# eIDAS-Node Migration Guide

For Version 2.2

## Document history

| Version | Date | Modification reason | Modified by |
|---|---|---|---|
| 2.2 | 17/09/2018 | Migration content relating to eIDAS-Node release 2.2 was created. | DIGIT |

## Table of contents

# 1. Introduction

This document is intended for a technical audience consisting of developers, administrators and those requiring detailed technical information on how to configure, build and deploy the eIDAS-Node application.

The purpose of this document is to facilitate migration from eIDAS-Node v2.1 to eIDAS-Node v2.2.

## 1.1. Document structure

This document is divided into the following sections:

Chapter 1 — *Introduction*: this section.

Chapter 2 — *Prerequisites*: Identifies any prerequisites that are required before migrating your eIDAS-Node to version 2.2.

Chapter 3 — Changes: Contains detailed information about the changes that should be taken into consideration when migrating to eIDAS-Node version 2.2.

## 1.2. Document aims

The main aim of this document is to provide information on all the changes requiring your action when migrating to eIDAS-Node version 2.2, including:

configuration changes; and

changes to code.

## 2. Prerequisites

Before starting your migration to eIDAS-Node version 2.2 you should have:

- already implemented eIDAS-Node version 2.1;
- downloaded the eIDAS-Node v2.2 Integration Package; and
- downloaded the latest documentation.

# 3. Changes

## 3.1. Summary of changes

In eIDAS-Node version 2.2 there are several changes that affect your installation. The main changes are:

- (EID-570) Key representation as ds:KeyValue/RSAKeyValue in ds:KeyInfo not supported
- (EID-606) Support of Sub-CA for Metadata Signer
- (EID-617) Error responses contains assertions with a false identity
- (EID-630) Missing Assertion in failed authentication response should be OK
- (EID-631) Issuer URL in SAML AuthnRequest can be manipulated
- (EID-643) Wrong character encoding in ConnectorMetadata

## 3.2. Implement usage of simple DSI keys in SAML messages : ENCRYPTION

### 3.2.1. Code changes

```
EIDAS-Encryption\..\eu\eidas\encryption\SAMLAuthnResponseEncrypter.java

     Logic changes to generate / marshall the <saml2:EncryptedAssertion> element(s)
     in an AuthenticatioNResponse to enclose <ds:KeyInfo> with <ds:RSAKeyValue>
     sub-element.

EIDAS-SAMLEngine\..\eidas\auth\engine\AbstractProtocolEngine.java
EIDAS-
SAMLEngine\..\eu\eidas\auth\engine\configuration\dom\EncryptionConfiguration.java
EIDAS-SAMLEngine\..\eu\eidas\auth\engine\configuration\dom\EncryptionKey.java
EIDAS-
SAMLEngine\..\eu\eidas\auth\engine\configuration\dom\KeyStoreEncryptionConfigurator
.java
EIDAS-SAMLEngine\..\eu\eidas\auth\engine\core\ProtocolCipherI.java
EIDAS-SAMLEngine\..\eu\eidas\auth\engine\core\ProtocolEncrypterI.java
EIDAS-SAMLEngine\..\eu\eidas\auth\engine\core\impl\AbstractProtocolCipher.java
EIDAS-SAMLEngine\..\eu\eidas\auth\engine\core\impl\AbstractProtocolEncrypter.java
EIDAS-SAMLEngine\..\eu\eidas\auth\engine\core\impl\AbstractProtocolSigner.java
EIDAS-
SAMLEngine\..\eu\eidas\auth\engine\core\impl\AbstractSamlEngineEncryption.java

     Changes in the API to accommodate the assertion.encrypt.with.key.value through
     an additional Boolean method argument

EIDAS-SAMLEngine\..\eu\eidas\auth\engine\core\impl\AbstractProtocolDecrypter.java

     Logic changes to un-marshall the <saml2:EncryptedAssertion> element(s) in an
     AuthenticatioNResponse enclosing <ds:KeyInfo> with <ds:RSAKeyValue> sub-
     element.
```

### 3.2.2. Configuration changes

```
EIDAS-Config\server\eidas.xml
```

| | |
|---|---|
| `assertion.encrypt.with.key.value` | When set to true, the eIDAS-Node marshals encrypted assertions, added to the Authentication Responses originates, to include the public RSA key instead of the full X509Certificate.<br><br>Otherwise, when set to false the eIDAS-Node keeps the behaviour of 2.0. |

## 3.3. Validate issuer of SAML request (and SAML response)

This change validates the issuer of the metadata against an accepted list of allowed url's aka. the whitelist. **An inexistent or empty whitelist for a component will not retrieve any metadata for that component.**

Specifically, the following two checks are performed during the eIDAS authentication flow:

The SAML Requests issued from Connector(s) into Proxy have the issuer checked against **connector.metadata.location.whitelist**

The SAML Responses issued from Proxy(s) back to Connector, have the issuer checked against a dynamic whitelist built from values of all the keys following the **pattern service1.metadata.url … service8.metadata.url**

### 3.3.1. Code changes

```
EIDAS-Node\src\main\java\eu\eidas\node\auth\connector\AUCONNECTORSAML.java
EIDAS-Node\src\main\java\eu\eidas\node\auth\service\AUSERVICESAML.java
EIDAS-Node\src\main\webapp\WEB-INF\applicationContext.xml
      Changes necessary to retrieve and populate corresponding metadata url whitelists that
      the node is allowed to contact back to retrieve metadata necessary to verify SAML
      request / SAML response signatures (based on connector.metadata.location.whitelist, and
      service[d].metadata.url values)
EIDAS-SAMLEngine\src\main\java\eu\eidas\auth\engine\ProtocolEngine.java
EIDAS-SAMLEngine\src\main\java\eu\eidas\auth\engine\ProtocolEngineI.java
      Core changes in the Protocol Engine to take into account the whitelists above when un-
      marshalling the SAML Request and Response and verify the signature of their originating
      issuer.

EIDAS-Commons\src\main\java\eu\eidas\util\WhitelistUtil.java
      Utility class for parsing a semicolumn separated list of ascii strings that represent a
      SET OF the urls.
```

### 3.3.2. Configuration changes

eidas.xml additions: semicolumn separated urls as values for following new keys

```
<entry key="connector.metadata.location.whitelist">
http://localhost:8080/EidasNode/ConnectorMetadata;
http://eidasnode:8888/EidasNode/ConnectorMetadata;

</entry>
```

## 3.4. Adapt Connector and Proxy Service Metadata to handle non-ASCII characters

When using non-ASCII characters in one of the metadata fields configurable through eidas.xml (eg. by specifying them in one of the *.contact.support.company entries) the servlet standard encoding of UTF-8 is now used for the generated XML metadata.

### 3.4.1. Code changes

```
EIDAS-Node\src\main\java\eu\eidas\node\connector\ConnectorMetadataGeneratorServlet.java
EIDAS-Node\src\main\java\eu\eidas\node\service\ProxyServiceMetadataGeneratorServlet.java
```

### 3.4.2. Configuration changes

No changes made.

## 3.5. Allow reception and generation of Non-successful SAML Response without assertion

In previous versions, generated non-successful SAML Responses always contained an assertion. If a node receives non-successful SAML Responses without an assertion, an exception is thrown and the flow stops.

In current version, this behaviour was changed and non-successful SAML Responses do not include by default an assertion anymore. However, a temporary property was added to eidas.xml, for interoperability reasons. It is to be used with eIDAS-Node versions 1.4.2 and below in the 1.4.x branch, and with eIDAS-Node versions 2.1 and below in the 2.x branch. For this, it uses the value of application identifier of protocol versioning, published in the requester's metadata.

The validation for non-successful SAML Responses, was also relaxed so that, non-successful SAML Responses, with or without assertion, are also accepted.

### 3.5.1. Code changes

The code was changed so that failure responses with or without assertion are accepted. Therefore AUCONNECTORSAML class changed to cope with the case when audience restriction can be null. AUSERVICESAML was also changed to call a new method in ProtocolEngineI that receives as parameter the list of application identifiers that responses to, must contains assertion. Some code was also extracted to methods generateResponseErrorMessage and getIncludeAssertionApplicationIdentifiers.

A new method was added to ProtocolProcessorI marshalErrorResponse, that overloads an existing one and contains an additional parameter the list of application identifiers of the requesters. The implementations of ProtocolProcessorI: EidasProtocolProcessor, EidasProtocolProcessorWithAutorization and StorkProtocolProcessor were changed to implement a new method. Similarly the ProtocolEngineI was also changed and the method generateResponseErrorMessage was overloaded with an extra parameter applicationIdentifiers. The implementation of the new method contained in the interface was done at ProtocolEngine. These new implementations allow generating non-successful error Response with or without assertion depending on the matching of request's application identifier with the ones in the parameter applicationIdentifiers.

Method extractVerifiedAssertion in ResponseUtil was changed so that assertion is only extracted in the case of success and when only one is in the response. Junit tests were added to cover this change at ResponseUtilTest. At EidasStringUtil, getTokens method was added to split a string into tokens separated by either semicolon or comma. EidasParameterKeys was also changed to contain the key for the new property added to eidas.xml: include.assertion.fail.response.application.identifiers.

The convertExtentions method at MetadataUtil was changed to allow setting the protocol version and application identifier at the EidasParametersI instance.

Other changes related to this topic occurred at EidasMetadata were protocol versioning related constants were changed to public.

### 3.5.2. Configuration changes

A new property was introduced in the eidas.xml:

```
        <entry key="include.assertion.fail.response.application.identifiers">CEF:eIDAS-ref:1.4.2;CEF:eIDAS-ref:2.1</entry>
```

The values of this property are application identifiers published in the metadata and related to the protocol versioning that non successful responses will include the assertion due to interoperability reasons. For other cases, non successful responses will not contain an assertion.

### 3.5.3. Clean-up dependencies

Several dependencies have been removed from application server:

xml-apis-1.4.01.jar
resolver-2.9.1.jar
xercesImpl-2.11.0.jar
xalan-2.7.2.jar
serializer-2.7.2.jar

There is no need to add manually these libraries to the application servers.

Several dependencies have been removed from the pom.xml:

xml-apis-1.4.01.jar
resolver-2.9.1.jar
xercesImpl-2.11.0.jar

In addition to the above libraries, also xalan and serialiser are no longer needed to be added manually to the application servers.

Xalan and serialiser are now packaged in the war and used at runtime.

com.sun.jersey/jersey-server has been removed from Node module. It was used for tomcat/glassfish.

com.sun.jersey.contribs/jersey-spring has been removed from Node and Updater Modules

cglib/cglib has been removed from Node module. It was used for tomcat/glassfish.

org.bouncycastle/bcprov-jdk15on has been moved out of pom profiles. Now it's used for all application servers.

Joda-time dependency has been refactored. It now comes only through Light-Commons.

```
Removed javax/javaee-api dependency from the Node.

Jboss-deployment-structure.xml configuration file for jboss/wildfly has been
refactored. Now we are keeping this file to exclude javaee.api from the server.
org.bouncycastle dependency has been moved from this file to MANIFEST.MF in order
to use wildfly without jboss-deplyment-structure.xml

org.owasp.esapi dependency has been replaced by apache.commons.lang.

The following profiles have been removed:
websphere
tomcat

Glassfish configuration file has been removed from EIDAS-Node.
```

## 3.6. Invalidate extra certificate published in metadata that do not belong to consumer's trustchain

In the previous implementation, if a certificate was published in the metadata not belonging to the trustchain, the metadata could be validated. This issue was fixed.

### 3.6.1. Code changes

CertificateUtil code was changed so that the validation of path done at isTrustValid method performs a correct validation of the credential to be validated against trusted credentials. Previous method checkTrust was renamed to checkChainTrust and a new method checkExplicitTrust was added that performs a check using the trust chain or the do direct check of certificate, as done in previous version that did not support validation against a trust chain. Several methods were added to CertificateUtil: getCertificates and getCertificates, that allow retrieval of certificates from credentials to be used by the AbstractProtocolSigner.

AbstractProtocolSigner was also changed, method checkValidTrust code was redone to allow coverage of all the test cases newly added at TestEidasNodeMetadataTrustChain. This test class now contains more cases to test the several cases of produced metadata containing the different combinations published certificates. And also different combination of for trusted certificates while validating metadata. Tests

- testValidMetadataSignatureWithoutIntermediateCaPublishedTrustingIntermediateCARootCA
- testValidMetadataSignatureWithoutIntermediateCaPublishedTrustingIntermediateCA

were added to cover missing cases to test the validation of metadata signature.

## 3.7. Prevent a DoS attack when it reads data without limiting the amount of characters.

The application uses methods that are susceptible to a DoS attack because it reads data without limiting the amount of characters. This can cause the application to run out of memory in cases of large texts, affecting the availability of the software.

The solution was to use a method that enforces a read length limit, such as BufferedReader.read().

### 3.7.1. Code changes

```
EIDAS-Node\src\main\java\eu\eidas\node\security\ContentSecurityPolicyReportServlet.java

This class has been removed as unused:
EIDAS-Node\src\main\java\eu\eidas\node\logging\integrity\HashFileChecker.java
```

### 3.7.2. Configuration changes

No changes made.

## 3.8. Cross-Site-Scripting: Sanitize input/output field

Ensure to apply adequate contextual encoding method regarding the context in which the data is displayed.

### 3.8.1. Code changes

```
EIDAS-Node\src\main\web\presentError.jsp
```

### 3.8.2. Configuration changes

No changes made.

## 3.9. Remove addressID and FullCvaddress

Removed FullCvaddress and AddressID attributes from PostalAddress to be align with the Specs.

### 3.9.1. Code changes

```
EIDAS-
Commons\src\main\java\eu\eidas\auth\commons\protocol\eidas\impl\AbstractPostalAddre
ssAttributeValueMarshaller.java
EIDAS-
Commons\src\main\java\eu\eidas\auth\commons\protocol\eidas\impl\PostalAddress.java
EIDAS-Commons\src
test\java\eu\eidas\auth\commons\protocol\eidas\impl\PostalAddressAttributeValueMars
hallerTest.java
```

### 3.9.2. Configuration changes

```
EIDAS-SAMLEngine\src\main\resources\eidas\CoreVocabularies-AggregateComponents-
1.1.xsd
```

## 3.10. Appends the trailing slash in websphere:

### 3.10.1. Configuration changes

For websphere, add the following property in server.xml file:

```
<webContainer com.ibm.ws.webcontainer.redirectcontextroot="true"/>
```

If set to true, and a request is made to the context root of an application with a missing trailing slash, the WebContainer appends the trailing slash. The WebContainer redirects to the URL with the appended slash before it applies any servlet filters defined in the application.

## 3.11. Cipher suites consolidation

### 3.11.1. Configuration changes

The eIDAS supported cipher suites have been consolidated according to the eIDAS-Crypto requirements. The eidas.xml file has been adapted and lists all supported cipher suites for Java7 and Java8. For authorized persons more details can be found at https://ec.europa.eu/cefdigital/wiki/x/6MXuAw

| | |
|---|---|
| tls.enabled.ciphers | For Java 7 supported Cipher suites are:<br><br>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,<br>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,<br>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,<br>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,<br>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,<br>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,<br>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,<br>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,<br>TLS_EMPTY_RENEGOTIATION_INFO_SCSV<br><br>For Java 8 supported Cipher suites are:<br><br>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,<br>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,<br>TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,<br>TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,<br>TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,<br>TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,<br>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,<br>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,<br>TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,<br>TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,<br>TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,<br>TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,<br>TLS_DHE_RSA_WITH_AES_128_CBC_SHA,<br>TLS_DHE_RSA_WITH_AES_256_CBC_SHA,<br>TLS_DHE_RSA_WITH_AES_128_CBC_SHA256,<br>TLS_DHE_RSA_WITH_AES_128_GCM_SHA256,<br>TLS_DHE_RSA_WITH_AES_256_CBC_SHA256,<br>TLS_DHE_RSA_WITH_AES_256_GCM_SHA384,<br>TLS_EMPTY_RENEGOTIATION_INFO_SCSV |

## 3.12. Other fixes/improvements requiring no action

### 3.12.1. Solving Threading issue in EIDAS-SpecificCommunicationDefinition

SpecificCommunicationApplicationContextProvider in package eu.eidas.specificcommunication was not thread safe. This class manages a static variable referencing the spring application context. The static reference was not protected against multi-threading despite the fact it has one getter and one setter. This could cause random issue while running unit test. Declaring volatile the application context reference solved the issue.

### 3.12.2. Refactoring of application context provider in EIDAS-Node

ApplicationContextProvider in eu.eidas.node package was renamed BeanProvider. The source code was updated accordingly. BeanProvider, just as ApplicationContextProvider before, is used for fetching instantiated beans managed by the Spring application context. Please note that the new getBean methods are using generics. This removes the need of typecasting the returned bean.

E.g here is a typical example of a BeanProvider.getBean call:

```
ConnectorControllerService connectorController = BeanProvider.getBean(ConnectorControllerService.class,
beanName);
```

The new implementation takes in account possible refresh of the Spring application context by implementing the Spring ApplicationListener<ContextRefreshedEvent>

### 3.12.3. Replace use of deprecated hash cryptographic algorithm SHA-1

At SecurityResponseHeaderHelper class the digest algorithm was changed from SHA-1 to SHA-256.

### 3.12.4. Metadata issuer whitelist URL was case insensitive, is now case sensitive

The issuer of either SAML Request or Response is tested against a whitelist of allowed URL's in case sensitive manner.

### 3.12.5. Cleanup of Colleague Request URL

At eidas.xml a still present obsolete entry <entry key="service2.url">http://eidasnode:8888/EidasNode/ColleagueRequest</entry>, was removed.

### 3.12.6. Added missing whitelist-based ESAPI encoder functions: encodeForHTML() for HTML content:

```
Change done at file EIDAS-Node\src\main\webapp\presentError.jsp
```

### 3.12.7. Build fails if tests not skipped

Cleaned up the failing tests in question.