EUROPEAN COMMISSION

DIGIT
Digital Europe Programme

Quick Start Guide

for the DomiSML (Business Document Metadata Service Location (BDMSL))

Version [2.8]

Status [Final]

Date: 07/09/2023

Document Approver(s):

Approver Name	Role
Joao RODRIGUES	eDelivery
Bogan DUMITRIU	Project Manager

Document Reviewers:

Reviewer Name	Role
Jože RIHTARŠIČ	Developer

Summary of Changes:

Version	Date	Created by	Short Description of Changes
1.5	26/08/2016	Adrien FERIAL	Version
1.6	30/09/2016	Yves ADAM	Align to new template
1.7	19/07/2017	Flavio SANTOS	Add configuration parameters
1.8	16/01/2018	Flavio SANTOS	Add encryption configuration parameters
1.9	27/03/2018	CEF Support	Reuse policy notice added, e-SENS profile replaced by eDelivery profile
1.10	24/04/2018	Flavio SANTOS	Configuration files in detail
2.0	08/05/2019	Jože RIHTARŠIČ	Update for SML 4.0
2.1	04/06/2019	Jože RIHTARŠIČ	configuration for weblogic/oracle and add chapter for BIND DNS configuration added
2.2	21/01/2020	Jože RIHTARŠIČ	Add configuration parameters
2.3	30/11/2020	Jože RIHTARŠIČ	Added description for configuration property: dnsClient.use.legacy.regexp
2.4	07/04/2022	Caroline AEBY	No more CEF
2.5	11/8/2022	Jože RIHTARŠIČ	SML 4.2 updates. Add reference to property table defined in Software Architecture Document. Update supported versions for Tomcat and MySQL
2.6	14/10/2022	Caroline AEBY	In heading, CEF replaced by Digital Europe Programme.
2.7	05/07/2023	Caroline AEBY	DomiSML 4.2.1 release
2.8	07/09/2023	Jože RIHTARŠIČ Caroline AEBY	DomiSML 4.3 release: Add support for JDK 11 Add examples for sig0 key generation

Table of Contents

1. INTRODUCTION

DomiSML was previously called BDMSL, which stands for Business Document Metadata Service Location. DomiSML is the sample implementation of the SML maintained by DG DIGIT. The version of the DomiSML/BDMSL refered in this document is 4.x versions. This version implements the eDelivery BDXL profile (see https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/eDelivery+BDXL).

1.1. Purpose of the Quick Start Guide

This document provides a brief description of the installation of the DomiSML component. Opposite to previous version, this version of the application does not use Liquibase as a database management tool. Before the installation, a database must be created using SQL scripts bundled in the sml-4.x-setup.zip file. The application bussines properties are stored in the database table BDMSL_CONFIGURATION. Application properties such as datasource JNDI, log folder, etc., are located in the smp.config.properties which must be located in the classpath of the server.

This guide illustrates the different steps to install the DomiSML application on a Tomcat server with a MySQL database and Weblogic 12.2.1.4 with an oracle database.

1.2. Pre-requisites

Please install the following software on the target system. For further information and installation details, please refer to the software owner's documentation.

- Java runtime environment (JRE) 8 and 11 **only**:
  http://www.oracle.com/technetwork/java/javase/downloads/index.html
- **One** of the supported Database Management Systems:
  - MySQL 8.0.x (tested version, future versions might also work)
  - Oracle 11g XE and Oracle 19c (tested versions, future versions might also work)
- **One** of the supported Application Servers:
  - Tomcat 9.x (tested with Adoptium JDK 11)
  - WebLogic 12.2 (tested with Oracle JDK 8)

1.3. Binaries repository

The eDelivery DomiSML artefacts can be downloaded from the Digital site[1].

1.4. Source Code Repository

The source code of eDelivery DomiSML is available in the **GIT** repository at the following location:

https://ec.europa.eu/digital-building-blocks/code/projects/EDELIVERY/repos/bdmsl/browse

As mentioned in the prerequisites, the deployment of the eDelivery DomiSML was only tested on Tomcat 9 and WebLogic 12.2.1.4 application server.

The deployment of the eDelivery DomiSML is made of the following mandatory steps:

- Database configuration
- Application Server preparation
- DomiSML Initial configuration
- DomiSML file deployment

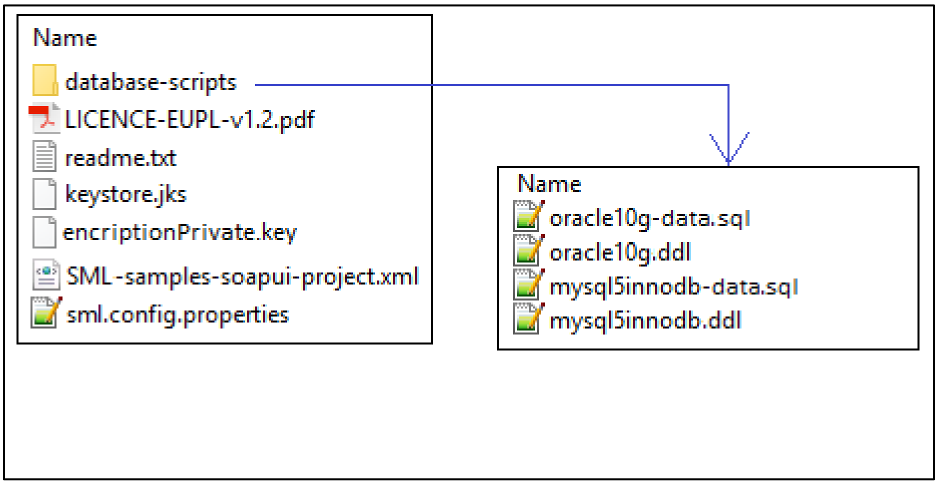
Remark:

> *The environment variable, **edelivery_path**, refers to the name of the folder where the DomiSML package is installed (**CATALINA_HOME for Tomcat** and **DOMAIN_HOME for Oracle Weblogic**).*

1.5. Database Scripts

The scripts to create (or migrate) the Oracle or MySQL databases are included in the following downloadable zip file from the Digital site (section §1.3): sml-4.x-setup.zip.

[1] https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/SML



Name
database-scripts
LICENCE-EUPL-v1.2.pdf
readme.txt
keystore.jks
encriptionPrivate.key
SML-samples-soapui-project.xml
sml.config.properties

Name
oracle10g-data.sql
oracle10g.ddl
mysql5innodb-data.sql
mysql5innodb.ddl

2. Database Creation

This section describes the steps necessary to create the database, the tables and the DomiSML database user (**dbuser** used for database connection purpose).

For this step you need to use the script included in the zip file downloaded in section §1.5.

2.1. MySQL database

1. Download and copy the mysql5innoDb.ddl script to edelivery_path/database-scripts

2. Open a command prompt and navigate to the edelivery_path/database-scripts folder

3. Execute the following MySQL commands **(WARNING: this step will <u>delete</u> the user schema if it already exists in the database)**:

```
mysql -h localhost -u root_user --password=root_password -e "drop schema if
exists bdmsl_schema;create schema bdmsl_schema;alter database bdmsl_schema
charset=utf8; create user sml_dbuser identified by 'sml_password';grant all
on bdmsl_schema.* to sml_dbuser;"
```

This creates the bdmsl_*schema* and a *bdmsl database_dbuser* with (all) privileges for the bdmsl_schema**.**

Execute the following command to create the required objects (tables, etc.) in the database:

```
mysql -h localhost -u root_user -proot_password bdmsl_schema <
mysql5innoDb.ddl
```

Execute the following command to set up the initial data:

```
mysql -h localhost -u root_user -proot_password bdmsl_schema <
mysql5innoDb-data.sql
```

2.2. Oracle database

1. Download and copy the **oracle10g.ddl script** to *edelivery_path/sql-scripts*

2. Navigate to *edelivery_path*/**sql-scripts** directory

3. Execute the following commands:

```
sqlplus sys as sysdba (password should be the one assigned during the
Oracle installation)
```

```
=================================================================== Once
logged in Oracle: create user sml_dbuser identified by sml_dbpassword;
grant all privileges to sml_dbuser;
connect sml_dbuser
show user
-- run the scripts with the @ sign from the location of the scripts
@oracle10g.ddl  (the Oracle database creation)
@oracle10g-data.sql  (the Oracle init data)

exit
```

3. Tomcat Configuration

In order to deploy the DomiSML on Tomcat, the steps below need to be completed.

3.1. Configuring the Extra CLASSPATH for Tomcat

In this Tomcat example, directories:

- **keystores**
- **logs**
- **classes**

will be created in the root path of the Tomcat installation (**CATALINA_HOME**). The content of the folder ***keystores*** will contain all security artifacts such as Keystore, Truststore, and encryption key.

The content of the folder ***classes*** is the DomiSML configuration file "sml.config.properties". And the folder must be added as **CLASSPATH** variable in the Tomcat batch file (CATALINA_HOME/bin/setenv.[sh|bat]). If the file *setenv.[sh|bat]* does not exist it must be created.

For Linux:

Edit the CATALINA_HOME/bin/setenv.sh file

```
#!/bin/sh
# Set CLASSPATH to include sml environment property file:
# sml.config.properties
export CLASSPATH=$CATALINA_HOME/classes
```

For Windows:

Edit the %CATALINA_HOME%/bin/setenv.bat file

```
REM Set CLASSPATH to include sml environment property file:
REM sml.config.properties


set classpath=%classpath%;%catalina_home%\classes
```

Place the **sml.config.properties** (DomiSML environment property file) in the folder classes.

Example can be downloaded from the Digital site (section §1.3): sml-4.x-setup.zip. Detailed description of environment properties is in section §5.1.

For tomcat/mysql configuration the file must have following properties and values:

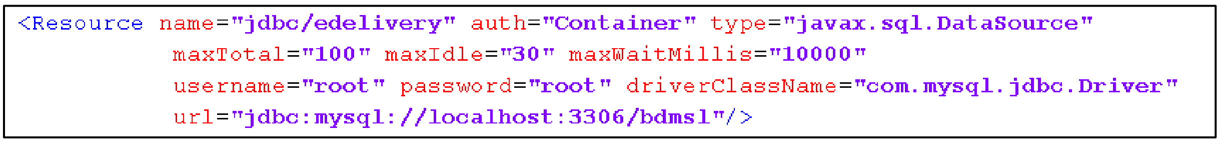
```
sml.hibernate.dialect=org.hibernate.dialect.MySQLDialect
sml.datasource.jndi=java:comp/env/jdbc/edelivery
sml.jsp.servlet.class=org.apache.jasper.servlet.JspServlet
# (Absolute/Relative) path to logs folder. Update the value!
sml.log.folder=/opt/tomcat/logs/
# Optional parameter(s) to set the init data at first startup
# The properties are copied to database table BDMSL_CONFIGURATION

# configurationDir:  set the absolute path to the “keystores” folder
configurationDir=/opt/tomcat/keystores/
```

3.2. Configuring the Datasource for Tomcat

Create a new data source in Tomcat named: java:comp/env/jdbc/edelivery.

For that go to TOMCAT_HOME/conf/context.xml and add the block:



```
<Resource name="jdbc/edelivery" auth="Container" type="javax.sql.DataSource"
          maxTotal="100" maxIdle="30" maxWaitMillis="10000"
          username="root" password="root" driverClassName="com.mysql.jdbc.Driver"
          url="jdbc:mysql://localhost:3306/bdmsl"/>
```

3.3. JDBC Driver

The JDBC driver needs to be downloaded from the manufacturer website:

- For Mysql: https://www.mysql.com/products/connector/

The JDBC driver (.jar file) must be copied to the following directory: edelivery_path/lib.

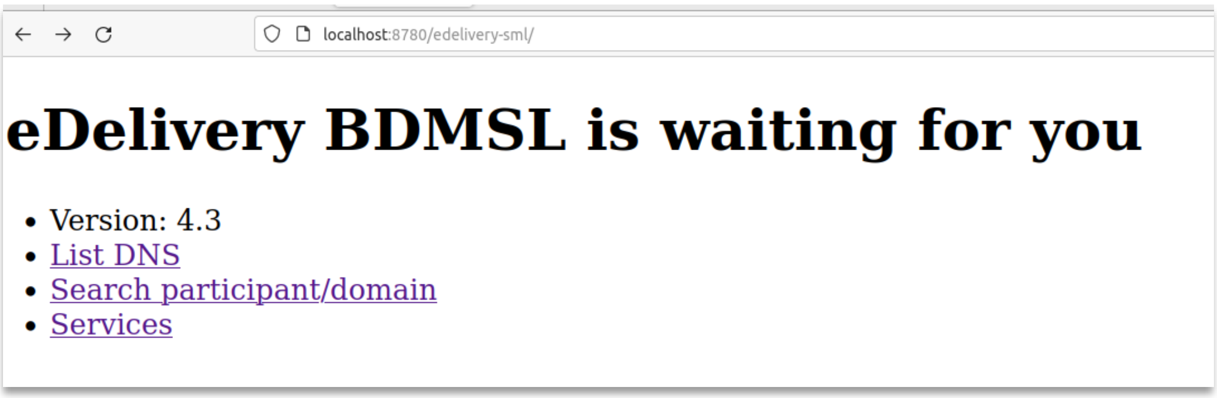
3.4. Deployment

Copy the **cef_bdmsl-webapp-4.X.war** file to the Tomcat **webapps** directory (edelivery path/webapps).

3.5. Verification of the Installation

Use your browser to go to the following address: **http://*[hostname]*:*[port]*/bdmsl-webapp-4.3/**
NOTE: URL context path must match the “war” filename. If the deployment filename is edelivery-sml.war, then DomisSML URL address is: http://[hostname]:[port]/edelivery-sml/

If the deployment is successful, the following page is displayed:



localhost:8780/edelivery-sml/

eDelivery BDMSL is waiting for you

- Version: 4.3
- List DNS
- Search participant/domain
- Services

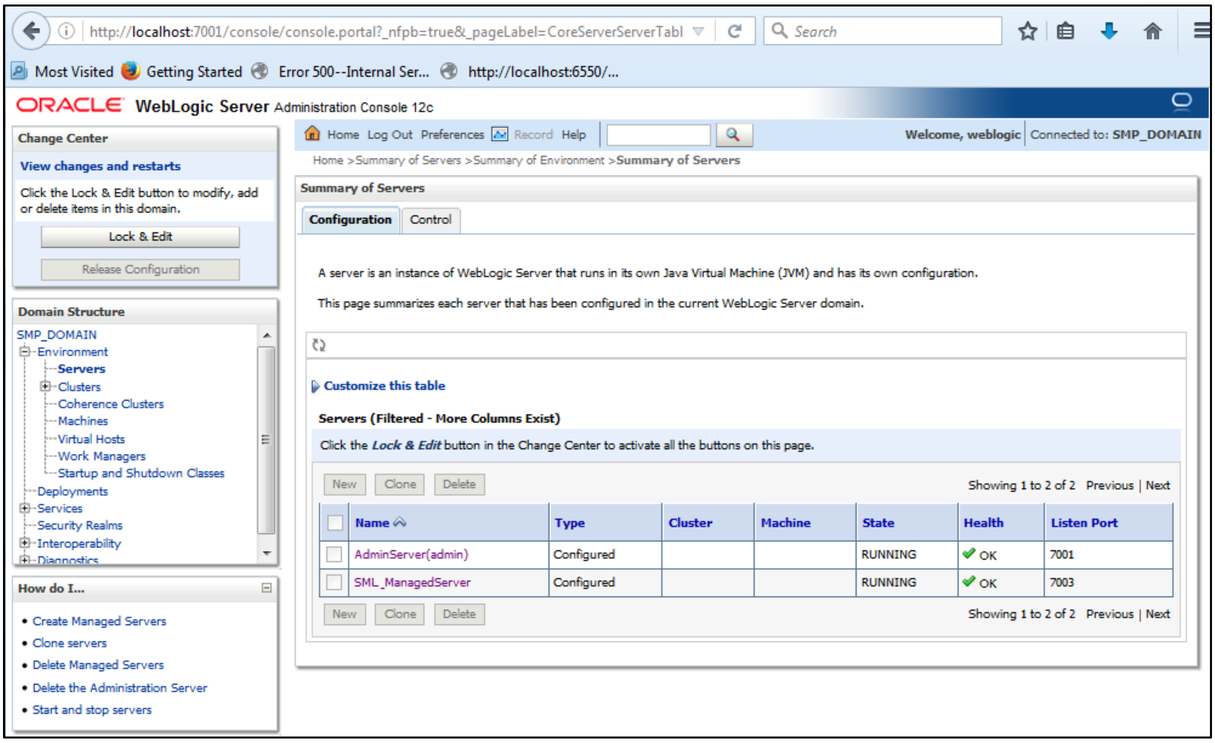
Important: Context path (example above: **/edeliery-sml/**) should be the same as is deployment WAR file. If the war file is called ***sml.war*** then the URL will be **http://*[hostname]*:*[port]*/*sml***.

4. WEBLOGIC CONFIGURATION

This section does not include the installation of a WebLogic 12.2.x application server. It is assumed that the WebLogic Server is installed, and a WebLogic domain is created with an administration server and a managed server on which the DomiSML will be deployed.
Hereafter the domain location will be referred as *DOMAIN_HOME* (user-defined name).
In the examples below, we will use the following Domain and Server names:

- Domain Name : SMLDOMAIN
- Administration Server : AdminServer
- SMP Managed Server : SML_ManagedServer

As shown below:


http://localhost:7001/console/console.portal?_nfpb=true&_pageLabel=CoreServerServerTabl
Search
Most Visited
Getting Started
Error 500--Internal Ser...
http://localhost:6550/...
ORACLE WebLogic Server Administration Console 12c
Home Log Out Preferences Record Help
Welcome, weblogic Connected to: SMP_DOMAIN
Home >Summary of Servers >Summary of Environment >Summary of Servers
Change Center
View changes and restarts
Lock & Edit
Release Configuration
Domain Structure
SMP_DOMAIN
Environment
Servers
Clusters
Coherence Clusters
Machines
Virtual Hosts
Work Managers
Startup and Shutdown Classes
Deployments
Services
Security Realms
Interoperability
Diagnostics
How do I...
Create Managed Servers
Clone servers
Delete Managed Servers
Delete the Administration Server
Start and stop servers
Summary of Servers
Configuration
Control
Customize this table
New Clone Delete
Showing 1 to 2 of 2 Previous | Next

Click the Lock & Edit button to modify, add or delete items in this domain.

A server is an instance of WebLogic Server that runs in its own Java Virtual Machine (JVM) and has its own configuration.

This page summarizes each server that has been configured in the current WebLogic Server domain.

Servers (Filtered - More Columns Exist)

Click the *Lock & Edit* button in the Change Center to activate all the buttons on this page.

□	Name	Type	Cluster	Machine	State	Health	Listen Port
□	AdminServer(admin)	Configured			RUNNING	✓ OK	7001
□	SML_ManagedServer	Configured			RUNNING	✓ OK	7003

In order to deploy the SMP on the WebLogic Application Server platform, two preliminary steps need to be completed:

- Configuring the Extra CLASSPATH for WebLogic
- Configure datasource

This is described in the following two sections.

4.1. Configuring the Extra CLASSPATH for WebLogic

Under the DOMAIN_HOME directory, create the following sub-directories:

- **keystores**
- **logs**

- **classes**

Edit the WebLogic DOMAIN_HOME/bin/setDomainEnv.sh.
For Linux:
Add the **EXPORT CLASSPATH=${CLASSPATH}:${DOMAIN_HOME}/classes/** statement at the end of the CLASSPATH definition as shown below:

```
../
if [ "${PRE_CLASSPATH}" != "" ] ; then
CLASSPATH="${PRE_CLASSPATH}${CLASSPATHSEP}${CLASSPATH}"
export CLASSPATH
fi
CLASSPATH=${CLASSPATH}:${DOMAIN_HOME}/classes
export CLASSPATH
/..
```

For Windows:

```
../
If NOT "%PRE_CLASSPATH%"=="" (
set CLASSPATH=%PRE_CLASSPATH%;%CLASSPATH%
)
set CLASSPATH=%CLASSPATH%;%DOMAIN_HOME%\classes
/..
```

Place the **sml.config.properties** (DOMISML environment property file) in the folder classes.

An example can be downloaded from the Digital site (section §1.3): sml-4.x-setup.zip. Detailed description of environment properties is in section §1.3.

For weblogic/oracle configuration, the file must have following properties and values:

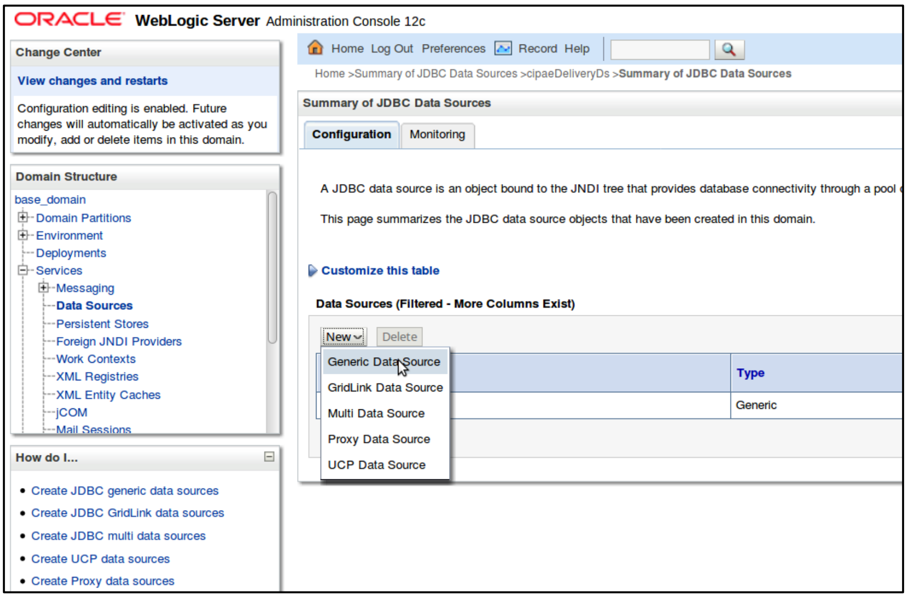
```
sml.hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
sml.datasource.jndi=jdbc/cipaeDeliveryDs
sml.jsp.servlet.class=weblogic.servlet.JSPServlet

# (Absolute/Relative) path to logs folder. Update the value!
sml.log.folder=/opt/tomcat/logs/

# Optional parameter(s) to set the init data at first startup
# The properties are copied to database table BDMSL_CONFIGURATION
# configurationDir:  set the absolute path to the “keystores” folder
configurationDir=/opt/tomcat/keystores/
```

4.2. Configuring datasource for WebLogic

Clik on Services/Data sources on left Domain structure panel. Then on configuration tab click on button 'New' and select 'Generic data source'.

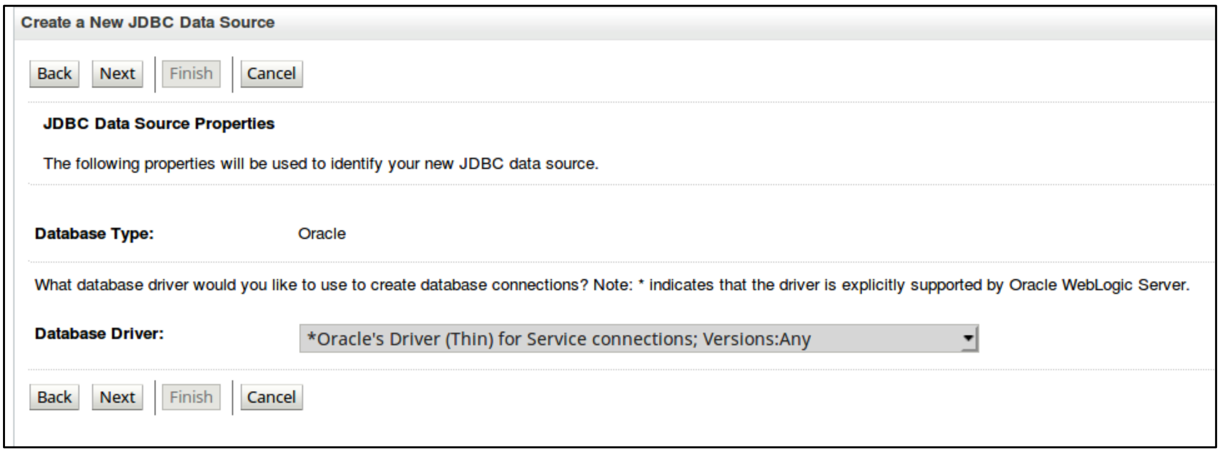

ORACLE WebLogic Server Administration Console 12c
Change Center
View changes and restarts
Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.
Domain Structure
base_domain
Domain Partitions
Environment
Deployments
Services
Messaging
Data Sources
Persistent Stores
Foreign JNDI Providers
Work Contexts
XML Registries
XML Entity Caches
jCOM
Mail Sessions
How do I...
Create JDBC generic data sources
Create JDBC GridLink data sources
Create JDBC multi data sources
Create UCP data sources
Create Proxy data sources
Home Log Out Preferences Record Help
Home >Summary of JDBC Data Sources >cipaeDeliveryDs >Summary of JDBC Data Sources
Summary of JDBC Data Sources
Configuration
Monitoring
A JDBC data source is an object bound to the JNDI tree that provides database connectivity through a pool
This page summarizes the JDBC data source objects that have been created in this domain.
Customize this table
Data Sources (Filtered - More Columns Exist)
New
Delete
Generic Data Source
GridLink Data Source
Multi Data Source
Proxy Data Source
UCP Data Source
Type
Generic

New datasource wizard ' *Create a New Data Source* ' is triggered which will guide you thought Datasource creation. In the first wizard page, enter the following values:

Set Name value: *cipaeDeliveryDS*
JNDI name: jdbc/*cipaeDeliveryDS*
Database Type: *oracle*

Click then on next.

In next wizard page select Database driver: Oracle's Driver (Thin) and click next twice.



JDBC Data Source Properties

The following properties will be used to identify your new JDBC data source.

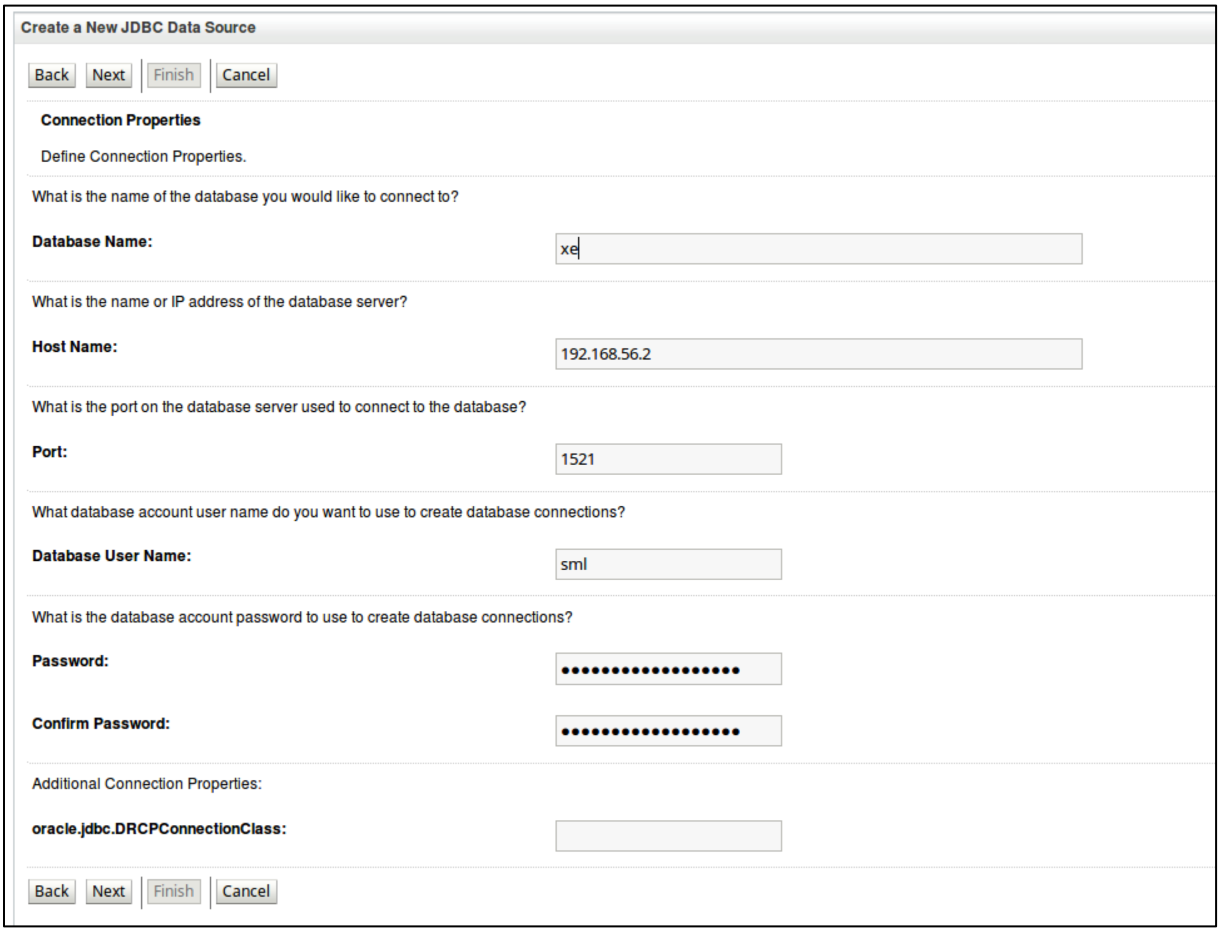
Database Type: Oracle

What database driver would you like to use to create database connections? Note: * indicates that the driver is explicitly supported by Oracle WebLogic Server.

Database Driver: *Oracle's Driver (Thin) for Service connections; Versions:Any

In the following wizard page, enter the datasource values (the values below are just an example: use the values from your oracle configuration):

Database Name: xe
Port: 1521
Database user sml_dbUser
Pasword: sml_dbPassword
Confirm password: sml_dbPassword



Create a New JDBC Data Source

Connection Properties

Define Connection Properties.

What is the name of the database you would like to connect to?

Database Name: xe

What is the name or IP address of the database server?

Host Name: 192.168.56.2

What is the port on the database server used to connect to the database?

Port: 1521

What database account user name do you want to use to create database connections?

Database User Name: sml

What is the database account password to use to create database connections?

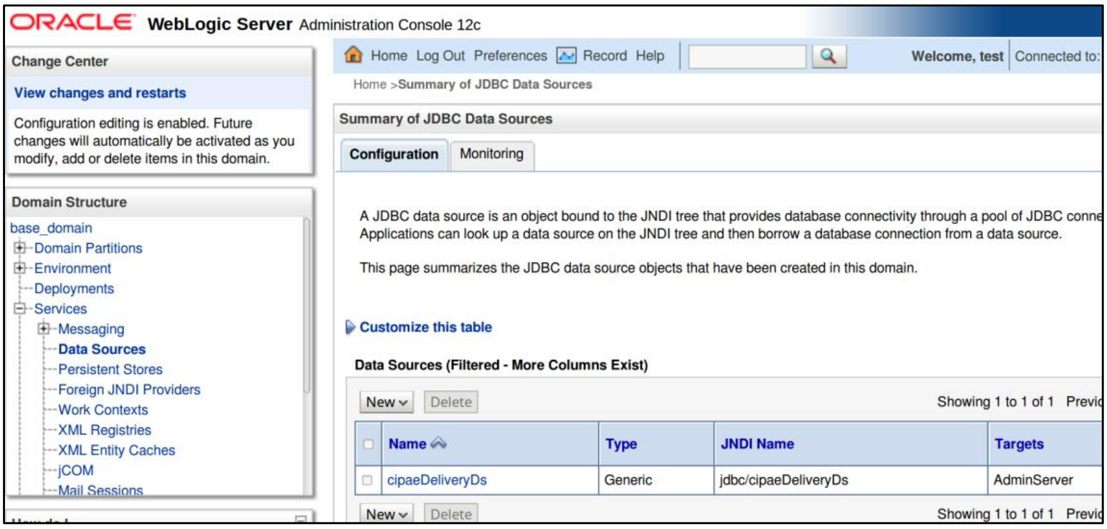
Password: ••••••••••••••••••

Confirm Password: ••••••••••••••••••

Additional Connection Properties:

oracle.jdbc.DRCPConnectionClass:

Then click 'Next' followed by click on 'Finish' button. Then a new Datasource configuration appears in the datasource table:



Configuration editing is enabled. Future changes will automatically be activated as you modify, add or delete items in this domain.

Summary of JDBC Data Sources

A JDBC data source is an object bound to the JNDI tree that provides database connectivity through a pool of JDBC conne
Applications can look up a data source on the JNDI tree and then borrow a database connection from a data source.

This page summarizes the JDBC data source objects that have been created in this domain.

Data Sources (Filtered - More Columns Exist)

□	Name	Type	JNDI Name	Targets
□	cipaeDeliveryDs	Generic	jdbc/cipaeDeliveryDs	AdminServer

4.3. Deployment

Deploy the **.war** file within WebLogic using the Oracle Weblogic deployer feature or using the Weblogic Administration Console.

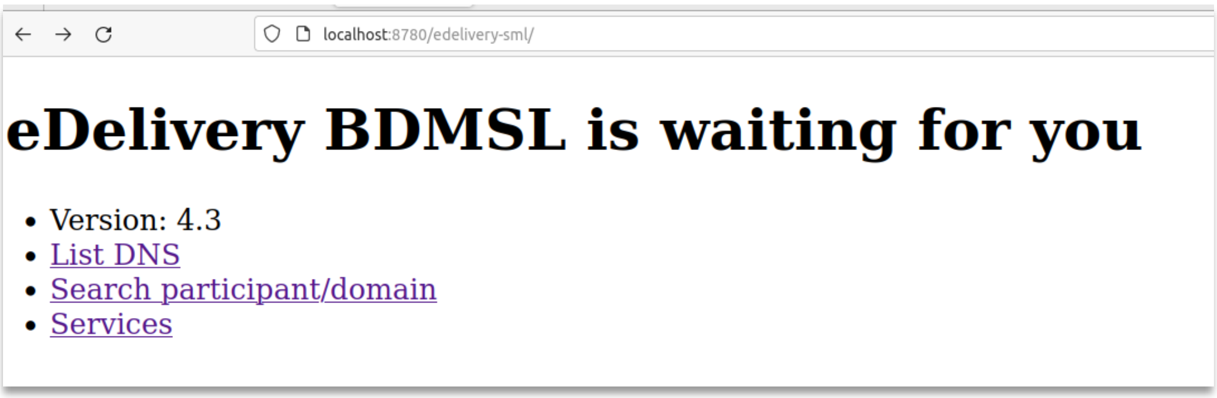
An example of using the Oracle the **weblogic.deployer** is shown below:

```
java weblogic.Deployer -adminurl
t3://${WebLogicAdminServerListenAddress}:${WebLogicAdminServerPort} \
-username ${WebLogicAdminUserName} \
-password ${WebLogicAdminUserPassword} \
-deploy -name edelivery-sml.war \
-targets ${SMP_ManagedServer} \
-source $TEMP_DIR/edelivery-sml.war
```

4.4. Verification of the Installation

Use your browser to navigate to the following address: http://*[hostname]*:*[port]*/edelivery-sml/

If the deployment is successful, the following page is displayed:



localhost:8780/edelivery-sml/

eDelivery BDMSL is waiting for you

- Version: 4.3
- List DNS
- Search participant/domain
- Services

5. Configuration

5.1. Environment parameters

DomiSML application has environment parameters stored in property file sml.config.properties. Configuration is in property file because they are required before database connection. In the setup bundle sml-4.x-setup.zip (section §1.5), there is example of configuration preset for Tomcat/MySql installation:

```
# **********************************
# Hibernate dialect configuration
# **********************************
# Oracle hibernate example
#sml.hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
# Mysql dialect
sml.hibernate.dialect=org.hibernate.dialect.MySQLDialect

# **********************************
# Datasource JNDI configuration
# **********************************
# weblogic datasource JNDI example
#sml.datasource.jndi=jdbc/cipaeDeliveryDs
# tomcat datasource JNDI example
sml.datasource.jndi=java:comp/env/jdbc/edelivery

# **********************************
# JSP implementation configuration
# **********************************
# Weblogic
#sml.jsp.servlet.class=weblogic.servlet.JSPServlet
# tomcat, jboss
sml.jsp.servlet.class=org.apache.jasper.servlet.JspServlet


# **********************************
# Logging implementation
# **********************************
sml.log.folder=./logs/
```

The configuration file has the following parameters:

- **sml.hibernate.dialect**: hibernate dialect for accessing the database
- **sml.datasource.jndi**: datasource JNDI name configured in sections §3.2and §4.2
- **sml.jsp.servlet.class**: application server implementation of the JSP framework
- **sml.log.folder**: logging folder.

5.2. DomiSML parameters

DomiSML application contains its parameters in database table BDMSL_CONFIGURATION. Parameters can be updated:

- via the sql script as showed below:

```
mysql -h localhost -u root_user -proot_password bdmsl_schema -e "update
bdmsl_configuration set value='true', last_updated_on=NOW() where
property='unsecureLoginAllowed'";
```

- or by calling the webservice operation: BDMSLAdminServices/SetProperty(). For more details, check the ICD document.

All properties are refreshed without server restart, except CRON schedule definitions: sml.property.refresh.cronJobExpression, certificateChangeCronExpression and dataInconsistencyAnalyzer.cronJobExpression.

Properties are refreshed as defined by the cron property: sml.property.refresh.cronJobExpression. By default, properties are refreshed (if changed) every hour. If a property is changed by the sql script, make sure that the value ***last_updated*** is also changed, otherwise the properties will not be updated.

For the list of properties and their description, please refer to the document "Software Architecture Document" on
https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/SML .

5.3. How to generate a private key file

DomiSML uses a private key for encrypting/decrypting passwords. If the key is not present in the folder defined in property "configurationDir" at the startup, it will automatically create and store it.

When deploying DomiSML (especially on production), make sure its unique encryption key is generated for the deployment. Below is an example of how to manually create the key.

To create a private key, please follow the steps below:

- Download one of the latest DomiSML war files (eg: bdmsl-webapp-4.0.x.war) from the repository https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/SML

- Extract the war file using any extracting tool

- Run the following commands to create a private key:

    1. cd bdmsl-webapp-4.3

    2. java -cp "WEB-INF/lib/*" eu.europa.ec.bdmsl.common.util.PrivateKeyGenerator c:\temp\encriptionPrivateKey.private

    Required parameter = Full directory path where the private key will be created

Example:

Printed result:

Private key created at c:\temp\encriptionPrivateKey.private

Once the private key is generated, please copy the private key file name to the value of the property `encriptionPrivateKey` in the table BDMSL_Configuration, and copy the private file to the path configured in the property `configurationDir`.

5.4. How to encrypt a password

DomiSML encrypts passwords automatically when setting the password property using the WebService “SetProperty”.

After generating a private key at item “§5.3- How to generate a private key file”, please configure the proxy or keystore (used to sign response) password if needed as follows:

- Inside the folder already extracted from BDMSL .war file, please run the command below:

    java -cp "WEB-INF/lib/*" eu.europa.ec.bdmsl.common.util.EncryptPassword c:\temp\privateKey.private Password123

    1st parameter = private key location

    2nd parameter = plain text password

- To configure the proxy password, please copy the printed encrypted and base64 encoded password to the value of the `httpProxyPassword` property in the table `BDMSL_CONFIGURATION`.

    Example:

    httpProxyPassword = vXA7JjCy0iDQmX1UEN1Qwg==

- To configure the keystore password, please copy the printed encrypted and base64 encoded password to the value of the `keystorePassword` property in the table BDMSL_CONFIGURATION.

Example:

keystorePassword = vXA7JjCy0iDQmX1UEN1Qwg==

5.5. Truststore for the certificate revocation list (CRL) download over the HTTPS

The DomiSML establishes the “HTTPS” trust to the server hosting the CLR list by using the application server system truststore, which is defined by system variables: javax.net.ssl.trustStrore, javax.net.ssl.trustStoreType, etc.

If the truststore is not set, the default java truststore is used in the following location: ${JAVA_HOME}/jre/lib/security/cacerts

To enable the download of the CRL files over HTTPS, please make sure the appropriate certificates are registered in the application server truststore.

Example of how to add/register the “crl-server” certificate to default java cacerts truststore:

```
"${JAVA_HOME}"/bin/keytool -importcert -alias crl-server -keystore
"${JAVA_HOME}/jre/lib/security/cacerts" -storepass changeit -file
/opt/smlconf/init-configuration/sml_crl_crl-server.cer -noprompt
```

5.6. Certificate to sign responses

If the flag `signResponse=true` in the table `BDMSL_CONFIGURATION`, a keystore file name, its alias and password must be provided in the same table.

For testing purposes only, the provided keystore.p12 (pass: test123) can be used (the keystore contains RSA, EC,ED25519 and ED448 key examples).

The keystore is located in the configuration bundle sml-setup-${VERSION}.zip.

5.7. Files to be copied under application server

In the configuration directory that you specified in the `configurationDir` property, you need to put the following files:

- `keystore.p12` (the name can be changed in the property `keystoreFileName`): this keystore must contain your private key with the alias and password defined in the `keystoreAlias` and `keystorePassword` properties.
- `sig0.private` (the name can be changed in the property `dnsClient.SIG0KeyFileName`): this file is only required if you use DNSSEC (i.e. property `dnsClient.SIG0Enabled` set to true).
- `encriptionPrivateKey.private` (the name can be changed in the property `encriptionPrivateKey`): this private key file is only required if you use Proxy or Sign Response.

Once the needed files have been copied, restart the server(s).

5.8. DNS integration

DomiSML was developed and tested with using a BIND9 DNS server. The DNS integration can be switched on/off by setting attribute **dnsClient.enabled** to *true/false*. If the property is set to true, the parameter **dnsClient.server** must contain the hostname/ip address of the DNS server.

To secure the DNS integration, DomiSML has implemented SIG(0). This option can be enabled/disabled by the following parameter: **dnsClient.SIG0Enabled**, with values: *true/false.*

If the option is set to false, the DNS should allow updates to **any** ip address (this is **NOT** advised in production environment) or restrict the update permission to the requester **ip address**.

Below is example of configuration for BIND9 zone example.edelivery.eu.local without the use of SIG(0) (in this case the DomiSML should have **dnsClient.SIG0Enabled=false**):

```
zone "example.edelivery.eu.local" {
    type master;
    file "/var/lib/bind/db.example.edelivery.eu.local ";
    allow-update { 10.22.1.3;}
    allow-transfer { 10.22.0.0/16; };
};
```

5.8.1. Securing DNS integration with SIG(0)

SIG0 are asymmetric key-pairs, usually with a filename ending with .key for a public key, and a filename ending with .private for a private key.

SIG(0) key pair can be created with dnssec-keygen utility (the tool is provided as part of a BIND9 DNS server)

Example of a command to generate the keys:

```
# DSA key
dnssec-keygen  -a DSA -b 1024 -n HOST -T KEY sig0.example.edelivery.eu.
local
# Example of bash command for generating the RSA type

dnssec-keygen -a RSASHA256 -b 4096 -T KEY -n HOST domisml-
rsasha256.test.edelivery.local

dnssec-keygen -a RSASHA512 -b 4096 -T KEY -n HOST domisml-
rsasha512.test.edelivery.local


# Example of command for generating the EC type key

dnssec-keygen -a ECDSAP256SHA256 -T KEY -n HOST domisml-
ecdsap256sha256.test.edelivery.local

dnssec-keygen -a ECDSAP384SHA384 -T KEY -n HOST domisml-
ecdsap384sha384.test.edelivery.local

# Example of command for generating the Edward curve type key

dnssec-keygen -a ED25519 -T KEY -n HOST domisml-
ed25519.test.edelivery.local
```

```
dnssec-keygen -a ED448 -T KEY -n HOST domisml-ed448.test.edelivery.local
```

Note: DSA algorithm is absolete on newer version of Bind9 server. Older version of Bind9 does not support ed25519 and ed448 keys.

The command produces the following files:

- Ksig0.example.edelivery.eu.local.+003+03054.key
- Ksig0.example.edelivery.eu.local.+003+03054.private

The content of the file is as follows:

```
Ksig0.example.edelivery.eu.+003+03054.key
```

It is the DNS Key entry, which should be put to DNS zone as in the example below:

```
sig0.example.edelivery.eu.local.      604800  IN      KEY     512 3 3
CLC4l6DtbztWAIJIMkYrv4MClWvj2BUclxqCd86vzX/f0ka+oS73dFCp
tb9Yv9oYjGmG1JLNv4EKuPiGPa8O/CQWrbJ5I7Yts3GDMgZNRswxMije
H6OoYkZ6ywRpjv8nommw6JMzDaDhcU5/tLQXhvz3U/c7W5QepAXfHb6Z
gGwL4TkqR/RGp5xcxayID4b/+DJvqi04BjNO9WR3XGRHWZ5aO0pRcRjx
imDtlnIjpsykE59o03UyQ+YT1CYNPjNlmOoT1JVgBEFGgouAm7yEZq3A
HWsqZEHCeucvQKBADmIk5rHwfZJwv7dzXrZR2U5AqE/AxqhrWyTpItRg
oGEkc+piGciuPRtwRZPkD6+GcFn/2knJ3YuRBOiog0+5mtbqaIPOew+B
+BtQk6X5E5tNnEuQJeRjjxznGYdzN7hTDFPvtwGEQvDUoU4SP/6YHoAd
AaH5Vs+YTRHjdISvnJIV6VRxIbQFJWaf3Z+UT4ns0+4pIGXm7C0ADA2a
1wGpj4QF8A37VAofcFWlUErtNv9YmVHQcA2l
```

When the public key is correctly registered to the DNS server, it can be tested with dig util as in the example below:

```
$dig sig0.example.edelivery.eu.local  @localhost KEY


ANSWER
; <<>> DiG 9.10.3-P4-Ubuntu <<>> sig0.example.edelivery.eu.local @localhost
KEY
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 36443
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
; sig0.example.edelivery.eu.local.             IN      KEY

;; ANSWER SECTION:
sig0.example.edelivery.eu.local.      604800  IN      KEY     512 3 3
CLC4l6DtbztWAIJIMkYrv4MClWvj2BUclxqCd86vzX/f0ka+oS73dFCp
```

```
tb9Yv9oYjGmG1JLNv4EKuPiGPa8O/CQWrbJ5I7Yts3GDMgZNRswxMije
H6OoYkZ6ywRpjv8nommw6JMzDaDhcU5/tLQXhvz3U/c7W5QepAXfHb6Z
gGwL4TkqR/RGp5xcxayID4b/+DJvqi04BjNO9WR3XGRHWZ5aO0pRcRjx
imDtlnIjpsykE59o03UyQ+YT1CYNPjNlmOoT1JVgBEFGgouAm7yEZq3A
HWsqZEHCeucvQKBADmIk5rHwfZJwv7dzXrZR2U5AqE/AxqhrWyTpItRg
oGEkc+piGciuPRtwRZPkD6+GcFn/2knJ3YuRBOiog0+5mtbqaIPOew+B
+BtQk6X5E5tNnEuQJeRjjxznGYdzN7hTDFPvtwGEQvDUoU4SP/6YHoAd
AaH5Vs+YTRHjdISvnJIV6VRxIbQFJWaf3Z+UT4ns0+4pIGXm7C0ADA2a
1wGpj4QF8A37VAofcFWlUErtNv9YmVHQcA2l

;; AUTHORITY SECTION:
example.edelivery.eu.local.            604800  IN      NS      ns.
example.edelivery.eu.local.

;; ADDITIONAL SECTION:
ns.example.com.local.         604800  IN      A       192.168.56.3
```

If we want to allow DNS updates on the zone “example.edelivery.eu.local " only by requests signed by private key of the **sig0.example.edelivery.eu.local ,** we have to update the DNS zone configuration as below:

```
zone "example.edelivery.eu.local" {
    type master;
    file "/var/lib/bind/db.example.edelivery.eu.local ";
    allow-update { key "sig0.example.edelivery.eu.local.";}
    allow-transfer { 10.22.0.0/16; };
};
```

5.8.2. Configuration of the SIG(0) in DomiSML

To configure DomiSML to use SIG(0), the following parameters must be set:

- **dnsClient.SIG0PublicKeyName:** must be DNS name of the DNS KEY entry. In the example above this value is: **dnsClient.SIG0PublicKeyName= sig0.example.edelivery.eu.local**
- **dnsClient.SIG0KeyFileName:** the private key must be put into to the BDMS configuration folder and Value of the parameter **dnsClient.SIG0KeyFileName** must be the name of the SIG(0) private key filename. As example: **dnsClient.SIG0KeyFileName= Ksig0.example.edelivery.eu.local.+003+03054.private**
- dnsClient.SIG0Enabled: to enable SIG(0) the configuration parameter must be set to true: **dnsClient.SIG0Enabled=true**

Note: The DomiSML does not use the SIG(0) to transfer the DNS records. The DNS records are retrieved from DBS server when generating inconsistency reports and when calling the resource web /listDNS. Above is an example of how to secure a transfer to a network: 10.22.0.0/16.

6. Contact information

eDelivery Support Team

By email: EC-EDELIVERY-SUPPORT@ec.europa.eu

Support Service: 8am to 6pm (Normal EC working Days)