(mode: GremlinMode)
45  @SerialVersionUID(1L)
46  final case class One(remoteAddress: Address, mode: GremlinMode)
47
48  sealed trait GremlinMode
49  @SerialVersionUID(1L)
```

main/scala/akka/remote/transport/AkkaProtocolTransport.scala, line 281 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ProtocolStateActor$InboundUnassociated
File: main/scala/akka/remote/transport/AkkaProtocolTransport.scala:281
Taint Flags:

```
278  extends ProtocolStateData
279
280  // The underlying transport is associated, but the handshake of the akka protocol is not yet finished
281  final case class InboundUnassociated(associationListener: AssociationEventListener, wrappedHandle: AssociationHandle)
282  extends InitialProtocolStateData
283
284  // Both transports are associated, but the handler for the handle has not yet been provided
```

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 156 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ActorTransportAdapter$AssociateUnderlying
File: main/scala/akka/remote/transport/AbstractTransportAdapter.scala:156
Taint Flags:

```
153  sealed trait TransportOperation extends NoSerializationVerificationNeeded
154
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 156 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

```
155 final case class ListenerRegistered(listener: AssociationEventListener) extends TransportOperation
156 final case class AssociateUnderlying(remoteAddress: Address, statusPromise: Promise[AssociationHandle])
157 extends TransportOperation
158 final case class ListenUnderlying(listenAddress: Address, upstreamListener: Future[AssociationEventListener])
159 extends TransportOperation
```

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 160 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ActorTransportAdapter$DisassociateUnderlying
File: main/scala/akka/remote/transport/AbstractTransportAdapter.scala:160
Taint Flags:

```
157 extends TransportOperation
158 final case class ListenUnderlying(listenAddress: Address, upstreamListener: Future[AssociationEventListener])
159 extends TransportOperation
160 final case class DisassociateUnderlying(info: DisassociateInfo = AssociationHandle.Unknown)
161 extends TransportOperation
162 with DeadLetterSuppression
163
```

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 158 (Code Correctness: Non-Static Inner Class Implements Serializable) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ActorTransportAdapter$ListenUnderlying
File: main/scala/akka/remote/transport/AbstractTransportAdapter.scala:158
Taint Flags:

```
155 final case class ListenerRegistered(listener: AssociationEventListener) extends TransportOperation
156 final case class AssociateUnderlying(remoteAddress: Address, statusPromise: Promise[AssociationHandle])
157 extends TransportOperation
158 final case class ListenUnderlying(listenAddress: Address, upstreamListener: Future[AssociationEventListener])
159 extends TransportOperation
160 final case class DisassociateUnderlying(info: DisassociateInfo = AssociationHandle.Unknown)
161 extends TransportOperation
```

Code Correctness: Non-Static Inner Class Implements Serializable — Low

Package: akka.remote.transport

main/scala/akka/remote/transport/AbstractTransportAdapter.scala, line 158 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala, line 252 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: ThrottlerManager$Checkin
File: main/scala/akka/remote/transport/ThrottlerTransportAdapter.scala:252
Taint Flags:

```
249  */
250 @nowarn("msg=deprecated")
251 private[transport] object ThrottlerManager {
252  final case class Checkin(origin: Address, handle: ThrottlerHandle) extends NoSerializationVerificationNeeded
253
254  final case class AssociateResult(handle: AssociationHandle, statusPromise: Promise[AssociationHandle])
255  extends NoSerializationVerificationNeeded
```

main/scala/akka/remote/transport/AkkaPduCodec.scala, line 38 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: AkkaPduCodec$Payload
File: main/scala/akka/remote/transport/AkkaPduCodec.scala:38
Taint Flags:

```
35  final case class Associate(info: HandshakeInfo) extends AkkaPdu
36  final case class Disassociate(reason: AssociationHandle.DisassociateInfo) extends AkkaPdu
37  case object Heartbeat extends AkkaPdu
38  final case class Payload(bytes: ByteString) extends AkkaPdu
39
40  final case class Message(
41  recipient: InternalActorRef,
```

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 58 (Code Correctness: Non-Static Inner Class Implements Serializable) — **Low**

Issue Details

Code Correctness: Non-Static Inner Class Implements Serializable	Low
Package: akka.remote.transport	
main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 58 (Code Correctness: Non-Static Inner Class Implements Serializable)	**Low**

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: Class: FailureInjectorTransportAdapter$Drop
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:58
Taint Flags:

```
55  def getInstance = this
56  }
57  @SerialVersionUID(1L)
58  final case class Drop(outboundDropP: Double, inboundDropP: Double) extends GremlinMode
59 }
60
61 /**
```

Command Injection (10 issues)

Abstract

Executing commands that include unvalidated user input can cause an application to execute malicious commands on behalf of an attacker.

Explanation

Command injection vulnerabilities take two forms: - An attacker can change the command that the program executes: the attacker explicitly controls what the command is. - An attacker can change the environment in which the command executes: the attacker implicitly controls what the command means. In this case, we are primarily concerned with the second scenario, the possibility that an attacker may be able to change the meaning of the command by changing an environment variable or by putting a malicious executable early in the search path. Command injection vulnerabilities of this type occur when: 1. An attacker modifies an application's environment. 2. The application executes a command without specifying an absolute path or verifying the binary being executed. 3. By executing the command, the application gives an attacker a privilege or capability that the attacker would not otherwise have. **Example:** The following code is from a web application that provides an interface through which users can update their password on the system. Part of the process for updating passwords in certain network environments is to run a `make` command in the `/var/yp` directory.

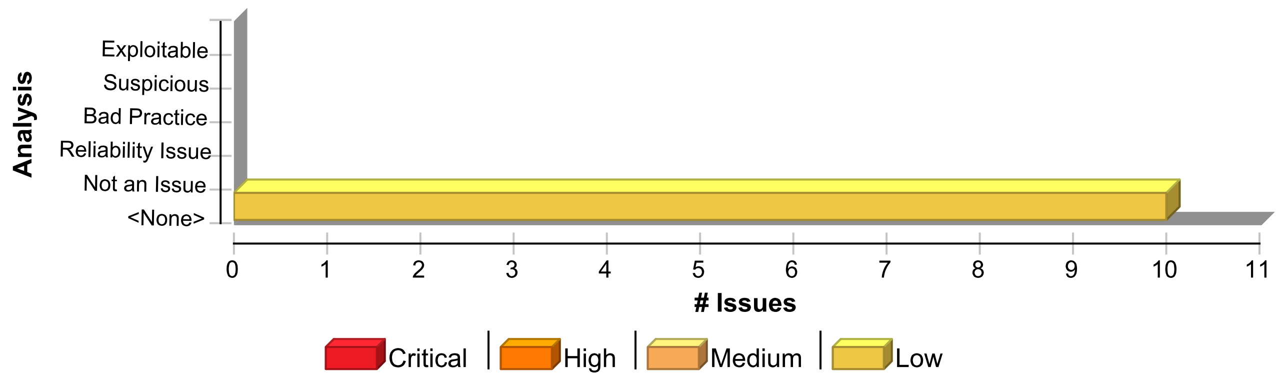
```
...
System.Runtime.getRuntime().exec("make");
...
```

The problem here is that the program does not specify an absolute path for make and fails to clean its environment prior to executing the call to `Runtime.exec()`. If an attacker can modify the `$PATH` variable to point to a malicious binary called `make` and then execute the application in their environment, the malicious binary will be loaded instead of the one intended. Because of the nature of the application, it runs with the privileges necessary to perform system operations, which means the attacker's `make` will now be run with these privileges, possibly giving them complete control of the system.

Recommendation

An attacker may indirectly control commands executed by a program by modifying the environment in which they are executed. The environment should not be trusted and precautions should be taken to prevent an attacker from using some manipulation of the environment to perform an attack. Whenever possible, commands should be controlled by the application and executed using an absolute path. In cases where the path is not known at compile time, such as for cross-platform applications, an absolute path should be constructed from trusted values during execution. Command values and paths read from configuration files or the environment should be sanity-checked against a set of invariants that define valid values. Other checks can sometimes be performed to detect if these sources may have been tampered with. For example, if a configuration file is world-writable, the program might refuse to run. In cases where information about the binary to be executed is known in advance, the program may perform checks to verify the identity of the binary. If a binary should always be owned by a particular user or have a particular set of access permissions assigned to it, these properties can be verified programmatically before the binary is executed. In the end it may be impossible for a program to fully protect itself from an imaginative attacker bent on controlling the commands the program executes. You should strive to identify and protect against every conceivable manipulation of input values and the environment. The goal should be to shut down as many attack vectors as possible.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Command Injection	10	0	0	10
Total	**10**	**0**	**0**	**10**

Command Injection — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 100 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:100
Taint Flags:

```
97  assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
98  assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
99  assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
100  assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
101  }
102 }
103
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 99 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:99
Taint Flags:

Command Injection — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 99 (Command Injection) — Low

```
println("===>>> Restoring network")
assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
}
}
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 98 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:98
Taint Flags:

```
def restoreIP() = {
println("===>>> Restoring network")
assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
}
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 97 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:97
Taint Flags:

```

def restoreIP() = {
println("===>>> Restoring network")
assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
```

Command Injection — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 97 (Command Injection) — Low

```
100 assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 92 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: enableTcpReset()
File: test/scala/akka/remote/NetworkFailureSpec.scala:92
Taint Flags:

```
89 def enableTcpReset() = {
90 restoreIP()
91 assert(
92 new ProcessBuilder("ipfw", "add", "1", "reset", "tcp", "from", "any", "to", "any", PortRange).start.waitFor == 0)
93 }
94
95 def restoreIP() = {
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 80 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:80
Taint Flags:

```
77 assert(
78 new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
79 assert(
80 new ProcessBuilder("ipfw", "pipe", "2", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
81 }
82
83 def enableNetworkDrop() = {
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 78 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation

Command Injection — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 78 (Command Injection) — Low

Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:78
Taint Flags:

```
75 assert(new ProcessBuilder("ipfw", "add", "pipe", "1", "ip", "from", "any", "to", "any").start.waitFor == 0)
76 assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77 assert(
78 new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
79 assert(
80 new ProcessBuilder("ipfw", "pipe", "2", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
81 }
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 76 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:76
Taint Flags:

```
73 def enableNetworkThrottling() = {
74 restoreIP()
75 assert(new ProcessBuilder("ipfw", "add", "pipe", "1", "ip", "from", "any", "to", "any").start.waitFor == 0)
76 assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77 assert(
78 new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
79 assert(
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 75 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:75

Command Injection — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 75 (Command Injection) — Low

Taint Flags:

```
72
73  def enableNetworkThrottling() = {
74  restoreIP()
75  assert(new ProcessBuilder("ipfw", "add", "pipe", "1", "ip", "from", "any", "to", "any").start.waitFor == 0)
76  assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77  assert(
78  new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 86 (Command Injection) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: ProcessBuilder(0)
Enclosing Method: enableNetworkDrop()
File: test/scala/akka/remote/NetworkFailureSpec.scala:86
Taint Flags:

```
83  def enableNetworkDrop() = {
84  restoreIP()
85  assert(
86  new ProcessBuilder("ipfw", "add", "1", "deny", "tcp", "from", "any", "to", "any", PortRange).start.waitFor == 0)
87  }
88
89  def enableTcpReset() = {
```

Dead Code: Expression is Always false (27 issues)

Abstract

This expression will always evaluate to `false`.

Explanation

This expression will always evaluate to `false`; the program could be rewritten in a simpler form. The nearby code may be present for debugging purposes, or it may not have been maintained along with the rest of the program. The expression may also be indicative of a bug earlier in the method. **Example 1:** The following method never sets the variable `secondCall` after initializing it to `false`. (The variable `firstCall` is mistakenly used twice.) The result is that the expression `firstCall && secondCall` will always evaluate to `false`, so `setUpDualCall()` will never be invoked.

```
public void setUpCalls() {
  boolean firstCall = false;
  boolean secondCall = false;

  if (fCall > 0) {
    setUpFCall();
    firstCall = true;
  }
  if (sCall > 0) {
    setUpSCall();
    firstCall = true;
  }

  if (firstCall && secondCall) {
    setUpDualCall();
  }
}
```

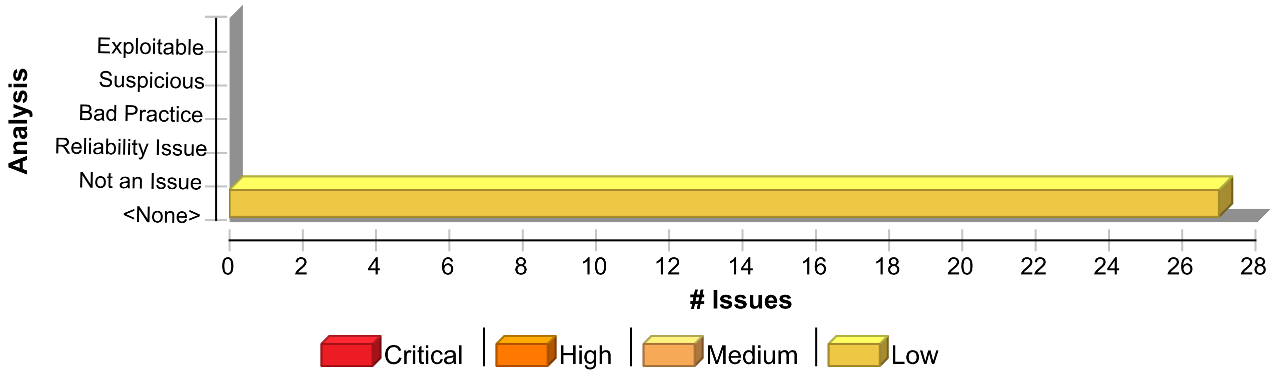
Example 2: The following method never sets the variable `firstCall` to `true`. (The variable `firstCall` is mistakenly set to `false` after the first conditional statement.) The result is that the first part of the expression `firstCall && secondCall` will always evaluate to `false`.

```
public void setUpCalls() {
  boolean firstCall = false;
  boolean secondCall = false;

  if (fCall > 0) {
    setUpFCall();
    firstCall = false;
  }
  if (sCall > 0) {
    setUpSCall();
    secondCall = true;
  }

  if (firstCall && secondCall) {
    setUpForCall();
  }
}
```

Recommendation

In general, you should repair or remove unused code. It causes additional complexity and maintenance burden without contributing to the functionality of the program.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Dead Code: Expression is Always false	27	0	0	27
Total	**27**	**0**	**0**	**27**

Dead Code: Expression is Always false — Low

Package: akka.remote

main/scala/akka/remote/RemoteWatcher.scala, line 205 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: quarantine()
File: main/scala/akka/remote/RemoteWatcher.scala:205
Taint Flags:

```
202
203 def quarantine(address: Address, uid: Option[Long], reason: String, harmless: Boolean): Unit = {
204 remoteProvider.transport match {
205 case t: ArteryTransport if harmless => t.quarantine(address, uid, reason, harmless)
206 case _ => remoteProvider.quarantine(address, uid, reason)
207 }
208 }
```

test/scala/akka/remote/Ticket1978CommunicationSpec.scala, line 198 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Dead Code: Expression is Always false — Low

Package: akka.remote

test/scala/akka/remote/Ticket1978CommunicationSpec.scala, line 198 (Dead Code: Expression is Always false) — Low

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/Ticket1978CommunicationSpec.scala:198
Taint Flags:

```
195 }
196
197 for (i <- 1 to 1000) here ! (("ping", i))
198 for (i <- 1 to 1000) expectMsgPF() { case (("pong", `i`), `testActor`) => true }
199 }
200
201 "support ask" in within(timeout.duration) {
```

test/scala/akka/remote/Ticket1978CommunicationSpec.scala, line 151 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/Ticket1978CommunicationSpec.scala:151
Taint Flags:

```
148 ("-") must {
149 if (cipherConfig.runTest && preCondition) {
150 other.actorOf(Props(new Actor {
151 def receive = { case ("ping", x) => sender() ! ((("pong", x), sender())) }
152 }), "echo")
153
154 val otherAddress =
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 376 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: actorOf()

Dead Code: Expression is Always false — Low

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 376 (Dead Code: Expression is Always false) — Low

File: main/scala/akka/remote/RemoteActorRefProvider.scala:376
Taint Flags:

```
373 deploy: Option[Deploy],
374 lookupDeploy: Boolean,
375 async: Boolean): InternalActorRef =
376 if (systemService) local.actorOf(system, props, supervisor, path, systemService, deploy, lookupDeploy, async)
377 else {
378
379 /*
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 409 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: lookupRemotes()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:409
Taint Flags:

```
406 p.headOption match {
407 case None => None
408 case Some("remote") => lookupRemotes(p.drop(3))
409 case Some("user") => deployer.lookup(p.drop(1))
410 case Some(_) => None
411 }
412 }
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 408 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: lookupRemotes()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:408
Taint Flags:

```
405 def lookupRemotes(p: Iterable[String]): Option[Deploy] = {
```

Dead Code: Expression is Always false — Low

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 408 (Dead Code: Expression is Always false) — Low

```
p.headOption match {
case None => None
case Some("remote") => lookupRemotes(p.drop(3))
case Some("user") => deployer.lookup(p.drop(1))
case Some(_) => None
}
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 686 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: getChild()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:686
Taint Flags:

```
val s = name.toStream
s.headOption match {
case None => this
case Some("..") => getParent.getChild(name)
case _ => new RemoteActorRef(remote, localAddressToUse, path / s, Nobody, props = None, deploy = None)
}
}
```

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 932 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: triggerCompressionAdvertisements()
File: main/scala/akka/remote/artery/ArteryTransport.scala:932
Taint Flags:

```
case OptionVal.Some(c) if actorRef || manifest =>
log.info("Triggering compression table advertisement for {}", c)
if (actorRef) c.runNextActorRefAdvertisement()
```

Dead Code: Expression is Always false — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 932 (Dead Code: Expression is Always false) — Low

```
if (manifest) c.runNextClassManifestAdvertisement()
case _ =>
}
}
```

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 18 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:18
Taint Flags:

```
var target: ActorRef = context.system.deadLetters

def receive = {
case "throwInvalidActorNameException" =>
// InvalidActorNameException is supported by akka-misc
throw InvalidActorNameException("wrong name")
case "throwException" =>
```

test/scala/akka/remote/artery/FlushOnShutdownSpec.scala, line 32 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/artery/FlushOnShutdownSpec.scala:32
Taint Flags:

```

val actorOnSystemB = remoteSystem.actorOf(Props(new Actor {
def receive = {
case "start" =>
context.actorSelection(rootActorPath(localSystem) / "user" / "receiver") ! Identify(None)

```

Dead Code: Expression is Always false — Low

Package: akka.remote.artery

test/scala/akka/remote/artery/FlushOnShutdownSpec.scala, line 32 (Dead Code: Expression is Always false) — Low

```
35 case ActorIdentity(_, Some(receiverRef)) =>
```

test/scala/akka/remote/artery/RemoteDeploymentSpec.scala, line 21 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/artery/RemoteDeploymentSpec.scala:21
Taint Flags:

```
18 case "throwInvalidActorNameException" =>
19 // InvalidActorNameException is supported by akka-misc
20 throw InvalidActorNameException("wrong name")
21 case "throwException" =>
22 // no specific serialization binding for Exception
23 throw new Exception("crash")
24 case x =>
```

test/scala/akka/remote/artery/RemoteSendConsistencySpec.scala, line 92 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/artery/RemoteSendConsistencySpec.scala:92
Taint Flags:

```
89 "be able to identify a remote actor and ping it" in {
90 systemB.actorOf(Props(new Actor {
91 def receive = {
92 case "ping" => sender() ! "pong"
93 }
94 }), "echo")
95
```

Dead Code: Expression is Always false — Low

Package: akka.remote.artery.tcp

main/scala/akka/remote/artery/tcp/SecureRandomFactory.scala, line 35 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: createSecureRandom()
File: main/scala/akka/remote/artery/tcp/SecureRandomFactory.scala:35
Taint Flags:

```
32
33 def createSecureRandom(randomNumberGenerator: String, log: MarkerLoggingAdapter): SecureRandom = {
34 val rng = randomNumberGenerator match {
35 case "" | GeneratorJdkSecureRandom =>
36 log.debug("Using platform default SecureRandom algorithm for SSL")
37 new SecureRandom
38 case custom =>
```

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 57 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/RemotingSpec.scala:57
Taint Flags:

```
54 case "ping" => sender() ! (("pong", sender()))
55 case a: ActorRef => a ! (("ping", sender()))
56 case ("ping", a: ActorRef) => sender() ! (("pong", a))
57 case ("pong", a: ActorRef) => a ! (("pong", sender().path.toSerializationFormat))
58 }
59 }
60
```

test/scala/akka/remote/classic/ActorsLeakSpec.scala, line 66 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality

Dead Code: Expression is Always false — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/ActorsLeakSpec.scala, line 66 (Dead Code: Expression is Always false) — Low

Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/ActorsLeakSpec.scala:66
Taint Flags:

```
63
64 class StoppableActor extends Actor {
65 override def receive = {
66 case "stop" => context.stop(self)
67 }
68 }
69
```

test/scala/akka/remote/classic/RemotingSpec.scala, line 54 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/RemotingSpec.scala:54
Taint Flags:

```
51
52 class Echo2 extends Actor {
53 def receive = {
54 case "ping" => sender() ! (("pong", sender()))
55 case a: ActorRef => a ! (("ping", sender()))
56 case ("ping", a: ActorRef) => sender() ! (("pong", a))
57 case ("pong", a: ActorRef) => a ! (("pong", sender().path.toSerializationFormat))
```

test/scala/akka/remote/classic/RemotingSpec.scala, line 56 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Dead Code: Expression is Always false — Low

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 56 (Dead Code: Expression is Always false) — Low

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/RemotingSpec.scala:56
Taint Flags:

```
53 def receive = {
54 case "ping" => sender() ! (("pong", sender()))
55 case a: ActorRef => a ! (("ping", sender()))
56 case ("ping", a: ActorRef) => sender() ! (("pong", a))
57 case ("pong", a: ActorRef) => a ! (("pong", sender().path.toSerializationFormat))
58 }
59 }
```

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 43 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:43
Taint Flags:

```
40
41 class Echo extends Actor {
42 override def receive = {
43 case "ping" => sender() ! "pong"
44 case x => sender() ! x
45 }
46 }
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 142 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()

Dead Code: Expression is Always false — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 142 (Dead Code: Expression is Always false) — Low

File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:142
Taint Flags:

```
139
140 here ! "Cleanup"
141 fishForMessage(5.seconds) {
142 case "Cleanup" => true
143 case Lost("Blackhole 3") => false
144 }
145 }
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 68 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:68
Taint Flags:

```
65 self ! "sendNext"
66 messageCount -= 1
67 }
68 case "pong" =>
69 received += 1
70 if (received >= MessageCount) controller ! (System.nanoTime() - startTime)
71 }
```

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 88 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala:88
Taint Flags:

```
85 override def preStart(): Unit = self ! "sendnext"
```

Dead Code: Expression is Always false — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/SystemMessageDeliveryStressTest.scala, line 88 (Dead Code: Expression is Always false) — Low

```
86
87 override def receive = {
88 case "sendnext" =>
89 targetRef.sendSystemMessage(Failed(child, null, counter))
90 counter += 1
91 burstCounter += 1
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 143 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:143
Taint Flags:

```
140 here ! "Cleanup"
141 fishForMessage(5.seconds) {
142 case "Cleanup" => true
143 case Lost("Blackhole 3") => false
144 }
145 }
146
```

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 62 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:62
Taint Flags:

```
59 case "start" =>
60 self ! "sendNext"
61 startTime = System.nanoTime()
62 case "sendNext" =>
```

Dead Code: Expression is Always false — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 62 (Dead Code: Expression is Always false) — Low

```
63 if (messageCount > 0) {
64 remote ! "ping"
65 self ! "sendNext"
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 60 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:60
Taint Flags:

```
57 var losses = 0
58
59 def receive = {
60 case "start" => self ! "sendNext"
61 case "sendNext" =>
62 if (nextSeq < limit) {
63 remote ! nextSeq
```

test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala, line 61 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/AkkaProtocolStressTest.scala:61
Taint Flags:

```
58
59 def receive = {
60 case "start" => self ! "sendNext"
61 case "sendNext" =>
62 if (nextSeq < limit) {
63 remote ! nextSeq
64 nextSeq += 1
```

Dead Code: Expression is Always false — Low

Package: akka.remote.classic.transport

test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala, line 59 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: applyOrElse()
File: test/scala/akka/remote/classic/transport/ThrottlerTransportAdapterSpec.scala:59
Taint Flags:

```
56 var startTime = 0L
57
58 override def receive = {
59 case "start" =>
60 self ! "sendNext"
61 startTime = System.nanoTime()
62 case "sendNext" =>
```

Package: test.scala.akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 228 (Dead Code: Expression is Always false) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: apply()
File: test/scala/akka/remote/classic/RemotingSpec.scala:228
Taint Flags:

```
225
226 "support ask" in {
227 Await.result(here ? "ping", timeout.duration) match {
228 case ("pong", _: akka.pattern.PromiseActorRef) => // good
229 case m => fail("" + m + " was not (pong, AskActorRef)")
230 }
231 }
```

Dead Code: Expression is Always true (6 issues)

Abstract

This expression will always evaluate to `true`.

Explanation

This expression will always evaluate to `true`; the program could be rewritten in a simpler form. The nearby code may be present for debugging purposes, or it may not have been maintained along with the rest of the program. The expression may also be indicative of a bug earlier in the method. **Example 1:** The following method never sets the variable `secondCall` after initializing it to `true`. (The variable `firstCall` is mistakenly used twice.) The result is that the expression `firstCall || secondCall` will always evaluate to `true`, so `setUpForCall()` will always be invoked.

```
public void setUpCalls() {
  boolean firstCall = true;
  boolean secondCall = true;

  if (fCall < 0) {
    cancelFCall();
    firstCall = false;
  }
  if (sCall < 0) {
    cancelSCall();
    firstCall = false;
  }

  if (firstCall || secondCall) {
    setUpForCall();
  }
}
```

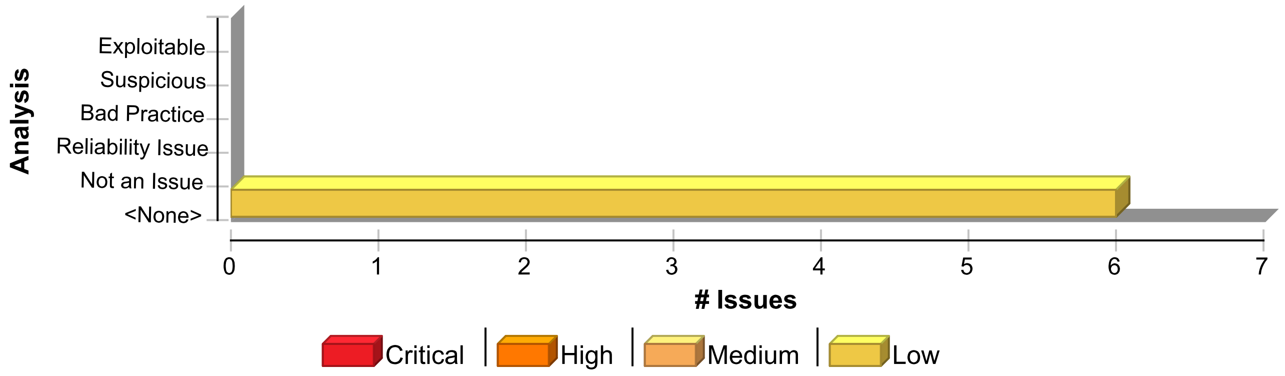
Example 2: The following method tries to check the variables `firstCall` and `secondCall`. (The variable `firstCall` is mistakenly set to `true` instead of being checked.) The result is that the first part of the expression `firstCall = true && secondCall == true` will always evaluate to `true`.

```
public void setUpCalls() {
  boolean firstCall = false;
  boolean secondCall = false;

  if (fCall > 0) {
    setUpFCall();
    firstCall = true;
  }
  if (sCall > 0) {
    setUpSCall();
    secondCall = true;
  }

  if (firstCall = true && secondCall == true) {
    setUpDualCall();
  }
}
```

Recommendation

In general, you should repair or remove unused code. It causes additional complexity and maintenance burden without contributing to the functionality of the program.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Dead Code: Expression is Always true	6	0	0	6
Total	**6**	**0**	**0**	**6**

Dead Code: Expression is Always true **Low**

Package: akka.remote

main/scala/akka/remote/RemoteActorRefProvider.scala, line 416 (Dead Code: Expression is Always true) **Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: actorOf()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:416
Taint Flags:

```
413
414 val elems = path.elements
415 val lookup =
416 if (lookupDeploy)
417 elems.head match {
418 case "user" | "system" => deployer.lookup(elems.drop(1))
419 case "remote" => lookupRemotes(elems)
```

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 929 (Dead Code: Expression is Always true) **Low**

Issue Details

Dead Code: Expression is Always true — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 929 (Dead Code: Expression is Always true) — Low

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: triggerCompressionAdvertisements()
File: main/scala/akka/remote/artery/ArteryTransport.scala:929
Taint Flags:

```
926 /** INTERNAL API: for testing only. */
927 private[remote] def triggerCompressionAdvertisements(actorRef: Boolean, manifest: Boolean) = {
928 inboundCompressionAccess match {
929 case OptionVal.Some(c) if actorRef || manifest =>
930 log.info("Triggering compression table advertisement for {}", c)
931 if (actorRef) c.runNextActorRefAdvertisement()
932 if (manifest) c.runNextClassManifestAdvertisement()
```

main/scala/akka/remote/artery/ArteryTransport.scala, line 931 (Dead Code: Expression is Always true) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: triggerCompressionAdvertisements()
File: main/scala/akka/remote/artery/ArteryTransport.scala:931
Taint Flags:

```
928 inboundCompressionAccess match {
929 case OptionVal.Some(c) if actorRef || manifest =>
930 log.info("Triggering compression table advertisement for {}", c)
931 if (actorRef) c.runNextActorRefAdvertisement()
932 if (manifest) c.runNextClassManifestAdvertisement()
933 case _ =>
934 }
```

Package: akka.remote.serialization

main/scala/akka/remote/serialization/ProtobufSerializer.scala, line 73 (Dead Code: Expression is Always true) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Dead Code: Expression is Always true — Low

Package: akka.remote.serialization

main/scala/akka/remote/serialization/ProtobufSerializer.scala, line 73 (Dead Code: Expression is Always true) — Low

Sink Details

Sink: IfStatement
Enclosing Method: parsingMethod()
File: main/scala/akka/remote/serialization/ProtobufSerializer.scala:73
Taint Flags:

```
70  case None =>
71  checkAllowedClass(clazz)
72  val unCachedParsingMethod =
73  if (method eq null) clazz.getDeclaredMethod("parseFrom", ProtobufSerializer.ARRAY_OF_BYTE_ARRAY: _*)
74  else method
75  if (parsingMethodBindingRef.compareAndSet(
76  parsingMethodBinding,
```

main/scala/akka/remote/serialization/ProtobufSerializer.scala, line 99 (Dead Code: Expression is Always true) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: IfStatement
Enclosing Method: toByteArrayMethod()
File: main/scala/akka/remote/serialization/ProtobufSerializer.scala:99
Taint Flags:

```
96  case Some(cachedtoByteArrayMethod) => cachedtoByteArrayMethod
97  case None =>
98  val unCachedtoByteArrayMethod =
99  if (method eq null) clazz.getMethod("toByteArray")
100  else method
101  if (toByteArrayMethodBindingRef.compareAndSet(
102  toByteArrayMethodBinding,
```

Package: test.scala.akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/SecureRandomFactorySpec.scala, line 40 (Dead Code: Expression is Always true) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Dead Code: Expression is Always true — Low

Package: test.scala.akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/SecureRandomFactorySpec.scala, line 40 (Dead Code: Expression is Always true) — Low

Sink: IfStatement
Enclosing Method: apply()
File: test/scala/akka/remote/artery/tcp/SecureRandomFactorySpec.scala:40
Taint Flags:

```
}

s"Artery's Secure Random support ($alg)" must {
if (isSupported) {
"generate random" in {
val bytes = Array.ofDim[Byte](16)
// Reproducer of the specific issue described at
```

Denial of Service (10 issues)

Abstract

An attacker could cause the program to crash or otherwise become unavailable to legitimate users.

Explanation

Attackers may be able to deny service to legitimate users by flooding the application with requests, but flooding attacks can often be defused at the network layer. More problematic are bugs that allow an attacker to overload the application using a small number of requests. Such bugs allow the attacker to specify the quantity of system resources their requests will consume or the duration for which they will use them. **Example 1:** The following code allows a user to specify the amount of time for which a thread will sleep. By specifying a large number, an attacker may tie up the thread indefinitely. With a small number of requests, the attacker may deplete the application's thread pool.

```
  int usrSleepTime = Integer.parseInt(usrInput);
  Thread.sleep(usrSleepTime);
```

Example 2: The following code reads a String from a zip file. Because it uses the `readLine()` method, it will read an unbounded amount of input. An attacker may take advantage of this code to cause an `OutOfMemoryException` or to consume a large amount of memory so that the program spends more time performing garbage collection or runs out of memory during some subsequent operation.

```
  InputStream zipInput = zipFile.getInputStream(zipEntry);
  Reader zipReader = new InputStreamReader(zipInput);
  BufferedReader br = new BufferedReader(zipReader);
  String line = br.readLine();
```

Recommendation

Validate user input to ensure that it will not cause inappropriate resource utilization. **Example 3:** The following code allows a user to specify the amount of time for which a thread will sleep just as in `Example 1`, but only if the value is within reasonable bounds.

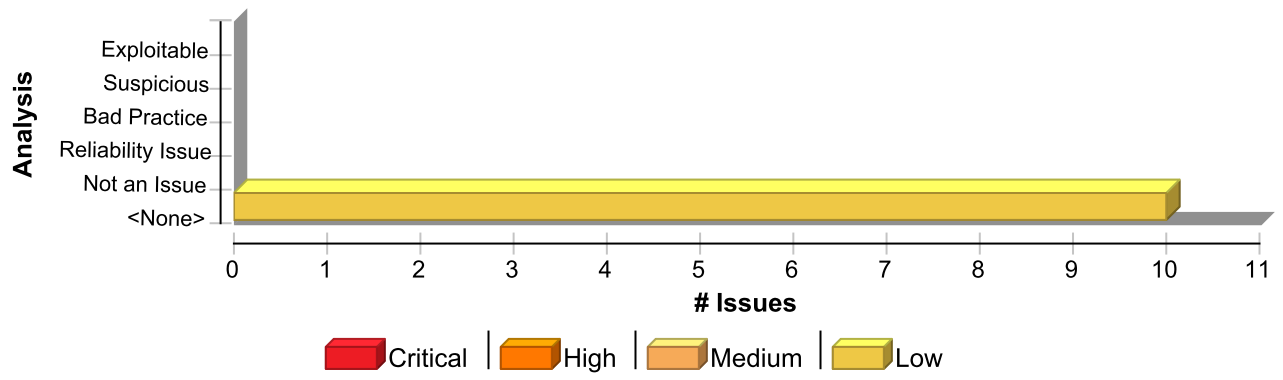
```
  int usrSleepTime = Integer.parseInt(usrInput);
  if (usrSleepTime >= SLEEP_MIN &&
      usrSleepTime <= SLEEP_MAX) {
    Thread.sleep(usrSleepTime);
  } else {
    throw new Exception("Invalid sleep duration");
  }
}
```

Example 4: The following code reads a String from a zip file just as in `Example 2`, but the maximum string length it will read is `MAX_STR_LEN` characters.

```
  InputStream zipInput = zipFile.getInputStream(zipEntry);
  Reader zipReader = new InputStreamReader(zipInput);
  BufferedReader br = new BufferedReader(zipReader);
  StringBuffer sb = new StringBuffer();
  int intC;
  while ((intC = br.read()) != -1) {
    char c = (char) intC;
    if (c == '\n') {
      break;
    }
    if (sb.length() >= MAX_STR_LEN) {
      throw new Exception("input too long");
    }
    sb.append(c);
  }
```

```
String line = sb.toString();
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Denial of Service	10	0	0	10
Total	**10**	**0**	**0**	**10**

Denial of Service — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 85 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: enableNetworkDrop()
File: test/scala/akka/remote/NetworkFailureSpec.scala:85
Taint Flags:

```
82
83 def enableNetworkDrop() = {
84 restoreIP()
85 assert(
86 new ProcessBuilder("ipfw", "add", "1", "deny", "tcp", "from", "any", "to", "any", PortRange).start.waitFor == 0)
87 }
88
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 100 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Denial of Service — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 100 (Denial of Service) — Low

Sink Details

Sink: waitFor()
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:100
Taint Flags:

```
97  assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
98  assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
99  assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
100  assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
101  }
102 }
103
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 99 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:99
Taint Flags:

```
96  println("===>>> Restoring network")
97  assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
98  assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
99  assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
100  assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
101  }
102 }
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 98 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:98
Taint Flags:

```
95  def restoreIP() = {
```

Denial of Service — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 98 (Denial of Service) — Low

```
96  println("===>>> Restoring network")
97  assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
98  assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
99  assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
100  assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
101  }
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 97 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: restoreIP()
File: test/scala/akka/remote/NetworkFailureSpec.scala:97
Taint Flags:

```
94
95  def restoreIP() = {
96  println("===>>> Restoring network")
97  assert(new ProcessBuilder("ipfw", "del", "pipe", "1").start.waitFor == 0)
98  assert(new ProcessBuilder("ipfw", "del", "pipe", "2").start.waitFor == 0)
99  assert(new ProcessBuilder("ipfw", "flush").start.waitFor == 0)
100  assert(new ProcessBuilder("ipfw", "pipe", "flush").start.waitFor == 0)
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 79 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:79
Taint Flags:

```
76  assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77  assert(
78  new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
79  assert(
80  new ProcessBuilder("ipfw", "pipe", "2", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
81  }
82
```

Denial of Service | Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 79 (Denial of Service) | Low

test/scala/akka/remote/NetworkFailureSpec.scala, line 77 (Denial of Service) | Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:77
Taint Flags:

```
74 restoreIP()
75 assert(new ProcessBuilder("ipfw", "add", "pipe", "1", "ip", "from", "any", "to", "any").start.waitFor == 0)
76 assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77 assert(
78 new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
79 assert(
80 new ProcessBuilder("ipfw", "pipe", "2", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 76 (Denial of Service) | Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:76
Taint Flags:

```
73 def enableNetworkThrottling() = {
74 restoreIP()
75 assert(new ProcessBuilder("ipfw", "add", "pipe", "1", "ip", "from", "any", "to", "any").start.waitFor == 0)
76 assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77 assert(
78 new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
79 assert(
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 75 (Denial of Service) | Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Denial of Service — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 75 (Denial of Service) — Low

Sink Details

Sink: waitFor()
Enclosing Method: enableNetworkThrottling()
File: test/scala/akka/remote/NetworkFailureSpec.scala:75
Taint Flags:

```
72
73  def enableNetworkThrottling() = {
74  restoreIP()
75  assert(new ProcessBuilder("ipfw", "add", "pipe", "1", "ip", "from", "any", "to", "any").start.waitFor == 0)
76  assert(new ProcessBuilder("ipfw", "add", "pipe", "2", "ip", "from", "any", "to", "any").start.waitFor == 0)
77  assert(
78  new ProcessBuilder("ipfw", "pipe", "1", "config", "bw", BytesPerSecond, "delay", DelayMillis).start.waitFor == 0)
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 91 (Denial of Service) — Low

Issue Details

Kingdom: Input Validation and Representation
Scan Engine: SCA (Semantic)

Sink Details

Sink: waitFor()
Enclosing Method: enableTcpReset()
File: test/scala/akka/remote/NetworkFailureSpec.scala:91
Taint Flags:

```
88
89  def enableTcpReset() = {
90  restoreIP()
91  assert(
92  new ProcessBuilder("ipfw", "add", "1", "reset", "tcp", "from", "any", "to", "any", PortRange).start.waitFor == 0)
93  }
94
```

Insecure Randomness (19 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.

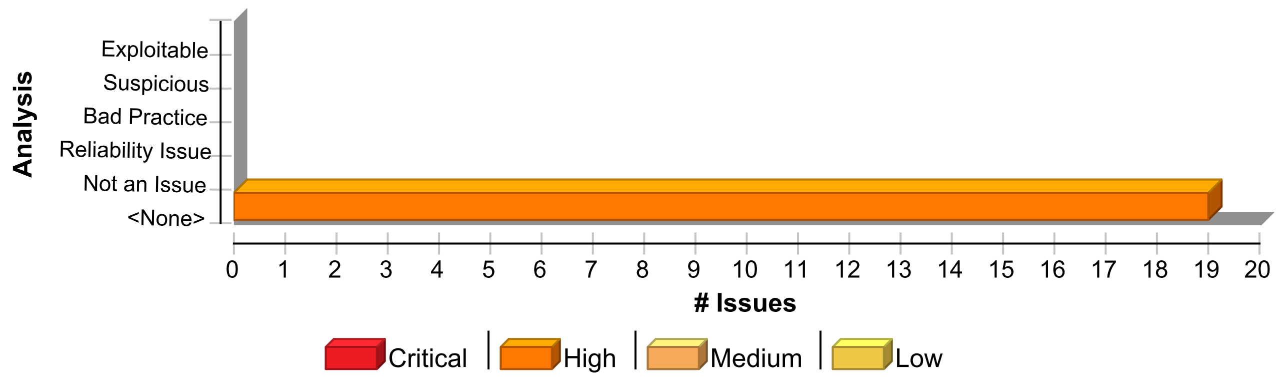
```
String GenerateReceiptURL(String baseUrl) {
    Random ranGen = new Random();
    ranGen.setSeed((new Date()).getTime());
    return (baseUrl + ranGen.nextInt(400000000) + ".html");
}
```

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	19	0	0	19
Total	**19**	**0**	**0**	**19**

Insecure Randomness — High

Package: akka.remote

test/scala/akka/remote/AckedDeliverySpec.scala, line 264 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextDouble()
Enclosing Method: happened()
File: test/scala/akka/remote/AckedDeliverySpec.scala:264
Taint Flags:

```
261
262  "SendBuffer and ReceiveBuffer" must {
263
264  def happened(p: Double) = ThreadLocalRandom.current().nextDouble() < p
265
266  @tailrec def geom(p: Double, limit: Int, acc: Int = 0): Int =
267  if (acc == limit) acc
```

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteMessageSerializationSpec.scala, line 89 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()

Insecure Randomness — High

Package: akka.remote.artery

test/scala/akka/remote/artery/RemoteMessageSerializationSpec.scala, line 89 (Insecure Randomness) — High

Enclosing Method: verifySend()
File: test/scala/akka/remote/artery/RemoteMessageSerializationSpec.scala:89
Taint Flags:

```
86 }
87
88 private def verifySend(msg: Any)(afterSend: => Unit): Unit = {
89 val bigBounceId = s"bigBounce-${ThreadLocalRandom.current.nextInt()}"
90 val bigBounceOther = remoteSystem.actorOf(Props(new Actor {
91 def receive = {
92 case x: Int => sender() ! byteStringOfSize(x)
```

test/scala/akka/remote/artery/RollingEventLogSimulationSpec.scala, line 132 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: chooseWriter()
File: test/scala/akka/remote/artery/RollingEventLogSimulationSpec.scala:132
Taint Flags:

```
129 var log: List[String] = Nil
130
131 @tailrec private def chooseWriter: Writer = {
132 val idx = Random.nextInt(writerCount)
133 val writer = writers(idx)
134 if (writer.isFinished) chooseWriter
135 else writer
```

test/scala/akka/remote/artery/TestContext.scala, line 35 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextDouble()
Enclosing Method: sendControl()
File: test/scala/akka/remote/artery/TestContext.scala:35
Taint Flags:

```
32 private val associationsByUid = new ConcurrentHashMap[Long, OutboundContext]()
```

Insecure Randomness — High

Package: akka.remote.artery

test/scala/akka/remote/artery/TestContext.scala, line 35 (Insecure Randomness) — High

```
33
34  override def sendControl(to: Address, message: ControlMessage) = {
35  if (ThreadLocalRandom.current().nextDouble() >= replyDropRate)
36  association(to).sendControl(message)
37  }
38
```

Package: akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/TcpFramingSpec.scala, line 39 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: next()
File: test/scala/akka/remote/artery/tcp/TcpFramingSpec.scala:39
Taint Flags:

```
36  override def hasNext: Boolean = remaining.nonEmpty
37
38  override def next(): ByteString = {
39  val chunkSize = rnd.nextInt(remaining.size) + 1 // no 0 length frames
40  val chunk = remaining.take(chunkSize)
41  remaining = remaining.drop(chunkSize)
42  chunk
```

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 166 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: verifySend()
File: test/scala/akka/remote/classic/RemotingSpec.scala:166
Taint Flags:

```
163  val here = RARP(system).provider.resolveActorRef("akka.test://remote-sys@localhost:12346/user/echo")
164
165  private def verifySend(msg: Any)(afterSend: => Unit): Unit = {
166  val bigBounceId = s"bigBounce-${ThreadLocalRandom.current.nextInt()}"
```

Insecure Randomness — High

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 166 (Insecure Randomness) — High

```
167 val bigBounceOther = remoteSystem.actorOf(Props(new Actor {
168 def receive = {
169 case x: Int => sender() ! byteStringOfSize(x)
```

Package: akka.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 149 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextDouble()
Enclosing Method: shouldDropOutbound()
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:149
Taint Flags:

```
146 chaosMode(remoteAddress) match {
147 case PassThru => false
148 case Drop(outboundDropP, _) =>
149 if (rng.nextDouble() <= outboundDropP) {
150 if (shouldDebugLog)
151 log.debug("Dropping outbound [{}] for [{}] {}", instance.getClass, remoteAddress, debugMessage)
152 true
```

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 138 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextDouble()
Enclosing Method: shouldDropInbound()
File: main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala:138
Taint Flags:

```
135 chaosMode(remoteAddress) match {
136 case PassThru => false
137 case Drop(_, inboundDropP) =>
138 if (rng.nextDouble() <= inboundDropP) {
139 if (shouldDebugLog)
140 log.debug("Dropping inbound [{}] for [{}] {}", instance.getClass, remoteAddress, debugMessage)
141 true
```

Insecure Randomness	High

Package: akka.remote.transport

main/scala/akka/remote/transport/FailureInjectorTransportAdapter.scala, line 138 (Insecure Randomness)	High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 154 (Insecure Randomness)	High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:154
Taint Flags:

```
watch(remoteRef)
unwatch(remoteRef)
}
Thread.sleep((idleTimeout - 10.millis).toMillis + rnd.nextInt(20))
}

watch(remoteRef)
```

test/scala/akka/remote/artery/ImmutableLongMapSpec.scala, line 149 (Insecure Randomness)	High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/ImmutableLongMapSpec.scala:149
Taint Flags:

```

(1 to 1000).foreach { i =>
withClue(s"seed=$seed, iteration=$i") {
val key = rnd.nextInt(100)
val value = String.valueOf(rnd.nextPrintableChar())
rnd.nextInt(3) match {
case 0 | 1 =>
```

Insecure Randomness — High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 342 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:342
Taint Flags:

```
339 watch(remoteRef)
340 unwatch(remoteRef)
341 }
342 Thread.sleep((idleTimeout - 10.millis).toMillis + rnd.nextInt(20))
343 }
344
345 watch(remoteRef)
```

test/scala/akka/remote/artery/LruBoundedCacheSpec.scala, line 249 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextString()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LruBoundedCacheSpec.scala:249
Taint Flags:

```
246 val cache = new TestCache(1024, 600, seed.toString)
247
248 // Fill up cache
249 for (_ <- 1 to 10000) cache.getOrCompute(Random.nextString(32))
250
251 val stats = cache.stats
252 // Have not seen lower than 890
```

test/scala/akka/remote/artery/ImmutableLongMapSpec.scala, line 150 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Insecure Randomness — High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/ImmutableLongMapSpec.scala, line 150 (Insecure Randomness) — High

Sink Details

Sink: nextPrintableChar()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/ImmutableLongMapSpec.scala:150
Taint Flags:

```
147 (1 to 1000).foreach { i =>
148 withClue(s"seed=$seed, iteration=$i") {
149 val key = rnd.nextInt(100)
150 val value = String.valueOf(rnd.nextPrintableChar())
151 rnd.nextInt(3) match {
152 case 0 | 1 =>
153 longMap = longMap.updated(key, value)
```

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 111 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextDouble()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:111
Taint Flags:

```
108 }
109
110 protected def randomDrop[T](dropRate: Double): Flow[T, T, NotUsed] = Flow[T].mapConcat { elem =>
111 if (ThreadLocalRandom.current().nextDouble() < dropRate) Nil
112 else List(elem)
113 }
114 }
```

test/scala/akka/remote/artery/ImmutableLongMapSpec.scala, line 151 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()

Insecure Randomness — High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/ImmutableLongMapSpec.scala, line 151 (Insecure Randomness) — High

File: test/scala/akka/remote/artery/ImmutableLongMapSpec.scala:151
Taint Flags:

```
148 withClue(s"seed=$seed, iteration=$i") {
149 val key = rnd.nextInt(100)
150 val value = String.valueOf(rnd.nextPrintableChar())
151 rnd.nextInt(3) match {
152 case 0 | 1 =>
153 longMap = longMap.updated(key, value)
154 reference = reference.updated(key, value)
```

test/scala/akka/remote/artery/LruBoundedCacheSpec.scala, line 243 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LruBoundedCacheSpec.scala:243
Taint Flags:

```
240
241 "maintain a good average probe distance" in {
242 for (_ <- 1 to 10) {
243 val seed = Random.nextInt(1024)
244 info(s"Variant $seed")
245 // Cache emulating 60% fill rate
246 val cache = new TestCache(1024, 600, seed.toString)
```

test/scala/akka/remote/artery/LruBoundedCacheSpec.scala, line 155 (Insecure Randomness) — High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LruBoundedCacheSpec.scala:155
Taint Flags:

```
152
```

Insecure Randomness High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/LruBoundedCacheSpec.scala, line 155 (Insecure Randomness) High

```
"work with a lower age threshold" in {
for (_ <- 1 to 10) {
val seed = Random.nextInt(1024)
info(s"Variant $seed")
val cache = new TestCache(4, 2, seed.toString)

```

test/scala/akka/remote/artery/LruBoundedCacheSpec.scala, line 75 (Insecure Randomness) High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextInt()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LruBoundedCacheSpec.scala:75
Taint Flags:

```

"evict oldest when full" in {
for (_ <- 1 to 10) {
val seed = Random.nextInt(1024)
info(s"Variant $seed")
val cache = new TestCache(4, 4, seed.toString)

```

Package: test.scala.akka.remote.serialization

test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala, line 119 (Insecure Randomness) High

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Semantic)

Sink Details

Sink: nextString()
Enclosing Method: apply()
File: test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala:119
Taint Flags:

```
}

"StringSerializer" must {
```

Insecure Randomness	High
Package: test.scala.akka.remote.serialization	
test/scala/akka/remote/serialization/PrimitivesSerializationSpec.scala, line 119 (Insecure Randomness)	**High**

```
val random = Random.nextString(256)
Seq("empty string" -> "", "hello" -> "hello", "árvíztrütvefúrógép" -> "árvíztrütvefúrógép", "random" -> random)
.foreach {
case (scenario, item) =>
```

J2EE Bad Practices: Sockets (7 issues)

Abstract

Socket-based communication in web applications is prone to error.

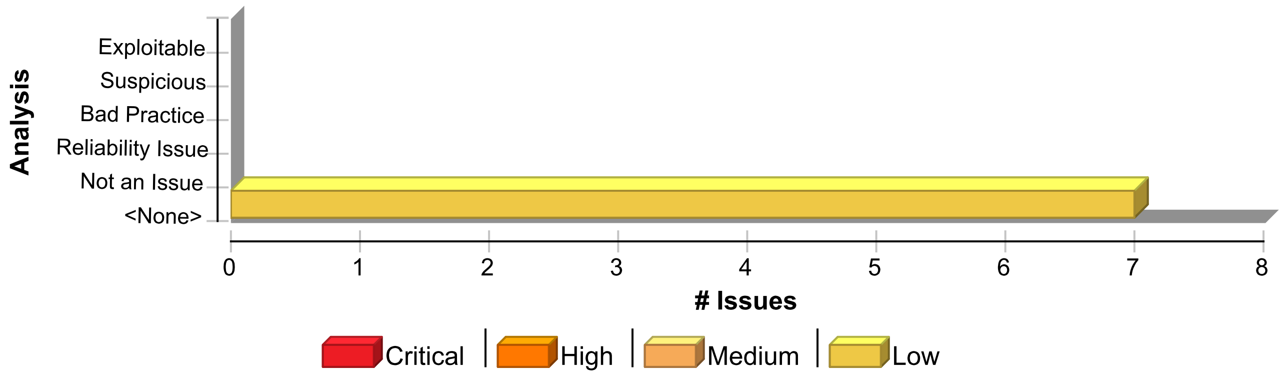
Explanation

The J2EE standard permits the use of sockets only for the purpose of communication with legacy systems when no higher-level protocol is available. Authoring your own communication protocol requires wrestling with difficult security issues, including: - In-band versus out-of-band signaling - Compatibility between protocol versions - Channel security - Error handling - Network constraints (firewalls) - Session management Without significant scrutiny by a security expert, chances are good that a custom communication protocol will suffer from security problems. Many of the same issues apply to a custom implementation of a standard protocol. While there are usually more resources available that address security concerns related to implementing a standard protocol, these resources are also available to attackers.

Recommendation

Replace a custom communication protocol with an industry standard protocol or framework. Consider whether you can use a protocol such as HTTP, FTP, SMTP, CORBA, RMI/IIOP, EJB, or SOAP. Consider the security track record of the protocol implementation you choose.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: Sockets	7	0	0	7
Total	**7**	**0**	**0**	**7**

J2EE Bad Practices: Sockets **Low**

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/ArteryAeronUdpTransport.scala, line 475 (J2EE Bad Practices: Sockets) **Low**

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

J2EE Bad Practices: Sockets — Low

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/ArteryAeronUdpTransport.scala, line 475 (J2EE Bad Practices: Sockets) — Low

Sink Details

Sink: InetSocketAddress()
Enclosing Method: autoSelectPort()
File: main/scala/akka/remote/artery/aeron/ArteryAeronUdpTransport.scala:475
Taint Flags:

```
472 import java.nio.channels.DatagramChannel
473
474 val socket = DatagramChannel.open().socket()
475 socket.bind(new InetSocketAddress(hostname, 0))
476 val port = socket.getLocalPort
477 socket.close()
478 port
```

Package: akka.remote.artery.tcp

main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala, line 122 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: createUnresolved()
Enclosing Method: outboundTransportSink()
File: main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala:122
Taint Flags:

```
119
120 val host = outboundContext.remoteAddress.host.get
121 val port = outboundContext.remoteAddress.port.get
122 val remoteAddress = InetSocketAddress.createUnresolved(host, port)
123
124 def connectionFlow: Flow[ByteString, ByteString, Future[Tcp.OutgoingConnection]] = {
125 val localAddress = settings.Advanced.Tcp.OutboundClientHostname match {
```

main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala, line 122 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: createUnresolved()

J2EE Bad Practices: Sockets — Low

Package: akka.remote.artery.tcp

main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala, line 122 (J2EE Bad Practices: Sockets) — Low

Enclosing Method: outboundTransportSink()
File: main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala:122
Taint Flags:

```
119
120 val host = outboundContext.remoteAddress.host.get
121 val port = outboundContext.remoteAddress.port.get
122 val remoteAddress = InetSocketAddress.createUnresolved(host, port)
123
124 def connectionFlow: Flow[ByteString, ByteString, Future[Tcp.OutgoingConnection]] = {
125 val localAddress = settings.Advanced.Tcp.OutboundClientHostname match {
```

main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala, line 127 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: InetSocketAddress()
Enclosing Method: connectionFlow()
File: main/scala/akka/remote/artery/tcp/ArteryTcpTransport.scala:127
Taint Flags:

```
124 def connectionFlow: Flow[ByteString, ByteString, Future[Tcp.OutgoingConnection]] = {
125 val localAddress = settings.Advanced.Tcp.OutboundClientHostname match {
126 case None => None
127 case Some(clientHostname) => Some(new InetSocketAddress(clientHostname, 0))
128 }
129 if (tlsEnabled) {
130 val sslProvider = sslEngineProvider.get
```

Package: akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 89 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: InetSocketAddress()
Enclosing Method: randomOpenServerSocket()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:89

J2EE Bad Practices: Sockets — Low

Package: akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 89 (J2EE Bad Practices: Sockets) — Low

Taint Flags:

```

def randomOpenServerSocket(address: String = InetAddress.getLocalHost.getHostAddress) = {
val ss = ServerSocketChannel.open().socket()
ss.bind(new InetSocketAddress(address, 0))
(ss, new InetSocketAddress(address, ss.getLocalPort))
}

```

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 90 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: InetSocketAddress()
Enclosing Method: randomOpenServerSocket()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:90
Taint Flags:

```
def randomOpenServerSocket(address: String = InetAddress.getLocalHost.getHostAddress) = {
val ss = ServerSocketChannel.open().socket()
ss.bind(new InetSocketAddress(address, 0))
(ss, new InetSocketAddress(address, ss.getLocalPort))
}

"bind to a specified port and remoting accepts from a bound port" in {
```

Package: main.scala.akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 496 (J2EE Bad Practices: Sockets) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: InetSocketAddress()
Enclosing Method: apply()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:496
Taint Flags:

J2EE Bad Practices: Sockets	Low

Package: main.scala.akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 496 (J2EE Bad Practices: Sockets)	Low

```
// TODO: This should be factored out to an async (or thread-isolated) name lookup service #2960
def addressToSocketAddress(addr: Address): Future[InetSocketAddress] = addr match {
case Address(_, _, Some(host), Some(port)) =>
Future { blocking { new InetSocketAddress(InetAddress.getByName(host), port) } }
case _ => Future.failed(new IllegalArgumentException(s"Address [$addr] does not contain host or port information."))
}

```

J2EE Bad Practices: Threads (23 issues)

Abstract

Thread management in a web application is forbidden in some circumstances and is always highly error prone.

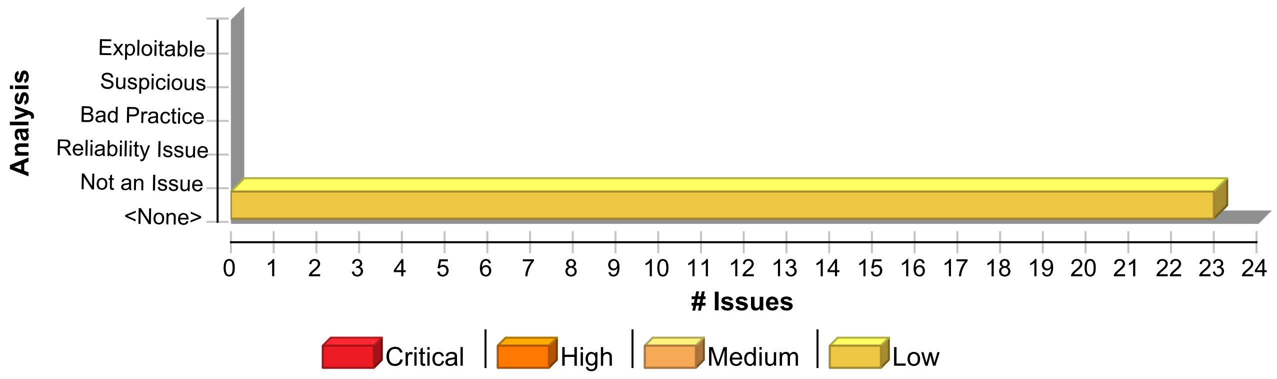
Explanation

Thread management in a web application is forbidden by the J2EE standard in some circumstances and is always highly error prone. Managing threads is difficult and is likely to interfere in unpredictable ways with the behavior of the application container. Even without interfering with the container, thread management usually leads to bugs that are hard to detect and diagnose like deadlock, race conditions, and other synchronization errors.

Recommendation

Avoid managing threads directly from within the web application. Instead use standards such as message driven beans and the EJB timer service that are provided by the application container.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
J2EE Bad Practices: Threads	23	0	0	23
Total	**23**	**0**	**0**	**23**

J2EE Bad Practices: Threads **Low**

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 70 (J2EE Bad Practices: Threads) **Low**

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: sleepFor()
File: test/scala/akka/remote/NetworkFailureSpec.scala:70
Taint Flags:

J2EE Bad Practices: Threads — Low

Package: akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 70 (J2EE Bad Practices: Threads) — Low

```
67
68 def sleepFor(duration: Duration) = {
69 println("===>>> Sleeping for [" + duration + "]")
70 Thread.sleep(duration.toMillis)
71 }
72
73 def enableNetworkThrottling() = {
```

main/scala/akka/remote/RemoteActorRefProvider.scala, line 698 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: interrupt()
Enclosing Method: applyOrElse()
File: main/scala/akka/remote/RemoteActorRefProvider.scala:698
Taint Flags:

```
695 case e: InterruptedException =>
696 remote.system.eventStream.publish(Error(e, path.toString, getClass, "interrupted during message send"))
697 remote.system.deadLetters.tell(message, sender)
698 Thread.currentThread.interrupt()
699 case NonFatal(e) =>
700 remote.system.eventStream.publish(Error(e, path.toString, getClass, "swallowing exception during message send"))
701 remote.system.deadLetters.tell(message, sender)
```

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 463 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: Thread()
Enclosing Method: ArteryTransport$$anon$1()
File: main/scala/akka/remote/artery/ArteryTransport.scala:463
Taint Flags:

```
460 }
461 }
462
```

J2EE Bad Practices: Threads — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 463 (J2EE Bad Practices: Threads) — Low

```
private lazy val shutdownHook = new Thread {
override def run(): Unit = {
if (!hasBeenShutdown.get) {
val coord = CoordinatedShutdown(system)
```

main/scala/akka/remote/artery/ArteryTransport.scala, line 369 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: addShutdownHook()
Enclosing Method: start()
File: main/scala/akka/remote/artery/ArteryTransport.scala:369
Taint Flags:

```
366
367 override def start(): Unit = {
368 if (system.settings.JvmShutdownHooks)
369 Runtime.getRuntime.addShutdownHook(shutdownHook)
370
371 startTransport()
372 flightRecorder.transportStarted()
```

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 134 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: run()
Enclosing Method: start()
File: main/scala/akka/remote/artery/aeron/TaskRunner.scala:134
Taint Flags:

```
131 m.withName(m.name + "-taskrunner")
132 case other => other
133 }
134 val thread = tf.newThread(this)
135 thread.start()
```

J2EE Bad Practices: Threads — Low

Package: akka.remote.artery.aeron

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 134 (J2EE Bad Practices: Threads) — Low

```
}

```

main/scala/akka/remote/artery/aeron/ArteryAeronUdpTransport.scala, line 256 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: retry()
File: main/scala/akka/remote/artery/aeron/ArteryAeronUdpTransport.scala:256
Taint Flags:

```
stopMediaDriver()
throw new RemoteTransportException("Inbound Aeron channel is in errored state. See Aeron logs for details.")
} else if (status == ChannelEndpointStatus.INITIALIZING && retries > 0) {
Thread.sleep(waitInterval)
retry(retries - 1)
} else {
aeronErrorLog.logErrors(log, 0L)
```

main/scala/akka/remote/artery/aeron/TaskRunner.scala, line 135 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: start()
Enclosing Method: start()
File: main/scala/akka/remote/artery/aeron/TaskRunner.scala:135
Taint Flags:

```
case other => other
}
val thread = tf.newThread(this)
thread.start()
}

def stop(): Future[Done] = {
```

J2EE Bad Practices: Threads — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 249 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: awaitCacheExpiration()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:249
Taint Flags:

```
246
247 // sleep to force the cache in sysB's instance to expire
248 def awaitCacheExpiration(): Unit = {
249 Thread.sleep((RotatingKeysSSLEngineProviderSpec.cacheTtlInSeconds + 1) * 1000)
250 }
251
252 def contact(fromSystem: ActorSystem, toPath: ActorPath): Unit = {
```

Package: akka.remote.serialization

main/scala/akka/remote/serialization/ActorRefResolveCache.scala, line 53 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: ThreadLocal()
Enclosing Method: ActorRefResolveThreadLocalCache$$anon$1()
File: main/scala/akka/remote/serialization/ActorRefResolveCache.scala:53
Taint Flags:

```
50 s"not with ${system.provider.getClass}")
51 }
52
53 private val current = new ThreadLocal[ActorRefResolveCache] {
54 override def initialValue: ActorRefResolveCache = new ActorRefResolveCache(provider)
55 }
56
```

Package: main.scala.akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 611 (J2EE Bad Practices: Threads) — Low

Issue Details

J2EE Bad Practices: Threads — Low

Package: main.scala.akka.remote.artery

main/scala/akka/remote/artery/ArteryTransport.scala, line 611 (J2EE Bad Practices: Threads) — Low

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: removeShutdownHook()
Enclosing Method: apply()
File: main/scala/akka/remote/artery/ArteryTransport.scala:611
Taint Flags:

```
608 if (hasBeenShutdown.compareAndSet(false, true)) {
609 log.debug("Shutting down [{}]", localAddress)
610 if (system.settings.JvmShutdownHooks)
611 Try(Runtime.getRuntime.removeShutdownHook(shutdownHook)) // may throw if shutdown already in progress
612 val allAssociations = associationRegistry.allAssociations
613 val flushing: Future[Done] =
614 if (allAssociations.isEmpty) Future.successful(Done)
```

Package: test.scala.akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 43 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:43
Taint Flags:

```
40 try {
41 enableNetworkThrottling()
42 println("===>>> Throttling network with [" + BytesPerSecond + ", " + DelayMillis + "] for [" + duration + "]")
43 Thread.sleep(duration.toMillis)
44 restoreIP()
45 } catch {
46 case e: Throwable =>
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 28 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

J2EE Bad Practices: Threads — Low

Package: test.scala.akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 28 (J2EE Bad Practices: Threads) — Low

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:28
Taint Flags:

```
25 try {
26 enableTcpReset()
27 println("===>>> Reply with [TCP RST] for [" + duration + "]")
28 Thread.sleep(duration.toMillis)
29 restoreIP()
30 } catch {
31 case e: Throwable =>
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 58 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:58
Taint Flags:

```
55 try {
56 enableNetworkDrop()
57 println("===>>> Blocking network [TCP DENY] for [" + duration + "]")
58 Thread.sleep(duration.toMillis)
59 restoreIP()
60 } catch {
61 case e: Throwable =>
```

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/LateConnectSpec.scala, line 41 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LateConnectSpec.scala:41
Taint Flags:

J2EE Bad Practices: Threads — Low

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/LateConnectSpec.scala, line 41 (J2EE Bad Practices: Threads) — Low

```
38 echoB ! "ping1"
39
40 // let the outbound streams be restarted (lazy), systemB is not started yet
41 Thread.sleep((RARP(system).provider.remoteSettings.Artery.Advanced.HandshakeTimeout + 1.second).toMillis)
42
43 // start systemB
44 systemB.actorOf(TestActors.echoActorProps, "echoB")
```

test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala, line 129 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala:129
Taint Flags:

```
126 regularRemote.tell(Ping(), probeSmall.ref)
127 Thread.sleep(50)
128 regularRemote.tell(Ping(), probeSmall.ref)
129 Thread.sleep(50)
130 regularRemote.tell(Ping(), probeSmall.ref)
131
132 // should be no problems sending regular small messages while large messages are being sent
```

test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala, line 127 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala:127
Taint Flags:

```
124 val largeBytes = 2000000
125 largeRemote.tell(Ping(ByteString.fromArray(new Array[Byte](largeBytes))), probeLarge.ref)
126 regularRemote.tell(Ping(), probeSmall.ref)
```

J2EE Bad Practices: Threads — Low

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/LargeMessagesStreamSpec.scala, line 127 (J2EE Bad Practices: Threads) — Low

```
127 Thread.sleep(50)
128 regularRemote.tell(Ping(), probeSmall.ref)
129 Thread.sleep(50)
130 regularRemote.tell(Ping(), probeSmall.ref)
```

test/scala/akka/remote/artery/RestartCounterSpec.scala, line 38 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/RestartCounterSpec.scala:38
Taint Flags:

```
35 counter.restart()
36 counter.restart()
37 counter.count() should ===(2)
38 Thread.sleep(600)
39 counter.restart()
40 counter.count() should ===(1)
41 }
```

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 342 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:342
Taint Flags:

```
339 watch(remoteRef)
340 unwatch(remoteRef)
341 }
342 Thread.sleep((idleTimeout - 10.millis).toMillis + rnd.nextInt(20))
343 }
344
```

J2EE Bad Practices: Threads — Low

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 342 (J2EE Bad Practices: Threads) — Low

```
345 watch(remoteRef)
```

test/scala/akka/remote/artery/RestartCounterSpec.scala, line 29 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/RestartCounterSpec.scala:29
Taint Flags:

```
26 val counter = new RestartCounter(3, 10.millis)
27 for (_ <- 1 to 10) {
28 counter.restart() should ===(true)
29 Thread.sleep(20)
30 }
31 }
32
```

test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala, line 154 (J2EE Bad Practices: Threads) — Low

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/SystemMessageDeliverySpec.scala:154
Taint Flags:

```
151 watch(remoteRef)
152 unwatch(remoteRef)
153 }
154 Thread.sleep((idleTimeout - 10.millis).toMillis + rnd.nextInt(20))
155 }
156
157 watch(remoteRef)
```

J2EE Bad Practices: Threads — Low

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/ActorRefResolveCacheQuarantineSpec.scala, line 45 (J2EE Bad Practices: Threads) — **Low**

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/ActorRefResolveCacheQuarantineSpec.scala:45
Taint Flags:

```
42  shutdown(clientSystem1)
43
44  // wait for it to be removed fully, remove-quarantined-association-after
45  Thread.sleep(4000)
46
47  val port1 = RARP(clientSystem1).provider.getDefaultAddress.getPort().get
48  val clientSystem2 =
```

Package: test.scala.akka.remote.artery.compress

test/scala/akka/remote/artery/compress/HandshakeShouldDropCompressionTableSpec.scala, line 108 (J2EE Bad Practices: Threads) — **Low**

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/compress/HandshakeShouldDropCompressionTableSpec.scala:108
Taint Flags:

```
105  (1 to messagesToExchange).foreach { i =>
106  echoSel ! s"hello-$i"
107  } // does not reply, but a hot receiver should be advertised
108  Thread.sleep(100)
109  }
110  waitForEcho(this, s"hello-$messagesToExchange", max = 10.seconds)
111  systemBTransport.triggerCompressionAdvertisements(actorRef = true, manifest = false)
```

J2EE Bad Practices: Threads	Low
Package: test.scala.akka.remote.artery.compress	
test/scala/akka/remote/artery/compress/ HandshakeShouldDropCompressionTableSpec.scala, line 96 (J2EE Bad Practices: Threads)	**Low**

Issue Details

Kingdom: Time and State
Scan Engine: SCA (Semantic)

Sink Details

Sink: sleep()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/compress/HandshakeShouldDropCompressionTableSpec.scala:96
Taint Flags:

```
93 shutdown(systemB)
94 systemB =
95 newRemoteSystem(name = Some("systemB"), extraConfig = Some(s"akka.remote.artery.canonical.port = $portB"))
96 Thread.sleep(1000)
97 log.info("SYSTEM READY {}...", systemB)
98
99 val aNewProbe = TestProbe()
```

Key Management: Hardcoded Encryption Key (3 issues)

Abstract

Hardcoded encryption keys can compromise security in a way that cannot be easily remedied.

Explanation

It is never a good idea to hardcode an encryption key because it allows all of the project's developers to view the encryption key, and makes fixing the problem extremely difficult. After the code is in production, a software patch is required to change the encryption key. If the account that is protected by the encryption key is compromised, the owners of the system must choose between security and availability. **Example 1:** The following code uses a hardcoded encryption key:

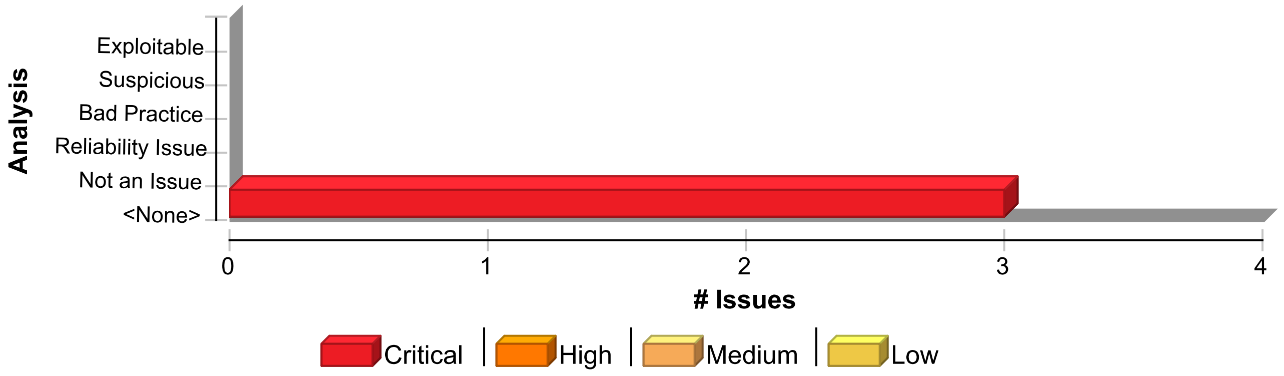
```
...
private static final String encryptionKey = "lakdsljkalkjlksdfkl";
byte[] keyBytes = encryptionKey.getBytes();
SecretKeySpec key = new SecretKeySpec(keyBytes, "AES");
Cipher encryptCipher = Cipher.getInstance("AES");
encryptCipher.init(Cipher.ENCRYPT_MODE, key);
...
```

Anyone with access to the code has access to the encryption key. After the application has shipped, there is no way to change the encryption key unless the program is patched. An employee with access to this information can use it to break into the system. If attackers had access to the executable for the application, they could extract the encryption key value.

Recommendation

Encryption keys should never be hardcoded and should be obfuscated and managed in an external source. Storing encryption keys in plain text anywhere on the system allows anyone with sufficient permissions to read and potentially misuse the encryption key.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Key Management: Hardcoded Encryption Key	3	0	0	3
Total	**3**	**0**	**0**	**3**

Key Management: Hardcoded Encryption Key — Critical

Package: akka.remote

main/scala/akka/remote/RemoteSettings.scala, line 110 (Key Management: Hardcoded Encryption Key) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: key
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:110
Taint Flags:

```
107
108 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
109 val LogBufferSizeExceeding: Int = {
110 val key = "akka.remote.classic.log-buffer-size-exceeding"
111 config.getString(key).toLowerCase match {
112 case "off" | "false" => Int.MaxValue
113 case _ => config.getInt(key)
```

main/scala/akka/remote/RemoteSettings.scala, line 110 (Key Management: Hardcoded Encryption Key) — Critical

Issue Details

Kingdom: Security Features
Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: key
Enclosing Method: RemoteSettings()
File: main/scala/akka/remote/RemoteSettings.scala:110
Taint Flags:

```
107
108 @deprecated("Classic remoting is deprecated, use Artery", "2.6.0")
109 val LogBufferSizeExceeding: Int = {
110 val key = "akka.remote.classic.log-buffer-size-exceeding"
111 config.getString(key).toLowerCase match {
112 case "off" | "false" => Int.MaxValue
113 case _ => config.getInt(key)
```

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/RemoteInstrumentsSpec.scala, line 30 (Key Management: Hardcoded Encryption Key) — Critical

Issue Details

Kingdom: Security Features

Key Management: Hardcoded Encryption Key — Critical

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/RemoteInstrumentsSpec.scala, line 30 (Key Management: Hardcoded Encryption Key) — Critical

Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: key
Enclosing Method: apply()
File: test/scala/akka/remote/artery/RemoteInstrumentsSpec.scala:30
Taint Flags:

```
27 "RemoteInstruments" must {
28
29 "combine and decompose single key and length" in {
30 val key: Byte = 17
31 val len = 812
32 val kl = RemoteInstruments.combineKeyLength(key, len)
33
```

Missing Check against Null (13 issues)

Abstract

The program can dereference a null-pointer because it does not check the return value of a function that might return `null`.

Explanation

Just about every serious attack on a software system begins with the violation of a programmer's assumptions. After the attack, the programmer's assumptions seem flimsy and poorly founded, but before an attack many programmers would defend their assumptions well past the end of their lunch break. Two dubious assumptions that are easy to spot in code are "this function call can never fail" and "it doesn't matter if this function call fails". When a programmer ignores the return value from a function, they implicitly state that they are operating under one of these assumptions.
Example 1: The following code does not check to see if the string returned by `getParameter()` is `null` before calling the member function `compareTo()`, potentially causing a `null` dereference.

```
String itemName = request.getParameter(ITEM_NAME);
    if (itemName.compareTo(IMPORTANT_ITEM)) {
        ...
    }
    ...
```

Example 2:. The following code shows a system property that is set to `null` and later dereferenced by a programmer who mistakenly assumes it will always be defined.

```
System.clearProperty("os.name");
...
String os = System.getProperty("os.name");
if (os.equalsIgnoreCase("Windows 95") )
    System.out.println("Not supported");
```

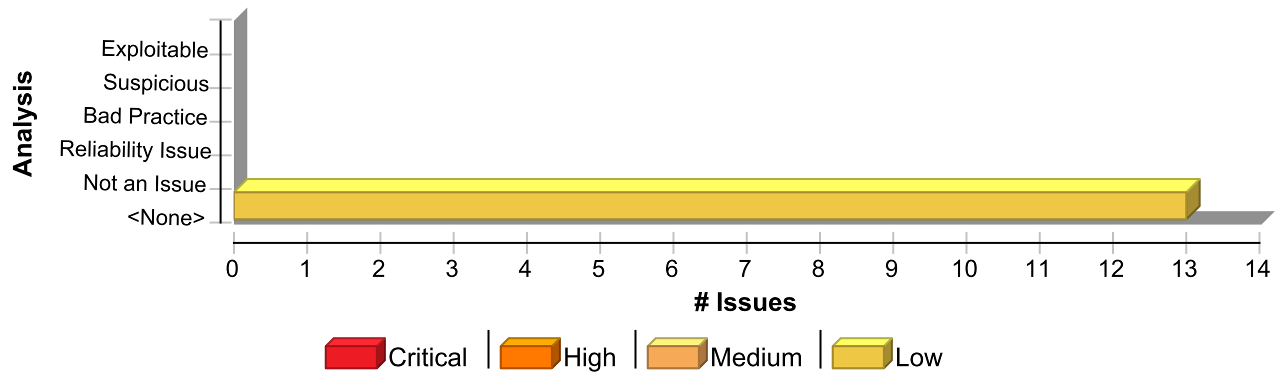
The traditional defense of this coding error is: "I know the requested value will always exist because.... If it does not exist, the program cannot perform the desired behavior so it doesn't matter whether I handle the error or simply allow the program to die dereferencing a `null` value." But attackers are skilled at finding unexpected paths through programs, particularly when exceptions are involved.

Recommendation

If a function can return an error code or any other evidence of its success or failure, always check for the error condition, even if there is no obvious way for it to occur. In addition to preventing security errors, many initially mysterious bugs have eventually led back to a failed method call with an unchecked return value. Create an easy to use and standard way for dealing with failure in your application. If error handling is straightforward, programmers will be less inclined to omit it. One approach to standardized error handling is to write wrappers around commonly-used functions that check and handle error conditions without additional programmer intervention. When wrappers are implemented and adopted, the use of non-wrapped equivalents can be prohibited and enforced by using custom rules.
Example 3: The following code implements a wrapper around `getParameter()` that checks the return value of `getParameter()` against `null` and uses a default value if the requested parameter is not defined.

```
String safeGetParameter (HttpRequest request, String name)
{
    String value = request.getParameter(name);
    if (value == null) {
        return getDefaultValue(name)
    }
    return value;
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Missing Check against Null	13	0	0	13
Total	**13**	**0**	**0**	**13**

Missing Check against Null — Low

Package: akka.remote

test/scala/akka/remote/Ticket1978CommunicationSpec.scala, line 31 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: Configuration()
File: test/scala/akka/remote/Ticket1978CommunicationSpec.scala:31
Taint Flags:

```
object Configuration {
 // set this in your JAVA_OPTS to see all ssl debug info: "-Djavax.net.debug=ssl,keymanager"
 // The certificate will expire in 2109
 private val trustStore = getClass.getClassLoader.getResource("truststore").getPath
 private val keyStore = getClass.getClassLoader.getResource("keystore").getPath
 private val conf = """
 akka {
```

test/scala/akka/remote/Ticket1978CommunicationSpec.scala, line 32 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Missing Check against Null — Low

Package: akka.remote

test/scala/akka/remote/Ticket1978CommunicationSpec.scala, line 32 (Missing Check against Null) — Low

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: Configuration()
File: test/scala/akka/remote/Ticket1978CommunicationSpec.scala:32
Taint Flags:

```
29 // set this in your JAVA_OPTS to see all ssl debug info: "-Djavax.net.debug=ssl,keymanager"
30 // The certificate will expire in 2109
31 private val trustStore = getClass.getClassLoader.getResource("truststore").getPath
32 private val keyStore = getClass.getClassLoader.getResource("keystore").getPath
33 private val conf = """
34 akka {
35 actor.provider = remote
```

Package: akka.remote.artery

test/scala/akka/remote/artery/ArterySpecSupport.scala, line 39 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: tlsConfig$lzycompute()
File: test/scala/akka/remote/artery/ArterySpecSupport.scala:39
Taint Flags:

```
36 // RotatingKeysSSLEngineProvider and, eventually, run tests twice
37 // (once for each provider).
38 lazy val tlsConfig: Config = {
39 val trustStore = getClass.getClassLoader.getResource("truststore").getPath
40 val keyStore = getClass.getClassLoader.getResource("keystore").getPath
41
42 ConfigFactory.parseString(s"""
```

test/scala/akka/remote/artery/ArterySpecSupport.scala, line 40 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL

Missing Check against Null	Low

Package: akka.remote.artery

test/scala/akka/remote/artery/ArterySpecSupport.scala, line 40 (Missing Check against Null)	Low

Enclosing Method: tlsConfig$lzycompute()
File: test/scala/akka/remote/artery/ArterySpecSupport.scala:40
Taint Flags:

```
37 // (once for each provider).
38 lazy val tlsConfig: Config = {
39 val trustStore = getClass.getClassLoader.getResource("truststore").getPath
40 val keyStore = getClass.getClassLoader.getResource("keystore").getPath
41
42 ConfigFactory.parseString(s"""
43 akka.remote.artery.ssl.config-ssl-engine {
```

Package: akka.remote.artery.tcp

test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala, line 64 (Missing Check against Null)	Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: resourcePath()
File: test/scala/akka/remote/artery/tcp/TlsTcpSpec.scala:64
Taint Flags:

```
61 """))
62
63 object TlsTcpSpec {
64 def resourcePath(name: String): String = getClass.getClassLoader.getResource(name).getPath
65
66 lazy val config: Config = {
67 ConfigFactory.parseString(s"""
```

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Missing Check against Null)	Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: RotatingKeysSSLEngineProviderSpec()

Missing Check against Null — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Missing Check against Null) — Low

File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
170 """
171
172 val resourcesConfig: String = baseConfig +
173 s"""
174 akka.remote.artery.ssl.rotating-keys-engine {
175 key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
170 """
171
172 val resourcesConfig: String = baseConfig +
173 s"""
174 akka.remote.artery.ssl.rotating-keys-engine {
175 key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: RotatingKeysSSLEngineProviderSpec()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:172
Taint Flags:

```
169 }
```

Missing Check against Null — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 172 (Missing Check against Null) — Low

```
"""

val resourcesConfig: String = baseConfig +
s"""
akka.remote.artery.ssl.rotating-keys-engine {
key-file = ${getClass.getClassLoader.getResource(s"$arteryNode001Id.pem").getPath}
```

test/scala/akka/remote/artery/tcp/ssl/PemManagersProviderSpec.scala, line 54 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: nameToPath()
File: test/scala/akka/remote/artery/tcp/ssl/PemManagersProviderSpec.scala:54
Taint Flags:

```
PemManagersProvider.loadCertificate(nameToPath(caCertFile)))
}

private def nameToPath(name: String): String = getClass.getClassLoader.getResource(name).getPath
}

undefined
```

test/scala/akka/remote/artery/tcp/ssl/TlsResourcesSpec.scala, line 86 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: toAbsolutePath()
File: test/scala/akka/remote/artery/tcp/ssl/TlsResourcesSpec.scala:86
Taint Flags:

```
object TlsResourcesSpec {

def toAbsolutePath(resourceName: String): String =
getClass.getClassLoader.getResource(resourceName).getPath
```

Missing Check against Null — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/TlsResourcesSpec.scala, line 86 (Missing Check against Null) — Low

```
87
88  private val certFactory = CertificateFactory.getInstance("X.509")
89  def loadCert(resourceName: String): X509Certificate = {
```

Package: akka.remote.classic

test/scala/akka/remote/classic/RemotingSpec.scala, line 68 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: RemotingSpec()
File: test/scala/akka/remote/classic/RemotingSpec.scala:68
Taint Flags:

```
65  }
66  }
67
68  val cfg: Config = ConfigFactory.parseString(s"""
69  common-ssl-settings {
70  key-store = "${getClass.getClassLoader.getResource("keystore").getPath}"
71  trust-store = "${getClass.getClassLoader.getResource("truststore").getPath}"
```

test/scala/akka/remote/classic/RemotingSpec.scala, line 68 (Missing Check against Null) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: RemotingSpec()
File: test/scala/akka/remote/classic/RemotingSpec.scala:68
Taint Flags:

```
65  }
66  }
67
68  val cfg: Config = ConfigFactory.parseString(s"""
69  common-ssl-settings {
70  key-store = "${getClass.getClassLoader.getResource("keystore").getPath}"
71  trust-store = "${getClass.getClassLoader.getResource("truststore").getPath}"
```

Missing Check against Null	Low

Package: test.scala.akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 199 (Missing Check against Null)	Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Control Flow)

Sink Details

Sink: getClassLoader() : Class.getClassLoader may return NULL
Enclosing Method: apply()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:199
Taint Flags:

```
196 private def deployResource(resourceName: String, to: Path): Unit = blocking {
197 // manually ensuring files are deleted and copied to prevent races.
198 try {
199 val from = new File(getClass.getClassLoader.getResource(resourceName).getPath).toPath
200 to.toFile.getParentFile.mkdirs()
201 Files.copy(from, to, StandardCopyOption.REPLACE_EXISTING)
202 } catch {
```

Null Dereference (2 issues)

Abstract

The program can potentially dereference a null-pointer, thereby causing a null-pointer exception.

Explanation

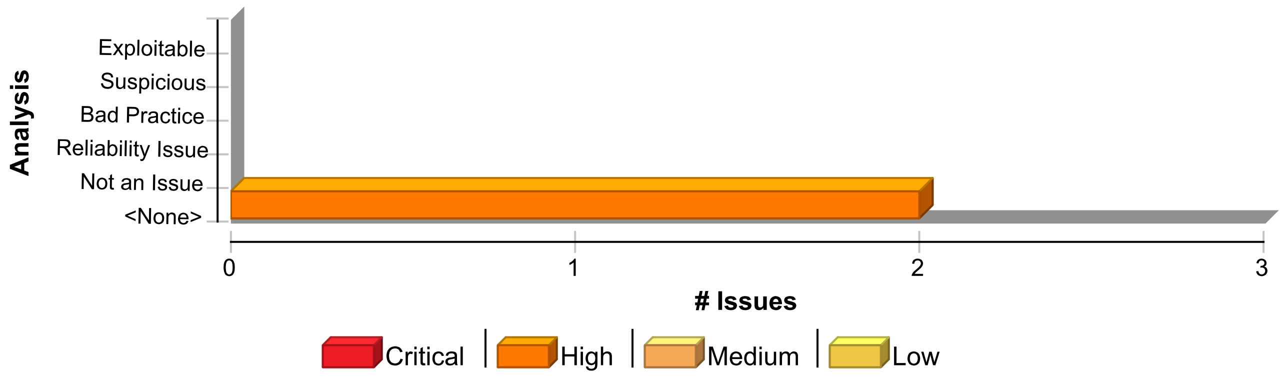
Null-pointer exceptions usually occur when one or more of the programmer's assumptions is violated. A dereference-after-store error occurs when a program explicitly sets an object to `null` and dereferences it later. This error is often the result of a programmer initializing a variable to `null` when it is declared. Most null-pointer issues result in general software reliability problems, but if attackers can intentionally trigger a null-pointer dereference, they can use the resulting exception to bypass security logic or to cause the application to reveal debugging information that will be valuable in planning subsequent attacks. **Example:** In the following code, the programmer explicitly sets the variable `foo` to `null`. Later, the programmer dereferences `foo` before checking the object for a `null` value.

```
Foo foo = null;
...
foo.setBar(val);
...
}
```

Recommendation

Implement careful checks before dereferencing objects that might be `null`. When possible, abstract `null` checks into wrappers around code that manipulates resources to ensure that they are applied in all cases and to minimize the places where mistakes can occur.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Null Dereference	2	0	0	2
Total	**2**	**0**	**0**	**2**

Null Dereference **High**

Package: akka.remote.serialization

main/scala/akka/remote/serialization/ProtobufSerializer.scala, line 64 (Null Dereference) **High**

Issue Details

Null Dereference High

Package: akka.remote.serialization

main/scala/akka/remote/serialization/ProtobufSerializer.scala, line 64 (Null Dereference) High

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Sink Details

Sink: Dereferenced : this.parsingMethod(this.parsingMethod$default$1$1(), clazz)
Enclosing Method: fromBinary()
File: main/scala/akka/remote/serialization/ProtobufSerializer.scala:64
Taint Flags:

```
61
62 override def fromBinary(bytes: Array[Byte], manifest: Option[Class[_]]): AnyRef = {
63 manifest match {
64 case Some(clazz) =>
65 @tailrec
66 def parsingMethod(method: Method = null): Method = {
67 val parsingMethodBinding = parsingMethodBindingRef.get()
```

main/scala/akka/remote/serialization/ProtobufSerializer.scala, line 109 (Null Dereference) High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Sink Details

Sink: Dereferenced : this.toByteArrayMethod(this.toByteArrayMethod$default$1$1(), clazz)
Enclosing Method: toBinary()
File: main/scala/akka/remote/serialization/ProtobufSerializer.scala:109
Taint Flags:

```
106 toByteArrayMethod(unCachedtoByteArrayMethod)
107 }
108 }
109 toByteArrayMethod().invoke(obj).asInstanceOf[Array[Byte]]
110 }
111
112 private def checkAllowedClass(clazz: Class[_]): Unit = {
```

Object Model Violation: Just one of equals() and hashCode() Defined (4 issues)

Abstract

This class overrides only one of `equals()` and `hashCode()`.

Explanation

Java objects are expected to obey a number of invariants related to equality. One of these invariants is that equal objects must have equal hashcodes. In other words, if `a.equals(b) == true` then `a.hashCode() == b.hashCode()`. Failure to uphold this invariant is likely to cause trouble if objects of this class are stored in a collection. If the objects of the class in question are used as a key in a Hashtable or if they are inserted into a Map or Set, it is critical that equal objects have equal hashcodes. **Example 1:** The following class overrides `equals()` but not `hashCode()`.

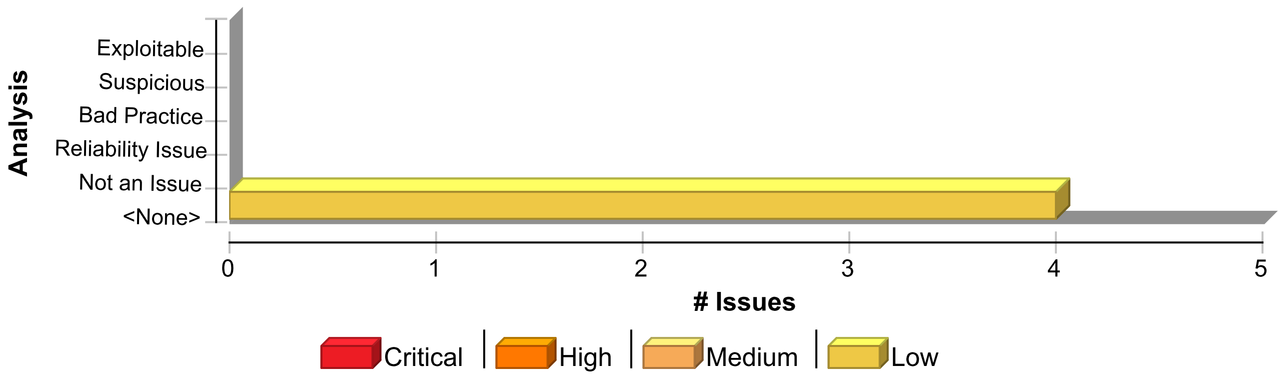
```
public class halfway() {
  public boolean equals(Object obj) {
    ...
  }
}
```

Recommendation

The FindBugs documentation recommends the following simple "starter" implementation of `hashCode()` [1]. It is highly inefficient, but it will produce correct results. If you do not believe that `hashCode()` is important for your program, consider using this implementation. **Example 2:** The code in `Example 1` could be rewritten in the following way:

```
public class halfway() {
  public boolean equals(Object obj) {
    ...
  }

public int hashCode() {
    assert false : "hashCode not designed";
    return 42; // any arbitrary constant will do
  }
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Object Model Violation: Just one of equals() and hashCode() Defined	4	0	0	4
Total	**4**	**0**	**0**	**4**

Object Model Violation: Just one of equals() and hashCode() Defined — Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 67 (Object Model Violation: Just one of equals() and hashCode() Defined) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: equals
Enclosing Method: equals()
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:67
Taint Flags:

```
64 }
65
66 class OtherException(msg: String) extends IllegalArgumentException(msg) with JavaSerializable {
67 override def equals(other: Any): Boolean = other match {
68 case e: OtherException => e.getMessage == getMessage
69 case _ => false
70 }
```

test/scala/akka/remote/serialization/SystemMessageSerializationSpec.scala, line 23 (Object Model Violation: Just one of equals() and hashCode() Defined) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: equals
Enclosing Method: equals()
File: test/scala/akka/remote/serialization/SystemMessageSerializationSpec.scala:23
Taint Flags:

```
20 val testConfig = ConfigFactory.parseString(serializationTestOverrides).withFallback(AkkaSpec.testConf)
21
22 class TestException(msg: String) extends RuntimeException(msg) with JavaSerializable {
23 override def equals(other: Any): Boolean = other match {
24 case e: TestException => e.getMessage == getMessage
25 case _ => false
26 }
```

Object Model Violation: Just one of equals() and hashCode() Defined — Low

Package: akka.remote.serialization

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 59 (Object Model Violation: Just one of equals() and hashCode() Defined) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: equals
Enclosing Method: equals()
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:59
Taint Flags:

```
}

class TestExceptionNoStack(msg: String) extends TestException(msg) with NoStackTrace {
override def equals(other: Any): Boolean = other match {
case e: TestExceptionNoStack =>
e.getMessage == getMessage && e.stackTrace == stackTrace
case _ => false
```

test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala, line 42 (Object Model Violation: Just one of equals() and hashCode() Defined) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Structural)

Sink Details

Sink: Function: equals
Enclosing Method: equals()
File: test/scala/akka/remote/serialization/MiscMessageSerializerSpec.scala:42
Taint Flags:

```
class TestException(msg: String, cause: Throwable) extends RuntimeException(msg, cause) {
def this(msg: String) = this(msg, null)

override def equals(other: Any): Boolean = other match {
case e: TestException =>
e.getMessage == getMessage && e.getCause == getCause &&
// on JDK9+ the stacktraces aren't equal, something about how they are constructed
```

Often Misused: Authentication (19 issues)

Abstract

Attackers may spoof DNS entries. Do not rely on DNS names for security.

Explanation

Many DNS servers are susceptible to spoofing attacks, so you should assume that your software will someday run in an environment with a compromised DNS server. If attackers are allowed to make DNS updates (sometimes called DNS cache poisoning), they can route your network traffic through their machines or make it appear as if their IP addresses are part of your domain. Do not base the security of your system on DNS names. **Example:** The following code uses a DNS lookup to determine whether an inbound request is from a trusted host. If an attacker can poison the DNS cache, they can gain trusted status.

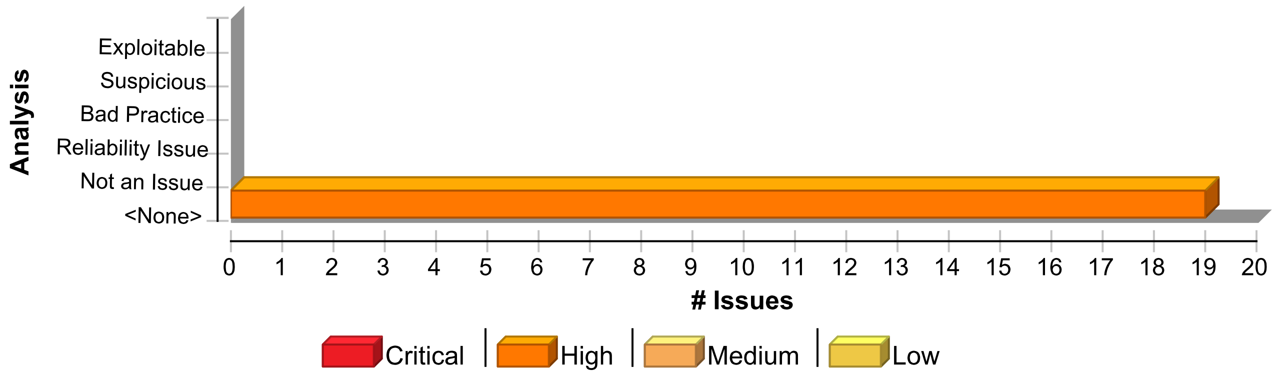
```
 String ip = request.getRemoteAddr();
 InetAddress addr = InetAddress.getByName(ip);
 if (addr.getCanonicalHostName().endsWith("trustme.com")) {
 trusted = true;
 }
```

IP addresses are more reliable than DNS names, but they can also be spoofed. Attackers may easily forge the source IP address of the packets they send, but response packets will return to the forged IP address. To see the response packets, the attacker has to sniff the traffic between the victim machine and the forged IP address. In order to accomplish the required sniffing, attackers typically attempt to locate themselves on the same subnet as the victim machine. Attackers may be able to circumvent this requirement by using source routing, but source routing is disabled across much of the Internet today. In summary, IP address verification can be a useful part of an authentication scheme, but it should not be the single factor required for authentication.

Recommendation

You can increase confidence in a domain name lookup if you check to make sure that the host's forward and backward DNS entries match. Attackers will not be able to spoof both the forward and the reverse DNS entries without controlling the nameservers for the target domain. This is not a foolproof approach however: attackers may be able to convince the domain registrar to turn over the domain to a malicious nameserver. Basing authentication on DNS entries is simply a risky proposition. While no authentication mechanism is foolproof, there are better alternatives than host-based authentication. Password systems offer decent security, but are susceptible to bad password choices, insecure password transmission, and bad password management. A cryptographic scheme like SSL is worth considering, but such schemes are often so complex that they bring with them the risk of significant implementation errors, and key material can always be stolen. In many situations, multi-factor authentication including a physical token offers the most security available at a reasonable price.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Often Misused: Authentication	19	0	0	19
Total	**19**	**0**	**0**	**19**

Often Misused: Authentication — High

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 294 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: getHostname()
File: main/scala/akka/remote/artery/ArterySettings.scala:294
Taint Flags:

```
291 }
292
293 def getHostname(key: String, config: Config): String = config.getString(key) match {
294 case "<getHostAddress>" => InetAddress.getLocalHost.getHostAddress
295 case "<getHostName>" => InetAddress.getLocalHost.getHostName
296 case other =>
297 if (other.startsWith("[") && other.endsWith("]")) other
```

main/scala/akka/remote/artery/ArterySettings.scala, line 295 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getLocalHost()
Enclosing Method: getHostname()
File: main/scala/akka/remote/artery/ArterySettings.scala:295
Taint Flags:

```
292
293 def getHostname(key: String, config: Config): String = config.getString(key) match {
294 case "<getHostAddress>" => InetAddress.getLocalHost.getHostAddress
295 case "<getHostName>" => InetAddress.getLocalHost.getHostName
296 case other =>
297 if (other.startsWith("[") && other.endsWith("]")) other
298 else if (AsyncDnsResolver.isIpv6Address(other)) {
```

Often Misused: Authentication — High

Package: akka.remote.artery

main/scala/akka/remote/artery/ArterySettings.scala, line 294 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getLocalHost()
Enclosing Method: getHostname()
File: main/scala/akka/remote/artery/ArterySettings.scala:294
Taint Flags:

```
291 }
292
293 def getHostname(key: String, config: Config): String = config.getString(key) match {
294 case "<getHostAddress>" => InetAddress.getLocalHost.getHostAddress
295 case "<getHostName>" => InetAddress.getLocalHost.getHostName
296 case other =>
297 if (other.startsWith("[") && other.endsWith("]")) other
```

main/scala/akka/remote/artery/ArterySettings.scala, line 295 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostName()
Enclosing Method: getHostname()
File: main/scala/akka/remote/artery/ArterySettings.scala:295
Taint Flags:

```
292
293 def getHostname(key: String, config: Config): String = config.getString(key) match {
294 case "<getHostAddress>" => InetAddress.getLocalHost.getHostAddress
295 case "<getHostName>" => InetAddress.getLocalHost.getHostName
296 case other =>
297 if (other.startsWith("[") && other.endsWith("]")) other
298 else if (AsyncDnsResolver.isIpv6Address(other)) {
```

Package: akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 34 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse

Often Misused: Authentication — High

Package: akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 34 (Often Misused: Authentication) — High

Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: toAkkaAddress()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:34
Taint Flags:

```
31
32  implicit class RichInetSocketAddress(address: InetSocketAddress) {
33  def toAkkaAddress(protocol: String)(implicit system: ActorSystem) =
34  Address(protocol, system.name, address.getAddress.getHostAddress, address.getPort)
35  }
36
37  implicit class RichAkkaAddress(address: Address) {
```

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 160 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getLocalHost()
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:160
Taint Flags:

```
157  }
158
159  val Hostname: String = getString("hostname") match {
160  case "" => InetAddress.getLocalHost.getHostAddress
161  case value => value
162  }
163
```

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 160 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Often Misused: Authentication — High

Package: akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 160 (Often Misused: Authentication) — High

Sink Details

Sink: getHostAddress()
Enclosing Method: NettyTransportSettings()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:160
Taint Flags:

```
157 }
158
159 val Hostname: String = getString("hostname") match {
160 case "" => InetAddress.getLocalHost.getHostAddress
161 case value => value
162 }
163
```

Package: main.scala.akka.remote.transport.netty

main/scala/akka/remote/transport/netty/NettyTransport.scala, line 496 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getByName()
Enclosing Method: apply()
File: main/scala/akka/remote/transport/netty/NettyTransport.scala:496
Taint Flags:

```
493 // TODO: This should be factored out to an async (or thread-isolated) name lookup service #2960
494 def addressToSocketAddress(addr: Address): Future[InetSocketAddress] = addr match {
495 case Address(_, _, Some(host), Some(port)) =>
496 Future { blocking { new InetSocketAddress(InetAddress.getByName(host), port) } }
497 case _ => Future.failed(new IllegalArgumentException(s"Address [$addr] does not contain host or port information."))
498 }
499
```

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala, line 76 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Often Misused: Authentication — High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala, line 76 (Often Misused: Authentication) — High

Sink: getLocalHost()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala:76
Taint Flags:

```
73 }
74
75 "bind to a specified port and remoting accepts from a bound port" in {
76 val address = SocketUtil.temporaryServerAddress(InetAddress.getLocalHost.getHostAddress, udp = isUDP)
77
78 val config = ConfigFactory.parseString(s"""
79 akka.remote.artery.canonical.port = 0
```

test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala, line 38 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getLocalHost()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala:38
Taint Flags:

```
35 }
36
37 "bind to a random port but remoting accepts from a specified port" in {
38 val address = SocketUtil.temporaryServerAddress(InetAddress.getLocalHost.getHostAddress, udp = isUDP)
39
40 val config = ConfigFactory.parseString(s"""
41 akka.remote.artery.canonical.port = ${address.getPort}
```

test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala, line 76 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala:76
Taint Flags:

Often Misused: Authentication — High

Package: test.scala.akka.remote.artery

test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala, line 76 (Often Misused: Authentication) — High

```
}

"bind to a specified port and remoting accepts from a bound port" in {
val address = SocketUtil.temporaryServerAddress(InetAddress.getLocalHost.getHostAddress, udp = isUDP)

val config = ConfigFactory.parseString(s"""
akka.remote.artery.canonical.port = 0
```

test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala, line 38 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/BindCanonicalAddressSpec.scala:38
Taint Flags:

```
}

"bind to a random port but remoting accepts from a specified port" in {
val address = SocketUtil.temporaryServerAddress(InetAddress.getLocalHost.getHostAddress, udp = isUDP)

val config = ConfigFactory.parseString(s"""
akka.remote.artery.canonical.port = ${address.getPort}
```

Package: test.scala.akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 87 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:87
Taint Flags:

```

}

```

Often Misused: Authentication — High

Package: test.scala.akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 87 (Often Misused: Authentication) — High

```
def randomOpenServerSocket(address: String = InetAddress.getLocalHost.getHostAddress) = {
val ss = ServerSocketChannel.open().socket()
ss.bind(new InetSocketAddress(address, 0))
(ss, new InetSocketAddress(address, ss.getLocalPort))
```

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 94 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getLocalHost()
Enclosing Method: apply()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:94
Taint Flags:

```
}

"bind to a specified port and remoting accepts from a bound port" in {
val address = SocketUtil.temporaryServerAddress(InetAddress.getLocalHost.getHostAddress, udp = false)

val bindConfig = ConfigFactory.parseString(s"""
akka.remote.artery.enabled = false
```

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 87 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getLocalHost()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:87
Taint Flags:

```

}

def randomOpenServerSocket(address: String = InetAddress.getLocalHost.getHostAddress) = {
val ss = ServerSocketChannel.open().socket()
ss.bind(new InetSocketAddress(address, 0))
(ss, new InetSocketAddress(address, ss.getLocalPort))
```

Often Misused: Authentication — High

Package: test.scala.akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 167 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: apply()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:167
Taint Flags:

```
164 null
165 }
166
167 val bindConfig = ConfigFactory.parseString(s"""
168 akka.remote.artery.enabled = false
169 akka.remote.classic {
170 netty.tcp {
```

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 94 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: apply()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:94
Taint Flags:

```
91 }
92
93 "bind to a specified port and remoting accepts from a bound port" in {
94 val address = SocketUtil.temporaryServerAddress(InetAddress.getLocalHost.getHostAddress, udp = false)
95
96 val bindConfig = ConfigFactory.parseString(s"""
97 akka.remote.artery.enabled = false
```

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 167 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Often Misused: Authentication — High

Package: test.scala.akka.remote.classic.transport.netty

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 167 (Often Misused: Authentication) — High

Sink Details

Sink: getHostAddress()
Enclosing Method: apply()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:167
Taint Flags:

```
null
}

val bindConfig = ConfigFactory.parseString(s"""
akka.remote.artery.enabled = false
akka.remote.classic {
netty.tcp {
```

test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala, line 138 (Often Misused: Authentication) — High

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: getHostAddress()
Enclosing Method: apply()
File: test/scala/akka/remote/classic/transport/netty/NettyTransportSpec.scala:138
Taint Flags:

```
s"bind to default tcp address" in {
val address = SocketUtil.temporaryServerAddress()

val bindConfig = ConfigFactory.parseString(s"""
akka.remote.artery.enabled = false
akka.remote.classic {
netty.tcp {
```

Poor Error Handling: Empty Catch Block (1 issue)

Abstract

Ignoring an exception can cause the program to overlook unexpected states and conditions.

Explanation

Just about every serious attack on a software system begins with the violation of a programmer's assumptions. After the attack, the programmer's assumptions seem flimsy and poorly founded, but before an attack many programmers would defend their assumptions well past the end of their lunch break. Two dubious assumptions that are easy to spot in code are "this method call can never fail" and "it doesn't matter if this call fails". When a programmer ignores an exception, they implicitly state that they are operating under one of these assumptions. **Example 1:** The following code excerpt ignores a rarely-thrown exception from `doExchange()`.

```
try {
  doExchange();
}
catch (RareException e) {
  // this can never happen
}
```

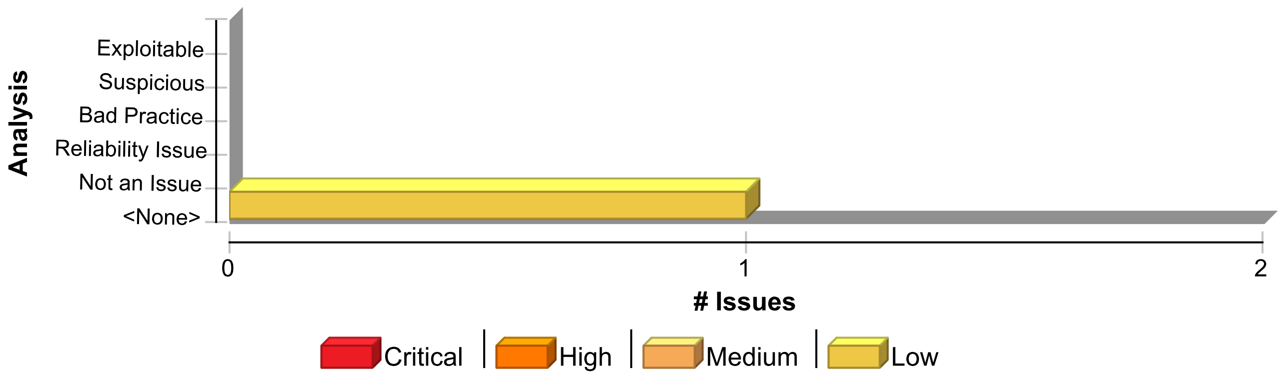
If a `RareException` were to ever be thrown, the program would continue to execute as though nothing unusual had occurred. The program records no evidence indicating the special situation, potentially frustrating any later attempt to explain the program's behavior.

Recommendation

At a minimum, log the fact that the exception was thrown so that it will be possible to come back later and make sense of the resulting program behavior. Better yet, abort the current operation. If the exception is being ignored because the caller cannot properly handle it but the context makes it inconvenient or impossible for the caller to declare that it throws the exception itself, consider throwing a `RuntimeException` or an `Error`, both of which are unchecked exceptions. As of JDK 1.4, `RuntimeException` has a constructor that makes it easy to wrap another exception.
Example 2: The code in `Example 1` could be rewritten in the following way:

```
try {
  doExchange();
}
catch (RareException e) {
  throw new RuntimeException("This can never happen", e);
}
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Error Handling: Empty Catch Block	1	0	0	1
Total	**1**	**0**	**0**	**1**

Poor Error Handling: Empty Catch Block — Low

Package: test.scala.akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 132 (Poor Error Handling: Empty Catch Block) — **Low**

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Sink Details

Sink: CatchBlock
Enclosing Method: apply()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:132
Taint Flags:

```
129 contact(remoteSysB.actorSystem, pathEchoC)
130 fail("The credentials under `ssl/rsa-client` are not valid for Akka remote so contact() must fail.")
131 } catch {
132 case _: java.lang.AssertionError =>
133 // This assertion error is expected because we expect a failure in contact() since
134 // the SSL credentials are invalid
135 }
```

Poor Error Handling: Overly Broad Catch (3 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:

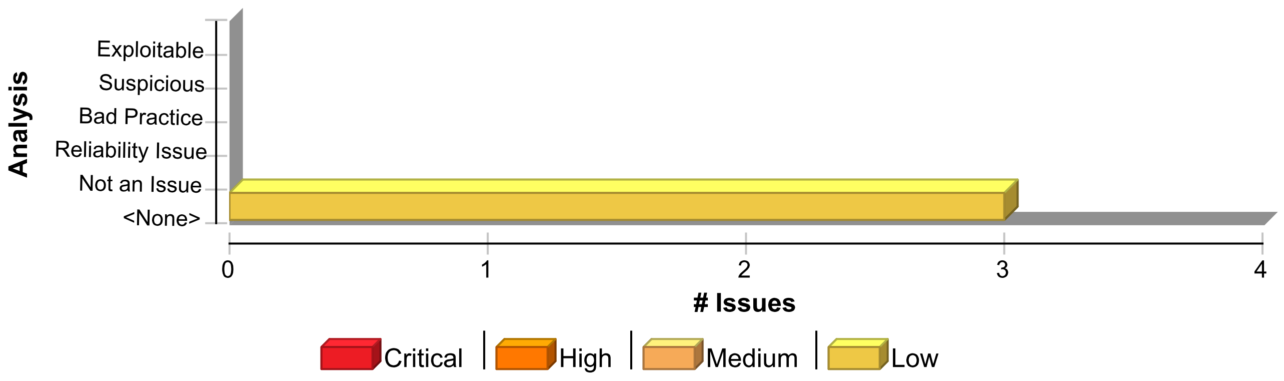
```
  try {
    doExchange();
  }
  catch (Exception e) {
    logger.error("doExchange failed", e);
  }
```

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	3	0	0	3
Total	**3**	**0**	**0**	**3**

Poor Error Handling: Overly Broad Catch — Low

Package: test.scala.akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 31 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Sink Details

Sink: CatchBlock
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:31
Taint Flags:

```
28 Thread.sleep(duration.toMillis)
29 restoreIP()
30 } catch {
31 case e: Throwable =>
32 dead.set(true)
33 e.printStackTrace
34 }
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 46 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Sink Details

Sink: CatchBlock

Poor Error Handling: Overly Broad Catch — Low

Package: test.scala.akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 46 (Poor Error Handling: Overly Broad Catch) — Low

Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:46
Taint Flags:

```
43 Thread.sleep(duration.toMillis)
44 restoreIP()
45 } catch {
46 case e: Throwable =>
47 dead.set(true)
48 e.printStackTrace
49 }
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 61 (Poor Error Handling: Overly Broad Catch) — Low

Issue Details

Kingdom: Errors
Scan Engine: SCA (Structural)

Sink Details

Sink: CatchBlock
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:61
Taint Flags:

```
58 Thread.sleep(duration.toMillis)
59 restoreIP()
60 } catch {
61 case e: Throwable =>
62 dead.set(true)
63 e.printStackTrace
64 }
```

Poor Style: Value Never Read (4 issues)

Abstract

The variable's value is assigned but never used, making it a dead store.

Explanation

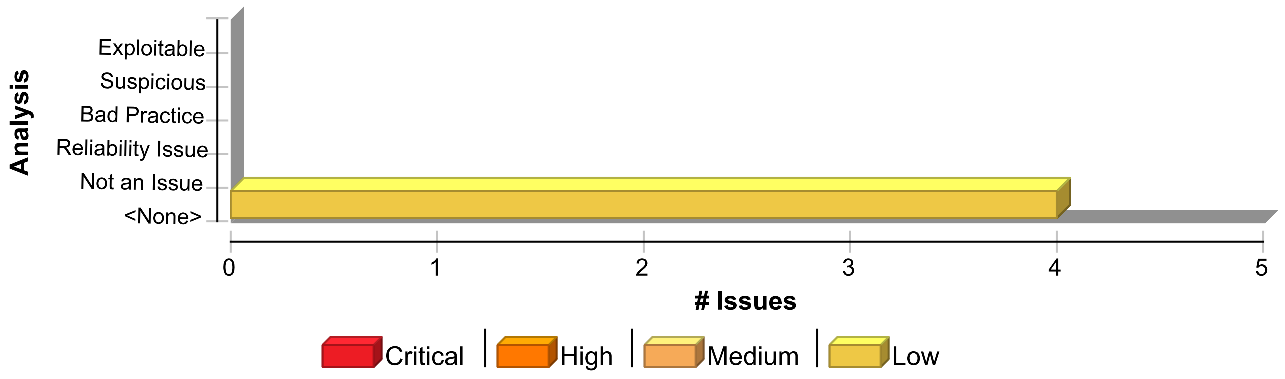
This variable's value is not used. After the assignment, the variable is either assigned another value or goes out of scope. **Example:** The following code excerpt assigns to the variable `r` and then overwrites the value without using it.

```
  r = getName();
  r = getNewBuffer(buf);
```

Recommendation

Remove unnecessary assignments in order to make the code easier to understand and maintain.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Poor Style: Value Never Read	4	0	0	4
Total	**4**	**0**	**0**	**4**

Poor Style: Value Never Read **Low**

Package: akka.remote.artery

main/scala/akka/remote/artery/RemoteInstrument.scala, line 195 (Poor Style: Value Never Read) **Low**

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: rewindPos
Enclosing Method: serialize()
File: main/scala/akka/remote/artery/RemoteInstrument.scala:195

Poor Style: Value Never Read — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/RemoteInstrument.scala, line 195 (Poor Style: Value Never Read) — Low

Taint Flags:

```
192 val dataPos = buffer.position()
193 var i = 0
194 while (i < instruments.length) {
195 val rewindPos = buffer.position()
196 val instrument = instruments(i)
197 try {
198 serializeInstrument(instrument, oe, buffer)
```

main/scala/akka/remote/artery/Association.scala, line 795 (Poor Style: Value Never Read) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: unused
Enclosing Method: getOrCreateQueueWrapper()
File: main/scala/akka/remote/artery/Association.scala:795
Taint Flags:

```
792
793 private def getOrCreateQueueWrapper(queueIndex: Int, capacity: Int): QueueWrapper = {
794 @nowarn("msg=never used")
795 val unused = queuesVisibility // volatile read to see latest queues array
796 queues(queueIndex) match {
797 case existing: QueueWrapper => existing
798 case _ =>
```

main/scala/akka/remote/artery/Association.scala, line 342 (Poor Style: Value Never Read) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: unused
Enclosing Method: send()
File: main/scala/akka/remote/artery/Association.scala:342
Taint Flags:

```
339
340 // volatile read to see latest queue array
341 @nowarn("msg=never used")
342 val unused = queuesVisibility
```

Poor Style: Value Never Read — Low

Package: akka.remote.artery

main/scala/akka/remote/artery/Association.scala, line 342 (Poor Style: Value Never Read) — Low

```
343
344 def dropped(queueIndex: Int, qSize: Int, env: OutboundEnvelope): Unit = {
345 val removed = isRemovedAfterQuarantined()
```

Package: test.scala.akka.remote.classic

test/scala/akka/remote/classic/RemoteInitErrorSpec.scala, line 49 (Poor Style: Value Never Read) — Low

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Structural)

Sink Details

Sink: VariableAccess: start
Enclosing Method: apply()
File: test/scala/akka/remote/classic/RemoteInitErrorSpec.scala:49
Taint Flags:

```
46
47 "Remoting" must {
48 "shut down properly on RemoteActorRefProvider initialization failure" in {
49 val start = currentThreadIds()
50 try {
51 ActorSystem("duplicate", ConfigFactory.parseString("akka.loglevel=OFF").withFallback(conf))
52 fail("initialization should fail due to invalid IP address")
```

System Information Leak (3 issues)

Abstract

Revealing system data or debugging information helps an adversary learn about the system and form a plan of attack.

Explanation

An information leak occurs when system data or debug information leaves the program through an output stream or logging function. **Example 1:** The following code writes an exception to the standard error stream:

```
try {
    ...
} catch (Exception e) {
    e.printStackTrace();
}
```

Depending upon the system configuration, this information can be dumped to a console, written to a log file, or exposed to a remote user. For example, with scripting mechanisms it is trivial to redirect output information from "Standard error" or "Standard output" into a file or another program. Alternatively, the system that the program runs on could have a remote logging mechanism such as a "syslog" server that sends the logs to a remote device. During development, you have no way of knowing where this information might end up being displayed. In some cases, the error message provides the attacker with the precise type of attack to which the system is vulnerable. For example, a database error message can reveal that the application is vulnerable to a SQL injection attack. Other error messages can reveal more oblique clues about the system. In `Example 1`, the leaked information could imply information about the type of operating system, the applications installed on the system, and the amount of care that the administrators have put into configuring the program. Here is another scenario, specific to the mobile world. Most mobile devices now implement a Near-Field Communication (NFC) protocol for quickly sharing information between devices using radio communication. It works by bringing devices to close proximity or simply having them touch each other. Even though the communication range of NFC is limited to just a few centimeters, eavesdropping, data modification and various other types of attacks are possible, since NFC alone does not ensure secure communication.
Example 2: The Android platform provides support for NFC. The following code creates a message that gets pushed to the other device within the range.

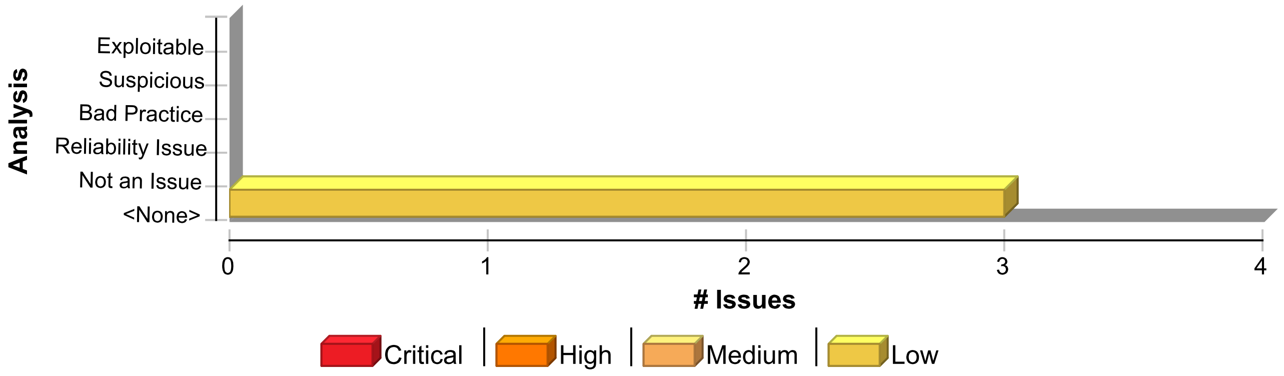
```
...
public static final String TAG = "NfcActivity";
private static final String DATA_SPLITTER = "__:DATA:__";
private static final String MIME_TYPE = "application/my.applications.mimetype";
...
public NdefMessage createNdefMessage(NfcEvent event) {
    TelephonyManager tm =
(TelephonyManager)Context.getSystemService(Context.TELEPHONY_SERVICE);
    String VERSION = tm.getDeviceSoftwareVersion();
    String text = TAG + DATA_SPLITTER + VERSION;
    NdefRecord record = new NdefRecord(NdefRecord.TNF_MIME_MEDIA,
            MIME_TYPE.getBytes(), new byte[0], text.getBytes());
    NdefRecord[] records = { record };
    NdefMessage msg = new NdefMessage(records);
    return msg;
}
...
```

NFC Data Exchange Format (NDEF) message contains typed data, a URI, or a custom application payload. If the message contains information about the application, such as its name, MIME type, or device software version, this information could be leaked to an eavesdropper. In `Example 2`, Fortify Static Code Analyzer reports a System Information Leak vulnerability on the return statement.

Recommendation

Write error messages with security in mind. In production environments, turn off detailed error information in favor of brief messages. Restrict the generation and storage of detailed output that can help administrators and programmers diagnose problems. Debug traces can sometimes appear in non-obvious places (embedded in comments in the HTML for an error page, for example). Even brief error messages that do not reveal stack traces or database dumps can potentially aid an attacker. For example, an "Access Denied" message can reveal that a file or user exists on the system. If you are concerned about leaking system data via NFC on an Android device, you could do one of the following three things. Do not include system data in the messages pushed to other devices in range, encrypt the payload of the message, or establish a secure communication channel at a higher layer.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
System Information Leak	3	0	0	3
Total	**3**	**0**	**0**	**3**

System Information Leak — Low

Package: test.scala.akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 63 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Sink Details

Sink: printStackTrace()
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:63
Taint Flags:

```
60 } catch {
61 case e: Throwable =>
62 dead.set(true)
63 e.printStackTrace
64 }
65 }
66 }
```

System Information Leak — Low

Package: test.scala.akka.remote

test/scala/akka/remote/NetworkFailureSpec.scala, line 48 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Sink Details

Sink: printStackTrace()
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:48
Taint Flags:

```
45 } catch {
46 case e: Throwable =>
47 dead.set(true)
48 e.printStackTrace
49 }
50 }
51 }
```

test/scala/akka/remote/NetworkFailureSpec.scala, line 33 (System Information Leak) — Low

Issue Details

Kingdom: Encapsulation
Scan Engine: SCA (Semantic)

Sink Details

Sink: printStackTrace()
Enclosing Method: apply()
File: test/scala/akka/remote/NetworkFailureSpec.scala:33
Taint Flags:

```
30 } catch {
31 case e: Throwable =>
32 dead.set(true)
33 e.printStackTrace
34 }
35 }
36 }
```

Unchecked Return Value (2 issues)

Abstract

Ignoring a method's return value can cause the program to overlook unexpected states and conditions.

Explanation

It is not uncommon for Java programmers to misunderstand `read()` and related methods that are part of many `java.io` classes. Most errors and unusual events in Java result in an exception being thrown. (This is one of the advantages that Java has over languages like C: Exceptions make it easier for programmers to think about what can go wrong.) But the stream and reader classes do not consider it unusual or exceptional if only a small amount of data becomes available. These classes simply add the small amount of data to the return buffer, and set the return value to the number of bytes or characters read. There is no guarantee that the amount of data returned is equal to the amount of data requested. This behavior makes it important for programmers to examine the return value from `read()` and other IO methods to ensure that they receive the amount of data they expect. **Example:** The following code loops through a set of users, reading a private data file for each user. The programmer assumes that the files are always exactly 1 kilobyte in size and therefore ignores the return value from `read()`. If an attacker can create a smaller file, the program will recycle the remainder of the data from the previous user and handle it as though it belongs to the attacker.

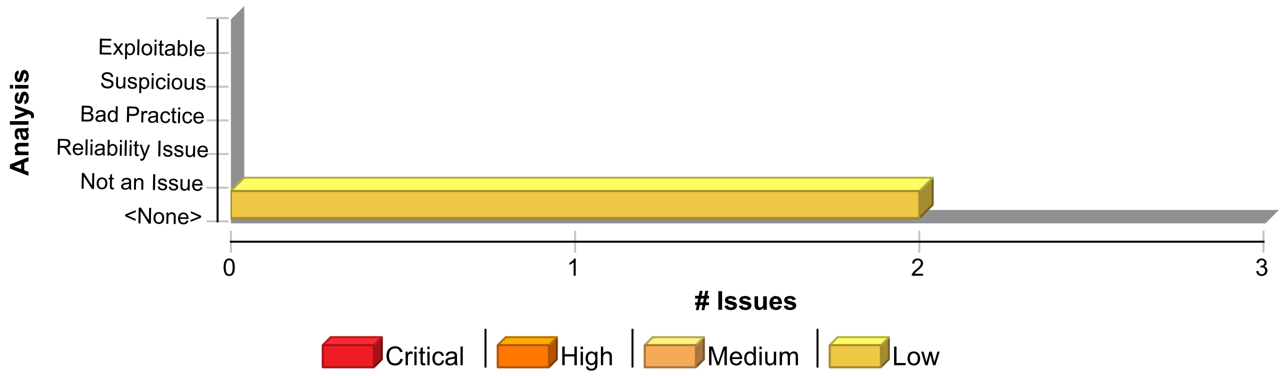
```
FileInputStream fis;
byte[] byteArray = new byte[1024];
for (Iterator i=users.iterator(); i.hasNext();) {
    String userName = (String) i.next();
    String pFileName = PFILE_ROOT + "/" + userName;
    FileInputStream fis = new FileInputStream(pFileName);
    fis.read(byteArray); // the file is always 1k bytes
    fis.close();
    processPFile(userName, byteArray);
}
```

Recommendation

```
  FileInputStream fis;
  byte[] byteArray = new byte[1024];
  for (Iterator i=users.iterator(); i.hasNext();) {
    String userName = (String) i.next();
    String pFileName = PFILE_ROOT + "/" + userName;
    fis = new FileInputStream(pFileName);
    int bRead = 0;
    while (bRead < 1024) {
        int rd = fis.read(byteArray, bRead, 1024 - bRead);
        if (rd == -1) {
          throw new IOException("file is unusually small");
        }
        bRead += rd;
    }
    // could add check to see if file is too large here
    fis.close();
    processPFile(userName, byteArray);
  }
```

Note: Because the fix for this problem is relatively complicated, you might be tempted to use a simpler approach, such as checking the size of the file before you begin reading. Such an approach would render the application vulnerable to a file system race condition, whereby an attacker could replace a well-formed file with a malicious file between the file size check and the call to read data from the file.

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unchecked Return Value	2	0	0	2
Total	**2**	**0**	**0**	**2**

Unchecked Return Value — Low

Package: akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 215 (Unchecked Return Value) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Sink Details

Sink: delete()
Enclosing Method: cleanupTemporaryDirectory()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:215
Taint Flags:

```
212  }
213  def cleanupTemporaryDirectory(): Unit = {
214  temporaryDirectory.toFile.listFiles().foreach { _.delete() }
215  temporaryDirectory.toFile.delete()
216  }
217 }
218
```

Package: test.scala.akka.remote.artery.tcp.ssl

test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 200 (Unchecked Return Value) — Low

Issue Details

Kingdom: API Abuse
Scan Engine: SCA (Semantic)

Unchecked Return Value	Low
Package: test.scala.akka.remote.artery.tcp.ssl	
test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala, line 200 (Unchecked Return Value)	**Low**

Sink Details

Sink: mkdirs()
Enclosing Method: apply()
File: test/scala/akka/remote/artery/tcp/ssl/RotatingKeysSSLEngineProviderSpec.scala:200
Taint Flags:

```
197 // manually ensuring files are deleted and copied to prevent races.
198 try {
199 val from = new File(getClass.getClassLoader.getResource(resourceName).getPath).toPath
200 to.toFile.getParentFile.mkdirs()
201 Files.copy(from, to, StandardCopyOption.REPLACE_EXISTING)
202 } catch {
203 case NonFatal(t) => throw new RuntimeException(s"Can't copy resource [$resourceName] to [$to].", t)
```

Unreleased Resource: Synchronization (1 issue)

Abstract

The program fails to release a lock it holds, which might lead to deadlock.

Explanation

The program can potentially fail to release a system resource. Resource leaks have at least two common causes: - Error conditions and other exceptional circumstances. - Confusion over which part of the program is responsible for releasing the resource. Most unreleased resource issues result in general software reliability problems. However, if an attacker can intentionally trigger a resource leak, the attacker may be able to launch a denial of service by depleting the resource pool. **Example 1:** The following code establishes a lock before `performOperationInCriticalSection()`, but fails to release the lock if an exception is thrown in that method.

```
  ReentrantLock  myLock = new ReentrantLock();

  myLock.lock();
  performOperationInCriticalSection();
  myLock.unlock();
```

This category was derived from the Cigital Java Rulepack.

Recommendation

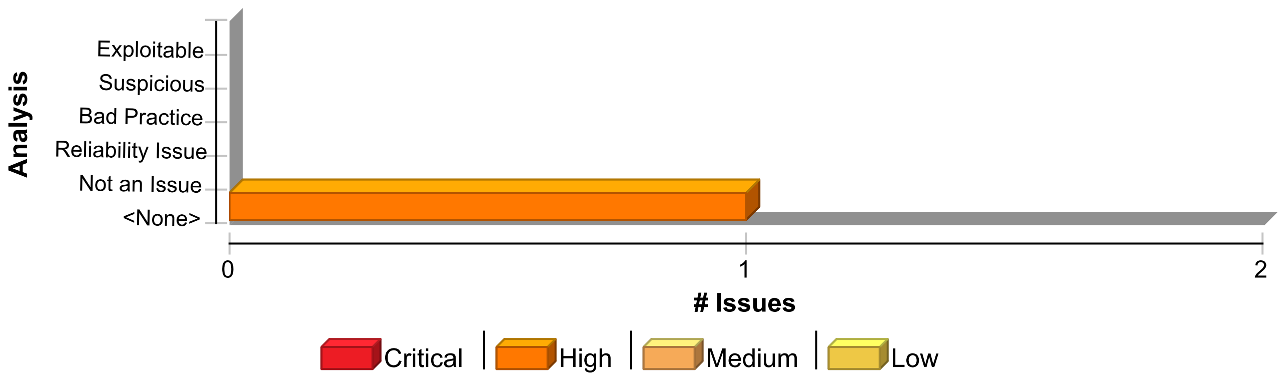
Because resource leaks can be hard to track down, establish a set of resource management patterns and idioms for your software and do not tolerate deviations from your conventions. One good pattern for addressing the error handling mistake in this example is to release the lock in a `finally` block. **Example 2:** The following code will always release the lock.

```
  ReentrantLock  myLock = new ReentrantLock();

  try {
    myLock.lock();
    performOperationInCriticalSection();
    myLock.unlock();
  }
  finally {
    if (myLock != null) {
      myLock.unlock();
    }
  }
```

Example 3: If using Kotlin, it is advisable to use the `withLock` function, removing the possibility of forgetting to unlock.

```
  val myLock = ReentrantLock()
  myLock.withLock {
    performOperationInCriticalSection()
  }
```

Issue Summary



Engine Breakdown

	SCA	WebInspect	SecurityScope	Total
Unreleased Resource: Synchronization	1	0	0	1
Total	**1**	**0**	**0**	**1**

Unreleased Resource: Synchronization — High

Package: akka.remote

main/scala/akka/remote/DefaultFailureDetectorRegistry.scala, line 41 (Unreleased Resource: Synchronization) — High

Issue Details

Kingdom: Code Quality
Scan Engine: SCA (Control Flow)

Sink Details

Sink: this.failureDetectorCreationLock().lock() : locked
Enclosing Method: heartbeat()
File: main/scala/akka/remote/DefaultFailureDetectorRegistry.scala:41
Taint Flags:

```
38 case Some(failureDetector) => failureDetector.heartbeat()
39 case None =>
40 // First one wins and creates the new FailureDetector
41 failureDetectorCreationLock.lock()
42 try {
43 // First check for non-existing key was outside the lock, and a second thread might just released the lock
44 // when this one acquired it, so the second check is needed.
```