European Commission

eIDAS-Node Demo Tools Installation and Configuration Guide

Version 1.4.3

Document history

Version	Date	Modification reason	Modified by
1.0	06/10/2017	Origination	DIGIT
1.4.1	15/06/2018	Document Version changed to correspond with the release. Reuse of document policy updated.	DIGIT
1.4.3	11/09/2018	Addition of a new Identity Provider Property field for not successful SAML Response generation.	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 a brief overview of technical nature and is not supplementing or amending terms and conditions of any procurement procedure; therefore, no compensation claim can be based on the contents of the present document.

Table of contents

List of tables

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 document describes the installation and configuration settings for Demo Tools (SP and IdP) supplied with the package for basic testing.

1.1. Purpose

The purpose of this document is to describe how to quickly install the Demo tools provided in the Integration Package (Service Provider (SP) and Identity Provider (IdP)) for testing purposes.

1.2. Document structure

This document is divided into the following sections:

- Chapter 1 – *Introduction*: this section.
- Chapter 2 – *Setting up the Demo Service Provider* provides information on the Demo SP properties to enable set up.
- Chapter 3 – *Setting up the Demo Identity Provider* provides information on the Demo IdP properties to enable set up.
- Chapter 4 – *Setting up the MS-Specific part* provides information on the Demo MS-Specific part properties to enable set up.

1.3. Other technical reference documentation

We recommend that you also familiarise yourself with the following eID technical reference documents which are available on **CEF Digital Home > eID > All eID services > eIDAS Node integration package > View latest version**:

- *eIDAS-Node Installation, Configuration and Integration Quick Start Guide* describes how to quickly install a Service Provider, eIDAS-Node Connector, eIDAS-Node Proxy Service and IdP from the distributions in the release package. The distributions provide preconfigured eIDAS-Node modules for running on each of the supported application servers.
- *eIDAS-Node Installation and Configuration Guide* describes the steps involved when implementing a Basic Setup and goes on to provide detailed information required for customisation and deployment.
- *eIDAS-Node National IdP and SP Integration Guide* provides guidance by recommending one way in which eID can be integrated into your national eID infrastructure.
- *eIDAS-Node and SAML* describes the W3C recommendations and how SAML XML encryption is implemented and integrated in eID. Encryption of the sensitive data carried in SAML 2.0 Requests and Assertions is discussed alongside the use of AEAD algorithms as essential building blocks.

- *eIDAS-Node Error and Event Logging* provides information on the eID implementation of error and event logging as a building block for generating an audit trail of activity on the eIDAS Network. It describes the files that are generated, the file format, the components that are monitored and the events that are recorded.
- *eIDAS-Node Security Considerations* describes the security considerations that should be taken into account when implementing and operating your eIDAS-Node scheme.
- *eIDAS-Node Error Codes* contains tables showing the error codes that could be generated by components along with a description of the error, specific behaviour and, where relevant, possible operator actions to remedy the error.

2. Setting up the Demo Service Provider

The Demo Service Provider (SP) can be used to simulate a SAML-based SP requesting authentication. It works with the default MS-Specific part using the same protocol language. Please note that this is not a guide for your national infrastructure, for implementation options please read the *eIDAS-Node National IdP and SP Integration Guide*.

The Basic Setup provides a preconfigured version of Demo Service Provider, however you may need to fine-tune some options.

The Service Provider `sp.properties` configuration details are described in the following table. The location of this file must be set by the `SP_CONFIG_REPOSITORY` environment variable or command line argument.

Table 1: Service Provider Properties

Key	Description
`provider.name`	Name for this Service Provider
`sp.sector`	Sector for this Service Provider
`sp.application`	Application for this Service Provider
`sp.return`	URL used when the eIDAS-Node Connector finishes the process. This must be the value of the machine running the Service Provider, its format is http://*sp.ip.address:sp.port.number/sp.deployment.name*/ReturnPage .
`sp.qaalevel`	QAA level of this Service Provider
`sp.country`	Country for this Service Provider
`sp.metadata.url`	The URL used to provide metadata (i.e. https://sp:8888/SP/metadata)
`sp.metadata.httpfetch`	Enables fetching of metadata from Specific-Connector. Enabled by default, if disabled, SP.metadata.repository.path must be set to a local metadata file to be processed.
`sp.metadata.repository.path`	If `httpfetch` is disabled, SP will read and monitor the specified directory for metadata files (any XML) instead of checking Specific-Connector provided metadata.
`sp.metadata.retention`	Retention period for SP internal Metadata cache in seconds.
`sp.metadata.validatesignature`	Check fetched or loaded metadata signature, before cached and used locally. Entity descriptors for URLs specified in `sp.metadata.trusteddescriptors` are not verified. Default: true.

Key	Description
`sp.metadata.trusteddescriptors`	List of entity descriptor URLs. Metadata from these sources are not verified by signature checking.
`eidas.protocol.version`	Value of eidas protocol version followed by the SP. When not empty, the value will be published in the SP's metadata URL.
`eidas.application.identifier`	Value of eidas protocol application identifier relative to the SdP code and version number. When not empty, the value will be published in the SP's metadata URL.

It is also possible to specify SP-provided metadata content in this file by setting:

- `contact.support.email, contact.support.company;`
- `contact.support.givenname, contact.support.surname;`
- `contact.support.phone, contact.technical.email;`
- `contact.technical.company;`
- `contact.technical.givenname, contact.technical.surname;`
- `contact.technical.phone;`
- `organization.name;`
- `organization.displayname;`
- `organization.url;`
- `signature.algorithm.whitelist;` and
- `encryption.algorithm.whitelist.`

The following table describes the available eIDAS-Node for this Service Provider.

Table 2: Available eIDAS-Node for Service Provider

Key	Description
`country.number`	The number of possible eIDAS-Nodes that can communicate with this SP
`countryX.name`	The name of the eIDAS-Node X(= positive integer)
`countryX.url`	The URL for the eIDAS-Node X. This must be the value of the machine running the eIDAS-Node using the format: http://*node.ip.address:node.port.number/node.deployment.name*/.
`countryX.countrySelector`	The URL for the `CountrySelector` of the eIDAS-Node X. This must have the value of the machine running the Service Provider, its format is http://*node.ip.address:node.port.number /sp.deployment.name*/CountrySelector

3. Setting up the Demo Identity Provider

In order to proceed with Basic Setup, you may need to modify the configuration of the Demo Identity Provider.

The `user.properties` holds the credentials for citizens who are able to log in. The format is: `<username>=<password>`.

If the citizen does not have an Attribute Provider (AP), the IdP can be used as a replacement. The `idp.properties` is used by the IdP to provide the attribute values in the format: `<username>.<attributeName>=<attributeValue>`.

Table 3: Sample of user.properties content

Key	Description
`myUser=myPassword`	A sample username and password
`myUser.LegalName=my legal name`	A sample attribute definition

The `idp.properties` holds configuration parameters about the application, especially metadata configuration. The location of this file must be set by the `IDP_CONFIG_REPOSITORY` environment variable or command line argument

Table 4: Identity Provider Properties

Key	Description
`idp.metadata.url`	The URL used to provide metadata (i.e. https://<idp.*yourHostname*>:<idp.*yourPort*>/IdP/metadata)
`idp.metadata.httpfetch`	Enables fetching of metadata from Specific-Proxy. Enabled by default, if disabled, `idp.metadata.repository.path` must be set to a local metadata file to be processed.
`idp.metadata.repository.path`	If `httpfetch` is disabled, IDP will read and monitor the specified directory for metadata files (any XML) instead of checking Specific-Proxy provided metadata.
`idp.metadata.retention`	Retention period for IdP internal Metadata cache, seconds.
`idp.metadata.validatesignature`	Check fetched or loaded metadata signature, before cached and used locally. Entity descriptors for URLs specified in `sp.metadata.trusteddescriptors` are not verified. Default: true.
`idp.metadata.trusteddescriptors`	List of entity descriptor URLs. Metadata from these sources are not being verified by signature checking.

Key	Description
`idp.ssos.post.location`	The URL for the metadata `<md:SingleSignOnService>` location attribute of the `SingleSignOnService` related to `Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST`. e.g. https://idp:8080/IdP/AuthenticateCitizen
`eidas.protocol.version`	Value of eidas protocol version followed by the IdP, e.g. "1.1". When not empty, the value will be published in the IdP's metadata URL.
`eidas.application.identifier`	Value of eIDAS protocol application identifier relative to the IdP code and version number, for example CEF:eIDAS-ref:1.4.3. When not empty, the value will be published in the IdP's metadata URL.
`include.assertion.fail.response.application.identifiers`	Values of the protocol versioning's application identifiers published in the metadata of the sender of the request, that need to receive not successful responses with an assertion. The values are comma or semicolon separated. For example, a possible value may be CEF:eIDAS-ref:1.4.2;CEF:eIDAS-ref:2.1, which will result in not successful response sent to CEF:eIDAS-ref:1.4.2 or CEF:eIDAS-ref:2.1 requesters will contain an assertion as before. Other cases will not. Please note that this property will eventually disappear in the future once transition period ends and not successful responses with an assertion are no longer needed.

As with the SP, the following IdP metadata content can be changed in the file `idp.properties`:

- `contact.support.email`;
- `contact.support.company`;
- `contact.support.givenname`;
- `contact.support.surname`;
- `contact.support.phone`;
- `contact.technical.email`;
- `contact.technical.company`;
- `contact.technical.givenname`;
- `contact.technical.surname`;
- `contact.technical.phone`;
- `organization.name`;
- `organization.displayname`;
- `organization.url`;
- `signature.algorithm.whitelist`; and

- `encryption.algorithm.whitelist`.

4. Setting up the MS-Specific part

The eIDAS-Node product contains a sample/demo Member State Specific part that is aligned with the use of Demo Tools.

There are some configuration items that might need to be customised according to the test environment. The configuration file name is `eidas_Specific.xml`, and is located by `SPECIFIC_EIDAS_REPOSITORY` environment variable or command line argument.

Table 5: Specific part properties

Key	Description
`check_certificate_validity_period`	Boolean value (true|false), which indicates if the application will disallow the use of obsolete certificates.
`disallow_self_signed_certificate`	Boolean value (true|false), which indicates if the application will disallow of the use of self-signed certificates.
`signature.algorithm`	The signing algorithm (SHA2 based) used by the default signer for outgoing requests. This optional setting is overriding ProtocolEngine configuration.
`signature.algorithm.whitelist`	The list of allowed signature algorithms (in incoming requests). It contains OpenSAML's supported signing algorithms, separated by ";".
`response.sign.assertions`	When set to true, the SAML Responses will have the attribute assertion signed
`encryption.algorithm.whitelist`	Contains the encryption algorithms allowed in the responses received.
`response.encryption.mandatory`	When set to 'true' the node will try to encrypt assertions in the generated SAML responses (provided that the encryption related configuration is in place).
`external.authentication`	Indicates if the authentication is external (set to true)
`idp.url`	URL of the Identity Provider
`idp.metadata.url`	URL of the Metadata of IdP

4.1. Derivation

The following is a feature demonstration of simple mapping between MS-Specific and eIDAS attributes. The code and configuration is in the provided sample MS-Specific.

Table 6: Derivation

Key	Description
`deriveAttr.number`	Number of derivation attributes
`deriveAttrX.id`	Id of the derivation attribute X(= positive integer)
`deriveAttrX.name`	Name of the derivation attribute X(= positive integer)

Table 7: Derivation attributes

Key	Description
`allow.unknowns`	If set to true will allow unknown values to be normalised.
`allow.derivation.all`	If set to true will allow the derivation of derived attributes, as opposed to, the derivation of the main attribute.

Table 8: Examples of derived attributes

Key	Description
`dateOfBirth.stork.format`	Defines the date format from the AP/IdP, currently 'yyyyMMdd'.
`dateOfBirth.hasSeparator`	Does the specific date have a separator char?
`dateOfBirth.specific.separator`	Defines the specific separator for the date.
`dateOfBirth.implemention`	Class that implements the `dateOfBirth` value normalization.

5. Demo Tools Migration

In this section it is briefly described, the relevant changes in the Demo Tools worth mentioning occurred from previous version related either to code or configuration.

5.1. Code changes

At Demo IdP module, ProcessLogin was changed so that it can send a not successful SAML Response with or without assertion based on the application identifier of the issuer of the related SAML Request.

5.2. Configuration changes

The idp.properties has a new property `include.assertion.fail.response.application.identifiers` that needs to be added. For further details, please find the description of this property in Table 4: Identity Provider Properties.