eIDAS-Node Migration Guide v2.7.1

Table of Contents

Document history

Version	Date	Modification reason	Modified by
1.0	14/11/2023	Created	DIGIT

Disclaimer

This document is for informational purposes only and the Commission cannot be held responsible for any use which may be made of the information contained therein. References to legal acts or documentation of the European Union (EU) cannot be perceived as amending legislation in force or other EU documentation.
The document contains information of a technical nature and does not supplement or amend the terms and conditions of any procurement procedure; therefore, no compensation claim can be based on the contents of this document.

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 v2.7 to eIDAS-Node v2.7.1.

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 2.7.1.
Chapter 3 — eIDAS Node 2.7.1 "Dashboard".

Chapter 4 — *Changes*: Contains detailed information about the changes that should be taken into consideration when migrating to eIDAS-Node version 2.7.1.

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 2.7.1, including:

- configuration changes; and
- changes to code.

Disclaimer: The users of the eIDAS-Node sample implementation remain fully responsible for its integration with back-end systems (Service Providers and Identity Providers), testing, deployment and operation. The support and maintenance of the sample implementation, as well as any other auxiliary services, are provided by the European Commission according to the terms defined in the European Union Public License (EUPL) at https://joinup.ec.europa.eu/sites/default/files/custom-page/attachment/eupl_v1.2_en.pdf

2 Prerequisites

Before starting your migration to eIDAS-Node version 2.7.1 you should have:

- Already implemented eIDAS-Node version 2.7
- Downloaded the eIDAS-Node v2.7.1 Integration Package; and
- Downloaded the latest documentation.

3 eIDAS Node 2.7.1 "Dashboard"

This version aims to improve transparency within the eIDAS network.

Connectors and proxy services now have the option to disclose the endpoints and trust anchors they connect with.
In doing so, they make it possible for anyone to identify connections.
Connections in the eIDAS network are bilateral, a single sided connection may indicate a new or deprecated connection.

The dashboard is a centralised website where member states can manually enter information about the nodes they operate. In order to maintain consistency, the dashboard now has the capability to retrieve and compare this information directly from the connectors and proxy services.
Initially the dashboard started keeping watch over expired certificates in the network. This has expanded to include protocol version, bilateral trust, whitelisting and other elements that can be found in the metadata.
The new elements introduced on the metadata include trust anchors, trusted metadata URLs and algorithms available for symmetric encryption of the message.

4 Changes

4.1 Improvements for the eIDAS Dashboard

To enhance transparency in the eIDAS network, additional information about the interconnections is now published in the node's metadata. This information includes the trust anchor and the URLs of connected metadata.

	Trust Anchors	URI
Connector	connector-md-signature-trust-store	connector-recognized-urls
Proxy Service	service-md-signature-trust-store	service-recognized-urls

Additionally, a new field has been introduced to reflect the symmetric encryption capabilities of the proxy service.

```
http://eidas.europa.eu/entity-attributes/supported-encryption-algorithms
```

4.1.1 Code changes

- EIDAS-Commons
  - EIDASValues.java
    - Enum INTERCONNECTION_GRAPH_ENABLED was added
  - InterconnectionGraphData.java
    - new class InterconnectionGraphData was added
  - WhitelistUtil.java
    - new methods isUseWhiteList, metadataWhiteListHashes, getWhiteListURLs, getKey and hashUrl were added
  - WhitelistUtilTest.java
    - new unit tests testUseWhiteListTrue, testUseWhiteListFalse, testUseWhiteListNull, testGetWhiteListURLsSingleUrl, testGetWhiteListURLsMultipleUrl, testGetWhiteListURLsPropNotFound, testMultipleMetadataWhiteListHashes, testBlankMetadataWhiteListHashes and testSingleMetadataWhiteListHashes were added, along with private method createTestWhiteList
- EIDAS-Node-Connector
  - AUCONNECTORUtil.java
    - new method loadConfigServiceMetadataURLs was added
  - EidasNodeMetadataGenerator.java
    - variable AUNODEUtil was renamed to AUCONNECTORUtil
    - new variable whiteListConfigProperties was added
    - method generateConnectorMetadata
      - call to method generateMetadata was modified to account for new method signature
    - method generateMetadata
      - boolean idpRole was removed from method signature

            - code was added to retrieve the value of the interconnection.graph.enabled property and set it to the interconnectionGraphEnabled boolean, which in turn determines whether or not interconnection graph data will be added to metadata parameters
        - new method createInterconnectionGraphMetadata was added
        - new method setWhiteListConfigProperties was added
    - Application context
        - added new property whiteListConfigProperties to bean connectorMetadataGeneratorSP
    - EidasNodeMetadataGeneratorTest.java
        - unit test testGenerateMetadata was modified to account for rename of AUNODEUtil to AUCONNECTORUtil
        - unit test testGenerateMetadataWithContacts was modified to account for rename of AUNODEUtil to AUCONNECTORUtil
        - unit test testGenerateMetadataDecryptionCertificateException was modified to account for rename of AUNODEUtil to AUCONNECTORUtil
- EIDAS-Node-Proxy
    - EidasNodeMetadataGenerator.java
        - new variable whiteListConfigProperties was added
        - method generateMetadata
            - code was added to retrieve the value of the interconnection.graph.enabled property and set it to the interconnectionGraphEnabled boolean, which in turn determines whether or not interconnection graph data will be added to metadata parameters
        - new method createInterconnectionGraphMetadata was added
        - new boolean encryptionAlgorithmIsSupportedBySecurityProviders was added
        - new method setWhiteListConfigProperties was added
    - Application context
        - added new property whiteListConfigProperties to bean serviceMetadataGeneratorIDP
- EIDAS-SAMLEngine
    - AbstractProtocolSigner.java
        - new methods getTrustedCredentialGraphIdentifiers and getCertificateEncodedForm were added
    - ProtocolSignerI.java
        - new method getTrustedCredentialGraphIdentifiers was added
- EIDAS-SAML-Metadata
    - BaseMetadataFetcher.java
        - method isMetadataUrlWhitelisted
            - modified call to WhiteListUtil in order to get white list properties from application context when calling the class
    - EidasMetadataParameters.java

    - new variable interconnectionGraphData was added
    - Constructor was modified to include interconnectionGraphData
    - new methods getInterconnectionGraphData and setInterconnectionGraphData were added
    - method equals was modified to include a clause for interconnectionGraphData
    - method hashCode was modified to include a clause for interconnectionGraphData
- EidasMetadata.java
    - new constants were added:
        - INTERCONNECTION_GRAPH_CONNECTOR_TRUST_STORE_ATTRIBUTE_NAME
        - INTERCONNECTION_GRAPH_CONNECTOR_RECOGNISED_URLS_NAME
        - INTERCONNECTION_GRAPH_SERVICE_TRUST_STORE_ATTRIBUTE_NAME
        - INTERCONNECTION_GRAPH_SERVICE_RECOGNISED_URLS_NAME
        - INTERCONNECTION_GRAPH_SERVICE_SUPPORTED_ENCRYPTION_ALGORITHMS_NAME
    - Constructor was modified to account for change in inner class Generator of idpRole to isProxyService and of spRole to isConnector
    - Inner class Generator
        - boolean idpRole changed to isProxyService and boolean spRole changed to isConnector
        - variable interconnectionGraphData added to constructor
    - method generateEntityAttributes
        - Code block was added to add trust store signatures and recognised urls to metadata parameters if not null
        - Code block was added to add supported encryption algorithms to metadata parameters for the proxy service only
    - new methods generateTrustStoreAttribute, generateRecognizedUrlsAttribute, generateSupportedEncryptionAlgorithmsAttribute, hideAttributeWithEmptyValues, buildSAMLAttributeFromBase64String, createSamlAttributeValueFromString and createSamlAttributeValueFromBase64String were added
    - method generateEidasProtocolVersionAttributes was modified to hide attributes with empty values
    - method buildSamlAttribute was modified to call new method createSAMLAttributeValueFromString where xsi:type should be xsd:String
    - method createSamlAttributeValue was modified to accept new argument attributeXsiTypeValue
- EidasMetadataParametersI.java
    - added new methods setInterconnectionGraphData and getInterconnectionGraphData

4.1.2 Configuration Changes

- EIDAS-Node-Connector
    - resources/default/eidas.xml
        - added property **interconnection.graph.enabled** with default value set to false
- EIDAS-Node-Proxy
    - resources/default/eidas.xml
        - added property **interconnection.graph.enabled** with default value set to false

4.2 Update of example signature keystore configuration

Before 2.7.0 the configuration for signing was done with one or two keystores, grouping message signing and trust anchors together. In order not to display the message signing certificate as a Trust Anchor in the metadata **connector-md-signature-trust-store** and **service-md-signature-trust-store**, we have updated the example to use the eidasTrustStore.p12 for trust as is possible since 2.7.0 with 1 to n keystores.

4.2.1 Configuration Changes

EIDAS-Config/server/connector/SignModule_Connector.xml
EIDAS-Config/server/connector/SignModule_Connector_EC.xml
EIDAS-Config/server/proxy/SignModule_Service.xml
EIDAS-Config/server/proxy/SignModule_Service_EC.xml

4.3 Metadata keystore configuration fallback removed

Generally we work with at least 2 keystores to keep the trust chain separated from the trust anchors. (chain can be partially public)
This is important to prevent accidentally trusting a certificate that is not created by a member state (like a public CA).

In previous versions the value defined for the non-metadata keystore would be used when the configuration in SignModule.xml for metadata keystore had a missing configuration key.
This allowed for some or all keys to be left out for the metadata keystore.

In 2.7.0 we introduced multiple N.keystores. The application will now iterate through all keystores in order to find the message signing and metadata signing key pairs.
To keep the trust anchors separate, we introduced the new key keyStorePurpose TRUSTSTORE.
However in the 2.7.0 example the metadata.keyStorePurpose implicitly falls back to TRUSTSTORE.
To remove this side effect, we have removed the configuration key fallback code for metadata keystore.

Note: The keyStorePath and metadata.keyStorePath notation will be removed in the next major version in favor of 1.keyStorePath, 2.keyStorePath,...

4.3.1 Code changes

Removed methods tryConfigurationKeyPreferPrefix, getKeyStoreConfigurator, fetchLegacyConfiguration from EIDAS-SAMLEngine/src/main/java/eu/eidas/auth/engine/configuration/dom/KeyStoreSignatureConfigurator.java

4.3.2 Configuration changes

The following SignModule.xml metadata keystore configuration parameters no longer use values from the default keyStore:

- metadata.keyStorePath
- metadata.keyStoreType
- metadata.keyStoreProvider
- metadata.keyStorePassword
- metadata.keyStorePurpose
- metadata.keyAlias
- metadata.keyPassword

This means you might need to add one or more of these configuration parameters with the same value as their counterpart used for the keystore without any prefix.