eIDAS-Node Installation, Configuration and Integration

Quick Start Guide

Version 1.4.4

Document history

Version	Date	Modification reason	Modified by
1.0	26/11/2015	Modifications to align with the eIDAS technical specifications.	DIGIT
1.1.0	29/06/2016	Modifications due to installation changes related to architectural and stability improvements Update of the deployments configuration and related libraries	DIGIT
1.2.0	20/01/2017	Configuration and stability improvements, please see Version 1.2.0 Release Notes.	DIGIT
1.3.0	05/05/2017	Modifications to align with changes in Technical Specifications version 1.1. For details please see the Version 1.3.0 Release Notes.	DIGIT
1.4. Pre-Release	31/08/2017	Modifications to remove support for JBoss6. Support WebLogic 12.2 family of servers. Amend filename conventions to change '\' to '/'.	DIGIT
1.4. Official release	06/10/2017	Error corrections and improvements	DIGIT
1.4.1 Official release	15/06/2018	Document Version changed to correspond with the release. Reuse of document policy updated.	DIGIT
1.4.4 Official release	14/12/2018	New Deliverable and Description release content details were added. New Properties and Values were added to the eIDAS-Node hostname and port table. Demo eIDAS-Node set up and configuration now contains information on Weblogic and Websphere profiles. Paths for Glassfish and Tomcat additional files were changed.	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 abbreviations

The following abbreviations are used within this document.

Abbreviation	Meaning
eIDAS	electronic Identification and Signature. The Regulation (EU) N°910/2014 governs electronic identification and trust services for electronic transactions in the internal market to enable secure and seamless electronic interactions between businesses, citizens and public authorities.
IdP	Identity Provider. An institution that verifies the citizen's identity and issues an electronic ID.
LoA	Level of Assurance (LoA) is a term used to describe the degree of certainty that an individual is who they say they are at the time they present a digital credential.
MS	Member State.
SAML	Security Assertion Markup Language
SP	Service Provider

List of definitions

The following definitions are used within this document.

Term	Meaning
eIDAS-Node	An eIDAS-Node is an application component that can assume two different roles depending on the origin of a received request. See eIDAS-Node Connector and eIDAS-Node Proxy Service.
eIDAS-Node Connector	The eIDAS-Node assumes this role when it is located in the **Service Provider's** Member State. In a scenario with a Service Provider asking for authentication, the eIDAS-Node Connector receives the authentication request from the Service Provider and forwards it to the eIDAS-Node of the citizen's country.
eIDAS-Node Proxy Service	The eIDAS-Node assumes this role when it is located in the **citizen's** Member State. The eIDAS-Node Proxy Service receives authentication requests from an eIDAS-Node of another MS (their eIDAS-Node Connector). The eIDAS-Node Proxy-Service also has an interface with the national eID infrastructure and triggers the identification and authentication for a citizen at an identity and/or attribute provider.

1. Introduction

This document describes how to quickly install a Service Provider, eIDAS-Node Connector, eIDAS-Node Proxy Service and IdP from the distributions in this release package. The distributions provide preconfigured eIDAS-Node modules for running on each of the supported application servers (Glassfish, Tomcat, JBoss, WebLogic and WebSphere).

Detailed information on the setup and configuration of the sample eIDAS-Nodes, is included in the *eIDAS-Node Installation and Configuration Guide*.

Detailed information on integration of the eIDAS-Node into your national infrastructure is included in the *eIDAS-Node National IdP and SP Integration Guide*.

This document is divided into the following sections:

- Section 1 – *Introduction*: this section.
- Section 2 – *Release content*: lists the files delivered with this release and describes their contents;
- Section 3 – *Overview of the preconfigured demo eIDAS-Node packages*: illustrates the setup of the configurations provided with this distribution;
- Section 4 – *Demo eIDAS-Node set up and configuration*: describes step-by-step how to install the demo configuration;
- Section 5 – *Specific configuration*: provides information on how the setup can be changed to suit your needs;
- Section 6 – *Compiling the modules from the source*: describes how to rebuild the Maven project if necessary;
- Section *7* — *Enabling logging*: describes how to enable audit logging of the communications between eIDAS-Node Proxy Service and Connector.

2. Release content

For information on the changes in this release, please see the current Release Notes.

The deliverable consists of the following zip files:

Deliverable	Description
EIDAS-1.4.4.zip	Distribution version 1.4.4 of the sample eIDAS-Node
EIDAS-Sources-1.4.4.zip	Source files (Maven project) of the sample eIDAS-Node including an example of implementation of a SP (service provider) and IdP (Identity Provider).
EIDAS-Binaries-Glassfish-1.4.4.zip	Deployable war files of a preconfigured eIDAS-Node for a Glassfish server (including IdP.war, EidasNode.war, SP.war)
EIDAS-Binaries-Jboss-1.4.4.zip	Deployable war files of a preconfigured eIDAS-Node for a JBoss server (including IdP.war, EidasNode.war, SP.war)
EIDAS-Binaries-Tomcat-1.4.4.zip	Deployable war files of a preconfigured eIDAS-Node for a Tomcat server (including IdP.war, EidasNode.war, SP.war)
EIDAS-Binaries-Was-1.4.4.zip	Deployable war files of a preconfigured eIDAS-Node for a WebSphere server (including IdP.war, EidasNode.war, SP.war)
EIDAS-Binaries-Wls-1.4.4.zip	Deployable war files of a preconfigured eIDAS-Node for a WebLogic server (including IdP.war, EidasNode.war, SP.war)

3. Overview of the preconfigured demo eIDAS-Node packages

This distribution provides an example configuration in which each supported server represents one country providing an eID service. For the purpose of this demo, fictitious countries are used (CA, CB, CC, CD, CF).

The following table illustrates the setup of the configurations provided with this distribution.

Application Server	version	Default host	Default port	Country	Description
Tomcat	7*, 8	localhost	8080	CA	Country A
Glassfish	3, 4*	localhost	8081	CB	Country B
JBoss	7*	localhost	8085	CC	Country C
WebLogic	12.1.2 12.2.2	localhost	7001	CD	Country D
WebSphere/ WebSphere Liberty Profile	8.5.5* 8.5.5.4	localhost	9080	CF	Country F

* Default build server provided with the binaries

4. Demo eIDAS-Node set up and configuration

Each example eIDAS-Node package is preconfigured to use 'localhost' as hostname and a default http listening port; see the table in section 3. The http listening port of your application server must be adapted according to these default values.

If you need to change these default values, refer to section *5.1 — Changing the default hostname or http port* for details.

To set up and configure the demo, perform the following steps:

1. If Oracle provided JVM is going to be used, then it is necessary to apply the JCE Unlimited Strength Jurisdiction Policy Files, which contain no restriction on cryptographic strengths:

    a. Download the Java Cryptography Extension (JCE) Unlimited Strength Policy Files from Oracle:

        - For Java 7: http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html
        - For Java 8: http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html

    b. Uncompress and extract the downloaded zip file (it contains `README.txt` and two jar files).

    c. For the installation, please follow the instructions in the `README.txt` file.

2. Follow the steps below according to your type of server:

    If you are using Tomcat 7:

    a. Create a folder named `shared` in `$TOMCAT_HOME`.

    b. Create a subfolder named `lib` in `$TOMCAT_HOME/shared`.

    c. Edit the file `$TOMCAT_HOME/conf/catalina.properties` and change the property `shared.loader` so that it reads:

    ```
    shared.loader=${catalina.home}/shared/lib/*.jar
    ```

    d. Copy the files below to the new `shared/lib` directory:

    ```
    xml-apis-1.4.01.jar
    resolver-2.9.1.jar
    serializer-2.7.2.jar
    xalan-2.7.2.jar
    endorsed/xercesImpl-2.12.0.jar
    ```

If you are using Tomcat 8:

Copy the files below to the existing `lib` directory on the application server. These jars may be found under *AdditionalFiles/endorsed/* directory in the binary for your application server.

```
xml-apis-1.4.01.jar
resolver-2.9.1.jar
serializer-2.7.2.jar (rename this file to serializer.jar)
xalan-2.7.2.jar
xercesImpl-2.12.0.jar
```

If you are using Glassfish:

Copy the files below to the *$GLASSFISH_DOMAIN/lib/ext/* directory on the application server. These jars may be found under *AdditionalFiles/endorsed/* directory in the binary for your application server.

```
xml-apis-1.4.01.jar
resolver-2.9.1.jar
serializer-2.7.2.jar
xalan-2.7.2.jar
xercesImpl-2.12.0.jar
```

If you are using JBoss:

Copy the content of `AdditionalFiles/JBOSS7` directory into the `modules` directory on the application server.

If you are using Weblogic:

The xml-apis-1.4.01.jar file should be placed under `DOMAIN_HOME/lib/`

If you are using Websphere liberty profile:

The xml-apis-1.4.01.jar file should be placed under $SERVER_HOME/lib/global

3. It is necessary to increase the default JVM memory settings. Set the following JVM parameter in the startup script of your application server `-XX:MaxPermSize=512m`.

4. Local directory or directories must be defined in order to store the configuration files and the test keystores. These directories need to be defined either as OS/AS environment variables or command-line parameters:

   `EIDAS_CONFIG_REPOSITORY` for EidasNode,
   `SPECIFIC_CONFIG_REPOSITORY` for Specific,
   `SP_CONFIG_REPOSITORY` for SP
   `IDP_CONFIG_REPOSITORY` for IdP.

   It is also possible to use only one common directory for all the modules. JVM command line example:

```
-DEIDAS_CONFIG_REPOSITORY=c:/Pgm/projects/configEidas/glassfish/
-DSPECIFIC_CONFIG_REPOSITORY=c:/Pgm/projects/configEidas/glassfish/specific/
```

```
-DSP_CONFIG_REPOSITORY=c:/Pgm/projects/configEidas/glassfish/sp/
-DIDP_CONFIG_REPOSITORY=c:/Pgm/projects/configEidas/glassfish/idp/
```

By default the configuration file structure (e.g. Glassfish) must be as follows:

```
glassfish/eidas.xml
glassfish/encryptionConf.xml
glassfish/EncryptModule_Connector.xml
glassfish/EncryptModule_Service.xml
glassfish/hazelcast.xml
glassfish/saml-engine-additional-attributes.xml
glassfish/SamlEngine.xml
glassfish/SamlEngine_Connector.xml
glassfish/SamlEngine_Service.xml
glassfish/SignModule_Connector.xml
glassfish/SignModule_Service.xml
glassfish/idp/EncryptModule_IdP.xml
glassfish/idp/idp.properties
glassfish/idp/IdPSamlEngine.xml
glassfish/idp/saml-engine-additional-attributes.xml
glassfish/idp/saml-engine-eidas-attributes.xml
glassfish/idp/SamlEngine_IdP.xml
glassfish/idp/SignModule_IdP.xml
glassfish/idp/user.properties
glassfish/sp/EncryptModule_SP.xml
glassfish/sp/saml-engine-additional-attributes.xml
glassfish/sp/saml-engine-eidas-attributes.xml
glassfish/sp/SamlEngine_SP.xml
glassfish/sp/SignModule_SP.xml
glassfish/sp/sp.properties
glassfish/sp/SPSamlEngine.xml
glassfish/specific/eidas_Specific.xml
glassfish/specific/EncryptModule_SP-Specific.xml
glassfish/specific/EncryptModule_Specific-IdP.xml
glassfish/specific/saml-engine-additional-attributes.xml
glassfish/specific/SamlEngine_SP-Specific.xml
glassfish/specific/SamlEngine_Specific-IdP.xml
glassfish/specific/SignModule_SP-Specific.xml
glassfish/specific/SignModule_Specific-IdP.xml
glassfish/specific/SpecificSamlEngine.xml
keystore/demoKeys.jks
keystore/eidasKeyStore.jks
keystore/eidasKeyStore_Connector_CA.jks
keystore/eidasKeyStore_Connector_CB.jks
keystore/eidasKeyStore_Connector_CC.jks
keystore/eidasKeyStore_Connector_CD.jks
keystore/eidasKeyStore_Connector_CF.jks
keystore/eidasKeyStore_IDP_CA.jks
keystore/eidasKeyStore_IDP_CB.jks
keystore/eidasKeyStore_IDP_CC.jks
keystore/eidasKeyStore_IDP_CD.jks
keystore/eidasKeyStore_IDP_CF.jks
keystore/eidasKeyStore_METADATA.jks
keystore/eidasKeyStore_Service_CA.jks
keystore/eidasKeyStore_Service_CB.jks
keystore/eidasKeyStore_Service_CC.jks
keystore/eidasKeyStore_Service_CD.jks
keystore/eidasKeyStore_Service_CF.jks
keystore/eidasKeyStore_SP_CA.jks
keystore/eidasKeyStore_SP_CB.jks
keystore/eidasKeyStore_SP_CC.jks
keystore/eidasKeyStore_SP_CD.jks
keystore/eidasKeyStore_SP_CF.jks
```

Please note: all components in the binary distribution are preconfigured for the file system layout indicated above. Deviating from this layout will require changes to the configurations of the individual modules. Please refer to the *eIDAS-Node Installation and Configuration Guide* for more details.

5. Copy the server configuration files and test keystores into the local directories defined in step 4.

   Open the zip file (`config.zip` in the `EIDAS-Binaries-xxx-yyy.zip`) and copy the directory `keystore` and the directory of the application server as required (i.e. `glassfish`, `tomcat`, `jboss`, `wls`, `was`) into the configuration directory.

6. On WebSphere Liberty Profile the following features should be enabled:

   ```
   <feature>jsp-2.2</feature>
   <feature>servlet-3.0</feature>
   <feature>ssl-1.0</feature> (if planning to use https)
   ```

7. On WebSphere, use `BouncyCastle` provider instead of default IBM JVM default provider:

   a. Place `bouncycastle jar` in `$IBM_JRE/lib/ext` directory.

   b. Copy the IBM unrestricted JCE policy files provided in `AdditionalFiles` directory and put them under `$IBM_JRE/lib/security` to replace the existing ones.

   c. Add `bouncycastleprovider` in the list of providers in `$IBM_JRE/lib/security/java.security` file before the default provider, e.g.

   ```
   security.provider.1=com.ibm.crypto.pkcs11impl.provider.IBMPKCS11Impl
   security.provider.2=org.bouncycastle.jce.provider.BouncyCastleProvider
   security.provider.3=com.ibm.crypto.provider.IBMJCE
   ```

8. Add a static JCE for JBOSS 7:

   a. Locate and open in a text editor the file `$JRE_HOME/lib/security/java.security`.

   b. Add a line after the lines containing the security providers:
   `security.provider.N=`
   `org.bouncycastle.jce.provider.BouncyCastleProvider`
   (you should set N according to your config, to the next available index in the list of providers).

   c. Put `bcprov-jdk15on-1.51.jar` into the classpath (e.g. `$JRE_HOME/lib/ext`).

9. Deploy the applications according to your application server.

   - `EidasNode.war`
   - `SP.war`
   - `IdP.war`

You now have a Service Provider, eIDAS-Node Connector, eIDAS-Node Proxy Service and IdP configured to run on localhost:

- Tomcat: `http://localhost:8080/SP/`
- Glassfish: `http://localhost:8081/SP`
- Jboss: `http://localhost:8085/SP`
- WebLogic: `http://localhost:7001/SP`
- WebSphere, WebSphere Liberty Profile: `http://localhost:9080/SP/`

To validate the installation, a first test can be performed simulating that a citizen from a country accesses services in the same country.

1. Open the Service Provider URL : `http://localhost:defaultport/SP/`
2. Choose for both the SP and citizen country the fictitious country for which your application server has been configured (CA, CB, CC, CD or CF).
3. The generated `SAMLRequest` is displayed. Submit the form.
4. Click **Next** to give your consent to attributes being transferred.
5. Enter the user credentials. Type 'xavi' as **Username** and 'creus' as **Password** and submit the page.
6. Click **Submit** to validate the values to transfer.

   The `SAMLResponse` is displayed.
7. **Submit** the form.

   You should see **Login Succeeded**.

5. Specific configuration

5.1. Changing the default hostname or http port

The parameters below can be adapted to reflect your configuration.

Note: The application server must be restarted after changes have been made.

5.1.1. eIDAS-Node hostname and port

1. Edit the file `eidas.xml` located in the configuration directory as shown below.

Property	Value
`connector.assertion.url`	http://<connector.*yourHostname*>:<connector.*yourPort*>/EidasNode/ColleagueResponse
`connector.destination.url`	http://<connector.*yourHostname*>:<connector.*yourPort*>/EidasNode/ServiceProvider
`connector.node.url`	http://<connector.*yourHostname*>:<connector.*yourPort*>/EidasNode/
`connector.metadata.url`	http://<connector.*yourHostname*>:<connector.*yourPort*>/EidasNode/ConnectorMetadata
`service.node.url`	http://<service.*yourHostname*>:<service.*yourPort*>/EidasNode/
`service1.url`	http://<service.*yourHostname*>:<service.*yourPort*>/EidasNode/ColleagueRequest
`service.specificidpredirect.url`	http://<service.*yourHostname*>:<service.*yourPort*>/EidasNode/IdpResponse
`service.metadata.url`	http://<service.*yourHostname*>:<servicer.*yourPort*>/EidasNode/ServiceMetadata
`connector.responder.metadata.url`	The URL at which the metadata of the eIDAS-Node Connector (presenting itself as an IdP) will be made available, e.g. http://<connector.*yourHostname*>:<connector.*yourPort*>/EidasNode/ConnectorResponderMetadata. It will be used as Issuer in the responses that the Connector sends to the SP.
`service.requester.metadata.url`	The URL where the metadata of the Proxy Service (presenting itself as an SP) will be made available, e.g. http://<service.*yourHostname*>:<service.*yourPort*>/EidasNode/ServiceRequesterMetadata . It will be used as Issuer in the requests that the Connector sends to the IdP.
`ssos.serviceMetadataGeneratorIDP.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. http://<service.*yourHostname*>:<service.*yourPort*>/EidasNode/ColleagueRequest/

Property	Value
ssos.serviceMetadataGeneratorIDP.redirect.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-Redirect. e. g. http://*<service.yourHostname>*:*<service.yourPort>*/EidasNode/ColleagueRequest/
ssos.connectorMetadataGeneratorIDP.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. http://*<connector.yourHostname>*:*<connector.yourPort>*/EidasNode/ServiceProvider
ssos.connectorMetadataGeneratorIDP.redirect.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-Redirect. e.g. http://*<connector.yourHostname>*:*<connector.yourPort>*/EidasNode/ServiceProvider
security.header.CSP.report.uri	The value of this property should be the full url used by the reporting feature of CSP.
idp.metadata.location.whitelist	Semicolon separated urls of the IdP's that the Service Proxy is allowed to connect to obtain metadata/certificate to verify signature of SAML responses.
sp.metadata.location.whitelist	Semicolon separated urls of the SP's that the Connector is allowed to connect to obtain metadata/certificate to verify signature of SAML requests.
connector.metadata.location.whitelist	Semicolon separated urls of the Node Connectors that the ServiceProxy is allowed to connect to obtain metadata/certificate to verify signature of SAML requests.

2. Open and edit the file sp.properties as shown below.

Property	Value
country1.metadata.url	http://<connector.*yourHostname*>/EidasNode/ConnectorResponderMetadata

5.1.2. SP hostname and port

Open and edit the file sp.properties as shown below.

Property	Value
sp.return	http:// <sp.*yourHostname*>:<sp.*yourPort*>/SP/ReturnPage
sp.metadata.url	http:// <sp.*yourHostname*>:<sp.*yourPort*>/SP/metadata

5.1.3. IdP hostname and port

1. Edit the file eidas_Specific.xml located in the configuration folder as shown below.

Property	Value
idp.url	http://<idp.*yourHostname*>:<idp.*yourPort*>/IdP/AuthenticateCitizen
idp.metadata.url	https://<idp.*yourHostname*>:<idp.*yourPort*>/IdP/metadata

2. Open and edit the file idp.properties as shown below.

Property	Value
idp.metadata.url	http://<idp.*yourHostname*>:<idp.*yourPort*>/IdP/metadata
idp.ssos.redirect.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-Redirect. e.g. http://<idp.*yourHostname*>:<idp.*yourPort*>/IdP/AuthenticateCitizen
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. http://<idp.*yourHostname*>:<idp.*yourPort*>/IdP/AuthenticateCitizen

5.2. Changing the keystore location

By default the test keystores are located in the directory 'keystore' in the same directory as the configuration directory. You can change these values by editing the files below to reflect your configuration. All filenames and path information are relative to the configuration directory for the given module.

Keystore	Files
eIDAS-Node	SignModule_Service.xml SignModule_Connector.xml EncryptModule_Service.xml EncryptModule_Connector.xml
SP	SignModule_SP.xml EncryptModule_SP.xml
IdP	SignModule_IdP.xml EncryptModule_IdP.xml
Specific	SignModule_SP-Specific.xml SignModule_Specific-IdP.xml EncryptModule_SP-Specific.xml EncryptModule_Specific-IdP.xml

5.3. Changing keystore configuration

By default the preconfigured eIDAS components use the following extended configuration.

5.3.1. Extended configuration

In this configuration all stakeholders (SP/Connector /Proxy Service/IDP) use their own certificate for the signing and encrypting of SAML messages.

This setup is close to a real-life scenario, where the components are distributed across servers and Member States.

Example for country 'CA':

	Keystore		Certificate	Country
SP	eidasKeyStore_SP_CA.jks (SignModule_SP.xml, EncryptModule_SP.xml)	Key Pair	sp-ca-demo-certificate	CA
		Trusted	Metadata (signing certificate)	CA
Connector	eidasKeyStore_Connector_CA.jks (SignModule_SP-Specific.xml, SignModule_Connector.xml, EncryptModule_SP-Specific.xml EncryptModule_Connector.xml)	Key Pair	Connector-ca-demo-certificate	CA
		Trusted	Metadata (signing certificate)	CA
Proxy Service	eidasKeyStore_Service_CA.jks (SignModule_Service.xml, SignModule_Specific-IdP.xml, EncryptModule_Service.xml, EncryptModule_Specific-IdP.xml)	Key Pair	Service-ca-demo-certificate	CA
		Trusted	Metadata (signing certificate)	CA
IDP	eidasKeyStore_IDP_CA.jks (SignModule_IdP.xml, EncryptModule_IdP)	Key Pair	idp-ca-demo-certificate	CA
		Trusted	Metadata (signing certificate)	CA
Metadata	eidasKeyStore_METADATA.jks	Key Pair	Metadata (signing certificate)	CA

5.3.2. Basic configuration

In this configuration all stakeholders share the same certificate.

This setup is a simplified scenario for a lab environment, but corresponds less to a real-life situation.

In order to set up the basic scenario, all `SignModule` configuration files should be adapted to reference the common test keystore, `eidasKeyStore.jks`.

5.4. Preventing a citizen from authenticating in a country other than the requested one

1. By default the preconfigured Demo eIDAS-Node has a protection which does not allow citizens to authenticate in a country other than the one that has been requested.

2. If you need to disable this validation, edit the file `eidas.xml` located in the configuration directory.

Property	Value
`check.citizenCertificate.serviceCertificate`	false

5.5. eIDAS-Node compliance

For validation purposes the demo eIDAS Nodes do not use HTTPS and the configuration parameters are set as shown below. The parameters can be changed to be fully eIDAS compliant if required.

Parameter	Demo value	eIDAS value
`disallow_self_signed_certificate`	False	True: do not allow self-signed and expired certificates
`check_certificate_validity_period`	False	True: do not allow expired certificates
`metadata.activate`	True	True:specifies that metadata is generated by the Connector
`metadata.restrict.http`	False	True: metadata must be only available via HTTPS
`tls.enabled.protocols`	TLSv1.1,TLSv1.2	TLSv1.1,TLSv1.2: SSL/TLS enabled protocols
`metadata.check.signature`	True	True : metadata received from a communications partner must be signed
`metadata.validity.duration`	86400	Metadata validity period in seconds. Default=86400 (i.e. one day)
`response.encryption.mandatory`	True	True: do not allow response not encrypted
`validate.binding`	True	True: the bindings are validated
`security.header.csp.enabled`	True	True: the content-security and security

Parameter	Demo value	eIDAS value
		checks are enabled
disable.check.mandatory.eidas.attributes	False	False: check the eIDAS minimum dataset constraint. **Note:** this parameter is used by both Proxy Service and Connector.
disable.check.representative.attributes	False	True: disable the check of representative attributes in the request

6. Compiling the modules from the source

If you need to rebuild the Maven project, open EIDAS-Parent and execute the Maven commands described in the table below according to your application server.

Folder	Command line	
EIDAS-Parent	Tomcat/ Glassfish	`mvn clean install -P tomcat`
	jBoss7	`mvn clean install -P jBoss7`
	WebLogic	`mvn clean install -P weblogic`
	WebSphere	`mvn clean install -P websphere`

You can also rebuild each module separately by executing the commands below.

Folder - modules	Command line	
EIDAS-Light-Commons	`mvn clean install`	
EIDAS-Commons	`mvn clean install`	
Eidas-ConfigModule	`mvn clean install`	
EIDAS-SpecificCommunicationDefinition	`mvn clean install`	
EIDAS-SAMLEngine	`mvn clean install`	
EIDAS-Encryption	`mvn clean install`	
EIDAS-UPDATER	`mvn clean install`	
EIDAS-Specific	`mvn clean install`	
EIDAS-Node	Tomcat/ Glassfish	`mvn clean package -P tomcat`
	jBoss7	`mvn clean package -P jBoss7`
	WebLogic	`mvn clean package -P weblogic`
	WebSphere	`mvn clean package -P websphere`
EIDAS-SP	Tomcat/ Glassfish	`mvn clean package -P tomcat`
	jBoss7	`mvn clean package -P jBoss7`

Folder - modules	Command line	
	WebLogic	`mvn clean package -P weblogic`
	WebSphere	`mvn clean package -P websphere`
EIDAS-IdP-1.0	Tomcat/ Glassfish	`mvn clean package -P tomcat`
	jBoss7	`mvn clean package -P jBoss7`
	WebLogic	`mvn clean package -P weblogic`
	WebSphere	`mvn clean package -P websphere`

7. Enabling logging

The locations of the audit files are by default configured to use a Java system properties variable called `LOG_HOME`.

A value can be assigned to this variable by using: `-DLOG_HOME="<myDirectoryName>"` at server start-up.

Note: The eIDAS-Node logs may contain person identification data, hence these logs should be handled and protected appropriately in accordance with the European privacy regulations [Dir. 95/46/EC] and [Reg. 2016/679].

[Reg. 2016/679] REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC.

[Dir. 95/46/EC] Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data.