EUROPEAN COMMISSION

DIGIT
Connecting Europe Facility

Access Point

Administration Guide

Domibus 4.0.1

Version [2.9]

Status [Final]

Date: 10-01-2019

Document Approver(s):

Approver Name	Role
Adrien FERIAL	CEF Project Manager

Document Reviewers:

Reviewer Name	Role
Cosmin BACIU	CEF Technical Office
Catalin-Emanuel ENACHE	CEF Technical Office
Ioana DRAGUSANU	CEF Technical Officer

Summary of Changes:

Version	Date	Created by	Short Description of Changes
1.07	09/02/2018	Chaouki BERRAH Caroline AEBY	Update for version 3.3.2. MySQL and Oracle deletion scripts added + operational guidelines.
1.08	20/03/2018	Caroline AEBY	Reuse notice added, links to AS4 specifications updated.
1.09	04/04/2018	Chaouki BERRAH Caroline AEBY	Domibus 3.3.3
1.2	16/04/2018	Caroline AEBY	Domibus 3.3.4 PMode configuration moved to other section. Domibus properties: dynamic.discovery => dynamicdiscovery. 2 new properties added: domibus.dynamicdiscovery.partyid.responder.role & domibus.dynamicdiscovery.partyid.type
1.3	21/06/2018	Chaouki BERRAH	Domibus 3.3.4 Updates + binary files sources references updated
1.4	01/07/2018	Caroline AEBY	Updates for Domibus 4.0. Section on Plugin notification added.
1.5	26/07/2018	Chaouki BERRAH Ioana DRAGUSANU Cosmin BACIU Tiago MIGUEL Thomas DUSSART	Update PMODE Multitenancy Test Service and domibus.properties Alerts Section added
1.6	31/07/2018	Cosmin BACIU Caroline AEBY Thomas DUSSART	Updates
1.7	01/08/2018	Catalin ENACHE	Information on WildFly 12 added
1.8	16/08/2018	Cosmin BACIU Caroline AEBY	Additional information on Multitenancy
1.9	23/08/2018	AEBY Caroline Catalin ENACHE	Admin console 4.0 features Databse configuration updates
1.9.1	30/08/2018	Joze RIHTARSIC Caroline AEBY	domibus.pmode.dao.implementation removed from Domibus properties.
1.9.2	05/09/2018	Catalin ENACHE	Updates for 4.0 FR
1.9.3	07/09/2018	Cosmin BACIU Caroline AEBY	domibus.datasource.maxLifetime=30 Other changes
1.9.4	10/09/2018	Chaouki BERRAH	Oracle Script Change. Multitenancy Additonal Information including

			domain_name-domibus.properties example and screen shots.
1.9.5	13/09/2018	Catalin-Emanuel ENACHE Chaouki BERRAH	Update of Wildlfy configuration for Oracle
1.9.6	17/09/2018	Chaouki BERRAH Joze RIHTARSIC Ioana DRAGUSANU	Updates: Set XA Transaction Timeout ..messageIdPattern
1.9.7	17/09/2018	Caroline AEBY	Review
1.9.8	19/09/2018	Catalin-Emanuel ENACHE Chaouki Berrah	UIReplication Chapter added
1.9.9	20/09/2018	Cosmin BACIU	Corrected the retry timeout for the reception awareness configuration; removed the XA Transaction Timeout as it automatically set by the WSLT script
2.0	20/09/2018	Chaouki BERRAH	Alerts configuration example and screenshots
2.1	21/09/2018	Chaouki BERRAH	Weblogic Configuration changes
2.2	26/09/2018	Caroline AEBY	Contact information update
2.3	26/09/2018	Chaouki BERRAH	Prerequisites changes, 21.6 Troubleshooting section
2.4	05/10/2018	Chaouki BERRAH	Weblogic Server Multitenancy details added
2.5	05/10/2018	Caroline AEBY	Troubleshooting article section removed
2.6	08/10/2018	Thomas Dussart	Add missing alert properties
2.7	16/10/2018	Caroline AEBY	Add missing command lines in 16.1.1.2 Oracle
2.7.1	16/10/2018	Thomas Dussart Catalin Enache	Updates for 4.0.1 release Add super user alert information. Updated Multitenancy logging information.
2.7.2	05/11/2018	Catalin Enache	Added information on logging info per domain
2.7.3	21/11/2018	Caroline AEBY	Added missing line for Pull mode in Pmode
2.7.4	22/11/2018	Chaouki BERRAH	domain_name-domibus.dynamicdiscovery.transportprofileas4 Added
2.7.5	29/11/2018	Catalin-Emanuel ENACHE Chaouki BERRAH Caroiline AEBY	Weblogic Configuration Updates
2.7.6	05/12/2018	Maarten Daniels Chaouki BERRAH	Updates related to Dynamic Discovery in the PEPPOL Network
2.8	07/12/2018	Ioana DRAGUSANU	TLS configuration chapter updated
2.9	07/01/2019	Chaouki BERRAH	Link to WebLogic Script modified

Table of Contents

1. INTRODUCTION

This Administration Guide is intended for Server Administrators in charge of installing, managing and troubleshooting an eDelivery Access Point.

1.1. Purpose

The purpose of this guide is to provide detailed information on how to deploy and configure Domibus on WebLogic, Tomcat and WildFly with MySQL or Oracle. It also provides detailed descriptions of related Security Configurations (Policies, Certificates), Message Filtering, PMode Configuration, Application Monitoring, Custom Plugins Registration, JMS Monitoring, Data Archiving, Troubleshooting and TLS Configuration.

1.2. References

Ref.	Document	Content outline
[REF1]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/Domibus	Location of the release artefacts on the CEF Digital site
[REF2]	https://dev.mysql.com/downloads/connector/j/	Location to download the MySQL JDBC driver from the Official website
[REF3]	http://www.oracle.com/technetwork/database/features/jdbc/default-2280470.html	Location of the Oracle JDBC driver from the Official website
[REF4]	https://docs.jboss.org/author/display/WFLY9/WildFly+9+Cluster+Howto	Location to the Official documentation on how to setup a cluster on WildFly 9
[REF5]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/PKI+Service	CEF Public Key Infrastructure (PKI) Service Offering Document
[REF6]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/Domibus	Location of the latest Domibus release on the Single Web Portal

Ref.	Document	Content outline
[REF7]	https://access.redhat.com/documentation/en-US/Red_Hat_JBoss_Fuse/6.0/html/XML_Configuration_Reference/files/cxf-http-conf-2_7_0_xsd_Element_http-conf_tlsClientParameters.html	RedHat page for the XML Configuration Reference of the *http-conf:tlsClientParameters* element
[REF8]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eDelivery+SMP	SMP (Service Metadata Publisher) and Dynamic Discovery in AS4 Gateways
[REF9]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eDelivery+SMP	Space describing the SMP (Service Metadata Publisher)
[REF10]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/eDelivery+AS4	eDelivery AS4 Profile
[REF11]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/Domibus	Software Architecture Document (SAD)
[REF12]	https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/Domibus	JMS Plugin Interface Control Document (ICD)

2. CONVENTIONS

The commands and configuration files listed in this document usually contain a mix of reserved words (commands, instructions and system related special words) and user defined words (chosen by the user) as well as comments and preferred values for certain variables. The conventions used in this document, to distinguish between them, are the followings:

- To keep this document release agnostic as much as possible, the strings "x-y-z" or "x.y.z" are intended to refer to the version of Domibus discussed in this version of the document, in the present case "Domibus 4.0.1".
- **Bold** is used for "reserved" words and commands.
- *Normal italic* together with a short description of the argument is used for user-defined names (chosen by you to designate items like users, passwords, database etc.). Normally contains at least 2 words separated by "_". Could be highlighted at times.
- ***Bold and Italic*** is used for advisable values which can be changed by the user depending on their infrastructure. Could be highlighted at times.
- Comments are sometimes added to describe the purpose of the commands, usually enclosed in brackets ().

By default, non-OS specific paths will be described using Linux patterns.

2.1. Example 1: Sample Oracle Statement

create user *edelivery_user* **identified by** *edelivery_password*;

grant all privileges to *edelivery_user*;

(Where *edelivery_user* and *edelivery_password* are names chosen by the user)

2.2. Example 2: Sample Configuration file

jdbc.datasource.0.driver.name=com.mysql.jdbc.Driver

jdbc.datasource.0.driver.url=jdbc:mysql://*localhost:3306/* *domibus_schema*

jdbc.datasource.0.driver.password=*edelivery_password*

jdbc.datasource.0.driver.username=*edelivery_user*

(Where:

- *edelivery_user, domibus_schema* and *edelivery_password* are names chosen by the user.

- ***localhost:3306*** represents hostname:port parameters of the MySQL database.)

3. PREREQUISITES

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

- Java runtime environnement (JRE), version 8:

  http://www.oracle.com/technetwork/java/javase/downloads/index.html

- One of the supported Database Management Systems :
  - MySQL 5.6 or above
  - Oracle 10g+
- If you don't plan to deploy Domibus according to the Pre-Configured Single Server Deployment method, you must also install one of the supported application/web servers:
  - WebLogic 12c
  - WildFly 9 or WildFly 12
  - Apache Tomcat 8.0.x
- All Domibus installation resources, including full distributions and documentation can be found on the Single Web Portal :

  https://ec.europa.eu/cefdigital/wiki/display/CEFDIGITAL/Domibus

3.1. Binaries repository

All the artefacts can be directly downloaded from the CEF Digital site (cf.[REF1]).

4. DOMIBUS DEPLOYMENT

Remark:

*The variable **cef_edelivery_path** referring to the folder where the package is installed will be used later in this document.*

4.1. Database Configuration

For this step, you will have to use the following resources (see section §3.1–*"Binaries repository"* for the download location):

- **domibus-distribution-X.Y.Z-sql-scripts.zip**

4.1.1. MySQL configuration

1. Unzip **domibus-distribution-X.Y.Z-sql-scripts.zip** in *cef_edelivery_path***/sql-scripts.**
2. Open a command prompt and navigate to this directory: *cef_edelivery_path***/sql-scripts**.
3. Execute the following MySQL commands at the command prompt :

mysql -h *localhost* -u *root_user* **--password=***root_password* **-e "drop schema if exists** *domibus_schema***; create schema** *domibus_schema***; alter database** *domibus_schema* **charset=utf8 collate=utf8_bin; create user** *edelivery_user@localhost* **identified by '***edelivery_password***'; grant all on** *domibus_schema.** **to** *edelivery_user@localhost***;"**

The above script creates a schema (*domibus_schema*) and a user (*edelivery_user*) that have all the privileges on the schema**.**

mysql -h *localhost* -u *root_user* **--password=***root_password domibus_schema* **< mysql5innoDb-x.y.z.ddl**

The above script creates the required objects in *domibus_schema.*

mysql -h *localhost* -u *root_user* **--password=***root_password domibus_schema* **< mysql5innoDb-x.y.z-data.ddl**

The above script populates the tables with some predefined data in *domibus_schema*.

Please find below some information regarding the MySQL server configuration:

1. (Optional) Storing payload messages in a database with size over 30 MB.

   Domibus can temporarily store the messages in the database.

   Therefore, it is required to increase the maximum allowed size of packets. Update the default properties of **my.ini** (Windows) or **my.cnf** (Linux).

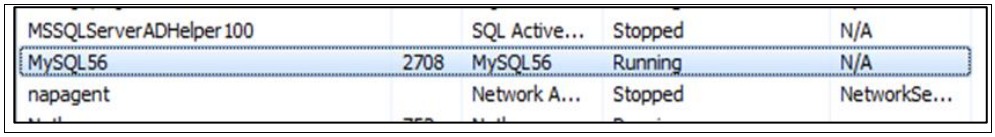
   - **max_allowed_packet** property

```
# The maximum size of one packet or any generated or intermediate string, or any
# parameter sent by the
# mysql_stmt_send_long_data() C API function.
max_allowed_packet=512M
```

- **innodb_log_file_size** property

```
# Size of each log file in a log group. You should set the combined size
# of log files to about 25%-100% of your buffer pool size to avoid
# unneeded buffer pool flush activity on log file overwrite. However,# note that larger logfile size will increase the time needed for the recovery process
innodb_log_file_size=5120M
```

- Restart MySQL service (Windows):



MySQL service

2. (Optional) For storing payload messages in a file system instead of a database see §5.2-Domibus Properties*).*

3. For MySQL 8 and Connector/J 8.0.x please set the database timezone. One way of setting the timezone is to modify the MySQL **my.ini** configuration file by adding the following property with the adjusted timezone. It is recommended that the database timezone is the same as the timezone of the machine where Domibus is installed.

```
default-time-zone='+00:00'
```

Remark:

If you are using Windows, make sure to have the parent directory of mysql.exe added to your PATH variable.

4.1.2. Oracle configuration

1. Unzip **domibus-distribution-X.Y.Z-sql-scripts.zip** in *cef_edelivery_path***/sql-scripts**

2. Open a command prompt and navigate to this directory: *cef_edelivery_path***/sql-scripts**.

3. Open a command line session, log in and execute the following commands :

```
sqlplus sys as sysdba  (password should be the one assigned during the Oracle installation )
==========================================================================
Once logged in Oracle:
CREATE USER <edelivery_user> IDENTIFIED BY <edelivery_password>
DEFAULT TABLESPACE <tablespace>
QUOTA UNLIMITED ON <tablespace>;
GRANT CREATE SESSION TO <edelivery_user>;
GRANT CREATE TABLE TO <edelivery_user>;
GRANT CREATE VIEW TO <edelivery_user>;
GRANT CREATE SEQUENCE TO <edelivery_user>;
GRANT EXECUTE ON DBMS_XA TO <edelivery_user>;
GRANT SELECT ON PENDING_TRANS$ TO <edelivery_user>;
GRANT SELECT ON DBA_2PC_PENDING TO <edelivery_user>;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO <edelivery_user>;


CONNECT <edelivery_user>
SHOW USER;  (should return : edelivery_user)
@oracle10g-x.y.z.ddl
@oracle10g-x.y.z-data.ddl

EXIT
==========================================================================
```

Remarks:

1. Replace <edelivery_user> and <edelivery_password> with corresponding values.

2. <tablespace> is created and assigned by your DBA; for local/test installations just replace it with users tablespace. The quota could be limited to a certain size.

3. DDL/SQL scripts must be run with the @ sign from the location of the scripts

4.1.3. MySQL and Oracle Deletion scripts

A deletion script for MySQL and Oracle Domibus DB is available in the **domibus-distribution-X.Y.Z-sql-scripts.zip**.

The purpose of the script is to delete all messages within a user defined period to recover disk space. The script requires a START_DATE parameter and an END_DATE parameter to be set.

The tables affected by the execution of this script are:

- TB_MESSAGING
- TB_ERROR_LOG
- TB_PARTY_ID
- TB_RECEIPT_DATA
- TB_PROPERTY
- TB_PART_INFO
- TB_RAWENVELOPE_LOG
- TB_ERROR
- TB_USER_MESSAGE
- TB_SIGNAL_MESSAGE
- TB_RECEIPT

- TB_MESSAGE_INFO
- TB_MESSAGE_LOG
- TB_MESSAGE_UI

Any information relevant to a message received or sent during the predefined period, will be removed from these tables.

In order to execute this script, it is advised to use a UI tool such as SQL developer or MySQL workbench.

Important: in order to keep the JMS queues synchronized with the DB data that will be deleted by this script, the Domibus Administrator should remove manually the associated JMS messages from the plugin notifications queues

4.2. Domibus on WebLogic 12.1.3

This section does not include the installation of WebLogic server 12.1.3. It is assumed that the WebLogic Server is installed and a Domain is created.
Hereafter the domain location will be referred as *DOMAIN_HOME (user defined name)*.

4.2.1. Single Server Deployment

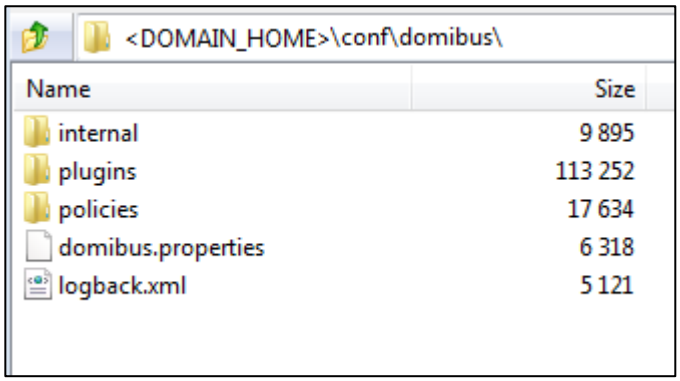
For this step, you will have to use the following resources (see section §3.1–*"Binaries repository "* for the download location):

- **domibus-distribution-X.Y.Z-weblogic-war.zip**
- **domibus-distribution-X.Y.Z-weblogic-configuration.zip**

At least one of the following plugins should be downloaded and installed:

- **domibus-distribution-X.Y.Z-default-ws-plugin.zip**
- **domibus-distribution-X.Y.Z-default-jms-plugin.zip**
- **domibus-distribution-X.Y.Z-default-fs-plugin.zip**

1. Download and unzip **domibus-distribution- X.Y.Z-weblogic-configuration.zip** in the directory *DOMAIN_HOME***/conf/domibus**



2. Download and unzip **domibus-distribution- X.Y.Z-weblogic-war.zip** in a temporary folder to prepare it for deployment.

3. Configure your Keystore based on section §5.1.2 – *"Certificates"*

4. Add the following lines in:

- For Windows : *DOMAIN_HOME***\bin\setDomainEnv.cmd**

  - Locate the **set DOMAIN_HOME** statement and add the following lines after:

```
…
set DOMAIN_HOME
# Added for Domibus
*****************************************************************
set EXTRA_JAVA_PROPERTIES=%EXTRA_JAVA_PROPERTIES% -
Ddomibus.config.location=%DOMAIN_HOME%/conf/domibus
#
*************************************************************************************
****
…
```

- For Linux : *DOMAIN_HOME***/bin/setDomainEnv.sh**

  - Locate the **export DOMAIN_HOME** statement and add the following lines after:

```
…
export DOMAIN_HOME
# Added for Domibus
*****************************************************************
EXTRA_JAVA_PROPERTIES="$EXTRA_JAVA_PROPERTIES -
Ddomibus.config.location=$DOMAIN_HOME/conf/domibus"
export EXTRA_JAVA_PROPERTIES
#
*************************************************************************************
****
…
```

5. Run the WebLogic Scripting Tool (WLST) in order to create the JMS resources and the Database datasources from the command line:

   - Download the WLST Package from following location: https://ec.europa.eu/cefdigital/artifact/content/repositories/public/eu/europa/ec/digit/ipcis/wslt-api/1.9.1/wslt-api-1.9.1.zip

   - Configure the WSLT API tool

     - Unzip the **wslt-api-1.9.1.zip**

     - Define the **WL_HOME** as a system environment variable to point to the WebLogic 'wlserver' directory as defined in the **DOMAIN_HOME/bin/setDomainEnv.[cmd|sh]**

       e.g. WL_HOME=/wls12130/wlserver

   - Take the script **WeblogicSingleServer.properties** from **domibus-distribution-X.Y.Z-weblogic-configuration.zip** under the scripts directory and copy the **WeblogicSingleServer.properties** file into the **wslt-api-1.9.1** directory and adapt the following properties :

     - Adapt the properties for connecting to the WebLogic domain:

```
domain.loading.type=connect
domain.connect.url=t3://localhost:7001
domain.connect.username=weblogic_name
domain.connect.password=weblogic_password
domain.name=my_domain1
```

- Adapt the jdbc.datasource properties for the datasources:
    - Common to Oracle and MySQL

```
################################################################
## Domain configuration
################################################################
## Variables
##---------Cross module--------
#Domibus application module target
application.module.target = AdminServer

##---------JMS configuration--------
#Domibus JMS application server name
jms.server.name = eDeliveryJMS
#Domibus JMS application module name
jms.module.name = eDeliveryModule
#Domibus JMS file store name
jms.server.store = eDeliveryFileStore
#Domibus JMS application module group
jms.queue.subdeployment.name = eDeliverySubD

##---------Database configuration--------
#Domibus database url
jdbc.datasource.driver.url=jdbc:oracle:thin:@127.0.0.1:1521:xe
#Domibus database user name
jdbc.datasource.driver.username=your_username
#Domibus database user password
jdbc.datasource.driver.password=your_password
```

    - For Oracle database:

```
jdbc.datasource.0.name=eDeliveryDs
jdbc.datasource.0.targets=${application.module.target}
jdbc.datasource.0.jndi.name=jdbc/cipaeDeliveryDs
jdbc.datasource.0.pool.capacity.max=50
jdbc.datasource.0.pool.connection.test.onreserv.enable=true
jdbc.datasource.0.pool.connection.test.onreserv.sql=SQL SELECT 1 FROM DUAL
jdbc.datasource.0.driver.name=oracle.jdbc.xa.client.OracleXADataSource
jdbc.datasource.0.driver.url=${jdbc.datasource.driver.url}
jdbc.datasource.0.driver.password=${jdbc.datasource.driver.password}
jdbc.datasource.0.driver.username=${jdbc.datasource.driver.username}
jdbc.datasource.0.driver.properties.items=0
jdbc.datasource.0.xa.transaction.timeout.branch=true

jdbc.datasource.1.name=edeliveryNonXA
```

```
jdbc.datasource.1.targets=${application.module.target}
jdbc.datasource.1.jndi.name=jdbc/cipaeDeliveryNonXADs
jdbc.datasource.1.transaction.protocol=None
jdbc.datasource.1.pool.capacity.max=50
jdbc.datasource.1.pool.connection.test.onreserv.enable=true
jdbc.datasource.1.pool.connection.test.onreserv.sql=SQL SELECT 1 FROM DUAL
jdbc.datasource.1.driver.name=oracle.jdbc.OracleDriver
jdbc.datasource.1.driver.url=${jdbc.datasource.driver.url}
jdbc.datasource.1.driver.password=${jdbc.datasource.driver.password}
jdbc.datasource.1.driver.username=${jdbc.datasource.driver.username}
jdbc.datasource.1.driver.properties.items=0
```

Remark:

MySQL configuration is commented by default. To enable MySQL, remove the comment (#) from the lines below. Don't forget to add the comment (#) for Oracle to disable it.

- For MySQL:

```
#jdbc.datasource.0.name=eDeliveryDs
#jdbc.datasource.0.targets=${application.module.target}
#jdbc.datasource.0.jndi.name=jdbc/cipaeDeliveryDs
#jdbc.datasource.0.transaction.protocol=LoggingLastResource
#jdbc.datasource.0.pool.connection.test.onreserv.enable=true
#jdbc.datasource.0.pool.connection.test.onreserv.sql=SQL SELECT 1
#jdbc.datasource.0.driver.name=com.mysql.jdbc.Driver
#jdbc.datasource.0.driver.url=${jdbc.datasource.driver.url}
#jdbc.datasource.0.driver.password=${jdbc.datasource.driver.password}
#jdbc.datasource.0.driver.username=${jdbc.datasource.driver.username}
#jdbc.datasource.0.driver.properties.items=0

#jdbc.datasource.1.name=edeliveryNonXA
#jdbc.datasource.1.targets=${application.module.target}
#jdbc.datasource.1.jndi.name=jdbc/cipaeDeliveryNonXADs
#jdbc.datasource.1.transaction.protocol=None
#jdbc.datasource.1.pool.capacity.max=50
#jdbc.datasource.1.pool.connection.test.onreserv.enable=true
#jdbc.datasource.1.pool.connection.test.onreserv.sql=SQL SELECT 1
#jdbc.datasource.1.driver.name=com.mysql.jdbc.Driver
#jdbc.datasource.1.driver.url=${jdbc.datasource.driver.url}
#jdbc.datasource.1.driver.password=${jdbc.datasource.driver.password}
#jdbc.datasource.1.driver.username=${jdbc.datasource.driver.username}
#jdbc.datasource.1.driver.properties.items=0
```

- Adapt the property for location of the filestore **persistent.filestore.0.location**.

  Example:

```
persistent.filestore.0.location=DOMAIN_HOME/filestore
```

Remark:

Make sure that the path for the filestore contains forward slashes (/).

- Adapt if necessary the JMX security configuration:

Example:

```
#################################################################
## Policy configuration
#################################################################
security.policies.0.mode = CREATE
security.policies.0.resource = type=<jmx>, operation=invoke, application=,
mbeanType=weblogic.management.runtime.JMSDestinationRuntimeMBean
security.policies.0.realm = myrealm
security.policies.0.authorizer = XACMLAuthorizer
security.policies.0.expression= Rol(Admin)|Grp(Administrators)|Grp(JMSManagers)
security.policies.items = 1
#################################################################
## Users configuration
#################################################################
security.users.0.realm=myrealm
security.users.0.name=jmsManager
security.users.0.password=jms_Manager1
security.users.0.comment=
security.users.0.authenticator=DefaultAuthenticator
security.users.items=1
#################################################################
## Groups configuration
#################################################################
security.groups.0.realm=myrealm
security.groups.0.name=JMSManagers
security.groups.0.description=
security.groups.0.authenticator=DefaultAuthenticator
security.groups.items=1
#################################################################
## Groups Membership configuration
#################################################################
security.group.member.0.user=jmsManager
security.group.member.0.groups=JMSManagers
security.group.member.0.realm=myrealm
security.group.member.0.authenticator=DefaultAuthenticator
security.group.member.items=1
```

- Start the WebLogic domain from within *DOMAIN_HOME:*
  - For Windows:

    startWebLogic.cmd
  - For Linux:

    startWebLogic.sh
- Execute the following command from within the **wlstapi-1.9.1/bin** directory:
  - For Windows:

    wlstapi.cmd ..\scripts\import.py --property ..\WeblogicSingleServer.properties
  - For Linux:

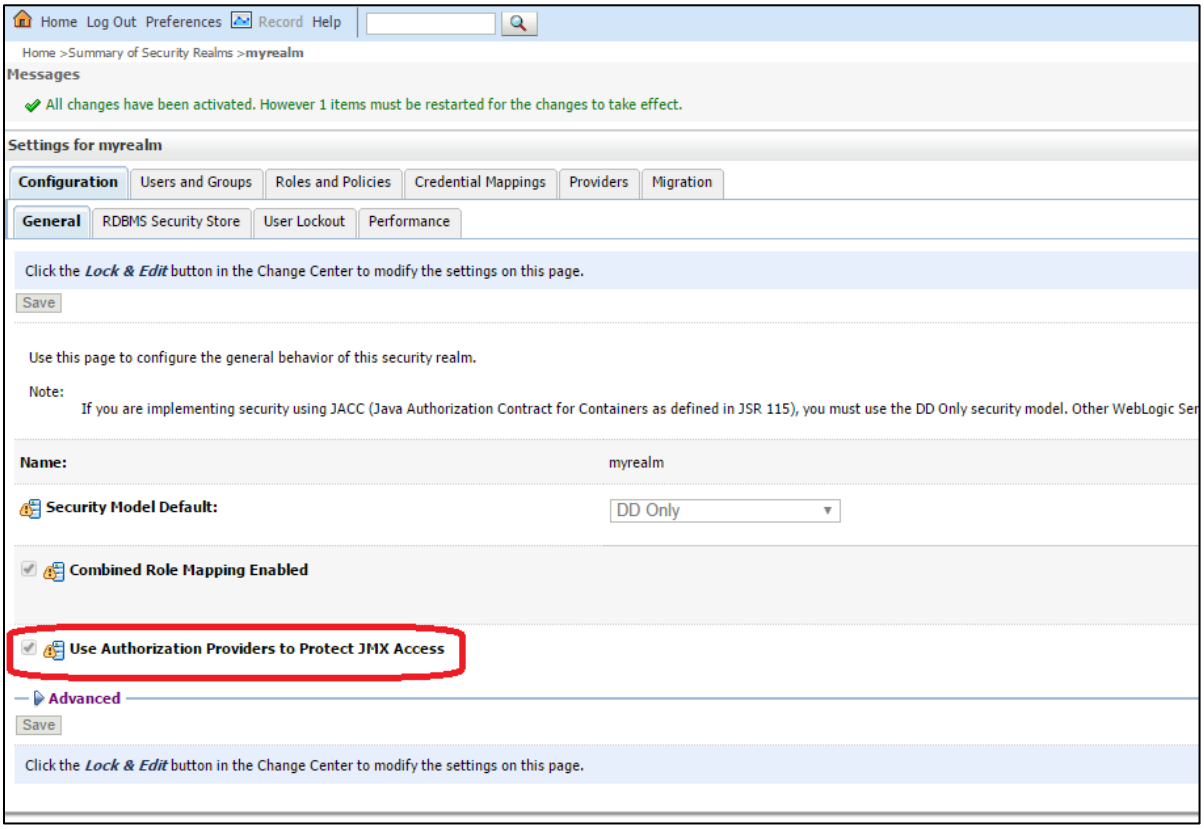
    wlstapi.sh ../scripts/import.py --property ../WeblogicSingleServer.properties

Expected result:

```
Saving all your changes ...
Saved all your changes successfully.
Activating all your changes, this may take a while ...
The edit lock associated with this edit session is released
once the activation is completed.
Activation completed
Location changed to serverRuntime tree. This is a read-only tree with DomainMBean as the root.
For more help, use help('domainConfig')

Disconnected from weblogic server: AdminServer
```

6. Activate the use of the authorization providers to protect the JMX access:



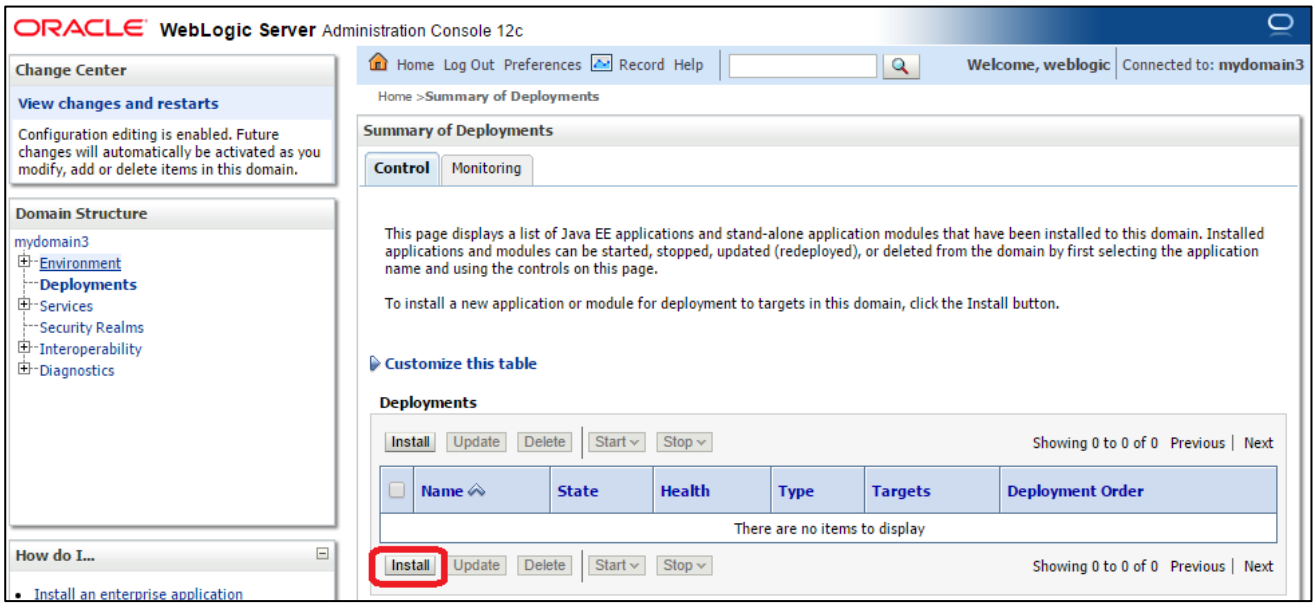
7. The database dialect is pre-configured to use the Oracle database. If you are using a MySQL database, you should adapt the following properties in <*DOMAIN_HOME*>**/conf/domibus/domibus.properties** as highlighted in the example below:

```
# ------------------------------ EntityManagerFactory ----------------------
domibus.entityManagerFactory.jpaProperty.hibernate.connection.driver_class=
com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
domibus.entityManagerFactory.jpaProperty.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDB
Dialect
```

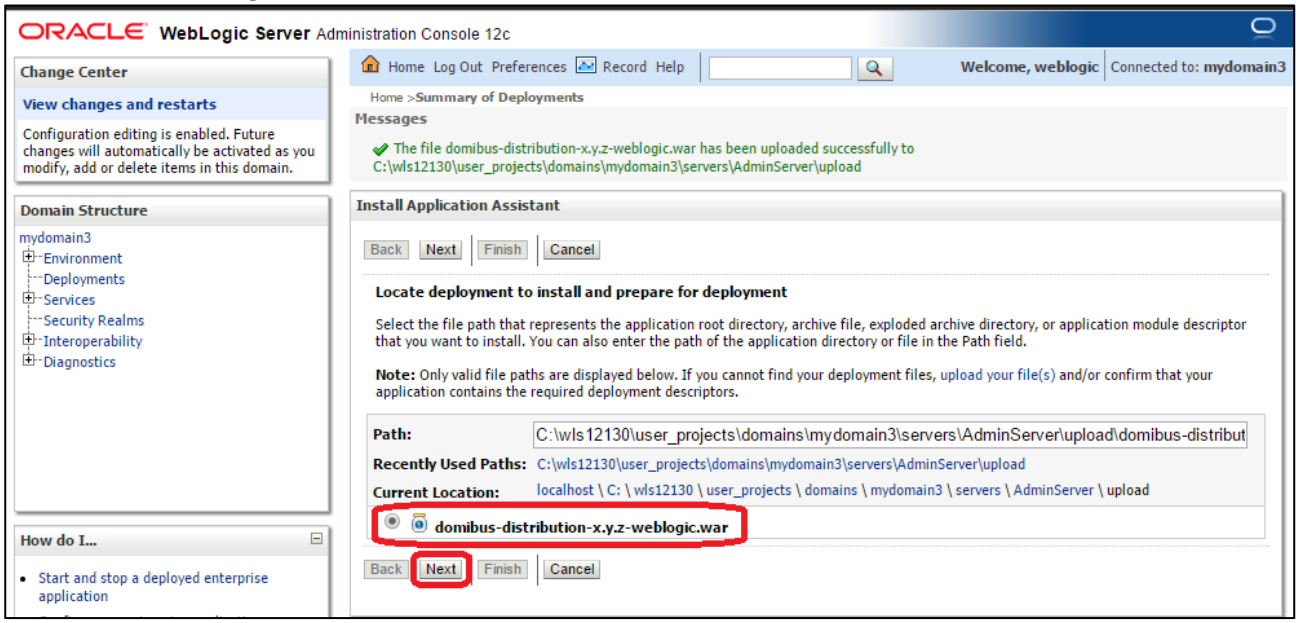
8. Install the WS Plugin. For more details, see section §6.2.1.2 – *"WebLogic"*.

9. Deploy **domibus-distribution-X.Y.Z-weblogic.war**

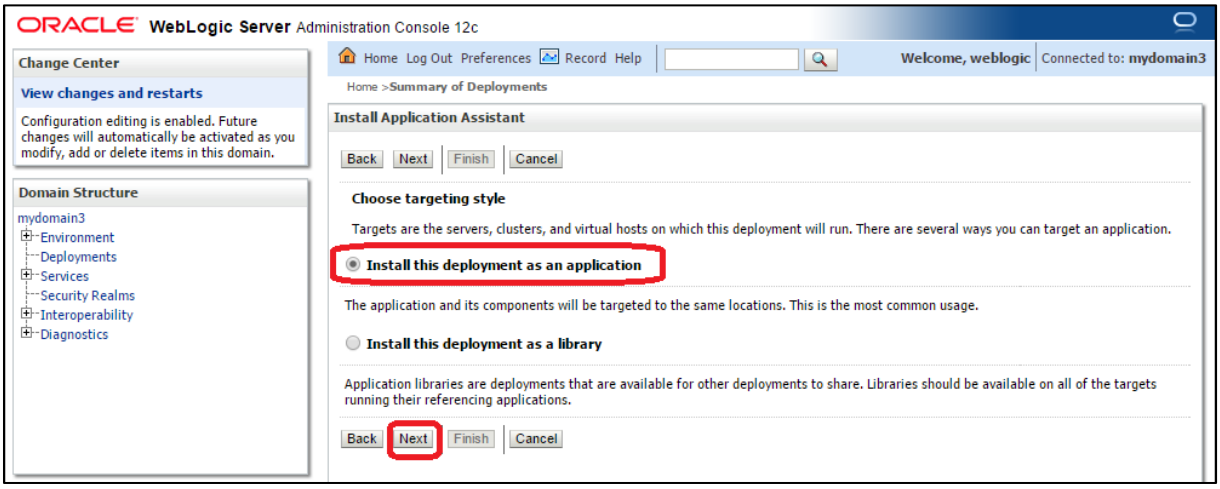
   - Click on **Install**:



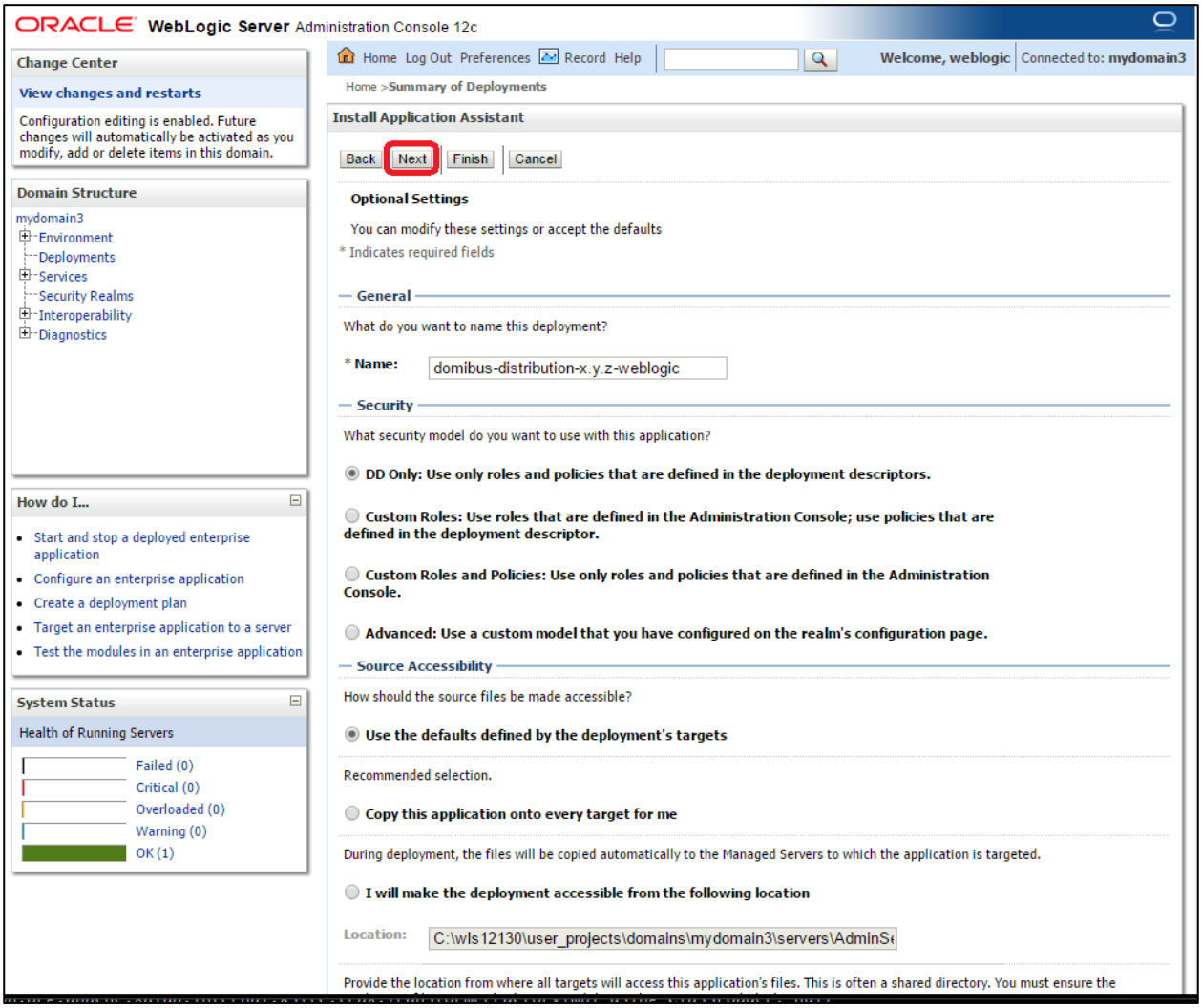
- Navigate to the location of the **.war** file and click **Next**:



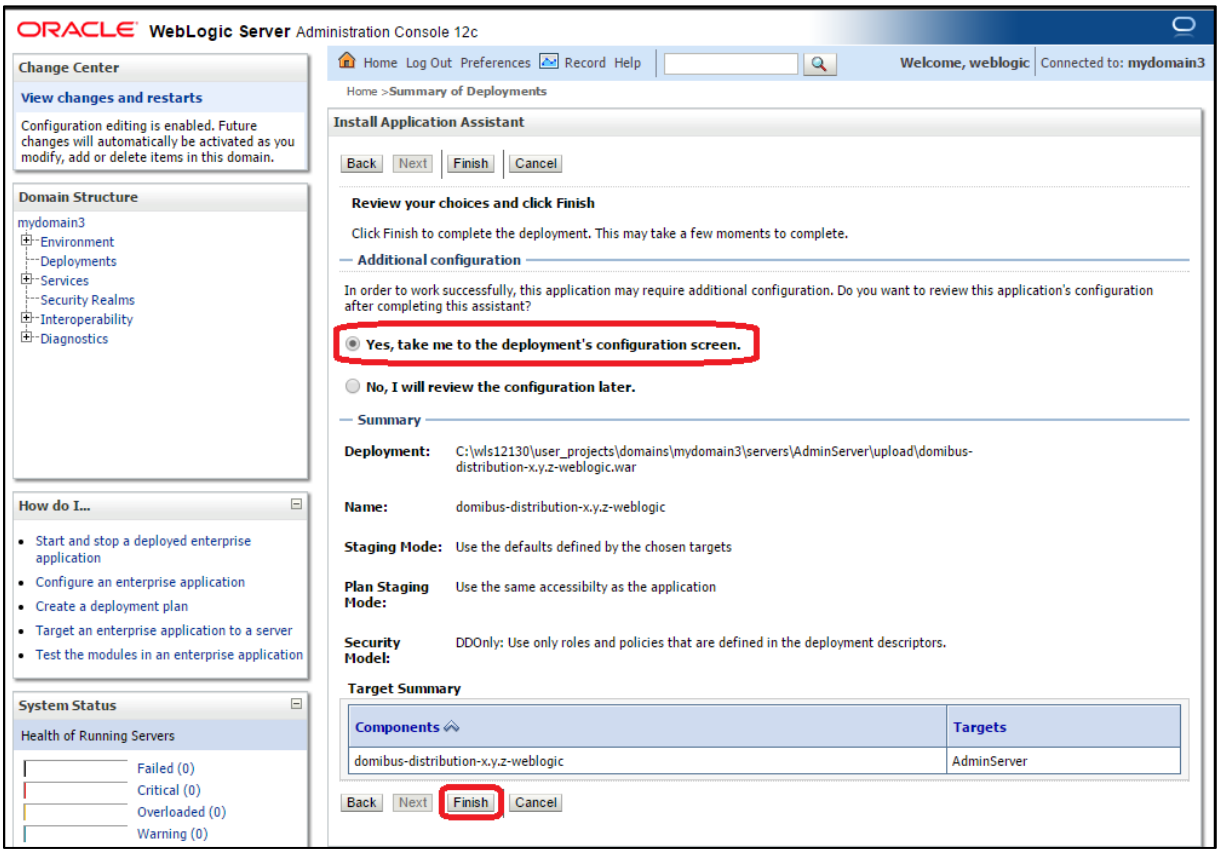
- Choose **Install this deployment as an application** and click **Next**:



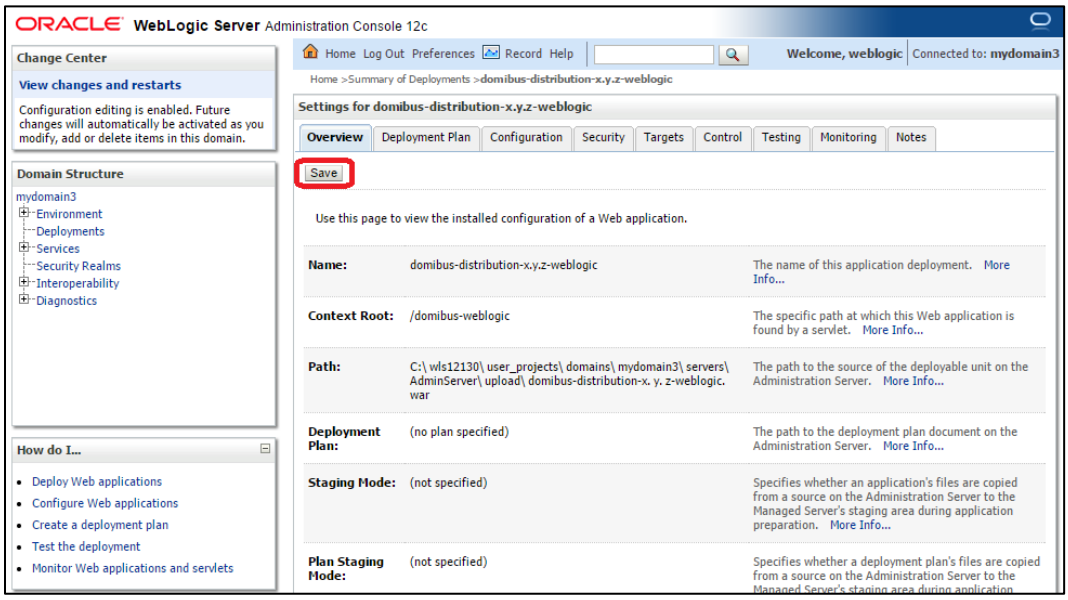
- Accept the default options and click **Next**:



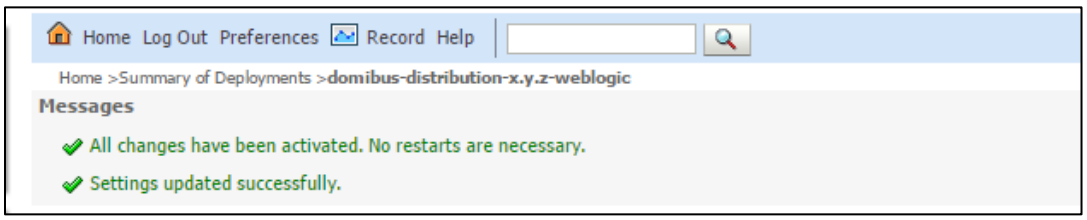
- Select the following option and click **Finish**:



- Here is an overview of the resulting settings, you can now click on the **Save** button:



The expected positive response to the deployment request should be the following:



10. Verify the installation by navigating with your browser to http://*localhost*:*7001*/domibus-weblogic: if you can access the page it means the deployment was successful.

    (By default: User = ***admin***; Password = ***123456)***

    Remark*:*

    It is recommended to change the passwords for the default users (See §9.1 – Administration for further information).

Expected result:

4.2.2. Clustered Deployment

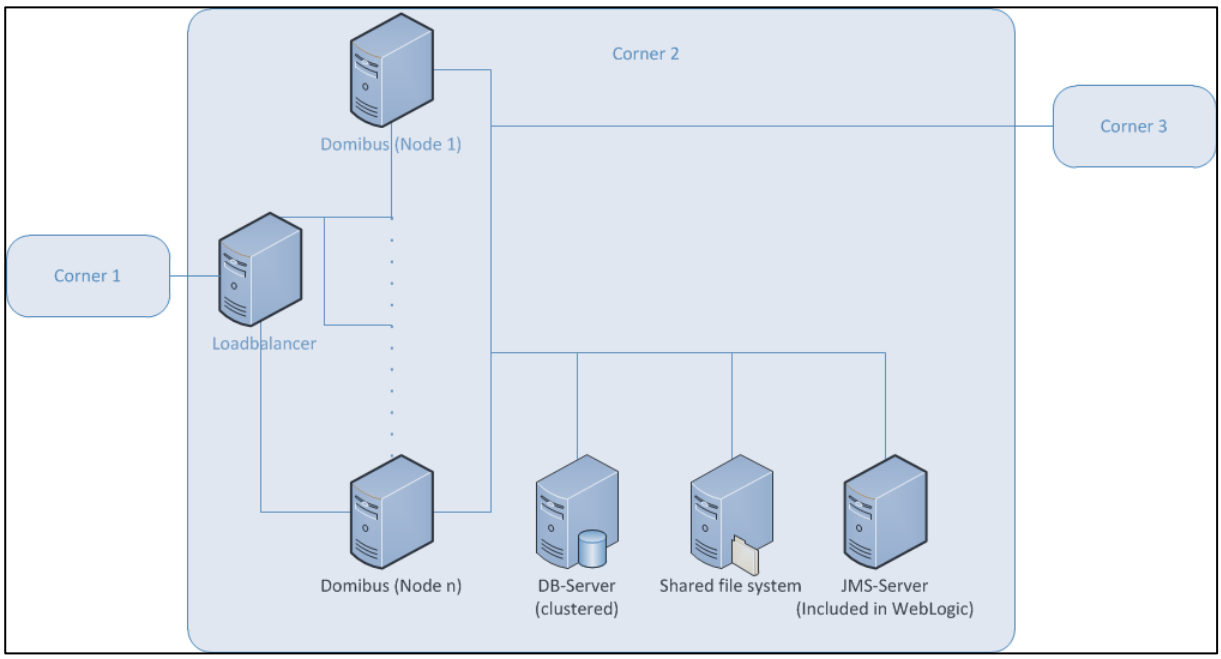


Figure 1 - Diagram representing the Deployment of Domibus in a Cluster on WebLogic

Remark:

In this section, we assume that a Domain and a WebLogic Cluster are already setup.

For this step, you will have to use the following resources (see section §3.1–*"Binaries repository"* for the download location):

- **domibus-distribution-X.Y.Z-weblogic-war.zip**
- **domibus-distribution-X.Y.Z-weblogic-configuration.zip**

At least one of the following plugins should be downloaded and installed:

- **domibus-distribution-X.Y.Z-default-ws-plugin.zip**
- **domibus-distribution-X.Y.Z-default-jms-plugin.zip**
- **domibus-distribution-X.Y.Z-default-fs-plugin.zip**

1. Download and unzip **domibus-distribution- X.Y.Z-weblogic-configuration.zip** in in a shared location that is accessible by all the nodes from the cluster. We will refer to this directory as *cef_shared_edelivery_path***/Domibus**.

2. Download and unzip **domibus-distribution- X.Y.Z-weblogic-war.zip** in a temporary folder to prepare it for deployment.

3. Configure your Keystore based on section §5.1.2 – *"Certificates"*.

4. Add the following lines in:

   - For Windows : *DOMAIN_HOME***\bin\setDomainEnv.cmd**

     - Locate the **set DOMAIN_HOME** statement and add the following lines after:

```
…
set DOMAIN_HOME
# Added for Domibus
*****************************************************************
set EXTRA_JAVA_PROPERTIES=%EXTRA_JAVA_PROPERTIES% -
Ddomibus.config.location=%DOMAIN_HOME%/conf/domibus
#
*********************************************************************************
****
…
```

   - For Linux : *DOMAIN_HOME***/bin/setDomainEnv.sh**

     - Locate the **export DOMAIN_HOME** statement and add the following lines after:

```
…
export DOMAIN_HOME
# Added for Domibus
*****************************************************************
EXTRA_JAVA_PROPERTIES="$EXTRA_JAVA_PROPERTIES -
Ddomibus.config.location=$DOMAIN_HOME/conf/domibus"
export EXTRA_JAVA_PROPERTIES
#
*********************************************************************************
****
…
```

5. Run the WebLogic Scripting Tool (WLST) in order to create the necessary JMS resources and Database datasources from the command line:

- Download the WLST Package from the following location: https://ec.europa.eu/cefdigital/artifact/content/repositories/eDelivery/eu/europa/ec/digit/ipcis/wslt-api/1.9.1/wslt-api-1.9.1.zip

- Configure the WSLT API tool:

  - Unzip the **wslt-api-1.9.1.zip**

  - Define the WL_HOME (SET or export command depending on your operating system) environment variable to point to the WebLogic **wlserver** directory

    e.g. WL_HOME=/wls12130/wlserver

- Take the script **WeblogicCluster.properties** from **domibus-distribution-X.Y.Z-weblogic-configuration.zip** under the scripts directory and copy the **WeblogicCluster.properties** file into the **wslt-api-1.9.1** directory and apply the following changes :

  - Adapt the properties for connecting to the WebLogic domain

```
domain.loading.type=connect
domain.connect.url=t3://localhost:7001
domain.connect.username=weblogic_user
domain.connect.password=weblogic_password
domain.name=mydomain1
```

  - Adapt the jdbc.datasource properties for the datasources

    For Oracle database:

```
jdbc.datasource.0.name= eDeliveryDs
jdbc.datasource.0.driver.name=oracle.jdbc.xa.client.OracleXADataSource
jdbc.datasource.0.driver.url=jdbc:oracle:thin:@127.0.0.1:1521:xe
jdbc.datasource.0.driver.password=edelivery_password
jdbc.datasource.0.driver.username=edelivery_username
jdbc.datasource.0.targets=cluster_name

jdbc.datasource.1.name=edeliveryNonXA
jdbc.datasource.1.driver.name= oracle.jdbc.OracleDriver
jdbc.datasource.1.driver.url=jdbc:oracle:thin:@127.0.0.1:1521:xe
jdbc.datasource.1.driver.password=edelivery_password
jdbc.datasource.1.driver.username=edelivery_username
jdbc.datasource.1.targets=cluster_name
```

    For MySQL database:

    Remark:

    MySQL configuration is commented by default. To enable MySQL, remove the comment (#) from the lines below. Don't forget to add the comment (#) for Oracle to disable it.

```
jdbc.datasource.0.name= eDeliveryDs
jdbc.datasource.0.driver.name=com.mysql.jdbc.Driver
# Connector/J 8.0.x
#jdbc.datasource.0.driver.name= com.mysql.cj.jdbc.Driver
jdbc.datasource.0.driver.url=jdbc:mysql://localhost:3306/domibus_schema
```

```
jdbc.datasource.0.driver.password=edelivery_password
jdbc.datasource.0.driver.username=edelivery_username
jdbc.datasource.0.targets=cluster_name
jdbc.datasource.0.transaction.protocol=LoggingLastResource
jdbc.datasource.0.pool.connection.test.onreserv.sql=SQL SELECT 1

jdbc.datasource.1.name= edeliveryNonXA
jdbc.datasource.1.driver.name=com.mysql.jdbc.Driver
# Connector/J 8.0.x
#jdbc.datasource.1.driver.name= com.mysql.cj.jdbc.Driver
jdbc.datasource.1.driver.url=jdbc:mysql://localhost:3306/domibus_schema
jdbc.datasource.1.driver.password=edelivery_password
jdbc.datasource.1.driver.username=edelivery_username
jdbc.datasource.1.targets=cluster_name
jdbc.datasource.1.transaction.protocol=None
jdbc.datasource.1.pool.connection.test.onreserv.sql=SQL SELECT 1
```

- Adapt the properties for target and location of the filestore:

```
persistent.filestore.0.target=cluster_name
persistent.filestore.0.location=DOMAIN_HOME/filestores
```

***Remark**:*

If you are using Windows, make sure that the paths for the filestore contain forward slash (/).

- Adapt if necessary the JMX security configuration:

Example:

```
#####################################################################
## Policy configuration
#####################################################################
security.policies.0.mode = CREATE
security.policies.0.resource = type=<jmx>, operation=invoke, application=,
mbeanType=weblogic.management.runtime.JMSDestinationRuntimeMBean
security.policies.0.realm = myrealm
security.policies.0.authorizer = XACMLAuthorizer
security.policies.0.expression= Rol(Admin)|Grp(Administrators)|Grp(JMSManagers)
security.policies.items = 1
#####################################################################
## Users configuration
#####################################################################
security.users.0.realm=myrealm
security.users.0.name=jmsManager
security.users.0.password=jms_Manager1
security.users.0.comment=
security.users.0.authenticator=DefaultAuthenticator
security.users.items=1
#####################################################################
## Groups configuration
#####################################################################
security.groups.0.realm=myrealm
security.groups.0.name=JMSManagers
```

```
security.groups.0.description=
security.groups.0.authenticator=DefaultAuthenticator
security.groups.items=1
#################################################################
## Groups Membership configuration
#################################################################
security.group.member.0.user=jmsManager
security.group.member.0.groups=JMSManagers
security.group.member.0.realm=myrealm
security.group.member.0.authenticator=DefaultAuthenticator
security.group.member.items=1
```

- Adapt the property for JMS Server:

Example:

```
jms.server.0.target=cluster_name
```

- Adapt the property for JMS Module:

Example:

```
jms.module.0.targets=cluster_name
```

- Start the WebLogic domain from within *DOMAIN_HOME:*
  - For Windows:

```
startWebLogic.cmd
```

  - For Linux:

```
startWebLogic.sh
```

- Execute the following command from within the **wlstapi-1.9.1/bin** directory:

For Windows:

```
wlstapi.cmd ..\scripts\import.py --
property ..\WeblogicCluster.properties
```

For Linux:

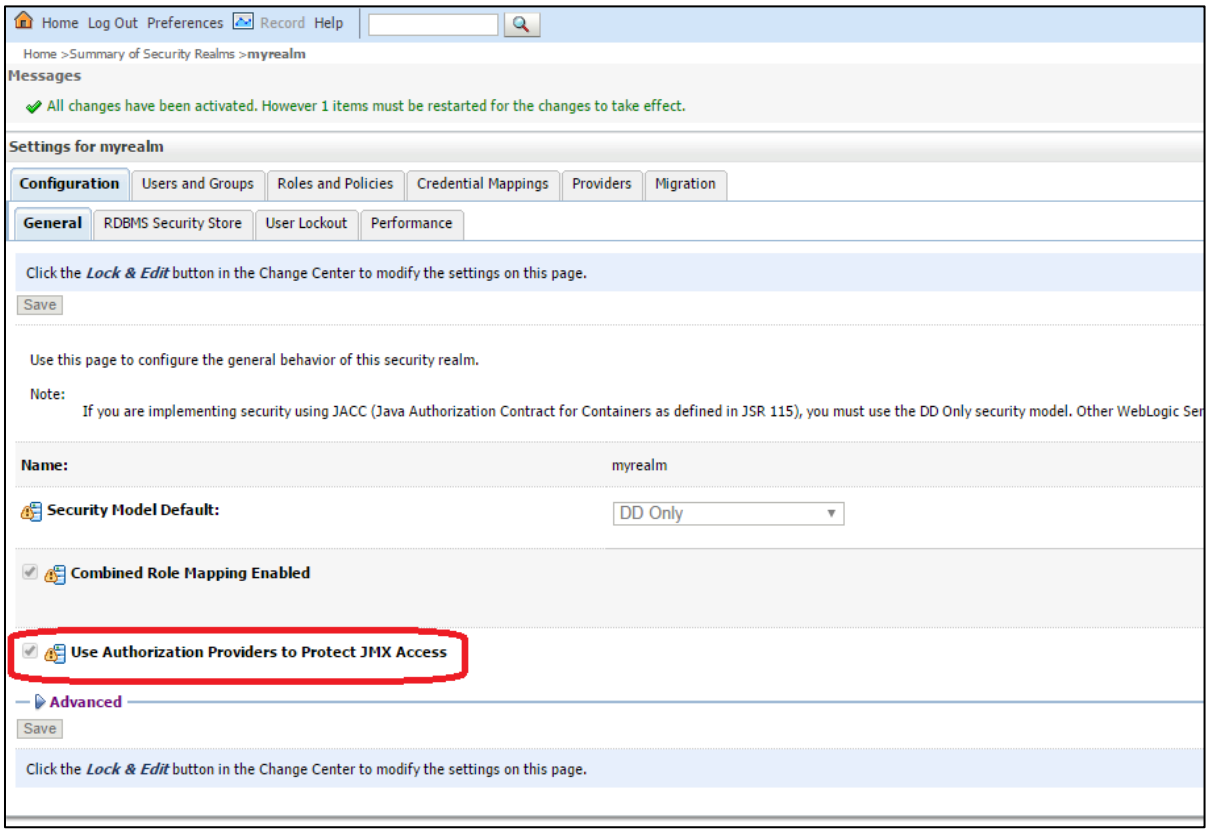
```
wlstapi.sh ../scripts/import.py --property ../WeblogicCluster.properties
```

Expected result:

```
Saving all your changes ...
Saved all your changes successfully.
Activating all your changes, this may take a while ...
The edit lock associated with this edit session is released
once the activation is completed.
Activation completed
Location changed to serverRuntime tree. This is a read-only tree with DomainMBean as the root.
For more help, use help('domainConfig')

Disconnected from weblogic server: AdminServer
```

6. Activate the use of the authorization providers to protect the JMX access:



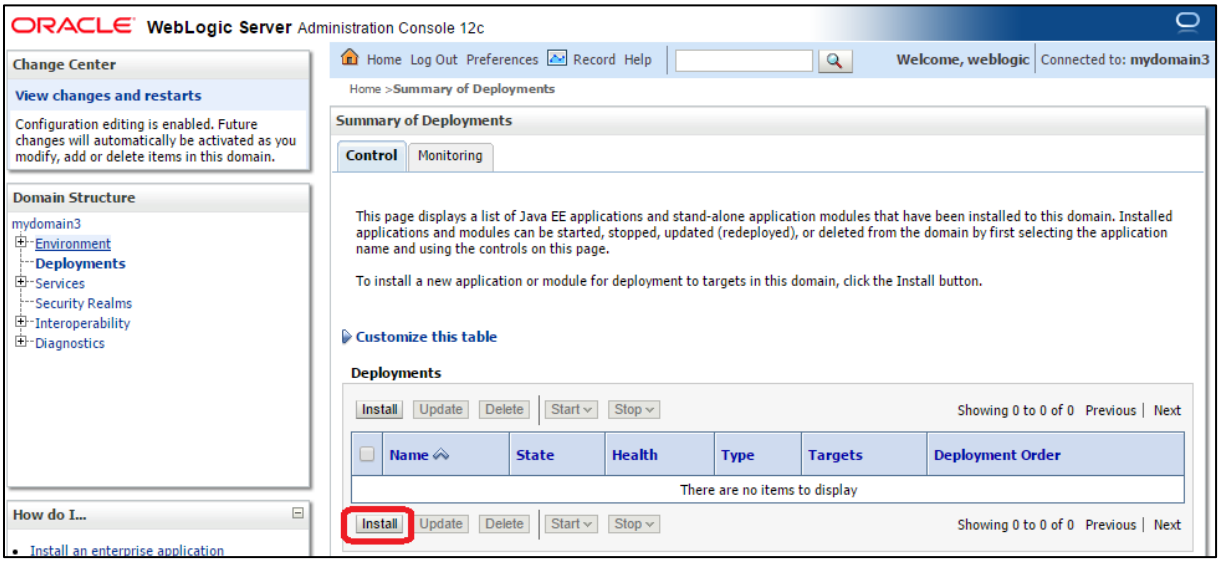
7. The database dialect is pre-configured to use the Oracle database. If you are using the MySQL database you should adapt the dialect as highlighted in the text below in <*DOMAIN_HOME*>**/conf/domibus/domibus.properties** file :

```
#EntityManagerFactory
domibus.entityManagerFactory.jpaProperty.hibernate.connection.driver_class=
com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
domibus.entityManagerFactory.jpaProperty.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect
```

8. Install the WS plugin. For more details, refer to chapter §6.2.1.2 – *"WebLogic"*.

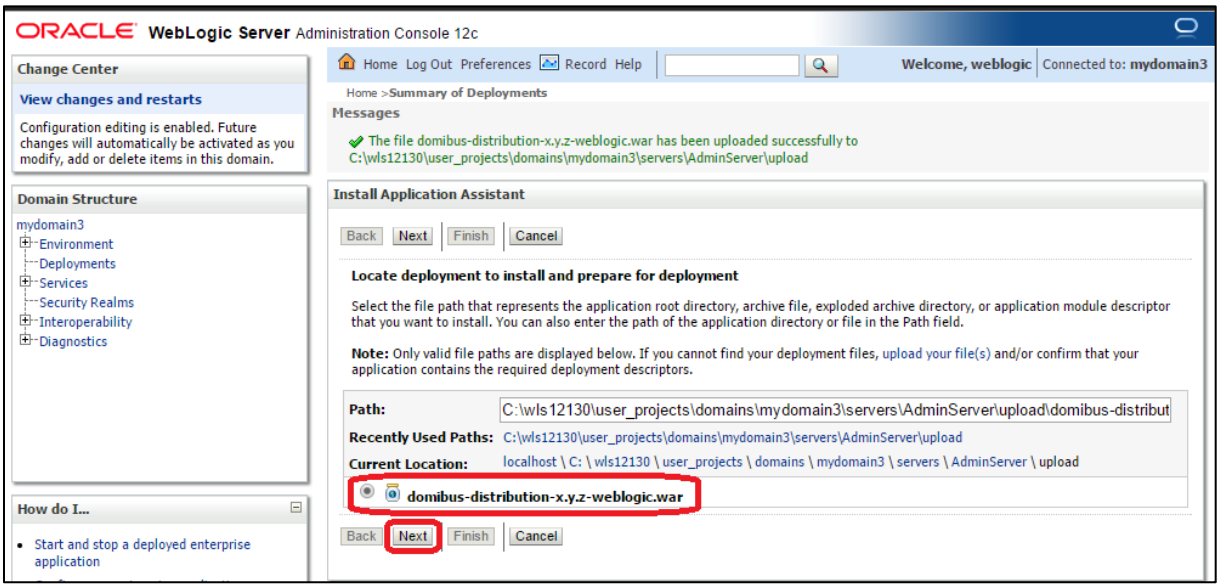
9. Deploy **domibus-distribution-X.Y.Z-weblogic.war**.

- Click **Install**

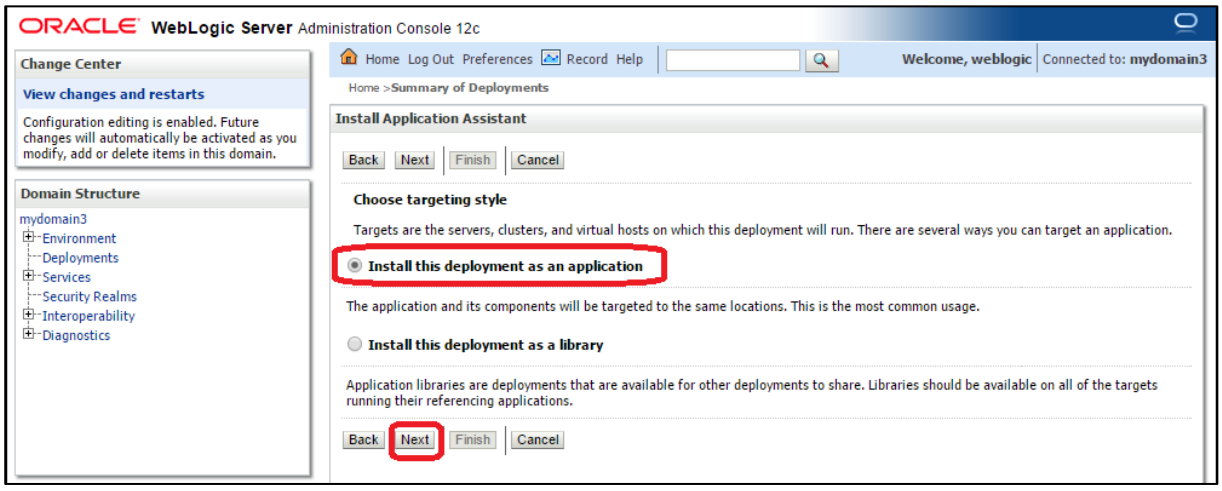


- Navigate to location DOMAIN_HOME**/conf/domibus** where the **domibus-distribution-X.Y.Z-weblogic.war** file has been previously copied.

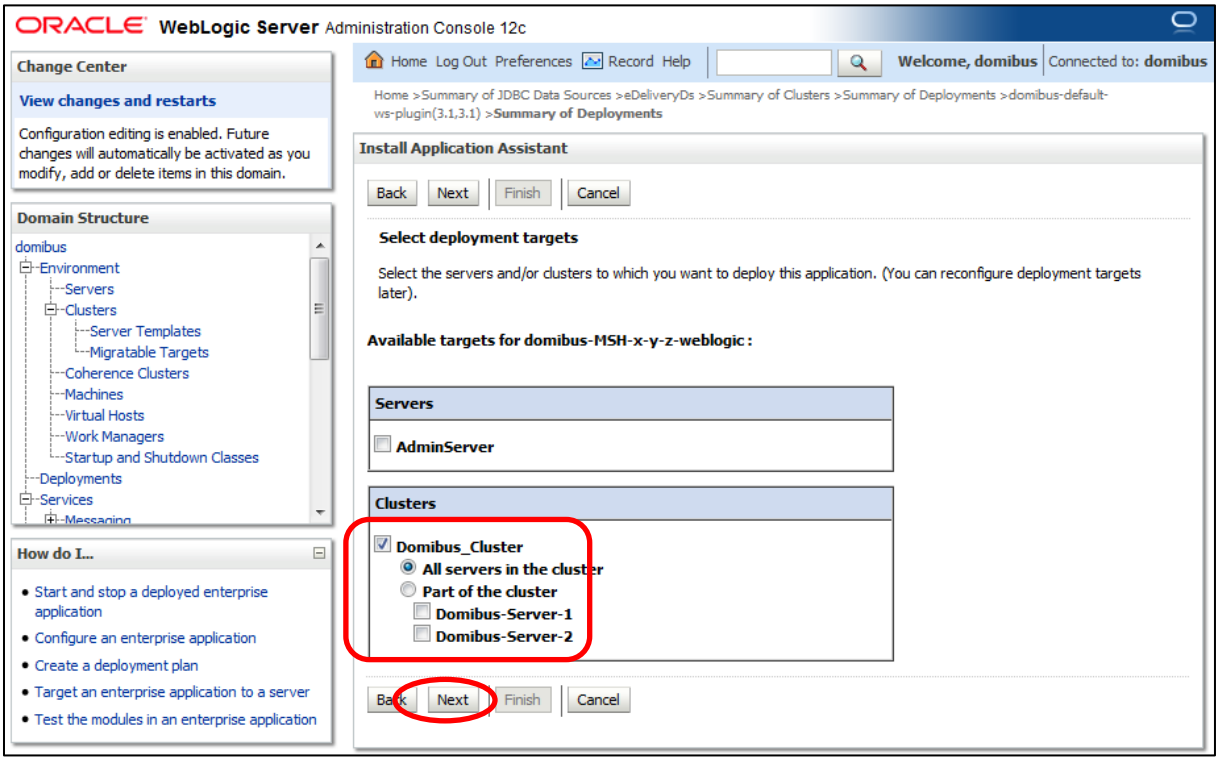
- Select the **domibus-distribution-X.Y.Z-weblogic.war** file and click **Next**:



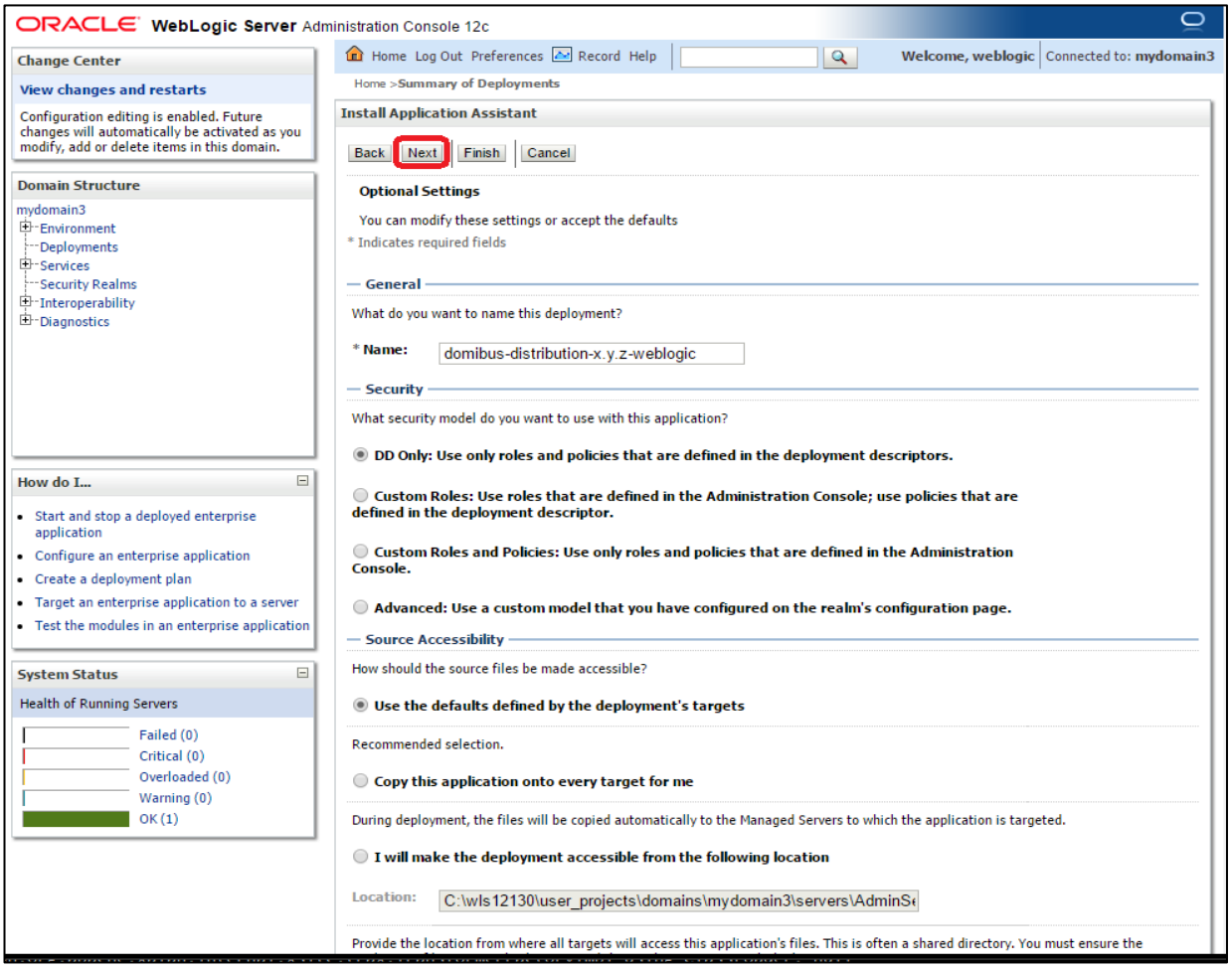
- Choose **Install this deployment as an application** and click **Next**:



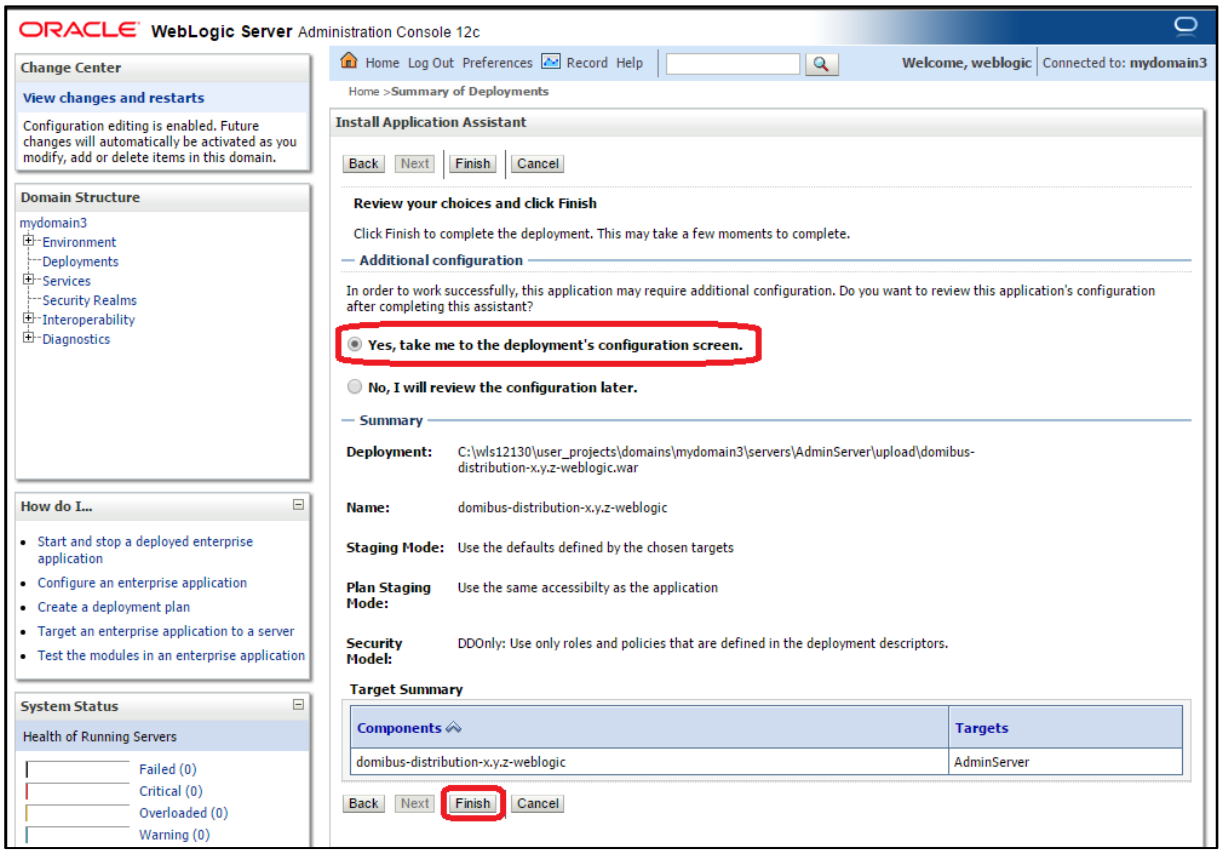
- Select your cluster for the deployment target and click **Next**:



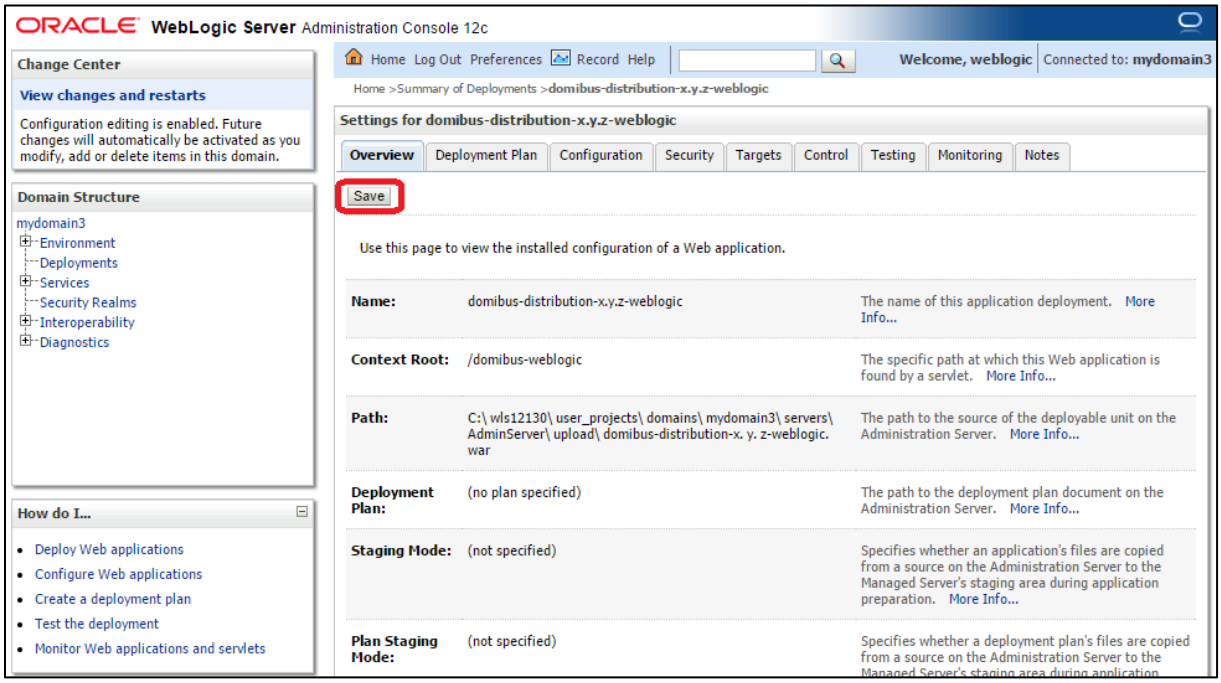
- Select the following options and click **Next**:



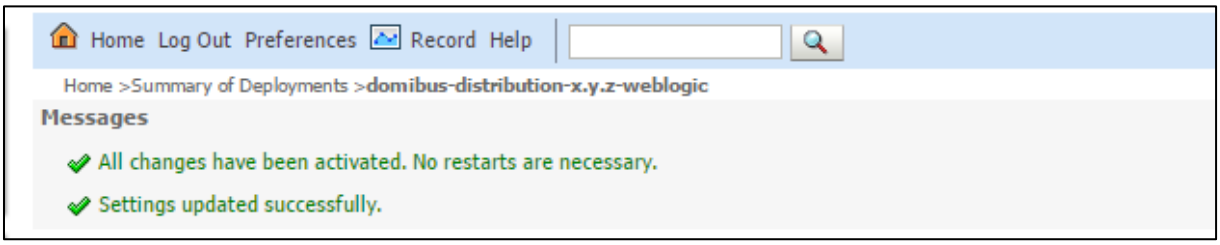
- Select the following option and click **Finish**:



- Here is an overview of the resulting settings, you can now click on the **Save** button:



The expected positive response to the deployment request should be the following:



10. Verify the installation by navigating with your browser to http://*localhost*:*7001*/domibus-weblogic

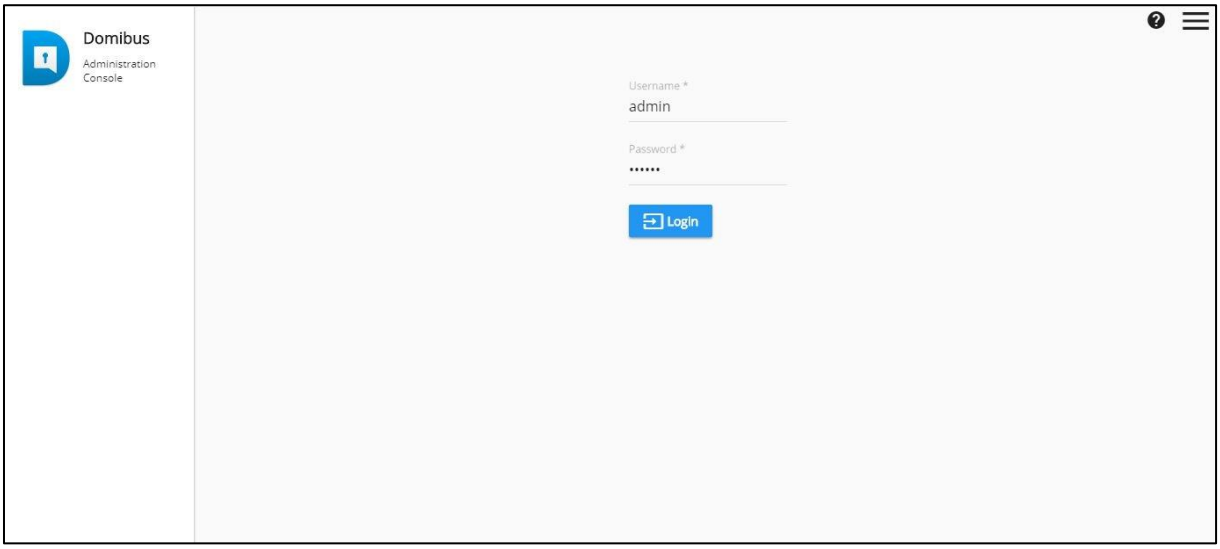
    If you can access the page it means the deployment was successful.

    (by default: user = ***admin***; password = ***123456)***

    Remark*:*

    It is recommended to change the passwords for the default users (See §9.1 – *"Administration " for further information).*

Expected result:



4.3. Domibus on Tomcat

Remark:

*As Tomcat is not a full Java EE application server and does not offer JMS capabilities by default, Domibus uses ActiveMQ as an in-memory JMS broker when deployed on a Tomcat servlet container. The configuration for the ActiveMQ JMS broker can be found in cef_edelivery_path****/domibus/internal/activemq.xml****.*

4.3.1. Pre-Configured Single Server Deployment

For this step, you will have to use the following resources (see section §3.1–*"Binaries repository"* for the download location):

- **domibus-distribution-X.Y.Z-tomcat-full.zip**

1. Unzip the archive:

    - Unzip **domibus-distribution-X.Y.Z-tomcat-full.zip** to a location on your physical machine: *cef_edelivery_path*.

2. Prepare the database:

    - For MySQL database:

Add MySQL JDBC driver (available on MySQL official web site cf. [REF2]) in the folder *cef_edelivery_path***/domibus/lib**.

Remark:

The version of the JDBC driver has to be mysql-connector-java-5.1.40.jar or higher.

Edit the properties file *cef_edelivery_path***/conf/domibus/domibus.properties** and adjust the highlighted parts in the text below according to your environment. The properties associated to the database configuration are pre-configured for the MySQL database:

```
# ------------------------------- Database -------------------------------
#Database server name
domibus.database.serverName=localhost

#Database port
domibus.database.port=3306

#XA properties
domibus.datasource.xa.property.user=edelivery_user
domibus.datasource.xa.property.password=edelivery_password
#MySQL
domibus.datasource.xa.property.url=jdbc:mysql://${domibus.database.serverName}:${domibus.database.port}/ domibus_schema?pinGlobalTxToPhysicalConnection=true
#Non-XA Datasource
domibus.datasource.url=jdbc:mysql://${domibus.database.serverName}:${domibus.database.port}/domibus_schema?useSSL=false
domibus.datasource.user=edelivery_user
domibus.datasource.password=edelivery_password
```

    - For Oracle database:

Add the Oracle JDBC driver (e.g. ***ojdbc7.jar***) (available on the Oracle official web site cf.[REF3]) in the *cef_edelivery_path***/domibus/lib folder**.

Edit the properties file *cef_edelivery_path***/conf/domibus/domibus.properties** and adjust the highlighted parts in the text below according to your environment:

```
# ---------------------------------- Database ----------------------------------

#Database server name
domibus.database.serverName=localhost

#Database port
domibus.database.port=1521

#XA Datasource
domibus.datasource.xa.xaDataSourceClassName=oracle.jdbc.xa.client.OracleXADataSource
#XA properties
domibus.datasource.xa.property.user=edelivery_user
domibus.datasource.xa.property.password=edelivery_password
domibus.datasource.xa.property.url=jdbc:oracle:thin:@${domibus.database.serverName}:${domibus.database.port}[:SID|/Service]

#Non-XA Datasource
domibus.datasource.driverClassName=oracle.jdbc.OracleDriver
domibus.datasource.URL=jdbc:oracle:thin:@${domibus.database.serverName}:${domibus.database.port}[:SID|/Service]
domibus.datasource.user=edelivery_user
domibus.datasource.password=edelivery_password
```

Remark:

Configure the database dialect as it is pre-configured for MySQL by default.

```
#EntityManagerFactory
domibus.entityManagerFactory.jpaProperty.hibernate.connection.driver_class=oracle.jdbc.xa.client.OracleXADataSource
domibus.entityManagerFactory.jpaProperty.hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
```

3. Configure your Keystore based on section §5.1.2 – *"Certificates"*.

4. Set JVM parameters:

Domibus expects a single environment variable **domibus.config.location**, pointing towards the *cef_edelivery_path***/conf/domibus** folder.

You can do this by editing the first command lines of *cef_edelivery_path***\domibus\bin\setenv.bat** (Windows) or *cef_edelivery_path***/domibus/bin/setenv.sh** (Linux). Set **CATALINA_HOME** equal to the absolute path of the installation *cef_edelivery_path***/domibus.**

- **For Windows** : edit *cef_edelivery_path***\domibus\bin\setenv.bat** by adding the following:

```
…
set CATALINA_HOME=cef_edelivery_path\domibus
set  JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8  -Xms128m -Xmx1024m -XX:PermSize=64m
set JAVA_OPTS=%JAVA_OPTS% -Ddomibus.config.location=%CATALINA_HOME%\conf\domibus
…
```

- **For Linux** : edit *cef_edelivery_path***/domibus/bin/setenv.sh** by adding the following:

```
…
export CATALINA_HOME=cef_edelivery_path/domibus
export JAVA_OPTS="$JAVA_OPTS -Xms128m -Xmx1024m "
export JAVA_OPTS="$JAVA_OPTS -Ddomibus.config.location=$CATALINA_HOME/conf/domibus"
…
```

5. Launch the Domibus application:

   - For Windows :

```
cd cef_edelivery_path\domibus\bin\
startup.bat
```

   - For Linux :

```
cd cef_edelivery_path /domibus/bin/chmod  u+x *.sh ./startup.sh
```

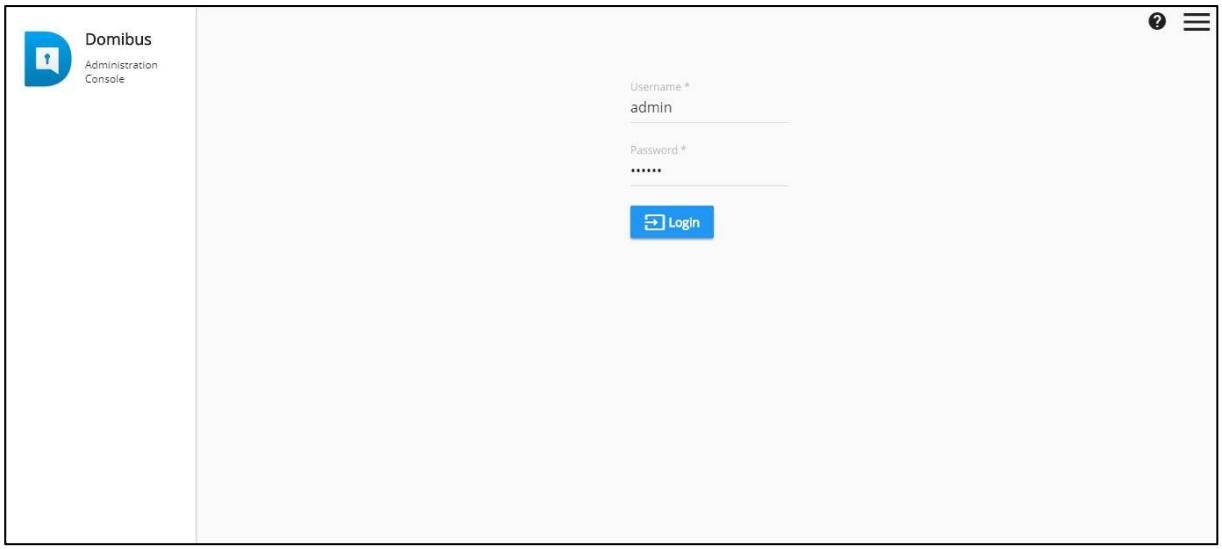
6. Display the Domibus home page on your browser: http://localhost:8080/domibus.
   (By default: User = ***admin***; Password = ***123456)***

   Remark*:*

   It is recommended to change the passwords for the default users. See §9.1 – "Administration " for further information.

   If you can access the page it means the deployment was successful.

Expected result:



4.3.2. Single Server Deployment

For this step, you will have to use the following resources (see §3.1–" *Binaries repository*" for the download location):

- **domibus-distribution-X.Y.Z-tomcat-configuration.zip**
- **domibus-distribution-X.Y.Z-tomcat-war.zip**

We assume that an Apache Tomcat 8.0.x is already installed and the installation location is now considered as your *cef_edelivery_path/domibus***.**

1. Download and unzip the artefact **domibus-distribution-X.Y.Z-tomcat-configuration.zip** into the directory *cef_edelivery_path/conf/domibus.*
2. Configure the MySQL or Oracle datasource as indicated in §4.3.1 – *"Pre-Configured Single Server Deployment"*
3. Configure your Keystore based on §5.1.2 – *"Certificates".*
4. Execute *step 4* from §4.3.1 – *"Pre-Configured Single Server Deployment"*.
5. If not already present, create a folder and name it **temp** under *cef_edelivery_path/conf/Domibus*.
6. Rename **domibus-MSH-X.Y.Z-tomcat.war** to **domibus.war** and deploy it to *cef_edelivery_path***/domibus /webapps**.

7. Launch the Domibus application:
   - For Windows :

```
cd cef_edelivery_path\domibus\bin\
startup.bat
```

   - For Linux :

```
cd cef_edelivery_path /domibus/bin/
chmod +x *.sh
./startup.sh
```

8. Display the Domibus home page on your browser: http://localhost:8080/domibus

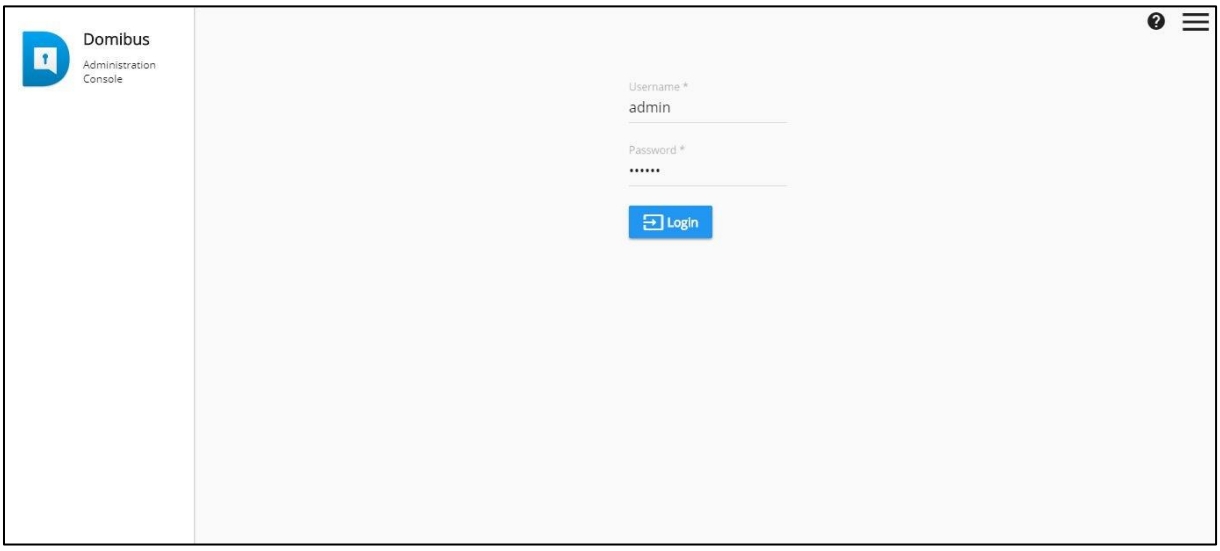
   (By default: User = ***admin***; Password = ***123456)****:*

   Remark*:*

   It is recommended to change the passwords for the default users. See §9.1 – "Administration " for further information.

   Accessing the page is an indication of a successful deployment.

Expected result:



4.3.3. Clustered Deployment

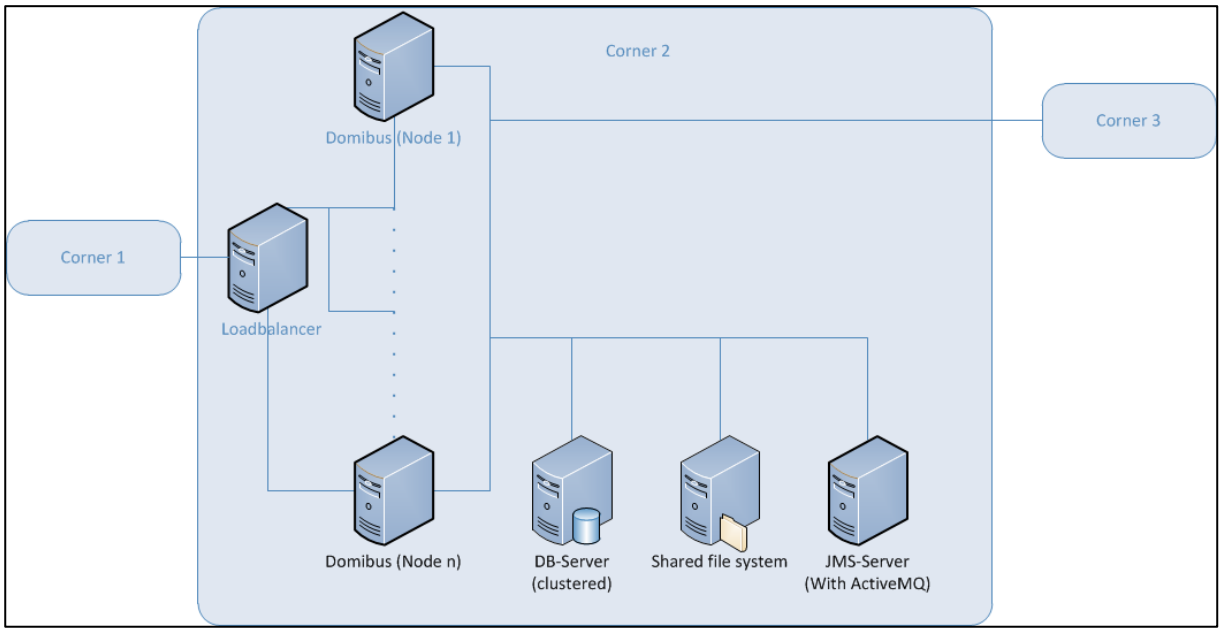


Figure 2 - Diagram representing the Deployment of Domibus in a Cluster on Tomcat

Remark:

In this section we assume that a JMS Broker and a Loadbalancer are configured separately (e.g. httpd).

For this step, you will have to use the following resources (see §3.1–" *Binaries repository* "for the download location):

- **domibus-distribution-X.Y.Z-tomcat-full.zip**
- **domibus-distribution-X.Y.Z-tomcat-war.zip**

1. Follow steps **1**, **2, 3, 4** and **5** from the §4.3.2 – *"Single Server Deployment"*

2. Set the JVM parameters:

Domibus expects a single JVM parameter **$domibus.config.location**, pointing towards the *cef_edelivery_path***/conf/domibus** folder.

You can do this by editing *cef_edelivery_path***\domibus\bin\setenv.bat** (Windows) or *cef_edelivery_path***/domibus/bin/setenv.sh** (Linux). Set **CATALINA_HOME** equal to the absolute path of the installation *cef_edelivery_path***/Domibus**.

- For Windows**:** edit *cef_edelivery_path***\domibus\bin\setenv.bat** by adding the following:

Remark:

your_node_id refers to the installed node in the cluster which starts normally at 01(then 02, etc.).

```
…
set CATALINA_HOME=cef_edelivery_path\domibus
set  JAVA_OPTS=%JAVA_OPTS% -Dfile.encoding=UTF-8 -Xms128m -Xmx1024m -XX:PermSize=64m
set JAVA_OPTS=%JAVA_OPTS% -Ddomibus.config.location=%CATALINA_HOME%\conf\domibus
set JAVA_OPTS=%JAVA_OPTS% -Ddomibus.node.id=your_node_id
…
```

- For Linux : edit *cef_edelivery_path***/domibus/bin/setenv.sh** by adding the following:

```
…
export CATALINA_HOME=cef_edelivery_path/domibus
export JAVA_OPTS=$JAVA_OPTS –Xms128m –Xmx1024m
export JAVA_OPTS="$JAVA_OPTS -Ddomibus.config.location=$CATALINA_HOME/conf/domibus"
export JAVA_OPTS="$JAVA_OPTS -Ddomibus.node.id=your_node_id"
…
```

3. Integrate the external JMS Broker with Domibus by adapting the following properties in *cef_edelivery_path***/conf/domibus/domibus.properties**.
   - Please note that the **activeMQ.embedded.configurationFile** property should be deleted as the JMS broker is external.

```
#ActiveMQ
activeMQ.broker.host=localhost
activeMQ.brokerName=localhost
#activeMQ.embedded.configurationFile=file:///${domibus.config.location}/internal/activemq.xml
activeMQ.connectorPort=1199
activeMQ.rmiServerPort=1200
activeMQ.transportConnector.uri=tcp://${activeMQ.broker.host}:61616
activeMQ.username=domibus
activeMQ.password=changeit
```

4. Change the following properties related to the **Atomikos** configuration in parameters in *cef_edelivery_path***/conf/domibus/domibus.properties**:

For clustered deployment:

Uncomment the following lines:

```
#com.atomikos.icatch.output_dir=${domibus.work.location:${domibus.config.location}}/work/transactions/${domibus.node.id}
#com.atomikos.icatch.log_base_dir=${domibus.work.location:${domibus.config.location}}/work/transactions/${domibus.node.id}/log
```

Comment the following lines:

```
com.atomikos.icatch.output_dir=${domibus.work.location:${domibus.config.location}}/work/transactions
com.atomikos.icatch.log_base_dir=${domibus.work.location:${domibus.config.location}}/work/transactions/log
```

5. Follow step **6** and **7** from the §4.3.2 – *"Single Server Deployment".*

4.4. Domibus on WildFly

4.4.1. Pre-Configured Single Server Deployment

In this section we assume that WildFly is installed at the location *cef_edelivery_path***/domibus**.

For this step, you will have to use the following resources (see section §3.1–*" Binaries repository"* for the download location):

- **domibus-distribution-X.Y.Z-wildfly-full.zip zip (WildFly 9 version)**
- **or domibus-distribution-X.Y.Z-wildfly12-full.zip (WildFly 12 version)**

Remark: below steps apply for both distributions of Domibus.

1. Download and unzip the **domibus-distribution-X.Y.Z-wildfly-full.zip** archive in your *cef_edelivery_path* location.

2. Configure the MySQL database (Option 1).

   - Drivers:

     Create the directory *cef_edelivery_path***/domibus/modules/system/layers/base/com/mysql/main** if it does not exist**.**
     Under this directory:

     - Download the MySQL JDBC driver available on MySQL official web site (cf.[REF2]) and copy it in the folder.

       Remark:

       The version of the driver has to be mysql-connector-java-5.1.40.jar or higher.

     - Create or edit the file *cef_edelivery_path***/domibus/modules/system/layers/base/com/mysql/main/module.xml** and copy the following module configuration. Make sure to put the name of the driver you are using as an argument of **resource-root** element. e.g. ***mysql-connector-java-5.1.40.jar***:

```
<module xmlns="urn:jboss:module:1.3" name="com.mysql">
        <resources>
                <resource-root path="mysql-connector-java-5.1.40.jar"/>
        </resources>
        <dependencies>
                <module name="javax.api"/>
                <module name="javax.transaction.api"/>
    </dependencies>
</module>
```

- Add your DBMS driver metadata to the Drivers section of the *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml**.

```
<subsystem xmlns="urn:jboss:domain:datasources:3.0">
  ……..……
  <datasources>
    ……..……
    <drivers>
          <driver name="com.mysql" module="com.mysql">
       <driver-class>com.mysql.jdbc.Driver</driver-class>
       <xa-datasource-class>
             com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
       </xa-datasource-class>
       <!--Connector/J 8.0.x
      <driver-class>com.mysql.cj.jdbc.Driver</driver-class>
      <xa-datasource-class>com.mysql.cj.jdbc.MysqlXADataSource</xa-datasource-class>
       -->
     </driver>

            <!--Oracle

            <driver name="com.oracle" module="com.oracle">

              <driver-class>oracle.jdbc.driver.OracleDriver</driver-class>

            </driver>
            -->
    <drivers>
    ……..……
  </datasources>
  ……..……
</subsystem>
```

- Datasources:
  - Add the datasources as indicated below to *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml**.

Remark:

*Please make sure you modify the connection details for the **MysqlXADS** datasource for MySQL according to your environment.*

```
<subsystem xmlns="urn:jboss:domain:datasources:3.0">
<datasources>
……..……
```

```
<xa-datasource jndi-name="java:/jdbc/cipaeDeliveryDs" pool-
name="eDeliveryMysqlXADS" enabled="true" use-ccm="true" statistics-enabled="true">
          <xa-datasource-property name="ServerName">localhost</xa-datasource-property>
        <xa-datasource-property name="DatabaseName">domibus_schema</xa-datasource-property>
        <xa-datasource-class>com.mysql.jdbc.jdbc2.optional.MysqlXADataSource</xa-datasource-class>
          <!--Connector/J 8.0.x
          <xa-datasource-class>com.mysql.cj.jdbc.MysqlXADataSource</xa-datasource-class>
         -->

          <driver>com.mysql</driver>
              <security>
                 <user-name>edelivery_user</user-name>
                 <password>edelivery_password</password>
              </security>
<validation>
<valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLValidConnectionChecker"/>
<background-validation>true</background-validation>
<exception-sorter class-name="org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLExceptionSorter"/>
                    </validation>
          </xa-datasource>
<datasource jndi-name="java:/jdbc/cipaeDeliveryNonXADs" pool-name="eDeliveryMysqlNonXADS"
enabled="true" use-ccm="true">
   <connection-url>jdbc:mysql://localhost:3306/domibus_schema</connection-url>
   <driver-class>com.mysql.jdbc.Driver</driver-class>
   <!--Connector/J 8.0.x
   <driver-class>com.mysql.cj.jdbc.Driver</driver-class>
    -->
   <driver>com.mysql</driver>
   <security>
      <user-name>edelivery_username</user-name>
      <password>edelivery_password</password>
   </security>
   <validation>
      <valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLValidConnectionChecker"/>
      <background-validation>true</background-validation>
      <exception-sorter class-
name="org.jboss.jca.adapters.jdbc.extensions.mysql.MySQLExceptionSorter"/>
   </validation>
</datasource>
……..
          </datasources>
</subsystem>
```

3. Configure the Oracle Database (option 2):
    - Drivers:

Create the directory *cef_edelivery_path***/domibus/modules/system/layers/base/com/oracle/main** if it does not exist. Under this directory:

- Download and copy the Oracle JDBC driver (e.g. ***ojdbc7.jar***, available on the Oracle official web site cf.[REF3]) in the folder.

- Create or edit the file *cef_edelivery_path***/domibus/modules/system/layers/base/com/oracle/main/module.xml** in the recently created folder.

  Add the following module configuration. Make sure to put the name of the driver you are using as an argument of **resource-root** element. e.g. ***ojdbc7.jar***:

```
<module xmlns="urn:jboss:module:1.3" name="com.oracle">
        <resources>
                <resource-root path="ojdbc7.jar"/>
        </resources>
        <dependencies>
                <module name="javax.api"/>
                <module name="javax.transaction.api"/>
        </dependencies>
</module>
```

- Uncomment Oracle paragraph from the Drivers section in *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml** .

```
<subsystem xmlns="urn:jboss:domain:datasources:3.0">
  ……..……
  <datasources>
    ……..……
    <drivers>
          <driver name="com.mysql" module="com.mysql">
       <driver-class>com.mysql.jdbc.Driver</driver-class>
       <xa-datasource-class>
     com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
       </xa-datasource-class>
       <!--Connector/J 8.0.x
      <driver-class>com.mysql.cj.jdbc.Driver</driver-class>
      <xa-datasource-class>com.mysql.cj.jdbc.MysqlXADataSource</xa-datasource-class>
       -->
      </driver>

            <!--Oracle

            <driver name="com.oracle" module="com.oracle">

              <driver-class>oracle.jdbc.driver.OracleDriver</driver-class>

            </driver>
            -->
    <drivers>
    ……..……
  </datasources>
  ……..……
</subsystem>
```

  - Datasources:

    - Uncomment the Oracle paragraph from the datasources section of *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml**.

Remark:

*Please make sure you modify the connection details for both **eDeliveryOracleNonXADS** and **eDeliveryOracleXADS** datasource for Oracle according to your environment.*

```
<!-- Oracle
<datasource jta="true" jndi-name="java:/jdbc/cipaeDeliveryNonXADs" pool-
name="eDeliveryOracleNonXADS" enabled="true" use-ccm="true">
   <connection-url>jdbc:oracle:thin:@localhost:1521[:SID|/Service]</connection-url>
   <driver-class>oracle.jdbc.OracleDriver</driver-class>
   <driver>com.oracle</driver>
   <security>
      <user-name>edelivery_username</user-name>
      <password>edelivery_password</password>
   </security>
   <validation>
      <valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConnectionChecker"/>
      <background-validation>true</background-validation>
      <stale-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleStaleConnectionChecker"/>
      <exception-sorter class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleExceptionSorter"/>
   </validation>
</datasource>
-->

…
<!-- Oracle
<xa-datasource jndi-name="java:/jdbc/cipaeDeliveryDs" pool-name="eDeliveryOracleXADS"
enabled="true" use-ccm="true">
   <xa-datasource-property name="URL">
      jdbc:oracle:thin:@localhost:1521[:SID|/Service]
   </xa-datasource-property>
   <xa-datasource-class>oracle.jdbc.xa.client.OracleXADataSource</xa-datasource-class>
   <driver>com.oracle</driver>
   <security>
      <user-name>edelivery_user</user-name>
      <password>edelivery_password</password>
   </security>
   <validation>
      <valid-connection-checker class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleValidConnectionChecker"/>
      <exception-sorter class-
name="org.jboss.jca.adapters.jdbc.extensions.oracle.OracleExceptionSorter"/>
   </validation>
</xa-datasource>
-->
```

- Edit the configuration file *cef_edelivery_path***/conf/domibus/domibus.properties** and configure the datasources as indicated below.

Remark:

Configure the database dialect as it is pre-configured for MySQL by default.

```
#EntityManagerFactory
domibus.entityManagerFactory.jpaProperty.hibernate.connection.driver_class=oracle.jdbc.xa.client.OracleXADataSource
domibus.entityManagerFactory.jpaProperty.hibernate.dialect=org.hibernate.dialect.Oracle10gDialect
```

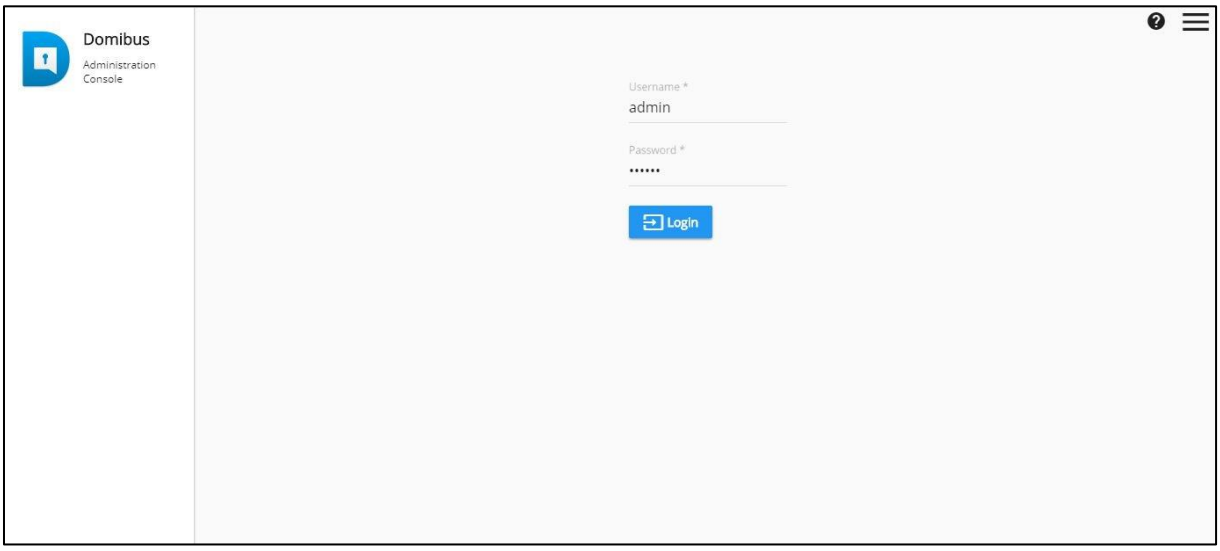
4. Configure your Keystore based on §5.1.2 – *"Certificates"*.
5. Run the standalone server:
   - For Windows under *cef_edelivery_path***\domibus\bin**
   - **standalone.bat --server-config=standalone-full.xml**
   - For Linux under *cef_edelivery_path***/domibus/bin/**
   - **standalone.sh --server-config=standalone-full.xml**
6. Display the Domibus home page in your browser: http://*localhost*:*8080*/domibus-wildfly (by default: User = ***admin***; Password = ***123456***).

   Remark:

   It is recommended to change the passwords for the default users. See §9.1 – *"Administration " for further information.*

   If you can access the page it means the deployment was successful.

Expected result:



4.4.2. Single Server Deployment

In this section we assume that WildFly (9 or 12 version) is installed at the location *cef_edelivery_path***/domibus**.

For this step, you will have to use the following resources (see section §3.1-"*Binaries repository* " for the download location):

- **domibus-distribution-X.Y.Z-wildfly-war.zip** (WildFly 9) **or domibus-distribution-X.Y.Z-wildfly12-war.zip** (WildFly 12)

- **domibus-distribution-X.Y.Z-wildfly-configuration.zip** (WildFly 9) **or domibus-distribution-X.Y.Z-wildfly12-configuration.zip** (WildFly 12)

1. Follow steps **2** (MySQL) or **3** (Oracle) from the §4.4.1 – *"Pre-Configured Single Server Deployment".*

2. Configure the environment variables under *cef_edelivery_path***/domibus/bin/standalone.conf**:

```
……….
JAVA_OPTS="-Xms128m –Xmx1024m
 java.net.preferIPv4Stack=true"
JAVA_OPTS="$JAVA_OPTS -Ddomibus.config.location=$JBOSS_HOME/conf/domibus
……….
```

3. Download and unzip **domibus-distribution-X.Y.Z-wildfly-configuration.zip** (WildFly 9) or **domibus-distribution-X.Y.Z-wildfly12-configuration.zip** (WildFly 12) in the directory *cef_edelivery_path***/conf/domibus**.
4. Configure your Keystore based on §5.1.2 – *"Certificates".*
5. Configure the JMS resources:

Configure the JMS resources in the configuration file *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml** by adding the **jms-connection-factories** and **jms-queues**.

```
<address-settings>
    <!--default for catch all-->
    <address-setting match="#">
        <dead-letter-address>jms.queue.DLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <max-size-bytes>10485760</max-size-bytes>
        <page-size-bytes>2097152</page-size-bytes>
        <message-counter-history-day-limit>10</message-counter-history-day-limit>
    </address-setting>
    <address-setting match="jms.queue.DomibusSendMessageQueue">
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>1000</redelivery-delay>
        <max-delivery-attempts>1</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusPullMessageQueue">
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>1000</redelivery-delay>
        <max-delivery-attempts>1</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusBusinessMessageOutQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusNotifyBackendJmsQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
```

```
    </address-setting>
    <address-setting match="jms.queue.DomibusErrorNotifyConsumerQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusErrorNotifyProducerQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
        <address-setting match="jms.queue.DomibusBusinessMessageInQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusPluginToBackendQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusNotifyBackendWebServiceQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusUnknownReceiverQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusNotifyBackendQueue">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>300000</redelivery-delay>
        <max-delivery-attempts>10</max-delivery-attempts>
    </address-setting>
    <address-setting match="jms.queue.DomibusClusterCommandTopic">
        <dead-letter-address>jms.queue.DomibusDLQ</dead-letter-address>
        <expiry-address>jms.queue.ExpiryQueue</expiry-address>
        <redelivery-delay>10000</redelivery-delay>
        <max-delivery-attempts>3</max-delivery-attempts>
    </address-setting>
</address-settings>
……….
<subsystem xmlns="urn:jboss:domain:messaging:3.0">
    <hornetq-server>
```

```
        <jmx-management-enabled>true</jmx-management-enabled>
        <jms-connection-factories>
……….
            <connection-factory name="edeliveryConnectionFactory">
                <connectors>
                    <connector-ref connector-name="in-vm"/>
                </connectors>
                  <entries>
                      <entry name="java:/jms/ConnectionFactory"/>
                  </entries>
                    <compress-large-messages>false
                    </compress-large-messages>
                      <failover-on-initial-connection>false
                      </failover-on-initial-connection>
                        <use-global-pools>true</use-global-pools>
            </connection-factory>
          ……….
        </jms-connection-factories>
          <jms-destinations>
          ……….
          <jms-queue name="DomibusBusinessMessageOutQueue">
            <entry name="java:/jms/domibus.backend.jms.outQueue"/>
            <entry name="java:/jms/queue/DomibusBusinessMessageOutQueue"/>
                <durable>true</durable>
          </jms-queue>
          <jms-queue name="DomibusNotifyBackendJmsQueue">
            <entry name="java:/jms/domibus.notification.jms"/>
            <entry name="java:/jms/queue/DomibusNotifyBackendJmsQueue"/>
    <durable>true</durable>
          </jms-queue>
          <jms-queue name="DomibusErrorNotifyConsumerQueue">
            <entry name="java:/jms/domibus.backend.jms.errorNotifyConsumer"/>
            <entry name="java:/jms/queue/DomibusErrorNotifyConsumerQueue"/>
                <durable>true</durable>
           </jms-queue>
           <jms-queue name="DomibusErrorNotifyProducerQueue">
             <entry name="java:/jms/domibus.backend.jms.errorNotifyProducer"/>
             <entry name="java:/jms/queue/DomibusErrorNotifyProducerQueue"/>
                <durable>true</durable>
           </jms-queue>
           <jms-queue name="DomibusBusinessMessageInQueue">
             <entry name="java:/jms/domibus.backend.jms.inQueue"/>
             <entry name="java:/jms/queue/DomibusBusinessMessageInQueue"/>
                <durable>true</durable>
           </jms-queue>
           <jms-queue name="DomibusPluginToBackendQueue">
             <entry name="java:/jms/domibus.backend.jms.replyQueue"/>
             <entry name="java:/jms/queue/DomibusPluginToBackendQueue"/>
                <durable>true</durable>
           </jms-queue>
           <jms-queue name="DomibusSendMessageQueue">
             <entry name="java:/jms/domibus.internal.dispatch.queue"/>
             <entry name="java:/jms/queue/DomibusSendMessageQueue"/>
```

```
                <durable>true</durable>
            </jms-queue>
            <jms-queue name="DomibusNotifyBackendWebServiceQueue">
              <entry name="java:/jms/domibus.notification.webservice"/>
              <entry name="java:/jms/queue/DomibusNotifyBackendWebServiceQueue"/>
                <durable>true</durable>
            </jms-queue>
            <jms-queue name="DomibusUnknownReceiverQueue">
              <entry name="java:/jms/domibus.internal.notification.unknown"/>
              <entry name="java:/jms/queue/DomibusUnknownReceiverQueue"/>
                <durable>true</durable>
            </jms-queue>
            <jms-queue name="DomibusNotifyBackendQueue">
              <entry name="java:/jms/domibus.internal.notification.queue"/>
              <entry name="java:/jms/queue/DomibusNotifyBackendQueue"/>
                <durable>true</durable>
            </jms-queue>
            <jms-queue name="DLQ">
              <entry name="java:/jms/domibus/ DLQ"/>
              <entry name="java:/jms/queue/DLQ"/>
                <durable>true</durable>
            </jms-queue>
            <jms-topic name="DomibusClusterCommandTopic">
              <entry name="java:/jms/domibus.internal.command"/>
              <entry name="java:/jms/topic/DomibusClusterCommandTopic"/>
            </jms-topic>
            ……….
            </jms-destinations>
    </hornetq-server>
</subsystem>
```

Remark:

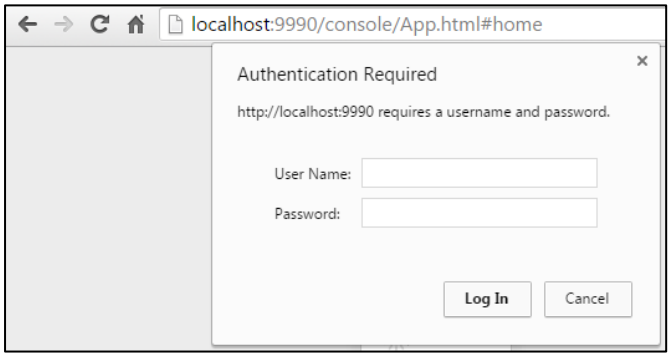
Please note that the JMX management also has to be enabled so the JMS resources can be monitored in the JMS Monitoring screen.

6. Configure the executor services:

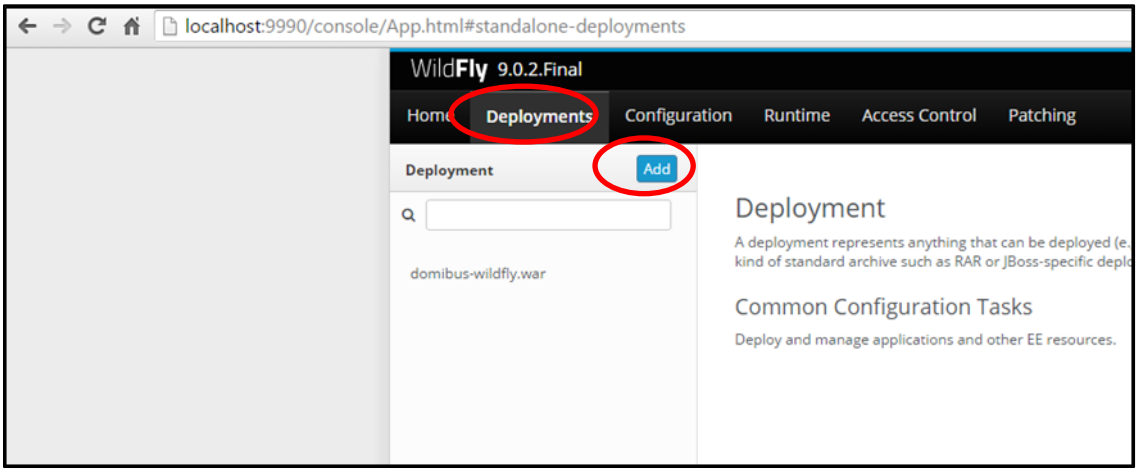
Configure the executor services in the configuration file *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml**.

```
<subsystem xmlns="urn:jboss:domain:ee:3.0">
   ..........................
   <concurrent>
      ..........................
      <managed-executor-services>
         <managed-executor-service name="domibusExecutorService" jndi-
name="java:jboss/ee/concurrency/executor/DomibusExecutorService" context-service="default" hung-
task-threshold="60000" core-threads="5" max-threads="25" keepalive-time="5000"/>
      </managed-executor-services>
      <managed-executor-services>
         <managed-executor-service name="quartzExecutorService" jndi-
name="java:jboss/ee/concurrency/executor/QuartzExecutorService" context-service="default" hung-
task-threshold="0" long-running-tasks="true" core-threads="5" max-threads="25" keepalive-
time="5000"/>
      </managed-executor-services>
       ..........................
   </concurrent>
    ..........................
<subsystem xmlns="urn:jboss:domain:ee:3.0">
```

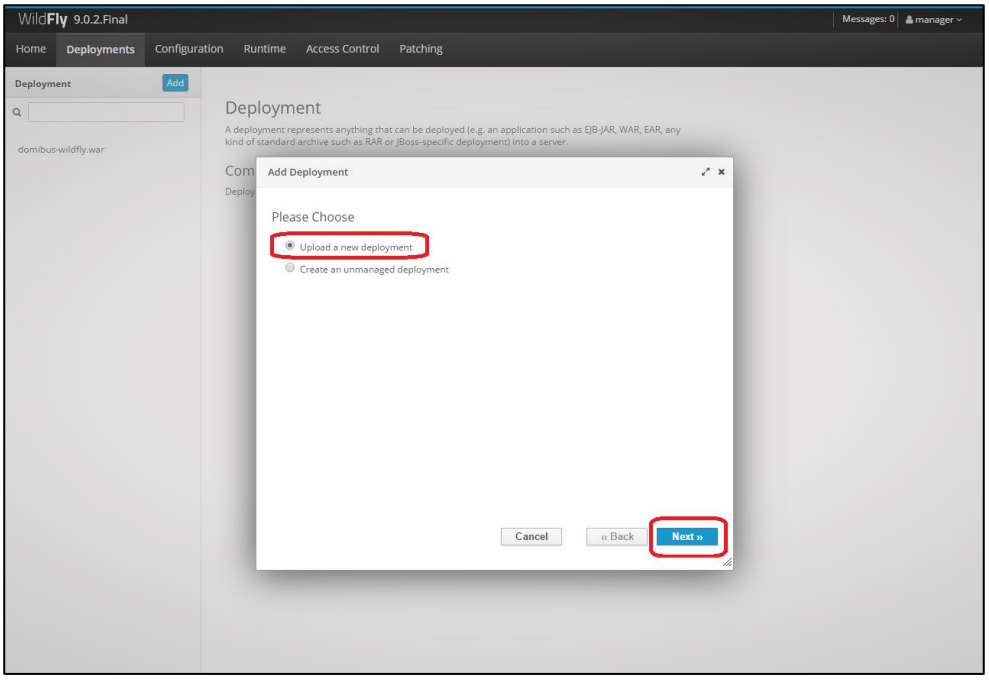
7. Connect to the Admin Console of WildFly at **http://*localhost*:*9990*/*console***:



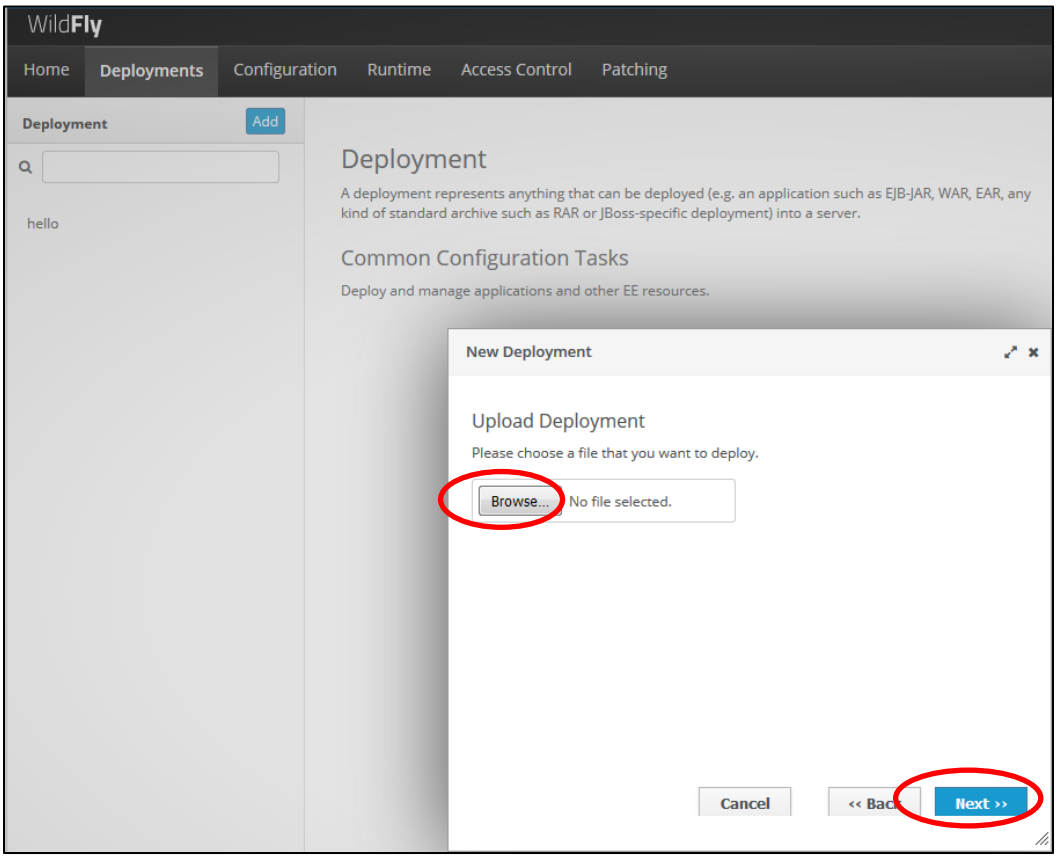
8. Click on **Deployments** in the console menu then click on **Add**:



9. Select **Upload a new deployment** then click **Next**:

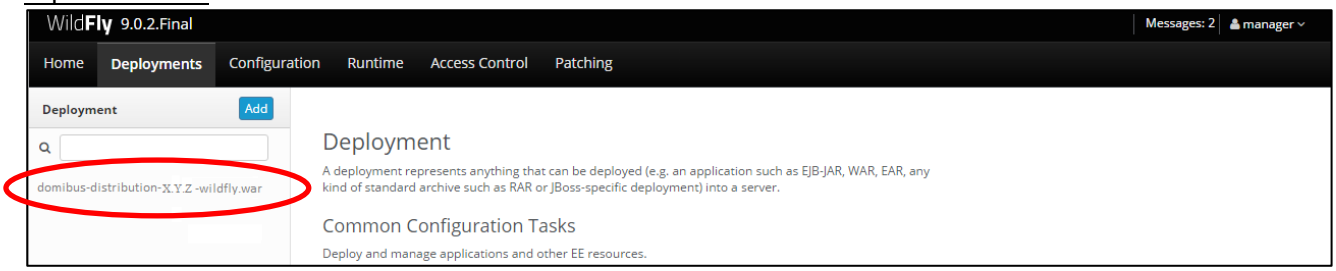


10. Browse to the location of the **domibus-distribution-X.Y.Z-wildfly.war** (WildFly 9) or **domibus-distribution-X.Y.Z-wildfly12.war** (WildFly 12) file, select it and click **Next** :



11. The deployment is successful when the name of the .war file appears in the Deployment column.

Expected result:



4.4.3. Clustered Deployment

For this step, you will have to use the following resources (see section §3.1–*"Binaries repository"* for the download location):

- **domibus-distribution-X.Y.Z-wildfly-configuration.zip** (WildFly 9) or **domibus-distribution-X.Y.Z-wildfly12-configuration.zip** (WildFly 12)
- **domibus-distribution-X.Y.Z-wildfly-war.zip** (WildFly 9) or **domibus-distribution-X.Y.Z-wildfly12-war.zip** (WildFly 12)

In this section we assume that the setup of WildFly (9 or 12 version) in domain mode has already been done and that the cluster has been enabled as described in the official documentation. For more details on how to perform an installation of WildFly (9 or 12 version) in domain mode, please refer to the official documentation (cf.[REF4]).

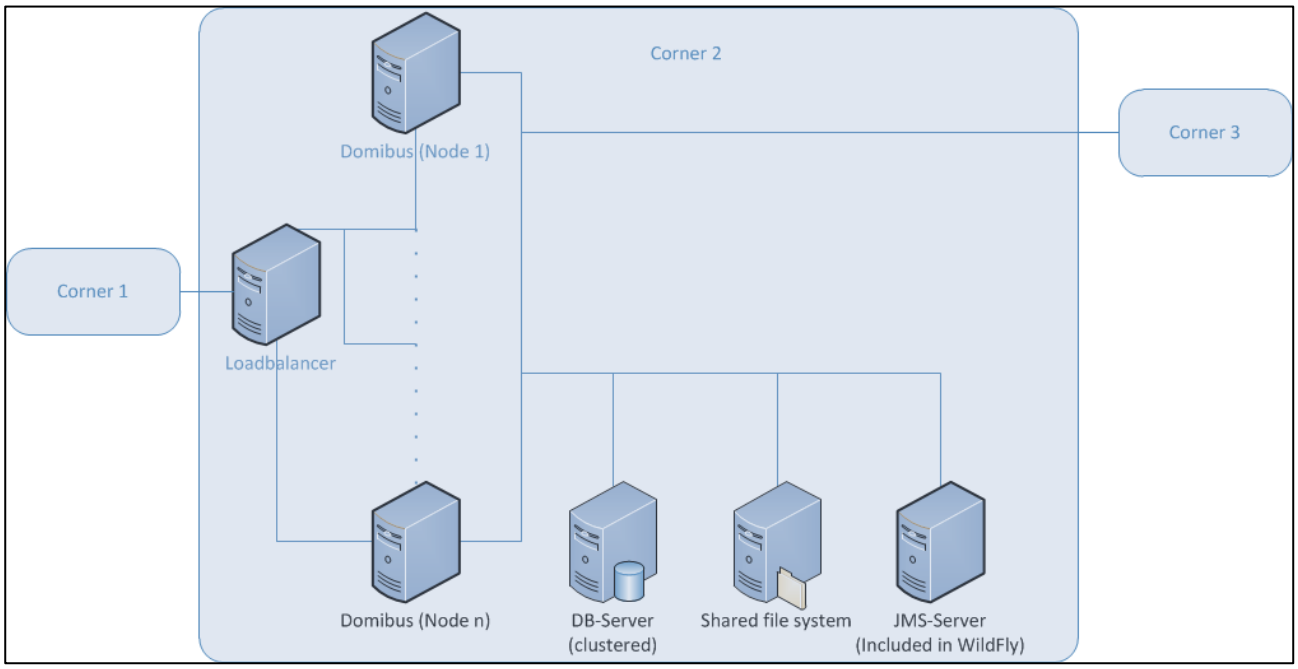


Figure 3 - Diagram representing the Deployment of Domibus in a Cluster on WildFly

In order to install Domibus in a WildFly cluster please follow the steps below:

1. Download and unzip **domibus-distribution-X.Y.Z-wildfly-configuration.zip** (for WildFly 9) or **domibus-distribution-X.Y.Z-wildfly12-configuration.zip** (for WildFly 12) in a shared location that is accessible by all the nodes from the cluster. We will refer to this directory as *cef_shared_edelivery_path***/Domibus**.

2. Follow steps **2** (MySQL) or **3** (Oracle) from the §4.4.1 – *"Pre-Configured Single Server Deployment"*.

   Remarks:

   - *This step needs to be performed on all the nodes from the cluster*
   - *In the following 2 steps we will edit the profile **full-ha** from the configuration file **domain/configuration/domain.xml** located in the master node*

3. Configure the JMS queues and topics as indicated in §4.4.2 point 5 – *"Configure the JMS resources"*.

4. Configure the database dialect as indicated in §4.4.1 point 3 - *Configure the Oracle Database (option 2)*.

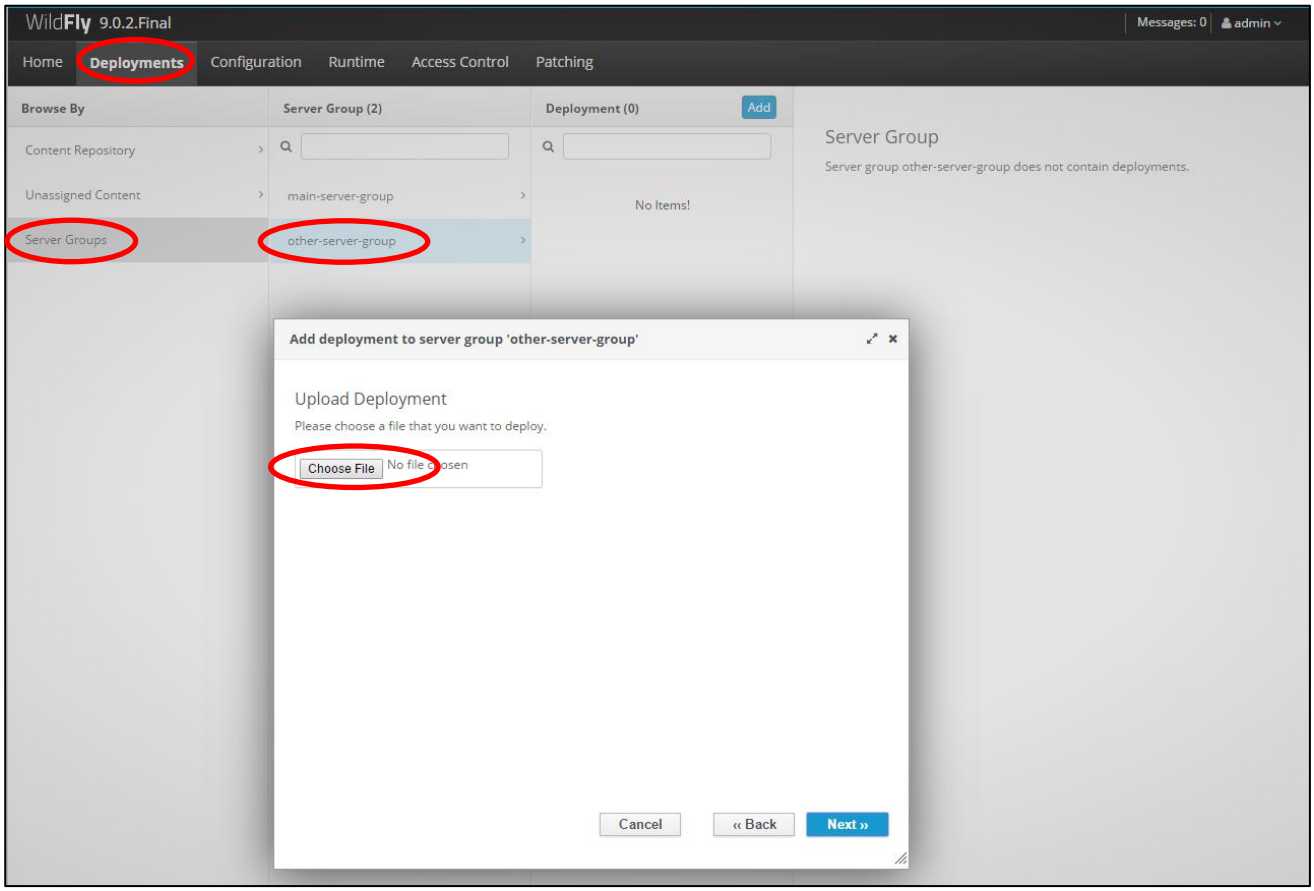
5. Configure the environment variables in the file **bin/domain.conf**.

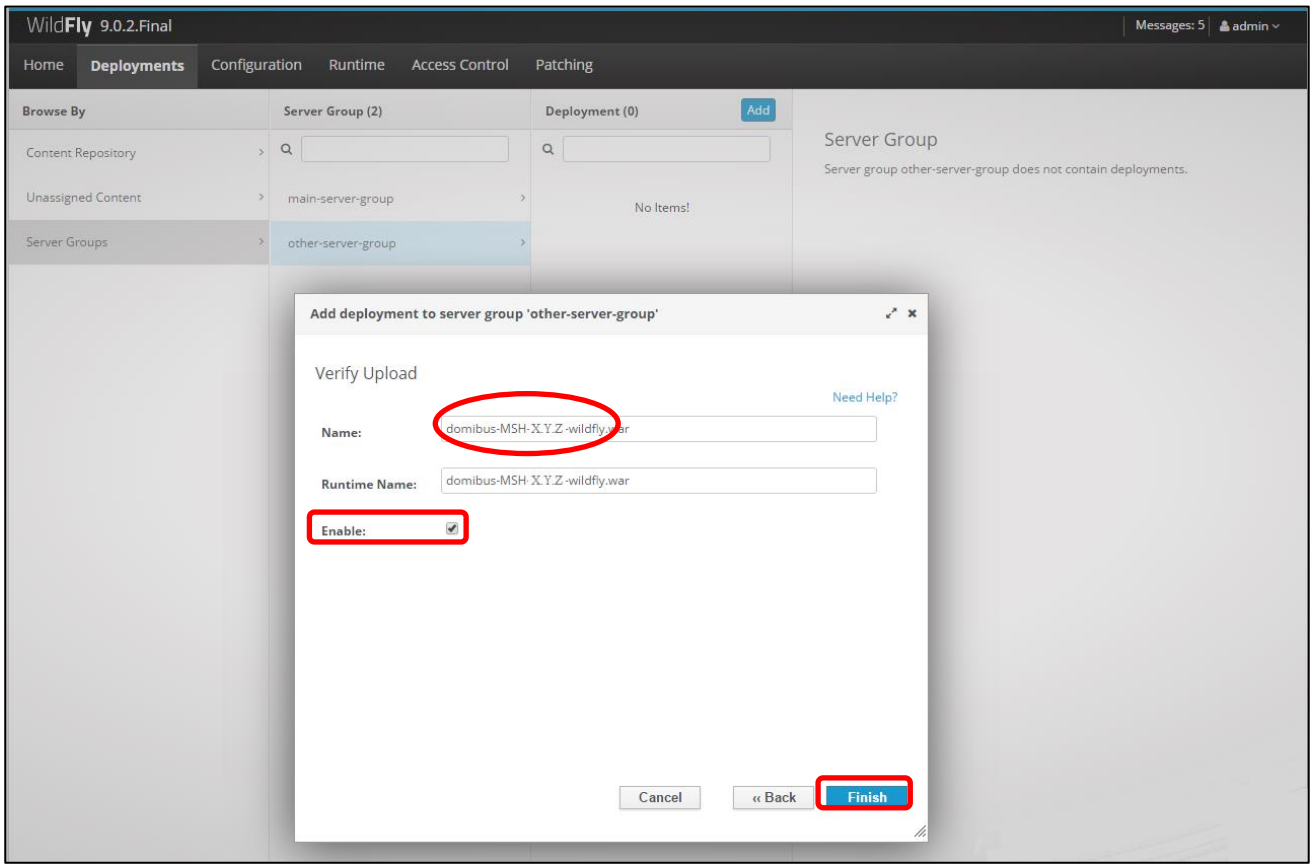
Remark:

bin/domain.conf is located in each WildFly node. The environment variable setting needs to be performed in every node from the cluster.

```
……
JAVA_OPTS="-Xms128m –Xmx1024m
-java.net.preferIPv4Stack=true"
JAVA_OPTS="$JAVA_OPTS -Ddomibus.config.location=cef_shared_edelivery_path/conf/domibus
……
```

6. Deploy the **domibus-distribution-X.Y.Z-wildfly.war** (for WildFly 9) or **domibus-distribution-X.Y.Z-wildfly12.war** (for WildFly 12) to the cluster. We will use the WildFly Administration console for performing the deployment. We will deploy the application on the **other-server-group** cluster which is configured step by step in the official documentation (cf.[REF4]).




WildFly 9.0.2.Final
Messages: 5
admin
Home
Deployments
Configuration
Runtime
Access Control
Patching
Browse By
Content Repository
Unassigned Content
Server Groups
Server Group (2)
main-server-group
other-server-group
Deployment (0)
Add
No Items!
Server Group
Server group other-server-group does not contain deployments.
Add deployment to server group 'other-server-group'
Verify Upload
Need Help?
Name:
domibus-MSH-X.Y.Z-wildfly.war
Runtime Name:
domibus-MSH-X.Y.Z-wildfly.war
Enable:
Cancel
« Back
Finish

5. DOMIBUS CONFIGURATION

Domibus exposes the Message Service Handler endpoint as **../services/msh**. Only this endpoint has to be reachable by the other AS4 Access Points and it is typically exposed on the internet.

If the Default WS Plugin (§6.1.2 – *"WS Plugin"*) is deployed, Domibus exposes the Default WS Plugin endpoint as **../services/backend**. This endpoint should ONLY be exposed to the backend client(s) within the trusted zone and it should not be exposed to the internet.

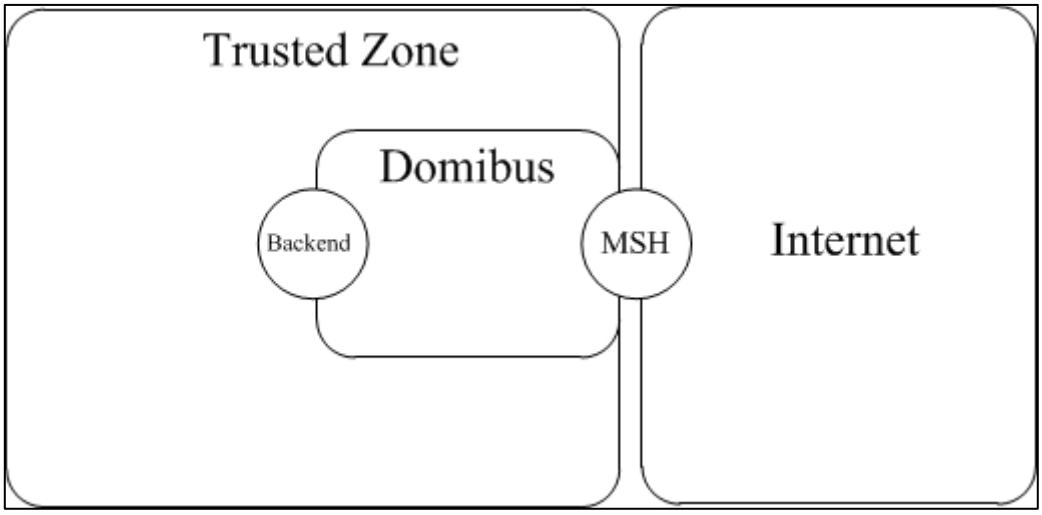


Figure 4 - Message Service Handler diagram

5.1. Security Configuration

5.1.1. *Security Policies*

The WS-Security policy used by Domibus when exchanging messages can be specified in the PMode configuration file (§7 – *"Pmode Configuration"*).

Security policy assertions are based on the ***WS-Policy framework***.

As requested by the eDelivery AS4 profile, Domibus supports all three mechanisms to reference a security token, as described below.

Domibus distribution includes one policy file for each mechanism (cef_edelivery_path/conf/domibus/policies/):

eDeliveryAS4Policy.xml - Reference to a Subject Key Identifier

The <wsse:SecurityTokenReference> element contains a <wsse:KeyIdentifier> element that specifies the token data by means of a X.509 SubjectKeyIdentifier reference. A subject key identifier MAY only be used to reference an X.509v3 certificate.

eDeliveryAS4Policy_BST.xml - Reference to a Binary Security Token

The <wsse:SecurityTokenReference> element contains a wsse:Reference> element that references a local <wsse:BinarySecurityToken> element or a remote data source that contains the token data itself.

eDeliveryAS4Policy_IS.xml - Reference to an Issuer and Serial Number

The <wsse:SecurityTokenReference> element contains a <ds:X509Data> element that contains a <ds:X509IssuerSerial> element that uniquely identifies an end entity certificate by its X.509 Issuer and Serial Number.

With the **eDeliveryAS4Policy.xml**, Domibus is able to receive messages with **all 3 referencing methods**. When eDeliveryAS4Policy_BST.xml or eDeliveryAS4Policy_IS.xml are used, the specific reference method becomes mandatory on both APs involved in the exchange.

For the connectivity with other APs, the three policies may be combined to obtain the required references for initiator/responder and signing/encryption tokens.

5.1.2. Certificates

The certificates that are used for signing and encrypting the messages when communicating with the other Access Points can be configured in the property file located under *cef_edelivery_path***/conf/domibus*/domibus.properties.***
By default Domibus is pre-configured to use self-signed certificates. Please note that self-signed certificates should be used only for testing purposes and are not intended for production use.

In order to configure Domibus to use custom certificates the following properties need to be modified:

```
#The location of the keystore
domibus.security.keystore.location=${domibus.config.location}/keystores/gateway_keystore.jks
#Type of the used keystore
domibus.security.keystore.type=jks
#The password used to load the keystore
domibus.security.keystore.password=test123

#Private key
#The alias from the keystore of the private key
domibus.security.key.private.alias=blue_gw
#The private key password
domibus.security.key.private.password=test123

#Truststore
#The location of the truststore
domibus.security.truststore.location=${domibus.config.location}/keystores/gateway_truststore.jks
#Type of the used truststore
domibus.security.truststore.type=jks
#The password used to load the trustStore
domibus.security.truststore.password=test123
```

1. Create, if not present, a folder *cef_edelivery_path***/conf/domibus/keystores**.
2. Get your key pair from an external provider. (Self-signed certificates should only be used for testing purposes, not production). If you are interested in using the CEF Public Key Infrastructure Solution (cf.[REF5]).
3. Create, if not present, the public and private keys containers (e.g. *truststore***.jks** *and keystore***.jks**).

4. Import your private key into your keystore.

Remarks:

- *Your private key and your keystore should always stay secret. Please never share them.*
- *The keystore alias has to be the same as the party*
- *It is strongly recommended to put your key pair (private and public key) and the public key of the other participants you trust in two separate containers.*

5.2. Domibus Properties

The following properties defined in the property file *cef_edelivery_path***/conf/domibus/domibus.properties** can be used to configure Domibus:

Configuration Property	Default value	Purpose
domibus.database.general.schema	general_schema	Multitenancy only: Schema used by Domibus to configure the association of users to domains, the super users and other things that are not related to a specific domain. This property is mandatory for Multitenancy mode.
domibus.msh.messageid.suffix	domibus.eu	This Property is used to generate the random Message id with a fixed suffix which is set by default to "domibus.eu". The resulting format will be UUID@$domibus.msh.messageid.suffix. This property is mandatory.
domibus.msh.retry.cron	0/5 * * * *	It is the retry cron job to send the messages. It is set by default to every 5 seconds. This property is mandatory
domibus.dispatch.ebms.error.unrecoverable.retry	true	This property should be set to true if Domibus needs to retry sending the failed messages. This property is mandatory
domibus.smlzone	acc.edelivery.tech.ec.europa.eu	Set the SMLZone if Domibus needs to be used under Dynamic discovery model. This property is only mandatory if an SML is used.
domibus.dynamicdiscovery.useDynamicDiscovery	false	Whether dynamic discovery is used or not.
domibus.dynamicdiscovery.client.specification	OASIS	The property specifies the dynamic discovery client to be used for the dynamic process. Possible values: OASIS and PEPPOL.
domibus.dynamicdiscovery.peppolclient.mode	TEST	This information is passed to the PEPPOL client that needs to know whether the usage is for PRODUCTION or TESTING mode.

domibus.dynamicdiscovery.oasisclient.regexCertificateSubjectValidation		Apart from validating response of signer certificates against the truststore, the Oasis Dynamic Discovery Client gives the possibility to add (optional) a regular expression to validate any certificate metadata related to the subject of the signer certificate. Example: domibus.dynamicdiscovery.oasisclient.regexCertificateSubjectValidation="^.*EHEALTH_SMP.*$"
domibus.dynamicdiscovery.partyid.responder.role	http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/responder	The role of the responder PartyId may be defined here for both PEPPOL and OASIS
domibus.dynamicdiscovery.partyid.type	urn:oasis:names:tc:ebcore:partyid-type:unregistered	The type of the PartyId may be defined here (default values are: urn:fdc:peppol.eu:2017:identifiers:ap for PEPPOL and urn:oasis:names:tc:ebcore:partyid-type:unregistered for OASIS)
domibus.dynamicdiscovery.transportprofileas4	bdxr-transport-ebms3-as4-v1p0	The AS4 transport profile by which the endpoint is identified in the SMP response. In PEPPOL the latest value is peppol-transport-as4-v2_0
domibus.jms.queue.pull	domibus.internal.pull.queue	Domibus internal queue used for dispatching the pull requests
domibus.deployment.clustered	false	If true the quartz scheduler jobs are clustered. This property is mandatory, it should be set to true if the deployment of Domibus is done in a cluster.
messageFactoryClass		The factory for creating SOAPMessage objects Default values - Tomcat/WebLogic: com.sun.xml.internal.messaging.saaj.soap.ver1_2.SOAPMessageFactory1_2Impl - WildFly: com.sun.xml.messaging.saaj.soap.ver1_2.SOAPMessageFactory1_2Impl
domibus.dispatcher.allowChunking	true	Allows chunking when sending messages to other Access Points
domibus.dispatcher.chunkingThreshold	104857600	If domibus.dispatcher.allowChunking is true, this property sets the threshold at which messages start getting chunked(in bytes). Messages under this limit do not get chunked. Defaults to 100 MB.

domibus.dispatcher.concurency	5-20	Specify concurrency limits via a "lower-upper" String, e.g. "5-10", or a simple upper limit String, e.g. "10" (the lower limit will be 1 in this case) #when sending messages to other Access Points.
domibus.msh.pull.cron	0 0 0/1 * * ?	Cron expression used for configuring the message puller scheduling.
domibus.pull.queue.concurency	1-1	Number of threads used to parallelize the pull requests.
domibus.pull.request.send.per.job.cycle	1	Number of pull requests executed every cron cycle.
domibus.retentionWorker.cronExpression	0/60 * * * * ?	Cron expression used for configuring the retention worker scheduling. The retention worker deletes the expired messages (downloaded and not-downloaded).
message.retention.downloaded.max.delete	50	This property is used to tweak the maximum downloaded messages to be deleted by the retention worker.
message.retention.not_downloaded.max.delete	50	This property is used to tweak the maximum not-downloaded messages to be deleted by the retention worker.
domibus.attachment.storage.location	-	It is possible to configure Domibus to save the message payloads on the file system instead of the database. This setting is recommended when exchanging payloads bigger than 30MB. In order to enable the file system storage please add the following property: domibus.attachment.storage.location= *your_file_system_location* where *your_file_system_location* is the location on the file system where the payloads will be saved. Remark: In a cluster configuration the file system storage needs to be accessible by all the nodes from the cluster.
domibus.taskExecutor.threadCount	50	Tomcat only: customize the task executor threads count.
domibus.jmx.user	jmsManager	WebLogic specific: the user that will be used to access the queues via JMX.
domibus.jmx.password	jms_Manager1	WebLogic specific: the associated password of the configured domibus.jmx.user.

domibus.sendMessage.messageIdPattern	^[\x20-\x7E]*$	When an initiator backend client submits messages to Domibus for transmission, with the message id field populated, then the message id should be RFC2822 compliant. The pattern specified here ensures this validation. This field is optional. In case the existing client does not match this message id pattern during submission, then this property can be omitted to skip the validation.
domibus.listPendingMessages.maxCount	10000 for Tomcat 500 for WildFly and Weblogic	This property specifies the maximum number of messages that would be served when the 'listPendingMessages' operation is invoked. Setting this property is expected to avoid timeouts due to huge result sets being served. A value of 0 would return all the pending messages. This property is optional. Omitting this property would default the resultset size to 500. Note: For Tomcat server, the maximum number of shown messages in queue monitoring is defined by the 'domibus.listPendingMessages.maxCount' property.
domibus.fourcornermodel.enabled	true	This property affects the GUI search and behaviour. If the property is set to false, 'Final Recipient' and 'Original Sender' criteria disappear from Messages Filter, Messages column picker and from Message details in the GUI. The internal SQL queries for User and Signal Message do not use TB_PROPERTY.NAME = 'finalRecipient' and 'originalSender' anymore.
domibus.dispatcher.connectionTimeout	240000	For connection between the access points – C2 & C3. Specifies the amount of time, in milliseconds, that the consumer will attempt to establish a connection before it times out. 0 is infinite.
domibus.dispatcher.receiveTimeout	240000	For connection between the access points – C2 & C3. Specifies the amount of time, in milliseconds, that the consumer will wait for a response before it times out. 0 is infinite.

domibus.msh.retry.tolerance	10800000	Timeout tolerance for retry messages (in miliseconds). Scheduled retries that, due to any reason, were not performed within this period will be timeout.
domibus.sendMessage.failure.delete.payload	false	Whether to delete the message payload or send failure. Defaults to false (the admin could put the message back in the send queue) .
domibus.auth.unsecureLoginAllowed	true	The property specifies if authentication is required or not.
compressionBlacklist	application/vnd.etsi.asic-s+zip,image/jpeg	The list of mime-types that will not be compressed (in outgoing messages) even if compression is turned on for the given message.
domibus.security.keystore.location	${domibus.config.location}/keystores/gateway_keystore.jks	The location of the keystore.
domibus.security.keystore.type	jks	The type of the used keystore.
domibus.security.keystore.password	test123	The password used to load the keystore.
domibus.security.key.private.alias	blue_gw	The alias from the keystore of the private key.
domibus.security.key.private.password	test123	The private key password.
domibus.security.truststore.location	${domibus.config.location}/keystores/gateway_truststore.jks	The location of the truststore.
domibus.security.truststore.type	jks	The type of the used keystore.
domibus.security.truststore.password	test123	The password used to load the trustStore.
domibus.entityManagerFactory.packagesToScan	eu.domibus	Packages to be scanned (comma separated) by the EntityManagerFactory.
domibus.entityManagerFactory.jpaProperty.hibernate.connection.driver_class		The JDBC driver class used for connecting to the database.
domibus.entityManagerFactory.jpaProperty.hibernate.dialect		This property makes Hibernate generate the appropriate SQL for the chosen database.
domibus.entityManagerFactory.jpaProperty.hibernate.format_sql	true	Pretty print the SQL in the log and console.

domibus.entityManagerFactory.jpaProperty.transaction.factory_class		The classname of a TransactionFactory to use with Hibernate Transaction API.
domibus.entityManagerFactory.jpaProperty.hibernate.transaction.manager_lookup_class		The classname of the TransactionManagerLookup.
com.atomikos.icatch.output_dir	${domibus.work.location:${domibus.config.location}}/work/transactions	Tomcat only: Specifies the directory in which to store the debug log files for Atomikos.
com.atomikos.icatch.log_base_dir	${domibus.work.location:${domibus.config.location}}/work/transactions/log	Tomcat only: Specifies the directory in which the log files should be stored.
com.atomikos.icatch.default_jta_timeout	60000	Tomcat only: The default timeout for JTA transactions.
com.atomikos.icatch.max_timeout	300000	Tomcat only: The default transaction max timeout for JTA transactions
domibus.jms.XAConnectionFactory.maxPoolSize	20	Tomcat only: The max pool size of the JMS connection factory.
activeMQ.broker.host	localhost	Tomcat only: The host of the JMS broker.
activeMQ.brokerName	localhost	Tomcat only: The name of the JMS broker.
activeMQ.embedded.configurationFile	file:///${domibus.config.location}/internal/activemq.xml	Tomcat only: The configuration file of the embedded ActiveMQ broker. In case an external broker is used this property is not needed and it should be deleted from the property file.
activeMQ.JMXURL	service:jmx:rmi://${activeMQ.broker.host}:${activeMQ.rmiServerPort}/jndi/rmi://${activeMQ.broker.host}:${activeMQ.connectorPort}/jmxrmi	Tomcat only: The service URL of the MBeanServer.
activeMQ.connectorPort	1199	Tomcat only: The port that the JMX connector will use for connecting to ActiveMQ.
activeMQ.rmiServerPort	1200	Tomcat only: The RMI server port.
activeMQ.transportConnector.uri	tcp://${activeMQ.broker.host}:61616	Tomcat only: The connection URI that the clients can use to connect to an ActiveMQ broker using a TCP socket.
activeMQ.username	domibus	Tomcat only: The username that is allowed to connect to the ActiveMQ broker.
activeMQ.password	changeit	Tomcat only: The password of the username defined in the **activeMQ.username** property.

domibus.datasource.xa.xaDataSourceClassName	com.mysql.jdbc.jdbc2.optional.MysqlXADataSource	Tomcat only(XA datasource): The fully qualified underlying XADataSource class name.
domibus.datasource.xa.maxLifetime	30	Tomcat only(XA datasource): Sets the maximum amount of seconds that a connection is kept in the pool before it is destroyed automatically.
domibus.datasource.xa.minPoolSize	5	Tomcat only(XA datasource): Sets the minimum pool size. The amount of pooled connections will not go below this value. The pool will open this amount of connections during initialization.
domibus.datasource.xa.maxPoolSize	100	Tomcat only(XA datasource): Sets the maximum pool size. The amount of pooled connections will not go above this value.
domibus.database.serverName	localhost	Tomcat only(XA datasource): The host name or the IP address of the database server.
domibus.database.port	3306	Tomcat only(XA datasource): The port number of the database server.
domibus.datasource.xa.property.user	edelivery	Tomcat only(XA datasource): A user who has access to the Domibus database schema.
domibus.datasource.xa.property.password	edelivery	Tomcat only(XA datasource): The password of the user defined in the **domibus.datasource.xa.property.user** property.
domibus.database.schema	domibus	Tomcat only: the Domibus database schema
domibus.datasource.xa.property.url	jdbc:mysql://${domibus.database.serverName}:${domibus.database.port}/domibus?pinGlobalTxToPhysicalConnection=true	Tomcat only(XA datasource): The JDBC URL connection. It re-uses the properties for the user and password defined above.
domibus.datasource.driverClassName	com.mysql.jdbc.Driver	Tomcat only(Non-XA datasource): the JDBC driver class name.
domibus.datasource.url	jdbc:mysql://localhost:3306/domibus?useSSL=false	Tomcat only(Non-XA datasource): The JDBC URL connection.
domibus.datasource.user	edelivery	Tomcat only(Non-XA datasource): A user who has access to the Domibus database schema.
domibus.datasource.password	edelivery	Tomcat only(Non-XA datasource): The password of the user defined in the **domibus.datasource.user** property.

domibus.receiver.certificate.validation.onsending	true	If activated Domibus will verify before sending a User Message if the receiver's certificate is valid and not revoked. If the receiver's certificate is not valid or it has been revoked Domibus will not send the message and it will mark it as SEND_FAILURE
domibus.sender.certificate.validation.onsending	true	If activated Domibus will verify before sending a User Message if his own certificate is valid and not revoked. If the certificate is not valid or it has been revoked, Domibus will not send the message and it will mark it as SEND_FAILURE (default is true)
domibus.sender.certificate.validation.onreceiving	true	If activated Domibus will verify before receiving a User Message if the sender's certificate is valid and not revoked. If the certificate is not valid or it has been revoked, Domibus will not accept the message (default is true)
domibus.sender.trust.validation.onreceiving	false	An extra security validation that requires that the party name reflected in the alias of the sender public key should also be contained in the subject of the certificate.
domibus.console.login.maximum.attempt	5	Maximum connection attempts before the account gets locked (suspended).
domibus.console.login.suspension.time	60	Property defining how many minutes the account remains locked (suspended) before it is automatically unlocked by the system.
domibus.UI.title.name=Domibus	Domibus	Property where you can specify the title in the Tab of Admin Console
domibus.jms.queue.ui.replication	domibus.internal.ui.replication.queue (Tomcat) or jms/ domibus.internal.ui.replication.queue for Weblogic/Wildfly	JNDI name of the UI replication queue. Value is different per type of server
domibus.ui.replication.sync.cron	0 0 2 * * ? *	Cron job (run every day at 2 AM) to check any unsynchronized data between native tables and TB_MESSAGE_UI table
domibus.ui.replication.sync.cron.max.rows	10000	Max number of rows which will be processed by the UI replication cron job – otherwise the REST resource must be called

Configuration Property	Default value	Purpose
Proxy Settings		In case your Access Point has to use a proxy server you can configure it with these properties.
domibus.proxy.enabled	false	true/false depending on whether you need to use proxy or not.
domibus.proxy.http.host	-	Host name of the proxy server.
domibus.proxy.http.port	-	Port of Proxy server
domibus.proxy.user	-	Username for authentication on the proxy server.
domibus.proxy.password	-	Password.
domibus.proxy.nonProxyHosts	-	Indicates the hosts that should be accessed without going through the proxy.
domibus.alert.sender.smtp.url		Smtp server url for sending alert.
domibus.alert.sender.smtp.port		Smtp server port.
domibus.alert.sender.smtp.user		Smtp server user.
domibus.alert.sender.smtp.password		Smtp server user password.
domibus.alert.sender.email		Alert sender email.
domibus.alert.receiver.email		Alert email receiver.
domibus.alert.cleaner.cron	0 0 0/1 * * ?	Cron configuration for cleaning alerts.
domibus.alert.cleaner.alert.lifetime	20	Lifetime in days of alerts before cleaning.
domibus.alert.active	TRUE	Enable/disable the entire alert module.
domibus.alert.mail.sending.active	FALSE	Allow to disable alert mail sending.
domibus.alert.mail.smtp.timeout	5000	SMTP Socket I/O timeout value in milliseconds
domibus.alert.queue.concurrency	1	Concurency to process the alerts.

domibus.alert.retry.cron	0 0/1 * * * ?	Frequency of failed alerts retry.
domibus.alert.retry.time	1	Elapsed time in minutes between alert retry.
domibus.alert.retry.max_attempts	2	Maximum number of attempts for failed alerts+
domibus.alert.msg.communication_failure.active	TRUE	Enable/disable the messaging alert module.
domibus.alert.msg.communication_failure.states	SEND_FAILURE	Message status change that should be notified by the messaging alert module. Comma separated.
domibus.alert.msg.communication_failure.level	HIGH	Alert levels corresponding to message status defined in previous property(domibus.alert.msg.communication_failure.states) .Should be (HIGH, MEDIUM OR LOW)
domibus.alert.msg.communication_failure.mail.subject	Message status change	Messaging alert module mail subject.
domibus.alert.user.login_failure.active	TRUE	Enable/disable the login failure alert of the authentication module.
domibus.alert.user.login_failure.level	LOW	Alert level for login failure.
domibus.alert.user.login_failure.mail.subject	Login failure	Login failure mail subject.
domibus.alert.user.account_disabled.active	TRUE	Enable/disable the account disable alert of the authentication module.
domibus.alert.user.account_disabled.level	HIGH	Alert level for account disabled.
domibus.alert.user.account_disabled.moment	WHEN_BLOCKED	When should the account disabled alert be triggered. 2 possible values: AT_LOGON: An alert will be triggered each a time user tries to login to a disabled account.
domibus.alert.user.account_disabled.subject	Account disabled	Account disabled mail subject.
domibus.alert.cert.imminent_expiration.active	TRUE	Enable/disable the imminent certificate expiration alert of certificate scanner module.

domibus.alert.cert.imminent_expiration.frequency_days	14	Frequency in days between alerts.
domibus.alert.cert.imminent_expiration.level	HIGH	Certificate imminent expiration alert level.
domibus.alert.cert.imminent_expiration.mail.subject	Certificate imminent expiration	Certificate imminent expiration mail subject.
domibus.alert.cert.expired.active	TRUE	Enable/disable the certificate expired alert of certificate scanner module.
domibus.alert.cert.expired.frequency_days	7	Frequency in days between alerts.
domibus.alert.cert.expired.duration_days	90	How long(in days) after the revocation should the system trigger alert for the expired certificate.
domibus.alert.cert.expired.level	HIGH	Certificate expired alert level.
domibus.alert.cert.expired.mail.subject	Certificate expired	Certificate expired mail subject.
domibus.alert.super.cleaner.cron	0 0 0/1 * * ?	Cron configuration for cleaning super user alerts.
domibus.alert.super.cleaner.alert.lifetime	20	Lifetime in days of super user alerts.
domibus.alert.super.active	true	Enable/disable the super user alert module.
domibus.alert.super.mail.sending.active	false	Enable/disable the super user alert mail sending.
domibus.alert.super.retry.cron	0 0/1 * * * ?	Frequency of failed super user alert retry.
domibus.alert.super.retry.time	1	Elapsed time in minutes between super user alert retry.
domibus.alert.super.retry.max_attempts	2	Maximum number of attempts for failed super user alert
domibus.alert.super.user.login_failure.active	true	Enable/disable the login failure super user alert of the authentication module.
domibus.alert.super.user.login_failure.level	LOW	Super user alert level for login failure.
domibus.alert.super.user.login_failure.mail.subject	Super user login failure	Super user login failure alert mail subject.

domibus.alert.super.user.account_disabled.active	true	Enable/disable the account disabled super user alert of the authentication module.
domibus.alert.super.user.account_disabled.level	HIGH	Super user alert level for account disabled.
domibus.alert.super.user.account_disabled.moment	WHEN_BLOCKED	#When should the account disabled super user alert be triggered. # 2 possible values: # AT_LOGON: An alert will be triggered each a time user tries to login to a disabled account. # WHEN_BLOCKED: An alert will be triggered once when the account got disabled.
domibus.alert.super.user.account_disabled.subject	Super user account disabled	Super user account disabled alert mail subject.

Table 1 - Domibus Properties

6. PLUGIN MANAGEMENT

This section describes the different types of plugins and their registration process.

6.1. Default Plugins

Domibus comes with three default plugins. The three Interface Control Documents (ICD) describe these three plugins (JMS, WS and File System Plugin) (cf.[REF6]).

6.1.1. JMS Plugin

For the JMS plugin, you will have to use the following resources (see section § 3.1-" *Binaries repository* " for the download location):

- **domibus-distribution-X.Y.Z-default-jms-plugin.zip**

6.1.2. WS Plugin

For the WS plugin, you will have to use the following resources (see section §3.1-"*Binaries repository*" for the download location):

- **domibus-distribution-X.Y.Z-default-ws-plugin.zip**

6.1.3. File System Plugin

For the File System plugin, you will have to use the following resources (see section §3.1-"*Binaries repository*" for the download location):

- **domibus-distribution-X.Y.Z-default-fs-plugin.zip**

6.2. Custom Plugin

Users can develop their own plugins. Please refer to the plugin cookbook for more details (cf.[REF6]).

6.2.1. Plugin registration

***Remark**:*

> *Please refer to section* 9.3 “Message Log” "" *for the routing of the specific plugin after registering the plugin on your specific Application Server.*

6.2.1.1. Tomcat

In order to install a custom plugin for Tomcat, please follow the steps below**:**

1. Stop Tomcat server
2. Copy the custom plugin jar file to the plugins folder *CATALINA_HOME***/conf/domibus/plugins/lib**
3. Copy the custom plugin XML configuration file to *CATALINA_HOME***/conf/domibus/plugins/config**

4. Start Tomcat server

Remark:

CATALINA_HOME is the folder where the Tomcat is installed.

6.2.1.2. *WebLogic*

In order to install a custom plugin for WebLogic please follow the steps below**:**

1. Stop the WebLogic server
2. Copy the custom plugin jar file to the plugins folder *DOMAIN_HOME***/conf/domibus/plugins/lib**
3. Copy the custom plugin XML configuration file to *DOMAIN_HOME***/conf/domibus/plugins/config**
4. Start the WebLogic server

Remark:

DOMAIN_HOME is the folder corresponding to the WebLogic domain.

6.2.1.3. WildFly

In order to install a custom plugin please follow the steps below**:**

1. Stop the WildFly server
2. Copy the custom plugin jar file to the plugins folder *cef_edelivery_path* **/conf/domibus/plugins/lib**
3. Copy the custom plugin XML configuration file to *cef_edelivery_path* **/conf/domibus/plugins/config**
4. Start the WildFly server

6.3. Plugin authentication

The plugins authentication is disabled by default for the default plugins. In order to enable the plugin authentication for the default plugins in Domibus the following steps must be followed:

1. Set the property "**domibus.auth.unsecureLoginAllowed**" to false in **domibus.properties**:

```
domibus.auth.unsecureLoginAllowed=false
```

2. Configure the application server to allow http(s) requests and pass the authentication credentials to Domibus.

6.4. Plugin notifications

Domibus core notifies the plugins on different events. The types of events are:

MESSAGE_RECEIVED, ***MESSAGE_SEND_FAILURE***, ***MESSAGE_RECEIVED_FAILURE***, ***MESSAGE_SEND_SUCCESS***, ***MESSAGE_STATUS_CHANGE***

For each plugin, it is possible in the configuration file (*-plugin.xml) to specify the list of events for which it requires notifications. This list is optional and passed as a constructor argument to the NotificationListener bean.

Example:

```
<util:list id="requiredNotificationsList" value-type="eu.domibus.common.NotificationType">
    <value>MESSAGE_RECEIVED</value>
    <value>MESSAGE_SEND_FAILURE</value>
    <value>MESSAGE_STATUS_CHANGE</value>
</util:list>
<bean id="webserviceNotificationListenerService"
      class="eu.domibus.plugin.NotificationListenerService"
      c:queue-ref="notifyBackendWebServiceQueue" c:mode="PULL"
      p:backendConnector-ref="backendWebservice"/>
      p:backendConnector-ref="backendWebservice">
    <constructor-arg ref="requiredNotificationsList"/>
</bean>
```

This list is optional. By default, PULL plugins receive notifications **for MESSAGE_RECEIVED, MESSAGE_SEND_FAILURE, MESSAGE_RECEIVED_FAILURE** while the **PUSH** plugins receive notification for all events.

7. PMode Configuration

Processing Modes (PModes) are used to configure Access Points. The PMode parameters are loaded into the Access Point via an XML file.

The features described in the PMode file are: Security, Reliability, Transport, Business Collaborations, Error Reporting, Message Exchange Patterns (MEPs) and Message Partition Channels (MPCs).

As different messages may be subject to various types of processing or, as different business domains may have several requirements, Access Points commonly support several PModes. Some PMode parameters are mandatory, others are optional. For more information, please refer to the Access Point Component Offering Document.

7.1. Configuration

In Domibus, PModes are XML files that you can create or edit. You can configure the two files given: *cef_edelivery_path***/conf/pmodes/domibus-gw-sample-pmode-***party_id_name1***.xml** and *cef_edelivery_path***/conf/pmodes/domibus-gw-sample-pmode-***party_id_name2***.xml.**

The "*party_id_name1" value must be replaced* with your own party name and *the "party_id_name2"* with your corresponding party name.

The party_id must match the alias of the certificate in the keystore and the endpoint must be the external access link to your instance.

***Remark**:*

This step could be managed by a PMode Configuration Manager, known to your Business Owner.

```
<party name="party_id_name2"
          endpoint="http:// party_id_name2_hostname:8080/domibus/services/msh">
   <identifier partyId="party_id_name2_1"
                    partyIdType="partyTypeUrn"/>
</party>
<party name="party_id_name1"
          endpoint="http:// party_id_name1_hostname:8080/domibus/services/msh">
   <identifier partyId="party_id_name1_1" partyIdType="partyTypeUrn"/>
</party>
```

Figure 5 - PMode view

7.1.1. Adding a new participant

If a new participant's Access Point is joining your network, you need to configure your PMode accordingly and re-upload it like mentioned in §7.1.4 – *"Upload new Configuration"*.

- Add a "new_party" element:

```
<party name="new_party_name"
                        endpoint="http://new_party_msh" >
        <identifier partyId="new_party_id" partyIdType="partyTypeUrn"/>
</party>
```

- Add your "new_party_name" as initiator:

The party with the role of initiator will be the sender of the messages:

```
<initiatorParties>
                ...
                <initiatorParty name="new_party_name"/>
        </initiatorParties>
```

- Add your "new_party_name" as responder:

The party with the role of responder will be the receiver of the messages:

```
<responderParties>
                ...
                <responderParty name="new_party_name"/>
        </responderParties>
```

7.1.2. Sample PMode file

Processing modes (PModes) describe how messages are exchanged between AS4 partners (in this case *Access Points blue_gw and red_gw*). These files contain the identifiers of each AS4 Access Point (identified as *parties* in the PMode file below).

Sender and Receiver Identifiers represent the organizations that send and receive the business documents. They are both used in the authorization process (PMode). Therefore, adding, modifying or deleting a participant implies modifying the corresponding PMode files.

Here is an example of a PMode XML file:

Remark:

In this setup, we have allowed each party (blue_gw or red_gw) to initiate the process. If only blue_gw is supposed to send messages, then put only blue_gw in <initiatorParties> and red_gw in <responderParties>.

```
<?xml version="1.0" encoding="UTF-8"?>
<db:configuration xmlns:db="http://domibus.eu/configuration" party="blue_gw">

        <mpcs>
                <mpc name="defaultMpc"
                        qualifiedName="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC"
                        enabled="true"
                        default="true"
                        retention_downloaded="0"
                        retention_undownloaded="14400"/>
        </mpcs>
        <businessProcesses>
                <roles>
```

```
                <role name="defaultInitiatorRole"
                        value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/initiator"/>
                <role name="defaultResponderRole"
                        value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/responder"/>
        </roles>
        <parties>
                <partyIdTypes>
                        <partyIdType name="partyTypeUrn"
value="urn:oasis:names:tc:ebcore:partyid-type:unregistered"/>
                </partyIdTypes>
                <party name="red_gw"
                        endpoint="http://<red_hostname>:8080/domibus/services/msh">
                        <identifier partyId="domibus-red" partyIdType="partyTypeUrn"/>
                </party>
                <party name="blue_gw"

        endpoint="http://<blue_hostname>:8080/domibus/services/msh">
                        <identifier partyId="domibus-blue" partyIdType="partyTypeUrn"/>
                </party>
        </parties>
        <meps>
                <mep name="oneway" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/oneWay"/>
                <mep name="twoway" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/twoWay"/>
                <binding name="push" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/push"/>
                <binding name="pull" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/pull"/>
                <binding name="pushAndPush" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/push-and-push"/>
        </meps>
        <properties>
                <property name="originalSenderProperty"
                        key="originalSender"
                        datatype="string"
                        required="true"/>
                <property name="finalRecipientProperty"
                        key="finalRecipient"
                        datatype="string"
                        required="true"/>
                <propertySet name="eDeliveryPropertySet">
                        <propertyRef property="finalRecipientProperty"/>
                        <propertyRef property="originalSenderProperty"/>
                </propertySet>
        </properties>
        <payloadProfiles>
                <payload name="businessContentPayload"
                        cid="cid:message"
                        required="true"
                        mimeType="text/xml"/>
```

```
                <payload name="businessContentAttachment"
                        cid="cid:attachment"
                        required="false"
                        mimeType="application/octet-stream"/>
                <payloadProfile name="MessageProfile" maxSize="40894464"> <!-- maxSize is
currently ignored -->
                    <attachment name="businessContentPayload"/>
                    <attachment name="businessContentAttachment"/>
                </payloadProfile>
            </payloadProfiles>
            <securities>
                <security name="eDeliveryAS4Policy"
                        policy="eDeliveryAS4Policy.xml"
                        signatureMethod="RSA_SHA256" />
            </securities>
            <errorHandlings>
                <errorHandling name="demoErrorHandling"
                        errorAsResponse="true"
                        businessErrorNotifyProducer="true"
                        businessErrorNotifyConsumer="true"
                        deliveryFailureNotifyProducer="true"/>
            </errorHandlings>
            <agreements>
                <agreement name="agreement1" value="A1" type="T1"/>
            </agreements>
            <services>
                <service name="testService1" value="bdx:noprocess" type="tc1"/>
                <service name="testService" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/service"/>
            </services>
            <actions>
                <action name="tc1Action" value="TC1Leg1"/>
                <action name="testAction" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/test"/>
            </actions>
            <as4>
                <receptionAwareness name="receptionAwareness" retry="12;4;CONSTANT"
duplicateDetection="true"/>
                <reliability name="AS4Reliability" nonRepudiation="true"
replyPattern="response"/>
            </as4>
            <legConfigurations>
                <legConfiguration name="pushTestcase1tc1Action"
                        service="testService1"
                        action="tc1Action"
                        defaultMpc="defaultMpc"
                        reliability="AS4Reliability"
                        security="eDeliveryAS4Policy"
                        receptionAwareness="receptionAwareness"
                        propertySet="eDeliveryPropertySet"
                        payloadProfile="MessageProfile"
                        errorHandling="demoErrorHandling"
                        compressPayloads="true"/>
```

```
                        <legConfiguration name="testServiceCase"
                                                service="testService"
                                                action="testAction"
                                                defaultMpc="defaultMpc"
                                                reliability="AS4Reliability"
                                                security="eDeliveryAS4Policy"
                                                receptionAwareness="receptionAwareness"
                                                propertySet="eDeliveryPropertySet"
                                                payloadProfile="MessageProfile"
                                                errorHandling="demoErrorHandling"
                                                compressPayloads="true"/>
                </legConfigurations>
<process name="tc1Process"
      mep="oneway"
      binding="push"
      initiatorRole="defaultInitiatorRole"
      responderRole="defaultResponderRole">
         <initiatorParties>
                <initiatorParty name="blue_gw"/>
                <initiatorParty name="red_gw"/>
         </initiatorParties>
         <responderParties>
                <responderParty name="blue_gw"/>
                <responderParty name="red_gw"/>
         </responderParties>
         <legs>
                <leg name="pushTestcase1tc1Action"/>
                <leg name="testServiceCase"/>
         </legs>
</process>
         </businessProcesses>
</db:configuration>
```

7.1.3. Domibus PMode configuration to ebMS3 PMode Mapping

The following table provides additional information concerning the Domibus PMode configuration files.

Domibus PMode Configuration	EbMS3 Specification [ebMS3CORE] [AS4-Profile]	Description
MPCs	-	Container which defines the different MPCs (Message Partition Channels).
MPC	PMode[1].BusinessInfo.MPC: The value of this parameter is the identifier of the MPC (Message Partition Channel) to which the message is assigned. It maps to the attribute **Messaging** / **UserMessage**	Message Partition Channel allows the partition of the flow of messages from a *Sending MSH* to a *Receiving MSH* into several flows, each of which is controlled separately. An MPC also allows merging flows from several *Sending MSHs* into a unique flow that will be treated as such by a *Receiving MSH*. The value of this parameter is the identifier of the MPC to which the message is assigned.
MessageRetentionDownloaded	-	Retention interval for messages already delivered to the backend.
MessageRetentionUnDownloaded	-	Retention interval for messages not yet delivered to the backend.
Parties	-	Container which defines the different PartyIdTypes, Party and Endpoint.
PartyIdTypes	maps to the attribute **Messaging/UserMessage/ PartyInfo**	Message Unit bundling happens when the Messaging element contains multiple child elements or Units (either User Message Units or Signal Message Units).
Party ID	maps to the element **Messaging/UserMessage/ PartyInfo**	The ebCore Party ID type can simply be used as an identifier format and therefore as a convention for values to be used in configuration and – as such – does not require any specific solution building block.

Endpoint	maps to **PMode[1].Protocol.Address**	The endpoint is a party attribute that contains the link to the MSH. The value of this parameter represents the address (endpoint URL) of the *Receiver MSH* (or *Receiver Party*) to which Messages under this PMode leg are to be sent. Note that a URL generally determines the transport protocol (e.g. if the endpoint is an email address, then the transport protocol must be SMTP; if the address scheme is "http", then the transport protocol must be HTTP).
AS4	-	Container.
Reliability [@Nonrepudiation] [@ReplyPattern]	Nonrepudiation maps to **PMode[1].Security.SendReceipt.NonRepudiation** ReplyPattern maps to **PMode[1].Security.SendReceipt.ReplyPattern**	PMode[1].Security.SendReceipt.NonRepudiation : value = 'true' (to be used for non-repudiation of receipt), value = 'false' (to be used simply for reception awareness). PMode[1].Security.SendReceipt.ReplyPattern: value = 'Response' (sending receipts on the HTTP response or back-channel). PMode[1].Security.SendReceipt.ReplyPattern: value = 'Callback' (sending receipts use a separate connection).
ReceptionAwareness [@retryTimeout] [@retryCount] [@strategy] [@duplicateDetection]	retryTimeout maps to **PMode[1].ReceptionAwareness.Retry=true** PMode[1].ReceptionAwareness.Retry.Parameters retryCount maps to **PMode[1].ReceptionAwareness.Retry.Parameters** strategy maps to **PMode[1].ReceptionAwareness.Retry.Parameters** duplicateDetection maps to **PMode[1].ReceptionAwareness.DuplicateDetection**	These parameters are stored in a composite string. • *retryTimeout* defines timeout in minutes. • *retryCount* is the total number of retries. • *strategy* defines the frequency of retries. The only *strategy* available as of now is *CONSTANT*. • *duplicateDetection* allows to check duplicates when receiving twice the same message. The only *duplicateDetection* available as of now is *TRUE*.
Securities	-	Container.
Security	-	Container.
Policy	PMode[1].Security.* NOT including PMode[1].Security.X509.Signature.Algorithm	The parameter defines the name of a WS-SecurityPolicy file.
SignatureMethod	PMode[1].Security.X509.Signature.Algorithm	This parameter is not supported by WS-SecurityPolicy and therefore it is defined separately.
BusinessProcessConfiguration	-	Container.

Agreements	maps to eb:Messaging/ UserMessage/ CollaborationInfo/ AgreementRef	This OPTIONAL element occurs zero times or once. The *AgreementRef* element is a string that identifies the entity or artifact governing the exchange of messages between the parties.
Actions	-	Container.
Action	maps to **Messaging/ UserMessage/ CollaborationInfo/Action**	This REQUIRED element occurs once. The element is a string identifying an operation or an activity within a Service that may support several of these
Services	-	Container.
ServiceTypes Type	maps to **Messaging/ UserMessage/ CollaborationInfo/ Service[@type]**	This REQUIRED element occurs once. It is a string identifying the service that acts on the message and it is specified by the designer of the service.
MEP [@Legs]	-	An ebMS MEP defines a typical choreography of ebMS User Messages which are all related through the use of the referencing feature (RefToMessageId). Each message of an MEP Access Point refers to a previous message of the same Access Point, unless it is the first one to occur. Messages are associated with a label (e.g. *request*, *reply*) that precisely identifies their direction between the parties involved and their role in the choreography.
Bindings	-	Container.
Binding	-	The previous definition of ebMS MEP is quite abstract and ignores any binding consideration to the transport protocol. This is intentional, so that application level MEPs can be mapped to ebMS MEPs independently from the transport protocol to be used.
Roles	-	Container.

Role	Maps to **PMode.Initiator.Role or PMode.Responder.Role** depending on where this is used. In ebMS3 message this defines the content of the following element: • For Initiator: **Messaging/UserMessage/PartyInfo/From/Role** • For Responder: **Messaging/UserMessage/PartyInfo/To/Role**	The required role element occurs once, and identifies the authorized role (*fromAuthorizedRole* or *toAuthorizedRole*) of the Party sending the message (when present as a child of the *From* element), or receiving the message (when present as a child of the *To* element). The value of the role element is a non-empty string, with a default value of *http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultRole* Other possible values are subject to partner agreement.
Processes	-	Container.
PayloadProfiles	-	Container.
Payloads	-	Container.
Payload	maps to **PMode[1].BusinessInfo.PayloadProfile**	This parameter allows specifying some constraint or profile on the payload. It specifies a list of payload parts. A payload part is a data structure that consists of five properties: 1. **name** (or Content-ID) that is the **part identifier**, and can be used as an index in the notation PayloadProfile; 2. **MIME data type** (text/xml, application/pdf, etc.); 3. **name of the applicable XML Schema file** if the MIME data type is text/xml; 4. **maximum size in kilobytes; (currently not used)** 5. **Boolean** string indicating whether the part is **expected** or **optional**, within the User message. The message payload(s) must match this profile.
ErrorHandlings	-	Container.
ErrorHandling	-	Container.

ErrorAsResponse	maps to **PMode[1].ErrorHandling.Report.AsResponse**	This Boolean parameter indicates (if *true*) that errors generated from receiving a message in error are sent over the back-channel of the underlying protocol associated with the message in error. If *false*, such errors are not sent over the back-channel.
ProcessErrorNotifyProducer	maps to **PMode[1].ErrorHandling.Report.ProcessErrorNotifyProducer**	This Boolean parameter indicates whether (if *true*) the Producer (application/party) of a User Message matching this PMode should be notified when an error occurs in the Sending MSH, during processing of the *User Message to be sent*.
ProcessErrorNotifyConsumer	maps to **PMode[1].ErrorHandling.Report.ProcessErrorNotifyProducer**	This Boolean parameter indicates whether (if *true*) the Consumer (application/party) of a User Message matching this PMode should be notified when an error occurs in the Receiving MSH, during processing of the *received User message*.
DeliveryFailureNotifyProducer	maps to **PMode[1].ErrorHandling.Report.DeliveryFailuresNotifyProducer**	When sending a message with this reliability requirement (*Submit* invocation), one of the two following outcomes shall occur: - The Receiving MSH successfully delivers (*Deliver* invocation) the message to the Consumer. - The Sending MSH notifies (*Notify* invocation) the Producer of a delivery failure.
Legs	-	Container.

Leg	-	Because messages in the same MEP may be subject to different requirements - e.g. the reliability, security and error reporting of a response may not be the same as for a request – the PMode will be divided into *legs*. Each user message label in an ebMS MEP is associated with a PMode leg. Each PMode leg has a full set of parameters for the six categories above (except for *General Parameters*), even though in many cases parameters will have the same value across the MEP legs. Signal messages that implement transport channel bindings (such as PullRequest) are also controlled by the same categories of parameters, except for *BusinessInfo group*.
Process	-	In *Process* everything is plugged together.

Table 2 - Domibus PMode configuration to ebMS3 mapping

7.1.4. Upload new Configuration

7.1.4.1. Upload the PMode file

Remark:

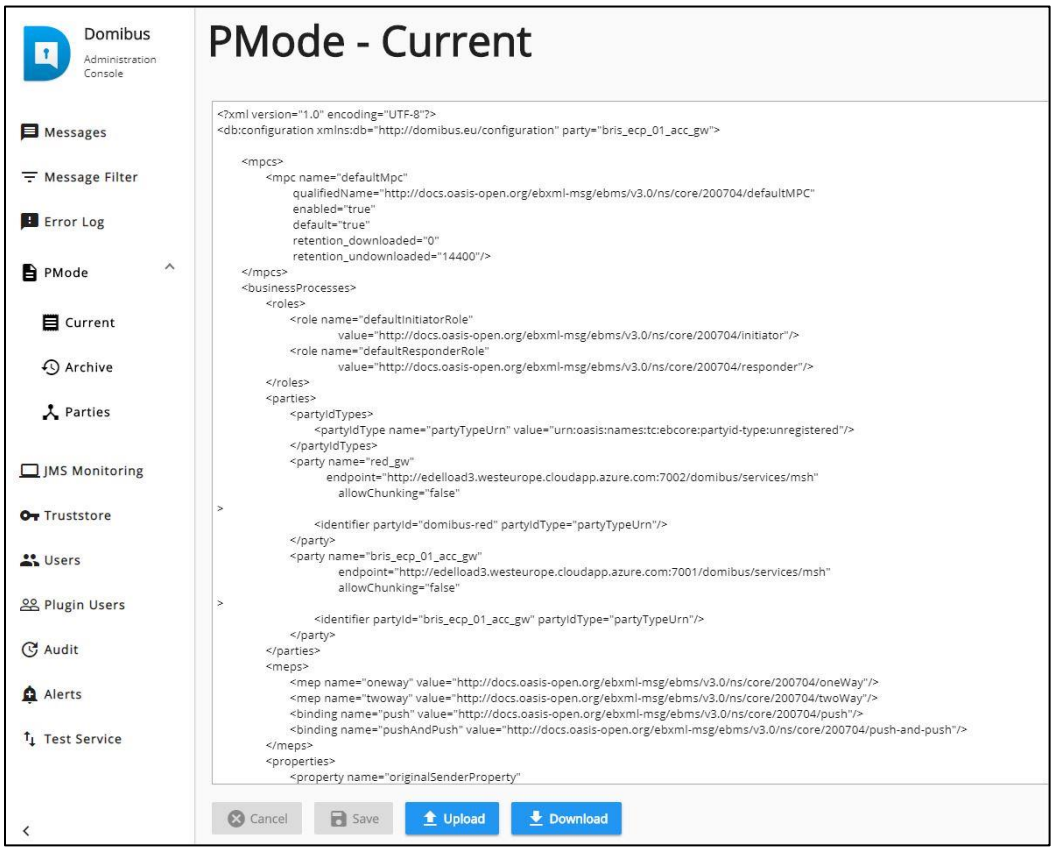
In case of a cluster environment, the PMode configuration is replicated automatically on all the nodes.

1. To update the PMode configuration and/or Truststore, connect to the Administration Console using the administrator's credentials (by default: User = ***admin***; Password = ***123456***) to http://*localhost*:*8080*/domibus.

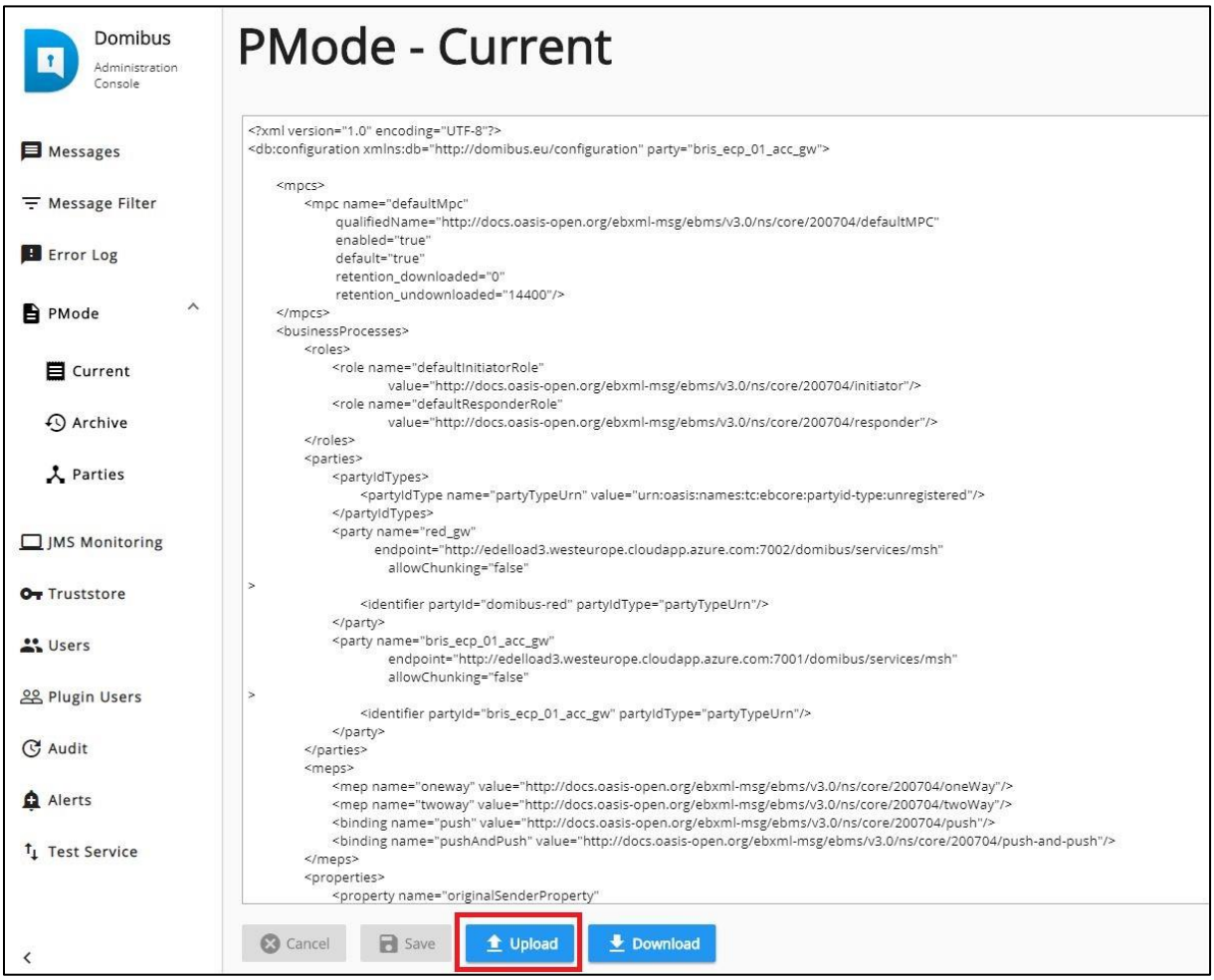
Remark*:*

It is recommended to change the passwords for the default users.
See §9.1 – "Administration " for further information.

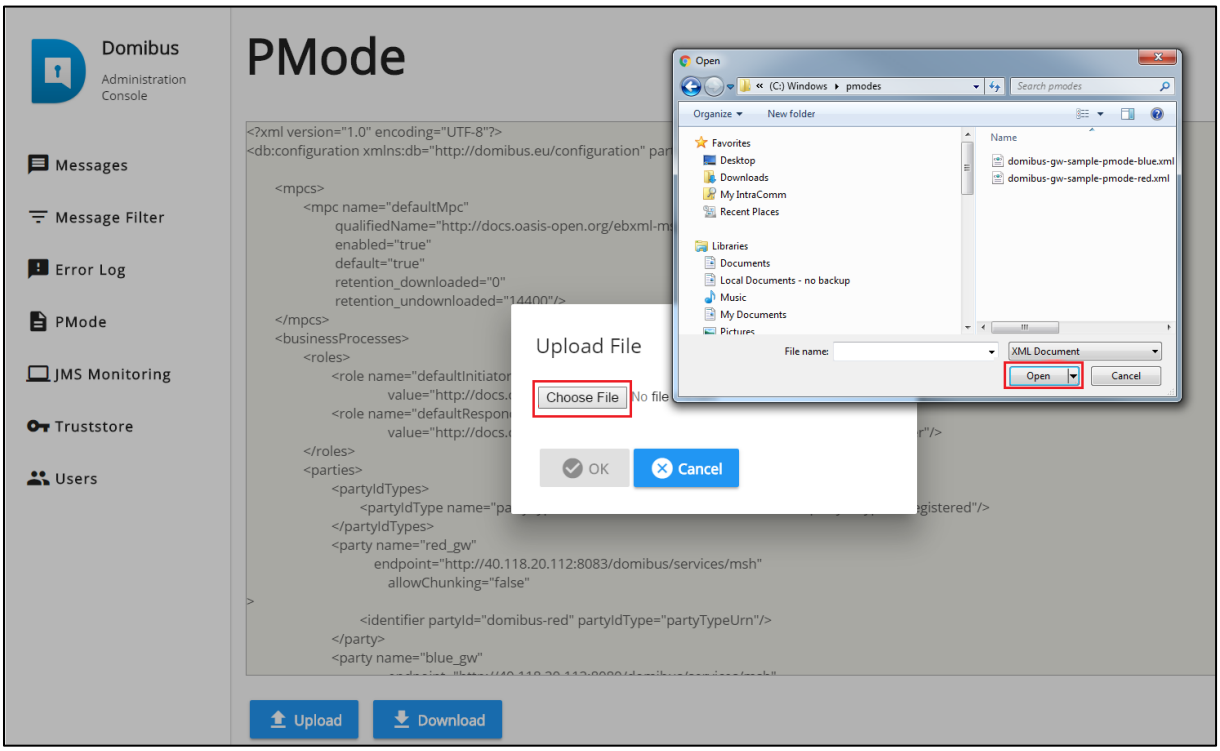
2. Click on the **PMode menu**:



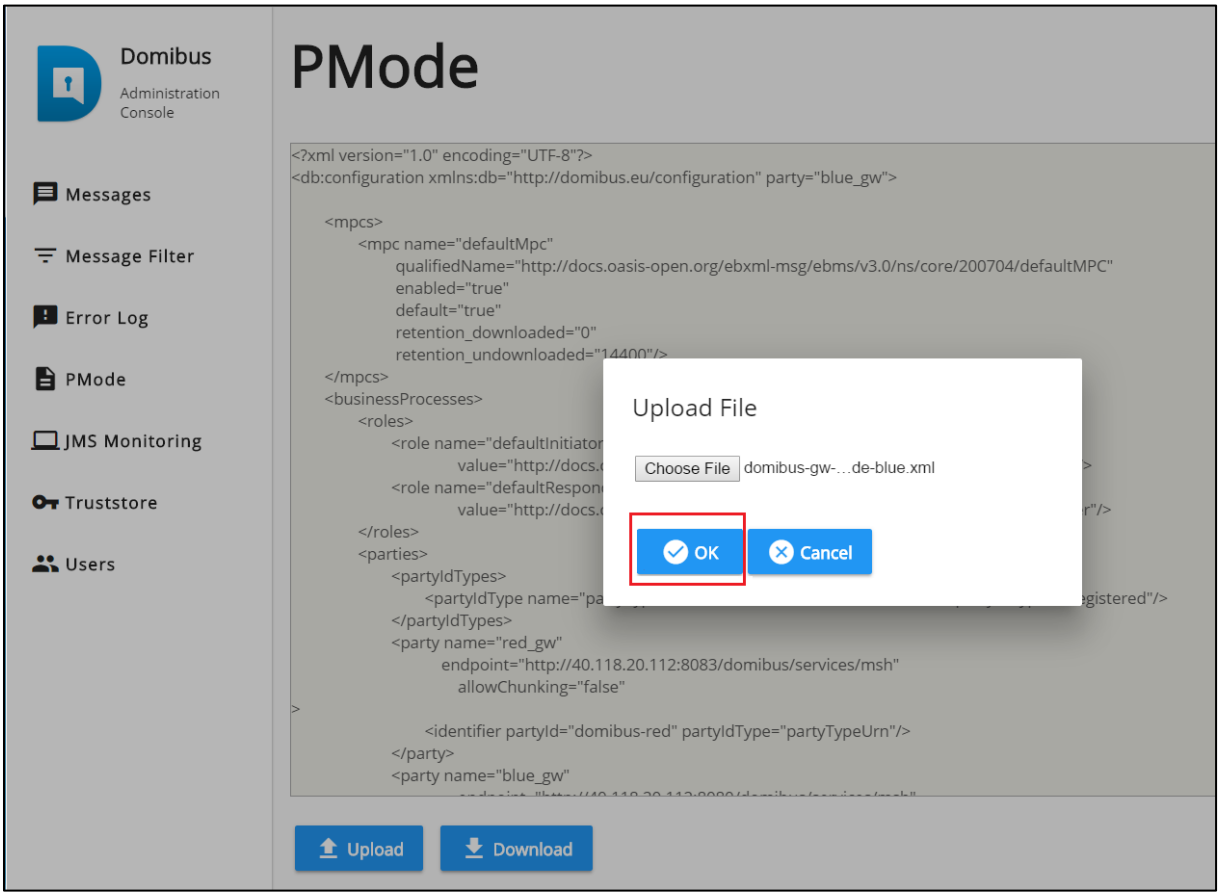
3. Press the **Upload** button:



4. Press the **Choose File** button, and navigate to the PMode file, select it and click on the **Open** button (or equivalent) in the standard dialog box:



5. Once the file has been selected, click "**OK**" to upload the PMode xml file:

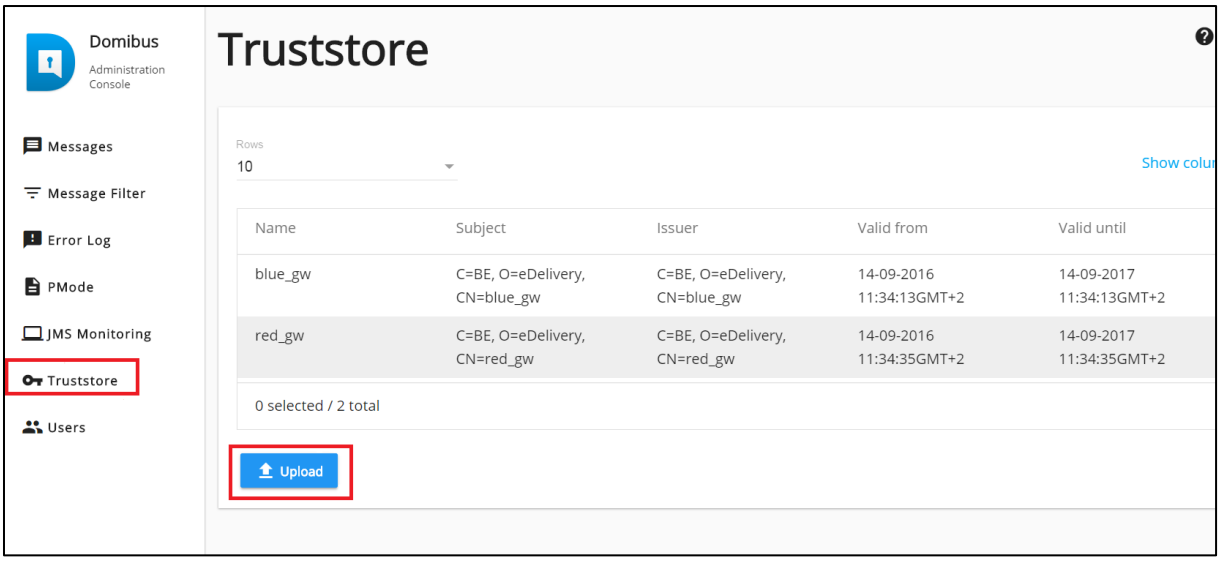


Remark:

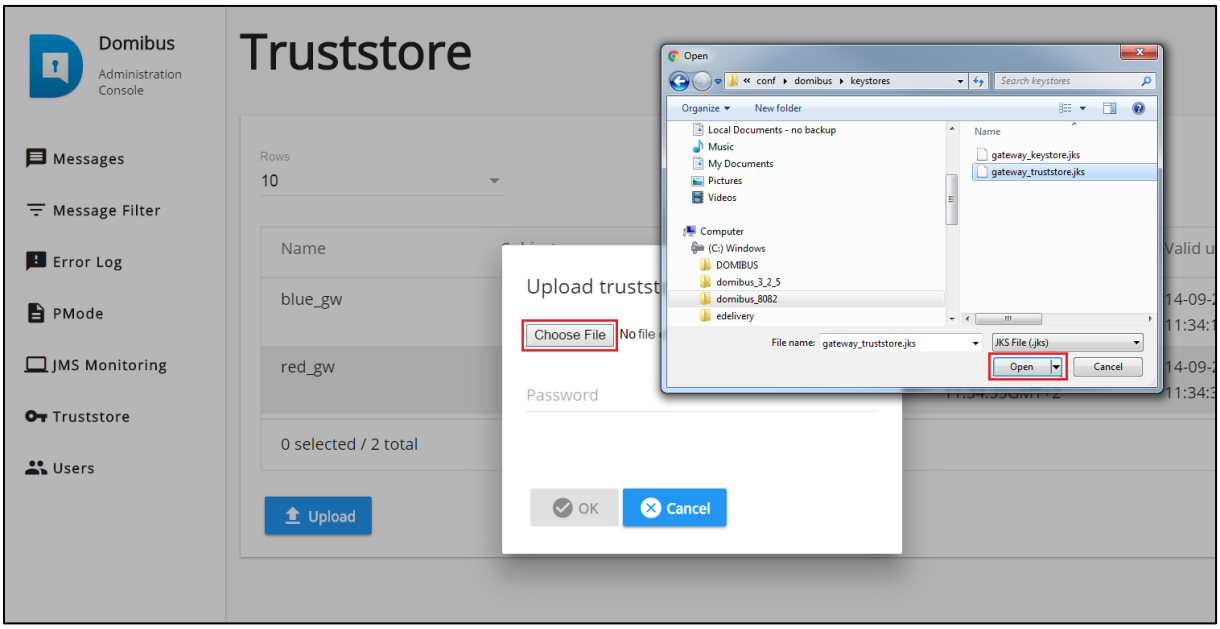
Every time a PMode is updated, the truststore is also reloaded from the filesystem.

7.1.4.2. Upload the Truststore

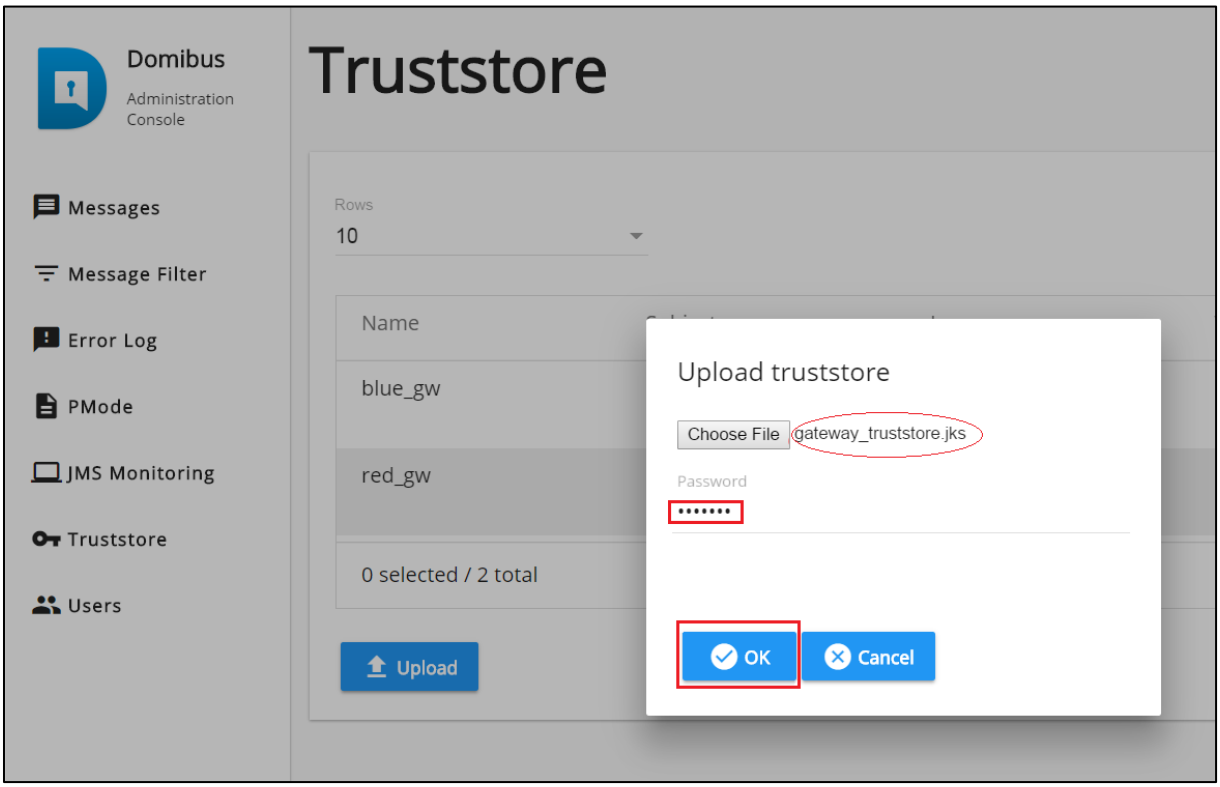
1. Select the "Truststore" menu and press the **Upload** button:



2. Navigate to the Truststore and select it by clicking on the **Open** button (or equivalent) of the standard file open dialog:



3. Once the file has been selected, enter the keystore password and click on the **OK** button to activate the new **truststore jks file:**


Domibus
Administration Console
Messages
Message Filter
Error Log
PMode
JMS Monitoring
Truststore
Users
Truststore
Rows
10
Name
blue_gw
red_gw
0 selected / 2 total
Upload
Upload truststore
Choose File
gateway_truststore.jks
Password
OK
Cancel

8. Special Scenario: Sender and Receiver are the same

In this special scenario, the Sender Access Point acts also as the Receiver Access Point. Multiple backends can exchange messages via the same Access Point using the same or different plugins.

8.1. PMode Configuration

A party (e.g. **blue_gw**) which is Sender and Receiver must be defined in both the <initiatorParties> and <responderParties> sections as shown below:

```
.....
        <initiatorParties>
                ……
                <initiatorParty name="blue_gw"/>
                ……
        </initiatorParties>
        <responderParties>
                ……
                <responderParty name="blue_gw"/>
                ……
        </responderParties>
.....
```

8.2. Message structure

A message that is sent to the same Access Point will have to contain the same party id in both **From** and **To** sections. Below there is an example of a message sent using the Default WS Plugin:

```
<ns:UserMessage>
…
 <ns:PartyInfo>
  <ns:From>
   <ns:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">domibus-blue</ns:PartyId>
   <ns:Role>http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/initiator</ns:Role>
  </ns:From>
  <ns:To>
   <ns:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">domibus-blue</ns:PartyId>
   <ns:Role>http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/responder</ns:Role>
  </ns:To>
</ns:PartyInfo>
….
```

8.3. Message ID convention

Due to some limitations related to the uniqueness of the message identifier, a convention has been defined in this scenario. The message ID used for the received message is derived from the message ID used for the sent message with the following rule: the suffix ***"_1"*** is added to the sent message id.

Example:
sent message ID is **ae15851e-78fb-4b51-aac8-333c08c450d6@domibus**
received message ID is **ae15851e-78fb-4b51-aac8-333c08c450d6@domibus_1**

9. ADMINISTRATION TOOLS

9.1. Administration Console

Domibus administration console can be used by adminsistrators and users to easily manage Domibus application.

The administration dashboard is reachable via the following URLs:

- Tomcat: http://your_server:your_port_number/domibus (Tomcat)
- WildFly: http://your_server:your_port_number/domibus-wildfly (WildFly)
- WebLogic: http://your_server:your_port_number/domibus-weblogic (WebLogic)

The admin console is made of several sections:

Messages

On this page, the administrator can see the details of the messages and re-process them if required. The administrator can also navigate through the messages history and download specific messages if needed.

Message Filter

On this page, the administrator can set defined filters and access them individually for edition directly in the list.

Error Log

On this page, the administrator can view the list of application errors, make searches on error messages and filter them.

PMode

On this page, the administrator can upload or download the PMde file. The administrator can edit the list of parties configured in the PMode and access them individually for modification purposes.

JMS Monitoring

On this page, the administrator can monitor and manage the contents of the JMS queues.

Truststore

On this page, the administrator can upload a new truststore to replace the current one.

Users

On this page, the administrator can create and manage users including: grant access rights, change passwords, assign roles, etc.

Plugin Users

On this page, the administrator can manage the plugin users: create, delete, edit, grant access rights and roles, etc.

Audit

On this page, the administrator has an overview of changes performed in the PMode, Parties, Message Filter and Users pages.

Alerts

This page displays the alers generated by Domibus in case of unusual behaviour of the application. The alerts are configured by the administrator.

Test Service

On this page the administrator can perform basic test of the communication configuration between two access points.

9.2. Multitenancy

In Multitenancy mode, each domain has its own set of configuration files: Keystore, Truststore, PMode, Domain properties, etc. Users are defined for each domain.

The user named **super** with role **ROLE_AP_ADMIN**, has the privileges to access all the available domains.

When logged as **super**, you are able to select a specific domain in the upper right part of the admin console in a drop-down list (default or dom50 domain ine the example below):

9.3. Message Log

Domibus administration dashboard includes a message logging page that gives the administrator information related to sent messages, received messages and their status (SENT, RECEIVED, FAILED, ACKNOWLEDGED,…):

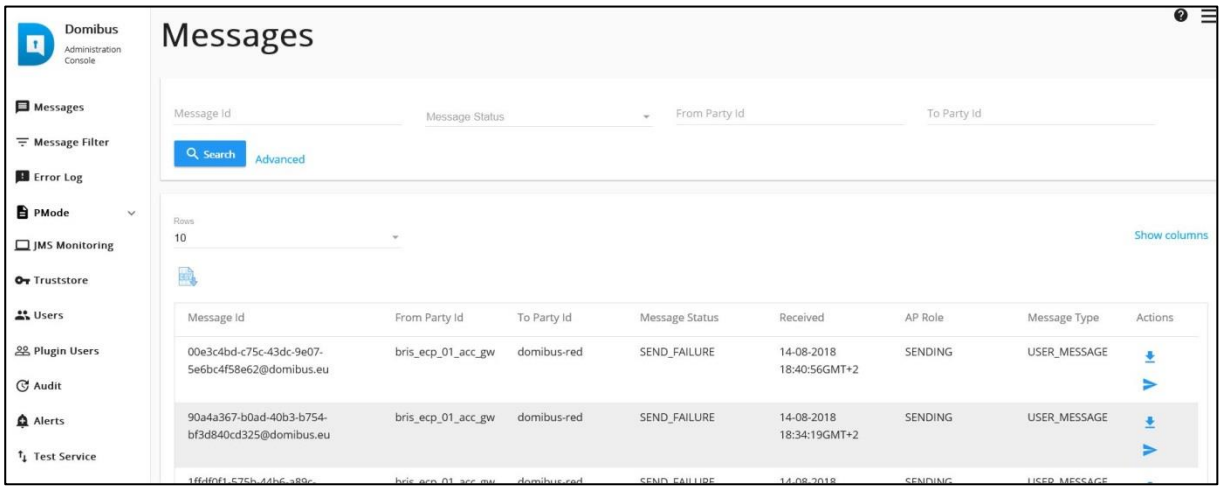


Figure 12 - Domibus Message Log

The following state machines illustrate the evolution of the processing of messages according to the encountered events:

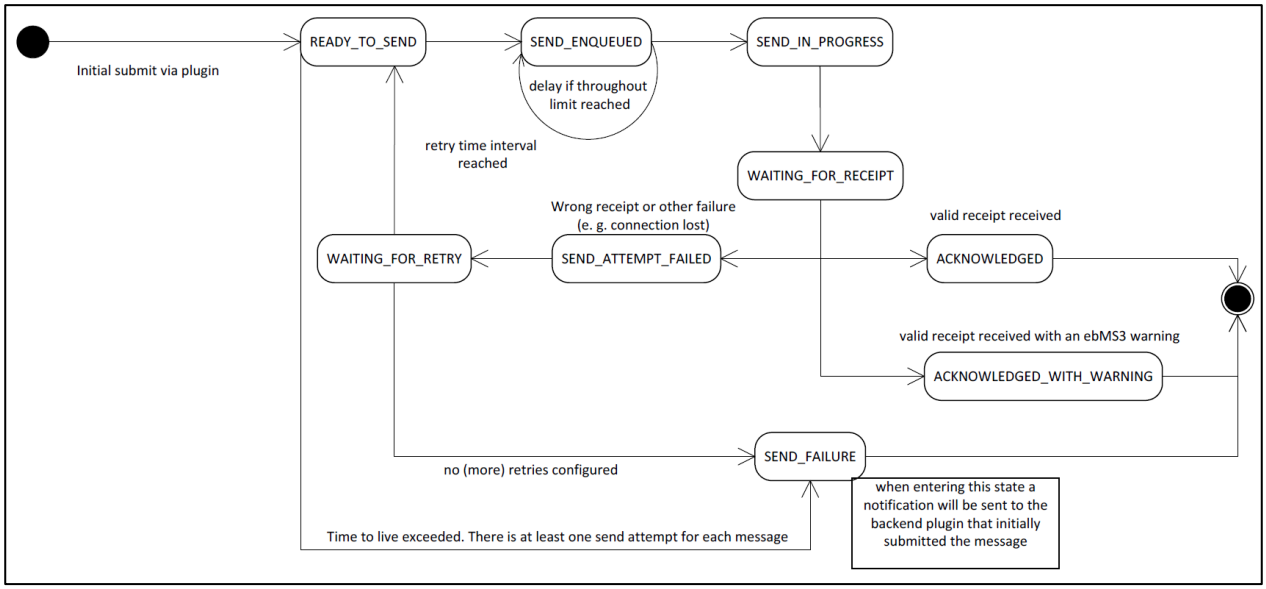


Figure 10 - State machine of Corner 2 (sending access point)

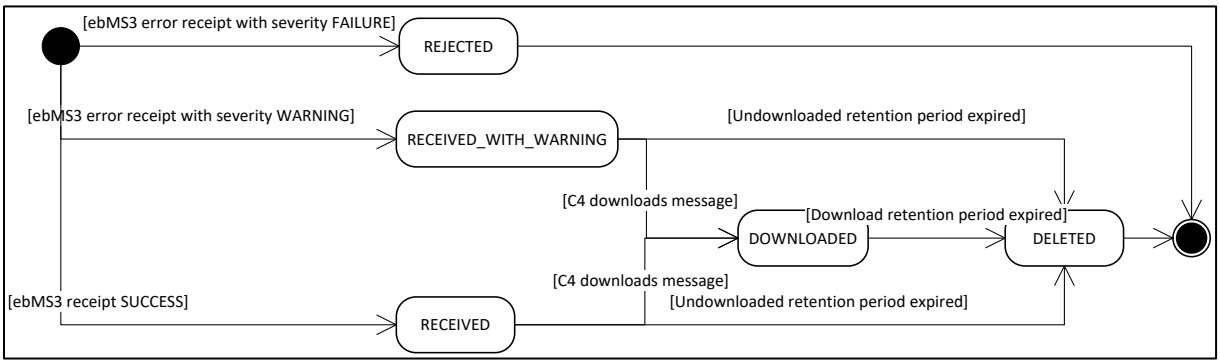


Figure 11 - State machine of Corner 3 (receiving access point)

9.4. Message Filtering

Domibus allows the routing of messages to different plugins, based on some messages attributes:

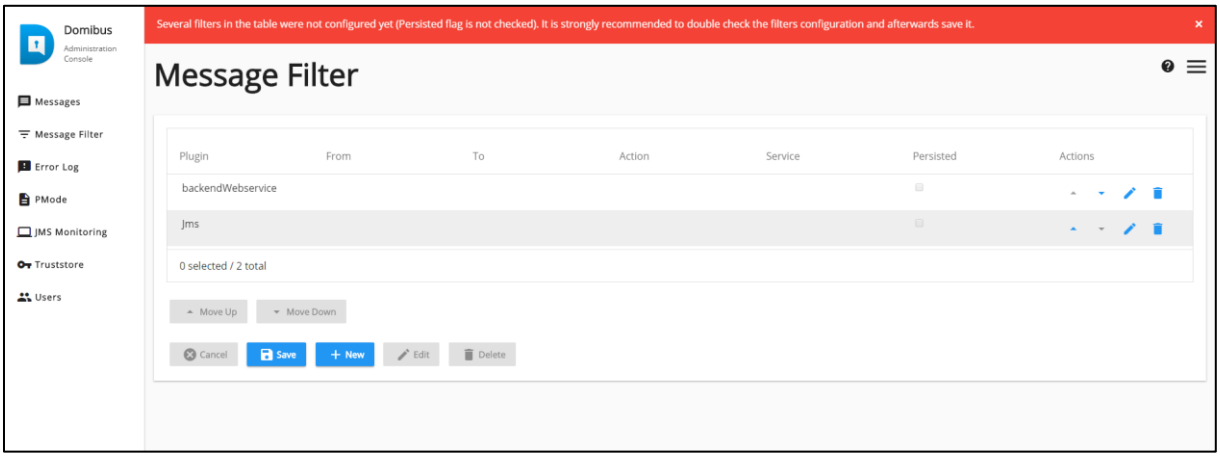
- **From** : initial sender (C1)
- **To** : final recipient (C4)
- **Action**: defined as 'Leg' in the PMode
- **Service**: as defined in the PMode

The following rules apply:

- Domibus considers the ordered list of 'filters' to route all messages. The first filter matching the filter criteria, will define the target plugin. The order of the plugin is therefore important in the routing process.

  Note 1: if the filters are all mutually exclusive, the order would not matter.

  Note 2: The 'Persisted' column indicates if the plugin filter configuration has already been saved. If a plugin filter configuration has not already been saved, the 'Persisted' value is unchecked and an error message is shown on the top of the screen. In this case, it is strongly recommended to review the filters configuration and save it afterwards.



- One plugin may be applied to multiple filters. This is done by the use of the 'OR' criteria.
  (cf. backendWebservice in the example below).
- Multiple attributes could also be defined in one filter. This is done by the use of the 'AND' criteria.
  (cf. the first filter in the example below).
- One filter may have no criteria, meaning that all messages (not matching previous filters) will be routed to the corresponding plugin automatically. As a result, subsequent filters will therefore not be considered for any incoming message. In the example below, the last filter routes all remaining messages to plugin 'backendWebservice'.

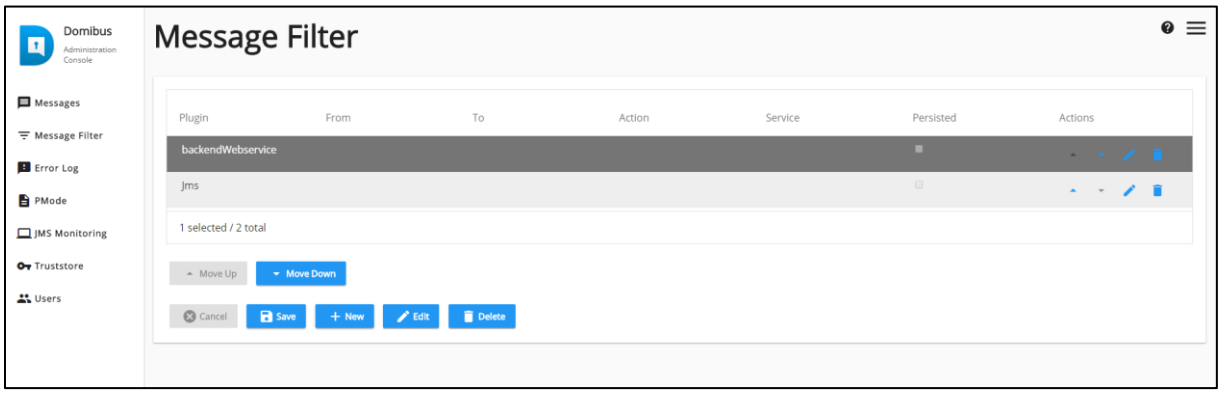


Figure 9 – Message Filter Page

Use the **New** and **Delete** buttons to create or delete a filter.

As the order matters, move up and down actions allow placing each filter in the right order:

Cf. **Move Up** and **Move Down** buttons.

After some changes have been applied to the filters, the **Cancel** and **Save** buttons become active:

- Press **Cancel** to cancel the changes
- Press **Save** to save the changes and activate them immediately.

The console will ask the user to confirm the operation, before proceeding.

Example of message attributes used for routing and matching the first filter used in the example above:

- **Action** : *TC1Leg1*
- **Service** : *bdx:noprocess:tc2*
- **From** : domibus-blue:urn:oasis:names:tc:ebcore:partyid-type:unregistered
- **To** : domibus-red:urn:oasis:names:tc:ebcore:partyid-type:unregistered

That information can be found in the incoming message received by Domibus (e.g. see below):

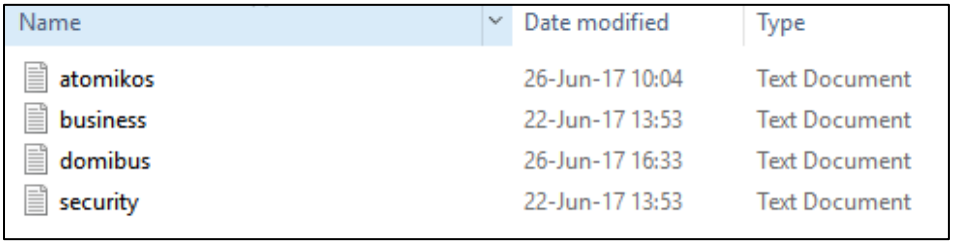
```
<ns:PartyInfo>
        <ns:From>
          <ns:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">domibus-
blue</ns:PartyId>
          <ns:Role>http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/initiator</ns:Role>
        </ns:From>
        <ns:To>
          <ns:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">domibus-
red</ns:PartyId>
          <ns:Role>http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/responder</ns:Role>
        </ns:To>
</ns:PartyInfo>
      <ns:CollaborationInfo>
        <ns:Service type="tc1">bdx:noprocess</ns:Service>
        <ns:Action>TC1Leg1</ns:Action>
      </ns:CollaborationInfo>
```

9.5. Application Logging

9.5.1. Domibus log files

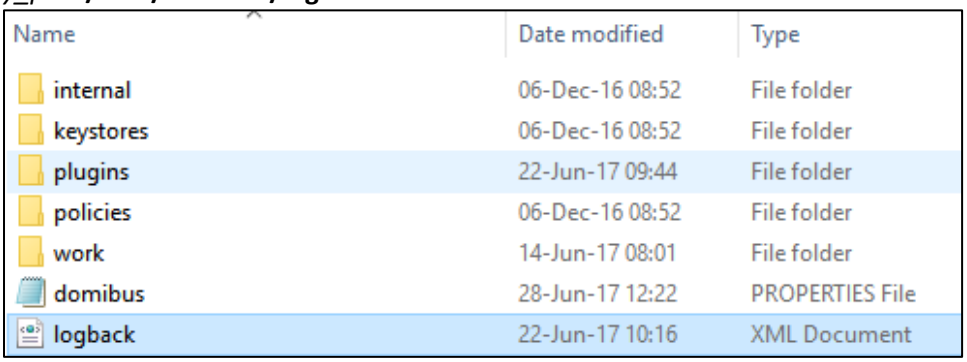
Domibus has three log files:

1. domibus-security.log : this log file contains all the security related information. For example, you can find information about the clients who connect to the application.
2. domibus-business.log: this log file contains all the business related information. For example, when a message is sent or received, etc.
3. domibus.log : this log file contains both the security and business logs plus miscellaneous logs like debug information, logs from one of the framework used by the application, etc.



9.5.2. Logging properties

It is possible to modify the configuration of the logs by editing the logging properties file: *cef_edelivery_path***/conf/domibus/logback.xml**:



9.5.3. Error Log page

To go to the error log page of the Domibus Admin Console, click on the **Error log** menu entry.

This option lists all the Message Transfers error logs and includes the **ErrorSignalMessageId**, **ErrorDetail** and **Timestamp**. You can sort messages by using the up or down arrow to search for a specific message.

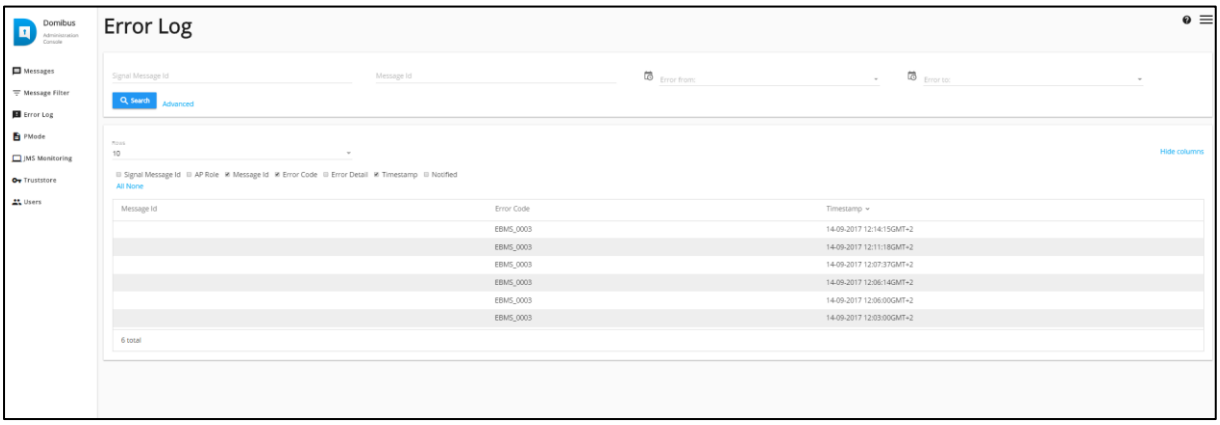
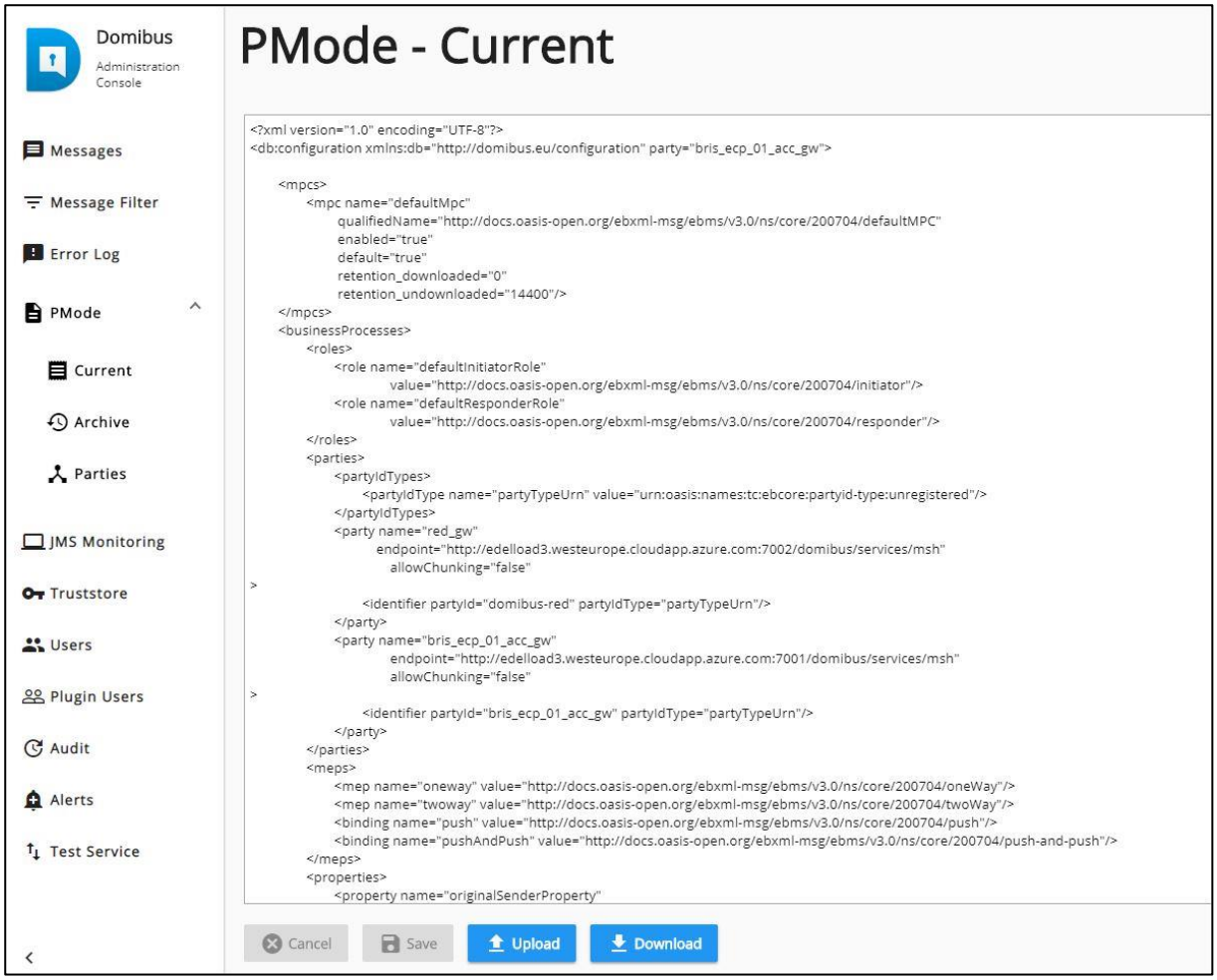


Figure 13 - Domibus – Error Log page

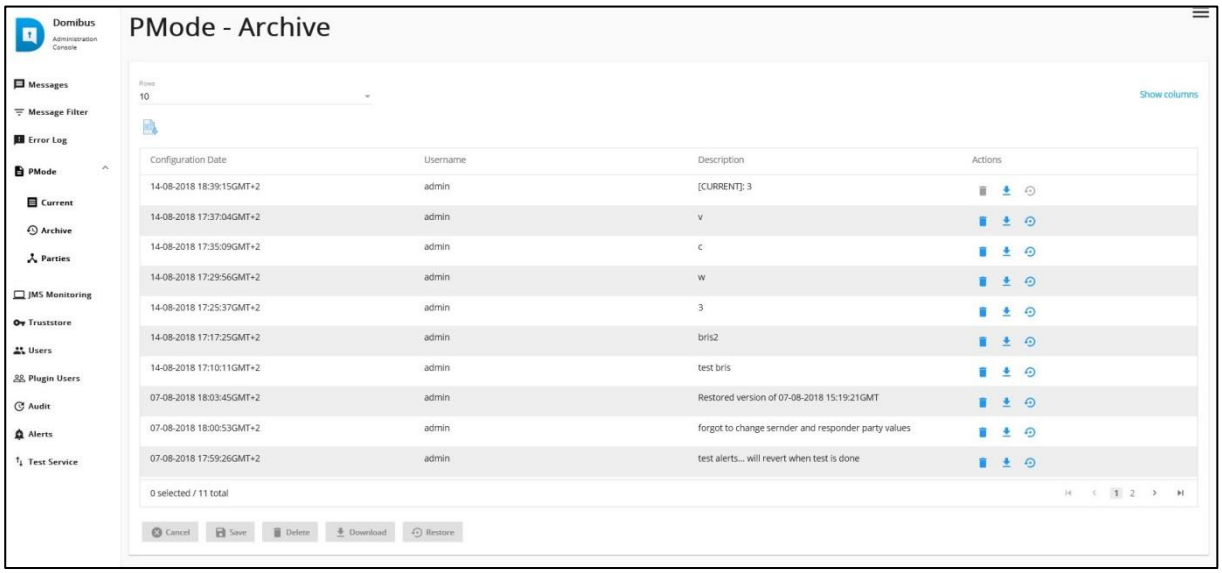
9.6. PMode

In the Administration console you can view the content of the current PMode:



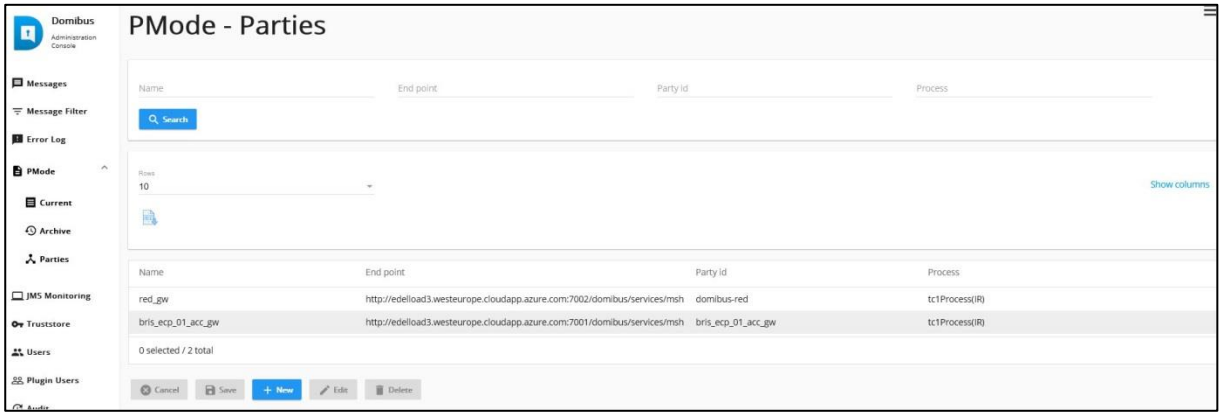
You can edit the content of your current PMode in the administration console and save the changes by clicking on **Save** or discard the changes by clicking on **Cancel**. You can **upload** a PMode file or **download** the current one.

Under **Archive** the history of the PMode changes is displayed:



Domibus keeps a snapshot of the Pmode each time the PMode is modified. The user can restore a particular version and make it the current PMode nu clicking on the restored button at the far right of the table.

Under Parties, the user can manage the parties in the PMode. Parties can be searched using filter criteria, they can be added, updated or deleted.



The PMode is updated and a new PMode snapshot is created when parties are added, updated or deleted.

9.7. Queue Monitoring

Domibus uses JMS queues to handle the messages:

Destination type	JNDI name	Comment	Description
Queue	jms/domibus.internal.dispatch.queue	No redelivery because redelivery of MSH messages is handled via ebMS3/AS4	This queue is used for scheduling messages for sending via the MSH.
Queue	jms/domibus.internal.notification.unknown		Notifications about received messages (by the MSH) that do not match any backend routing criteria will be sent to this queue. In production environment this queue should be monitored in order to handle those messages manually.
Topic	jms/domibus.internal.command		This topic is used for sending commands to all nodes in a cluster. For example, it is used after a PMode was uploaded in order to notify all nodes to update their PMode cache (in case caching is enabled).
Queue	jms/domibus.backend.jms.replyQueue		This queue is used for sending replies back to the sender of a message. Replies contain: a correlationId, ebMS3 messageId (if possible), error messages (if available).

Queue	jms/domibus.backend.jms.outQueue		Messages received by the MSH (that match the routing criteria for the JMS plugin) will be sent to this queue.
Queue	jms/domibus.backend.jms.inQueue		This queue is the entry point for messages to be sent by the sending MSH.
Queue	jms/domibus.backend.jms.errorNotifyConsumer		This queue is used to inform the receiver of a message that an error occurred during the processing of a received message.
Queue	jms/domibus.backend.jms.errorNotifyProducer		This queue is used to inform the sender of a message that an error occurred during the processing of a message to be sent.
Queue	jms/domibus.notification.jms		Used for sending notifications to the configured JMS plugin.
Queue	jms/domibus.internal.notification.queue		This queue is used to notify the configured plugin about the status of the message to be sent.
Queue	jms/domibus.notification.webservice		Used for sending notifications to the configured WS plugin.

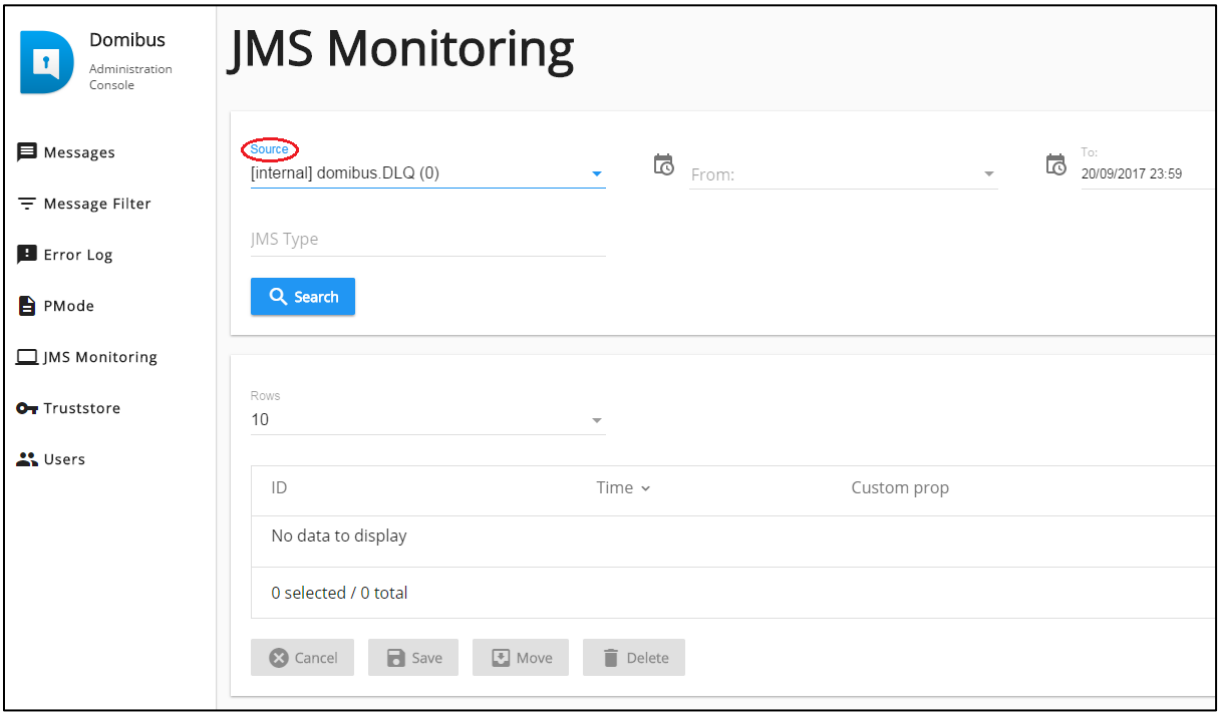
Queue	jms/domibus.DLQ		This is the Dead Letter Queue of the application. The messages from other queues that reached the retry limit are redirected to this queue.

Table 3 - Queue Monitoring

All these queues can be monitored and managed using the **JMS Monitoring** page, which is accessible from the **JMS Monitoring** menu of the administration console:

Warning:
For Tomcat server, the maximum number of shown messages in the queue monitoring is defined by the 'domibus.listPendingMessages.maxCount' property.

In the **Source** field, we have all the queues listed, along with the number of messages pending in each queue:



If a queue is used internally by the application core, its name will start with **[internal].** A regular expression is used to identify all the internal queues. The value for this regular expression can be adapted in the **domibus.jms.internalQueue.expression** property from the *cef_edelivery_path***/conf/domibus/domibus.properties** file.

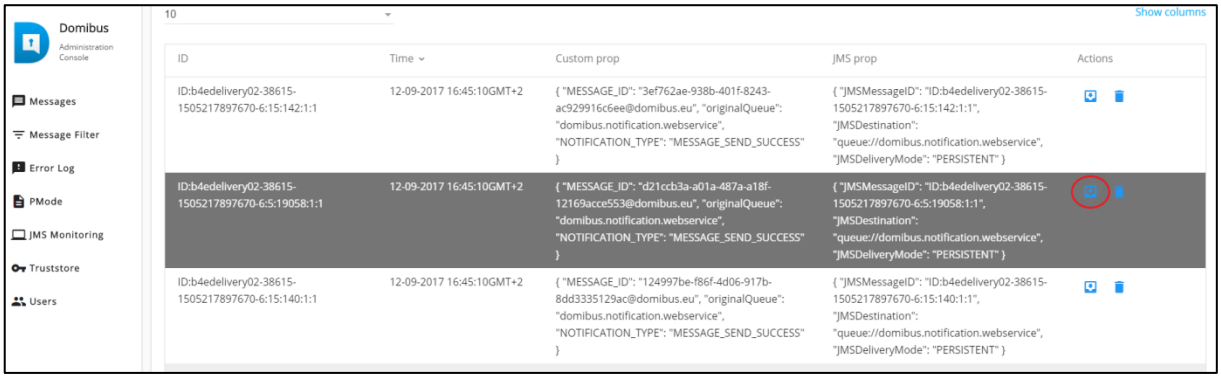
In the **JMS Monitoring** page the following operations can be performed:

1. Inspecting and filtering the messages from a queue based on the following fields:
    a. Signal Message id: identifier of an error signal message
    b. Message id: identifier of a message
    c. Error detail: text of the error (full)
    d. AP Role: role of the AP
    e. Error Code: structured code of the error
    f. Source: the source queue of the messages
    g. Error or Notified Time Period: time interval that will filter the messages based on the send dates
    h. JMS type: the JMS header **JMSType**
    i. Selector: the JMS message selector expression

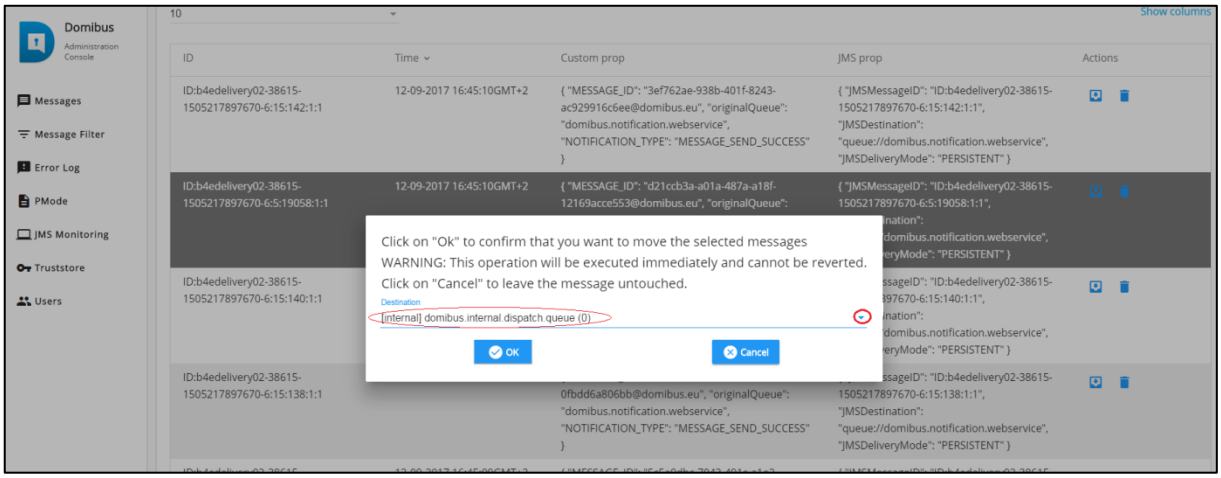
Remark:

For more information on the JMS message headers and the JMS message selector, please check the official documentation at https://docs.oracle.com/cd/E19798-01/821-1841/bnces/index.html.

2. Move a message:
    a. Move the message from the DLQ to the original queue:
        - Select the JMS message from the DLQ and press the **Move** icon (**in RED marker**):

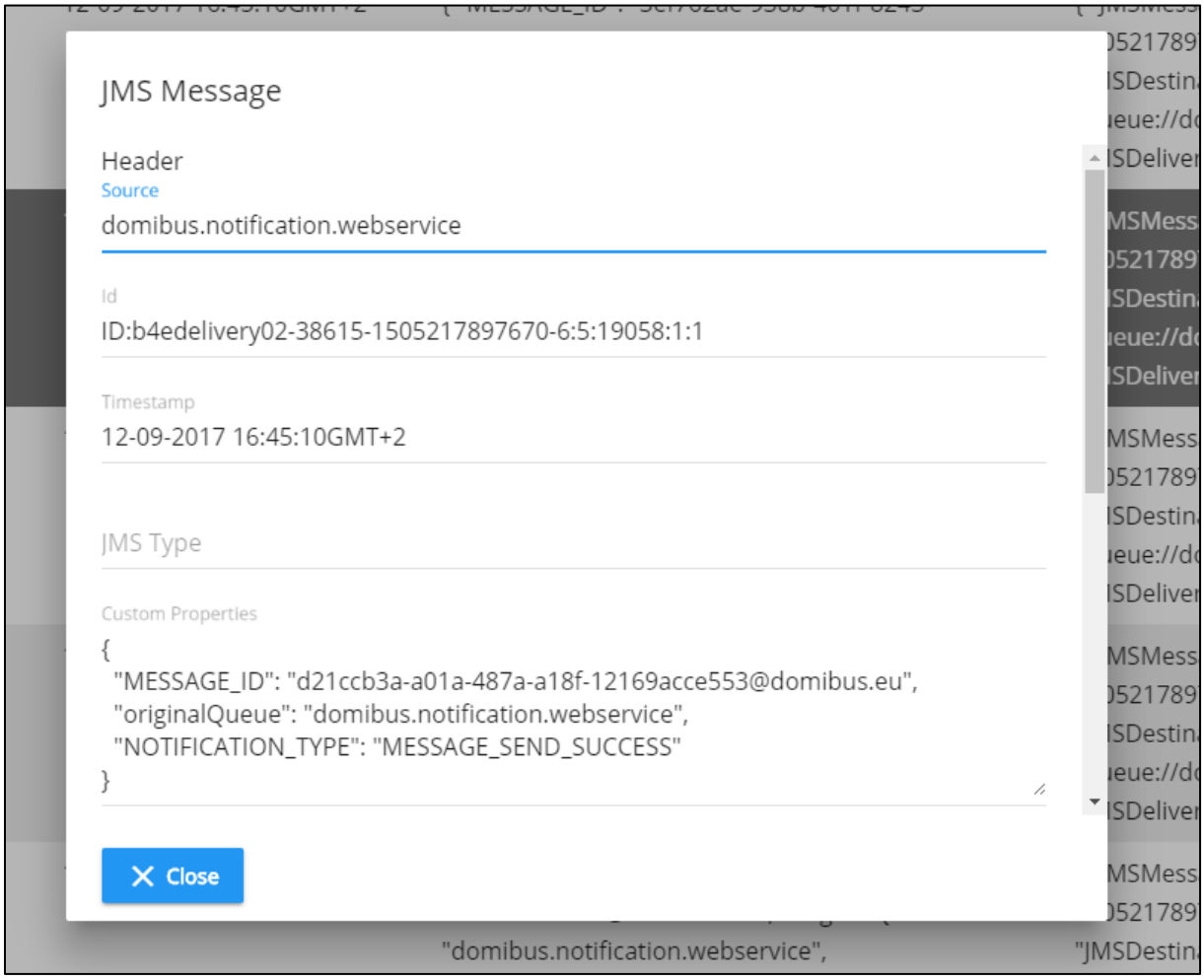


- Select the original queue from the **Destination** dropdown list in the dialog box:



- Press the **Ok** button in the dialog, and the message will be moved to the original queue.

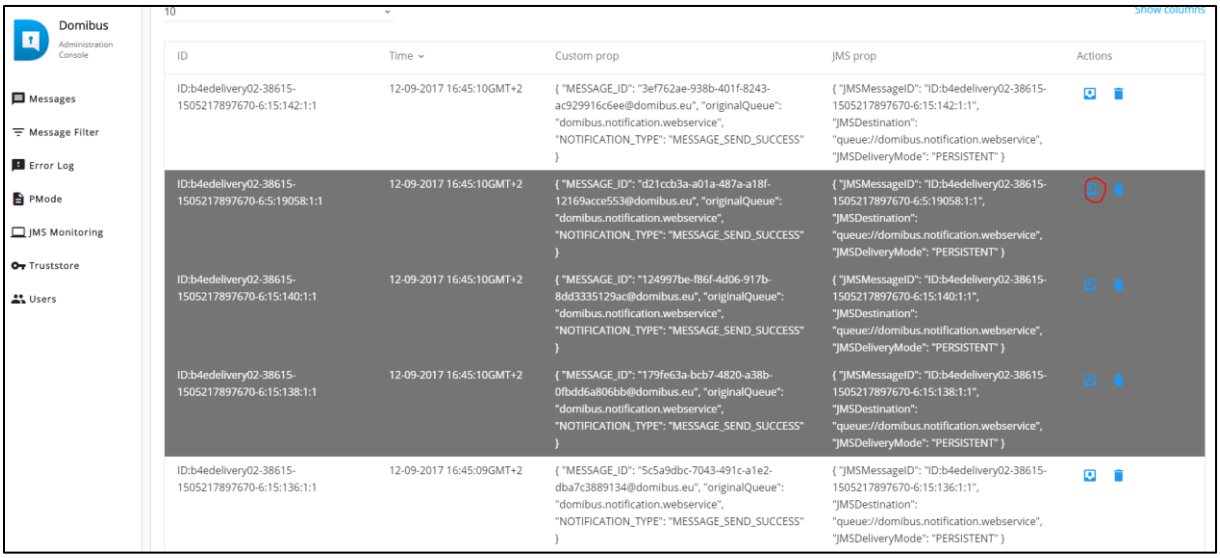
Note: the details of a message can be viewed by selecting it (double-clicking) from the message list:



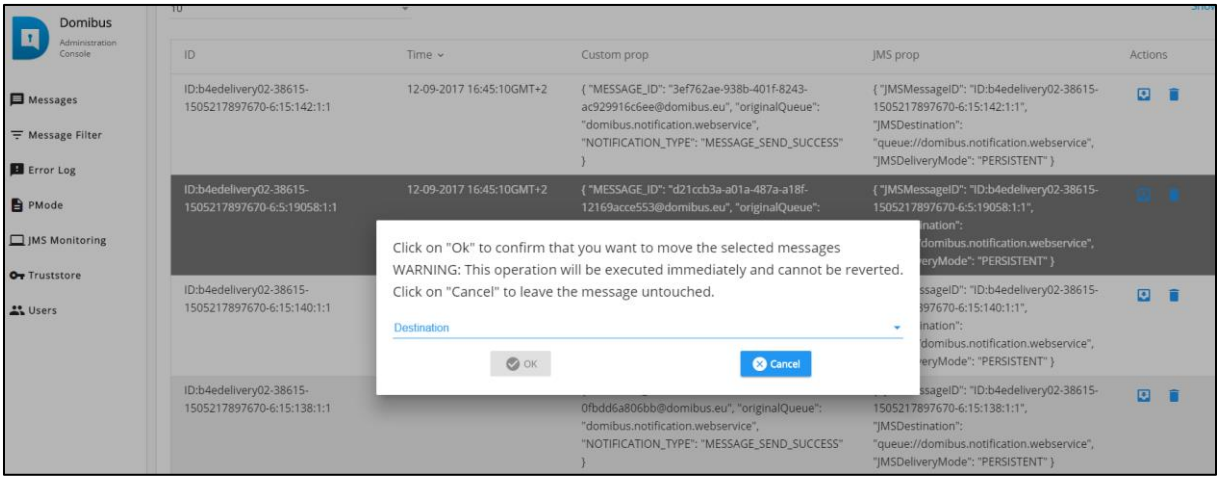
Click **Close** to exit the dialog box.

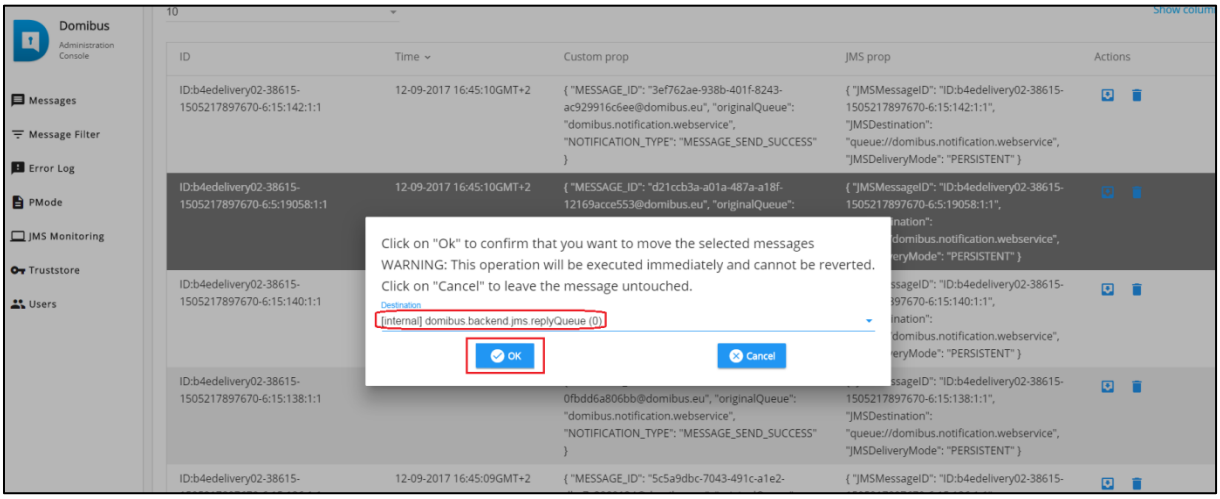
b. Move multiple messages from the DLQ to the original queue:

- Select multiple JMS messages from the DLQ and press the **Move** icon button:



- Select the original queue from the Destination dropdown list, and click **Ok**.

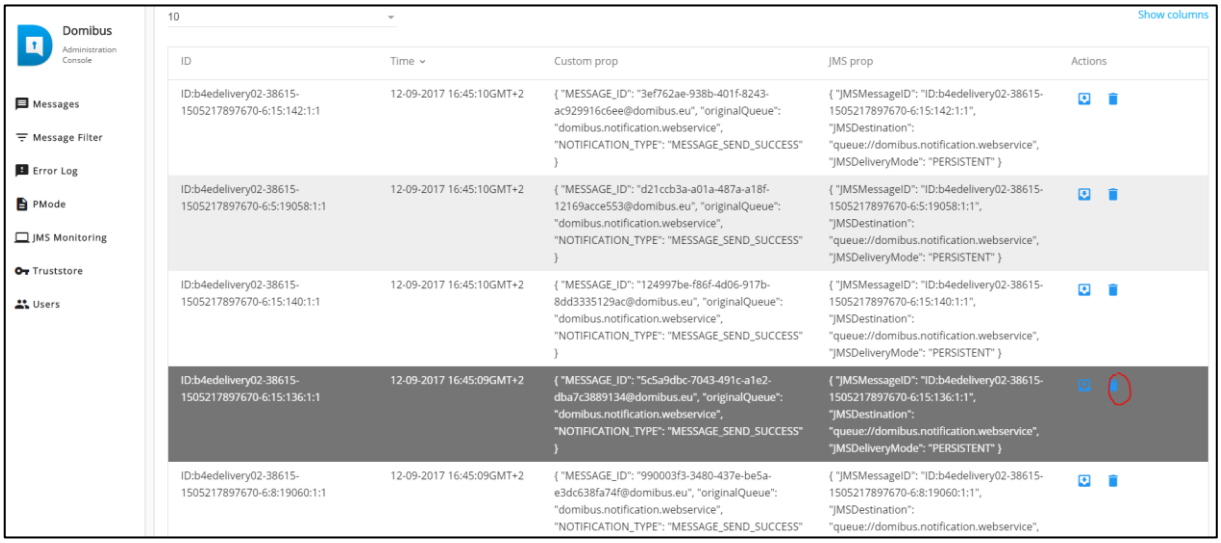




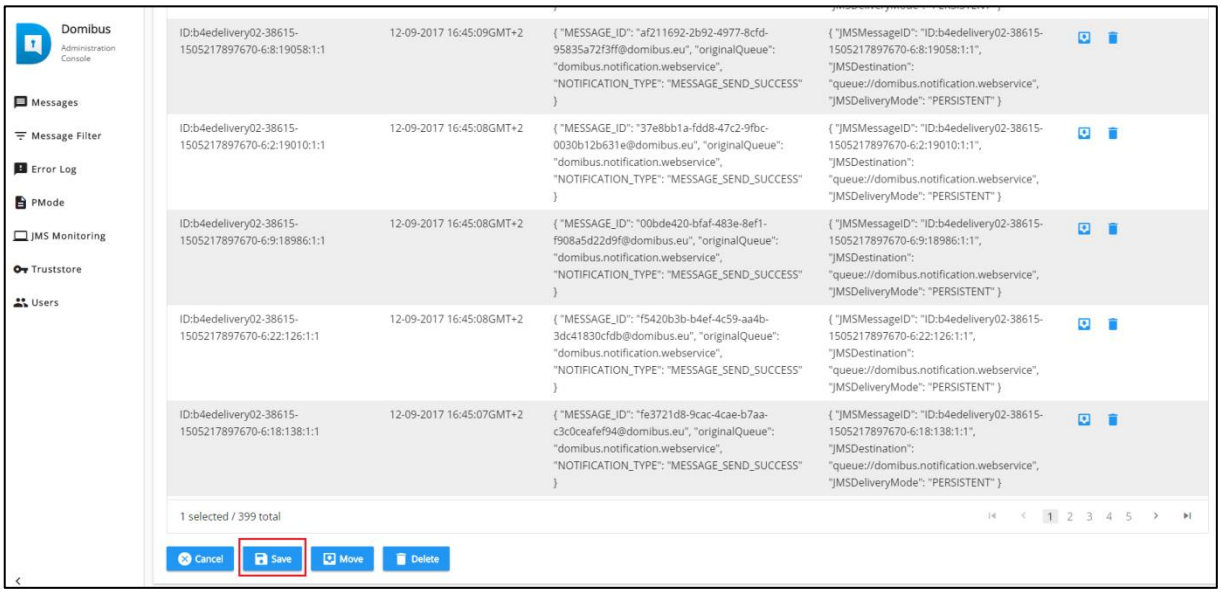
***Remark**:*

Please make sure that all the selected messages came from the same source queue. Use the filtering capabilities to ensure this.

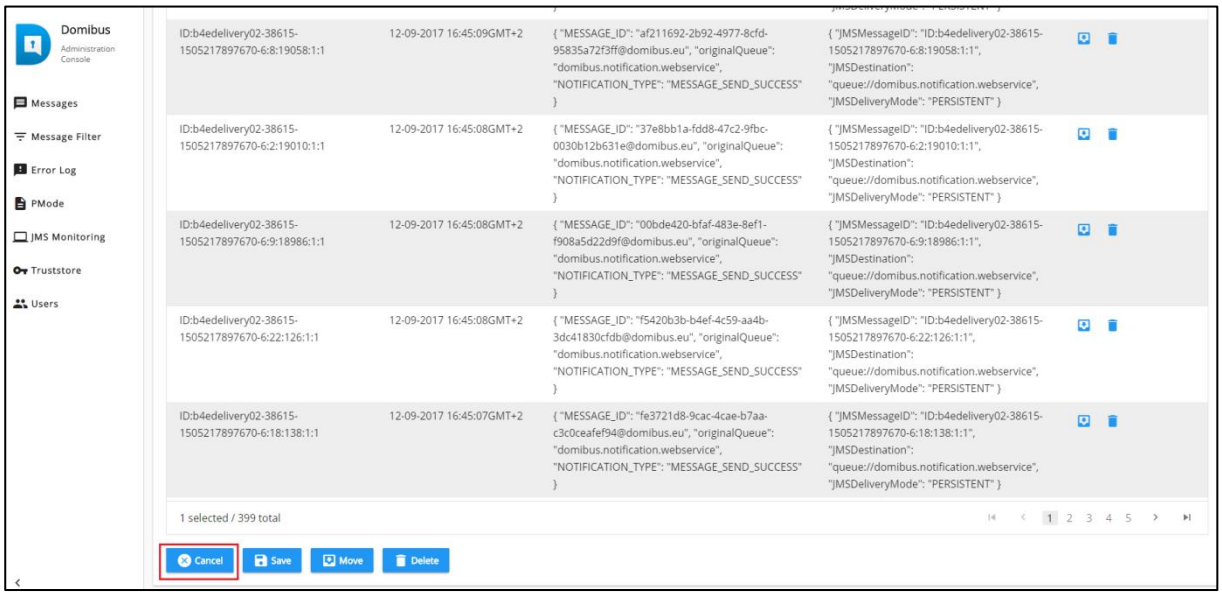
3. Delete message(s)
    a. Delete one or more messages from one queue:
        - Select one or several JMS messages from the source queue and press the **Delete** button:



        - By clicking the **Delete** button, the selected messages are removed from the screen, but you still have to confirm your changes by clicking on the **Save** button. As long as you have not clicked on the **Save** button, your changes are not taken into account in the system.



        - To cancel the changes you made, click on the **Cancel** button instead:



9.8. Configuration of the queues

Queues should be configured appropriately and according to the backend system needs and re-delivery policy.

9.8.1. Tomcat

Domibus uses ActiveMQ as JMS broker. The various queues are configured in the *cef_edelivery_path***/conf/domibus/internal/activemq.xml** file.
Please see ActiveMQ redelivery policy and configure the parameters below if needed:

```
<redeliveryPlugin fallbackToDeadLetter="true"
            sendToDlqIfMaxRetriesExceeded="true">
   <redeliveryPolicyMap>
      <redeliveryPolicyMap>
         <defaultEntry>
                        <!-- default policy-->
                        <redeliveryPolicy maximumRedeliveries="10"       redeliveryDelay="300000"/>
               </defaultEntry>
        <redeliveryPolicyEntries>
               <redeliveryPolicy queue="domibus.internal.dispatch.queue" maximumRedeliveries="0"/>
               <redeliveryPolicy queue="domibus.internal.pull.queue" maximumRedeliveries="0"/>
</redeliveryPolicyEntries>
      </redeliveryPolicyMap>
   </redeliveryPolicyMap>
</redeliveryPlugin>
```

Access to the JMS messaging subsystem is protected by a username and a password in clear text defined in the Domibus properties file *cef_edelivery_path***/conf/domibus/domibus.properties**.
It is recommended to change the password for the default user:

```
activeMQ.username=domibus
activeMQ.password=changeit
```

Remark:

*The user (**activeMQ.username**) and the password (**activeMQ.password**) defined in the **domibus.properties** file are referenced in the authentication section of the **activemq.xml** file provided.*

9.8.2. WebLogic

Please use the admin console of WebLogic to configure the re-delivery limit and delay if necessary.

9.8.3. WildFly

Please use the admin console of WildFly to configure the re-delivery limit and delay if necessary.

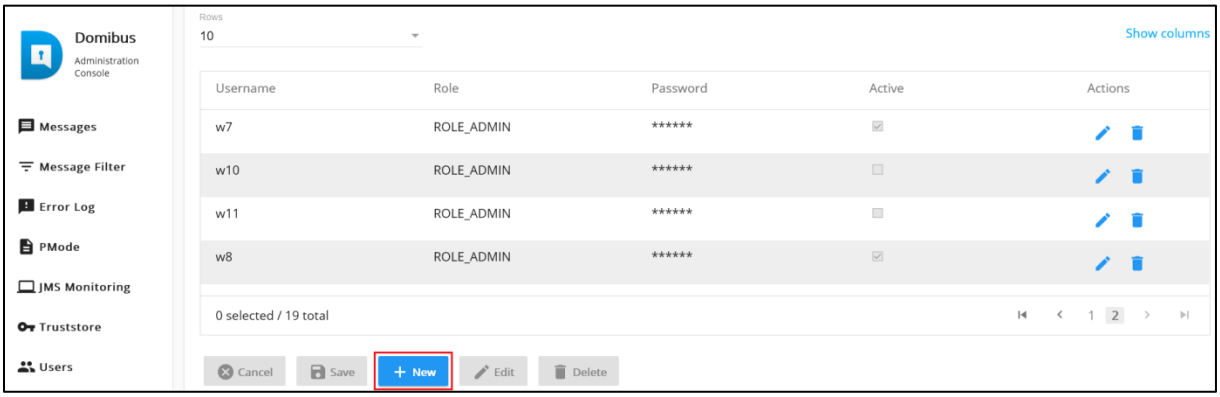
9.9. Truststore

In the Truststore screen, you can manage the trusted certificates. You can upload a new truststore to replace the current one and define its password.

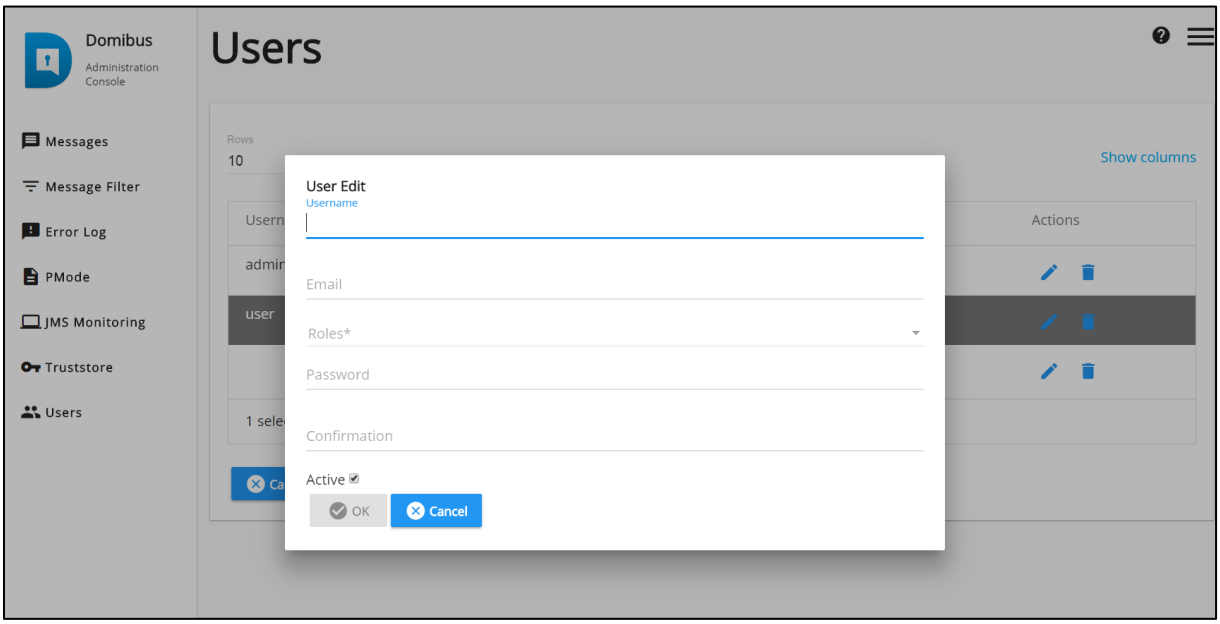
9.10. Users

9.10.1. Adding new users

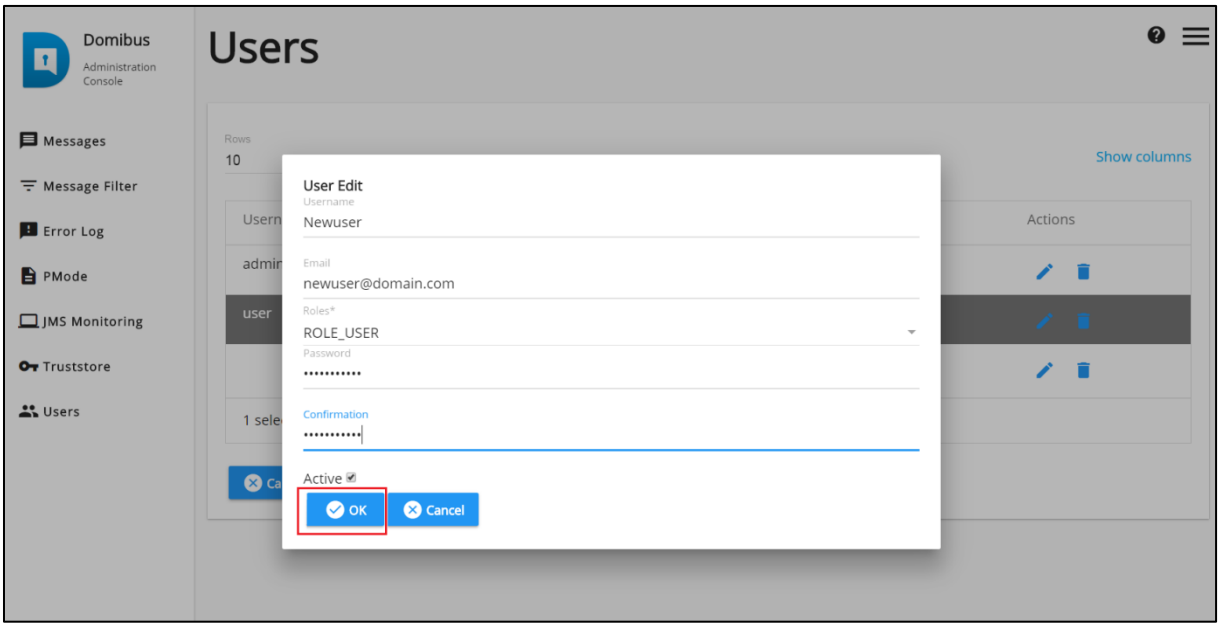
1. New users can be added to the existing default users (**admin** and **user**) by clicking on **New**:



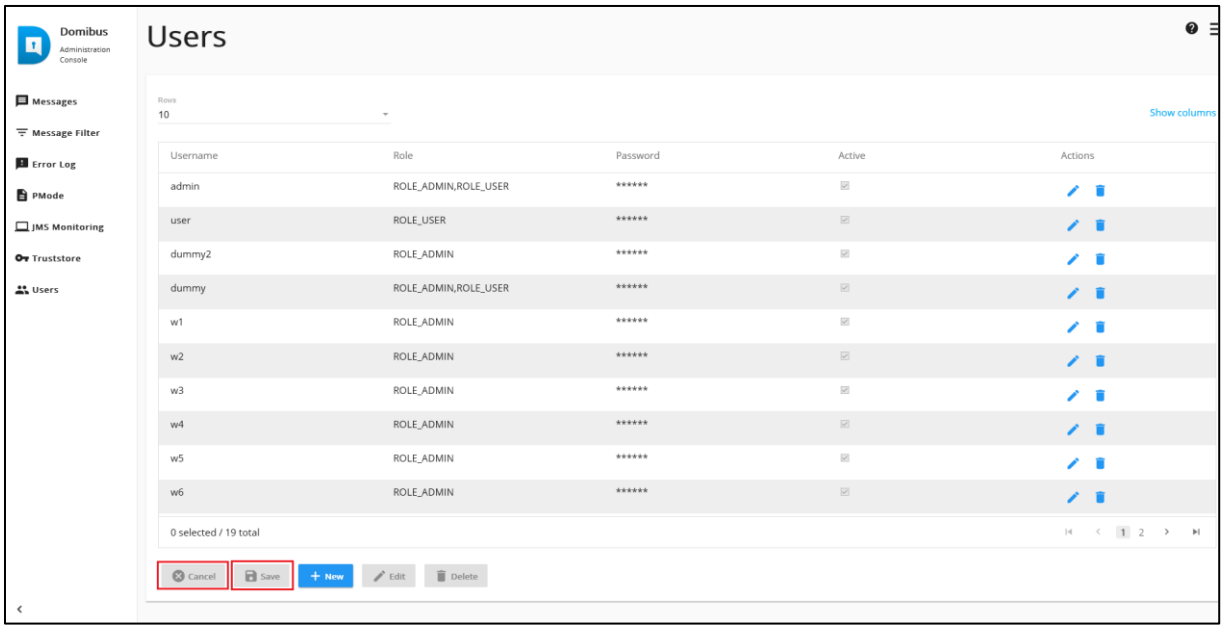
2. For each new user, you must enter a username, an email, a role and a password:


Domibus
Administration Console
Users
Messages
Message Filter
Error Log
PMode
JMS Monitoring
Truststore
Users
Rows
10
Show columns
Actions
User Edit
Username
Email
Roles*
Password
Confirmation
Active
OK
Cancel

3. Click on **OK**:


Domibus
Administration Console
Users
Messages
Message Filter
Error Log
PMode
JMS Monitoring
Truststore
Users
Rows
10
Show columns
Actions
User Edit
Username
Newuser
Email
newuser@domain.com
Roles*
ROLE_USER
Password
Confirmation
Active
OK
Cancel

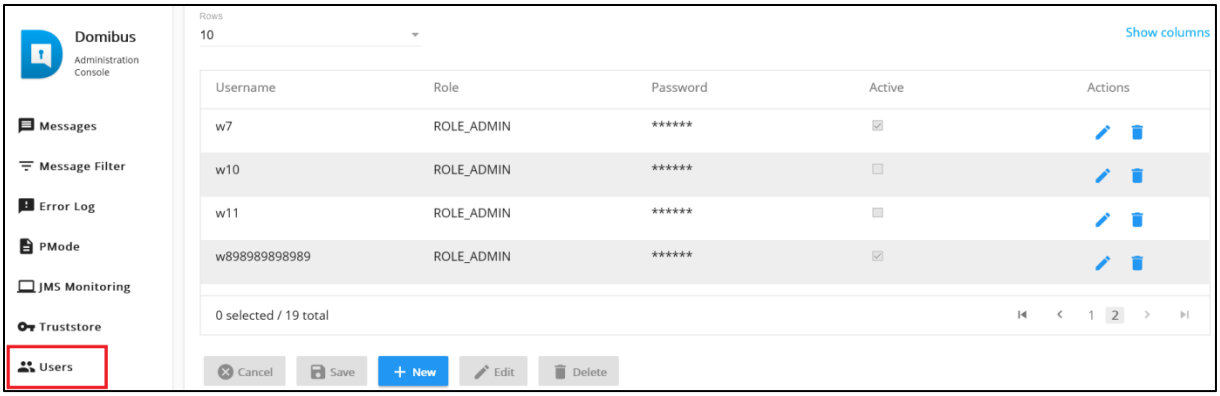
4. Again, once the user has been created, do not forget to click on the **Save** button on the **Users** page to register your changes on the system:



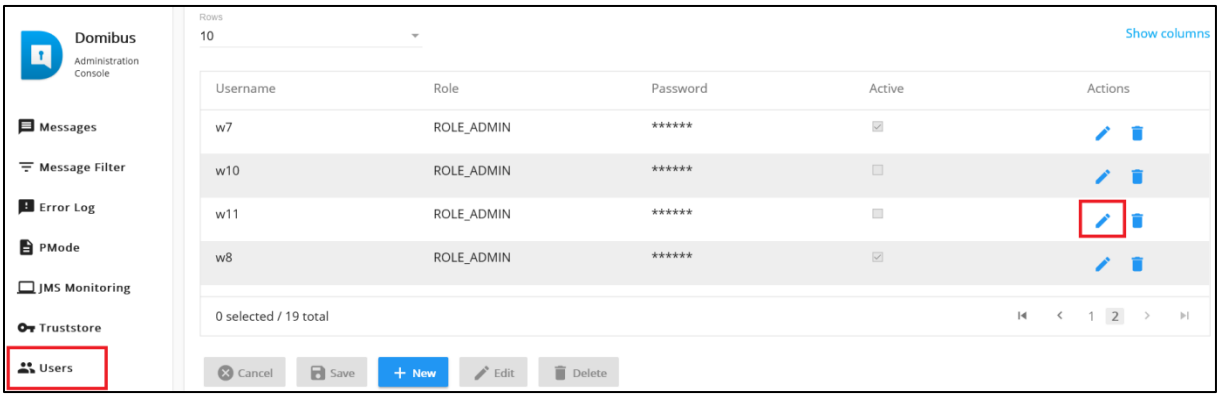
9.10.2. Changing passwords

It is recommended to change the passwords for the default users, who are allowed to have access to the Domibus Administration Console, mainly ***admin*** and ***user***.

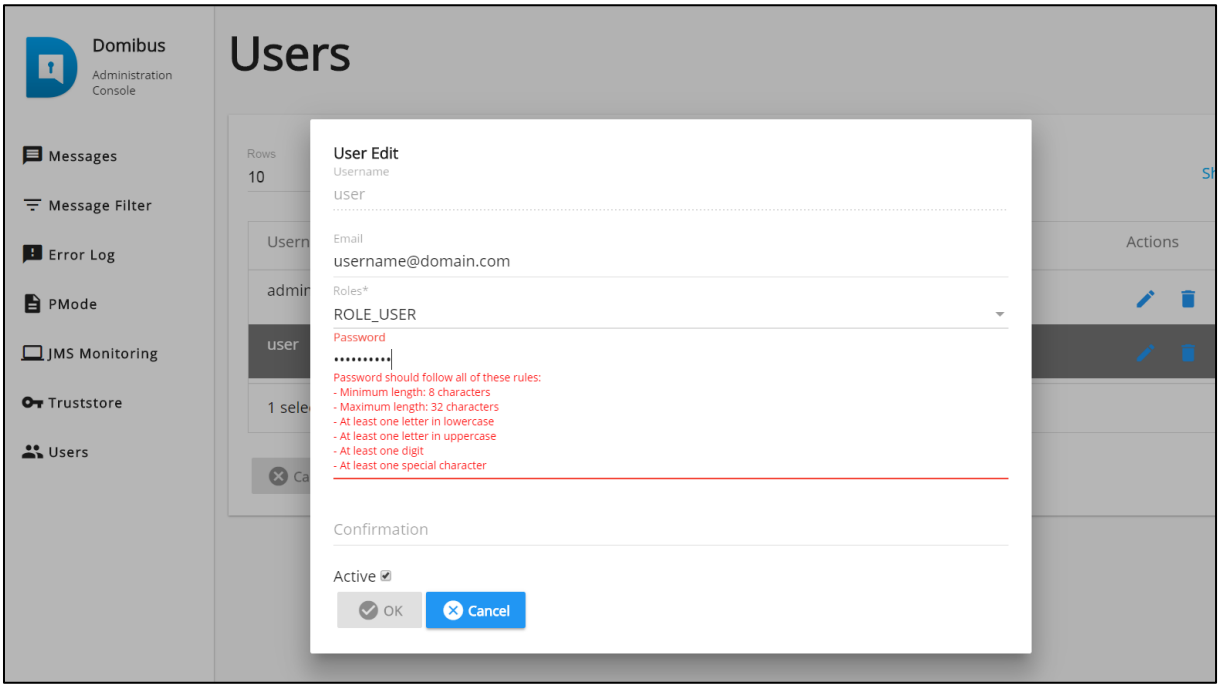
1. In order to change the password for a user, navigate to the **Users** menu entry to obtain the list of configured users:



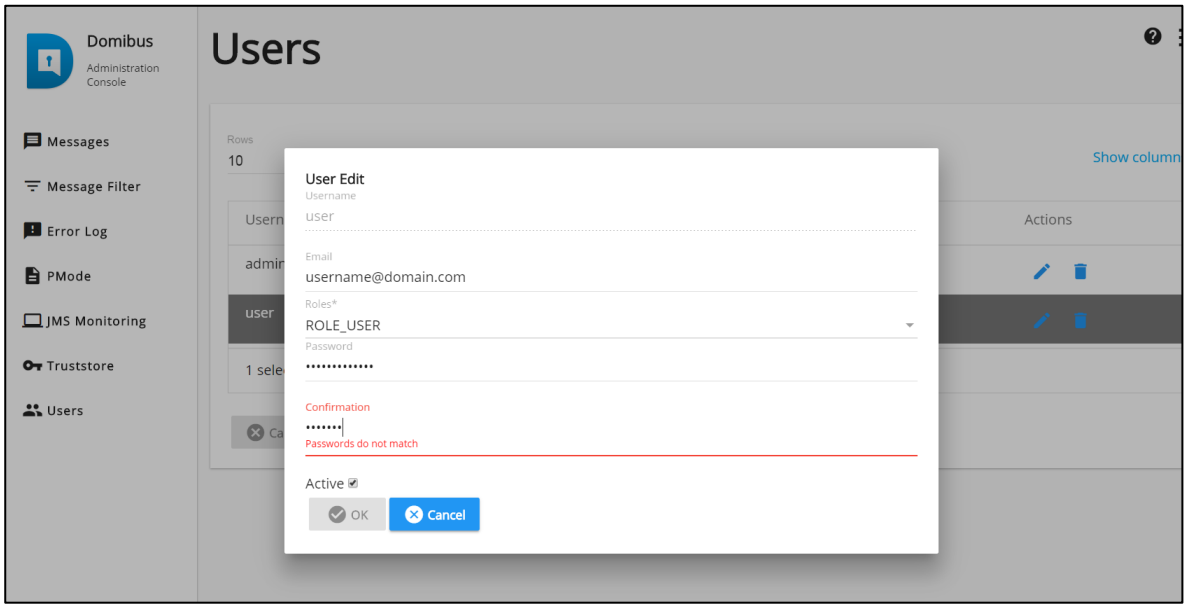
2. To edit the user details, click on the **EDIT** icon (in **RED**). DO NOT click on the BIN icon as this would DELETE the record.



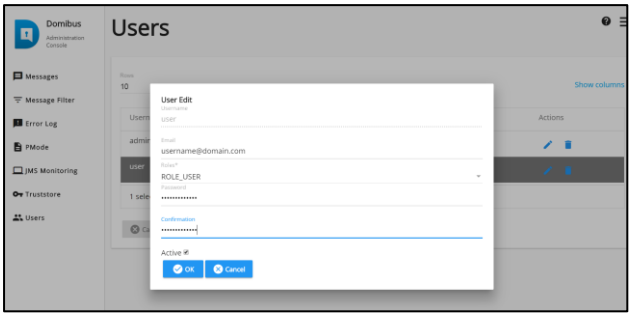
3. In the popup window, choose a new password using the rules shown:



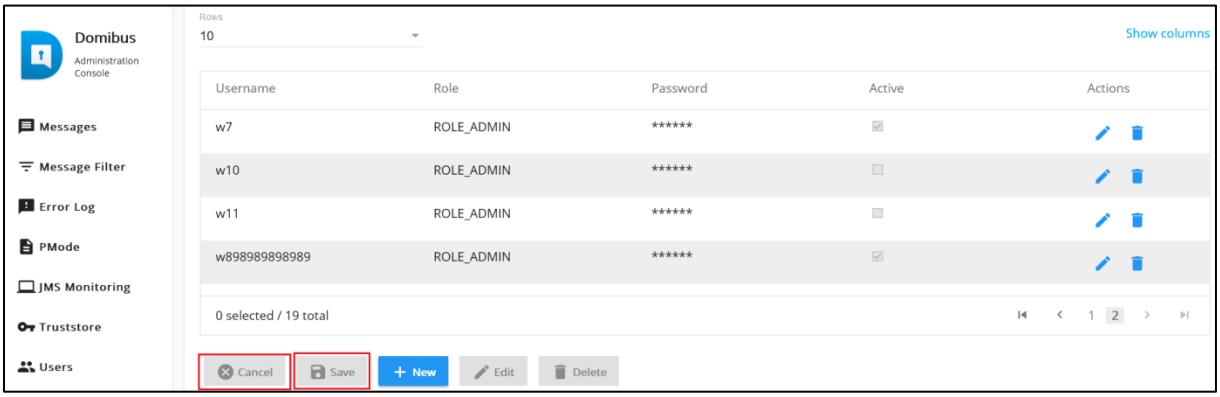
4. Confirm the password:



5. Click on **OK**:

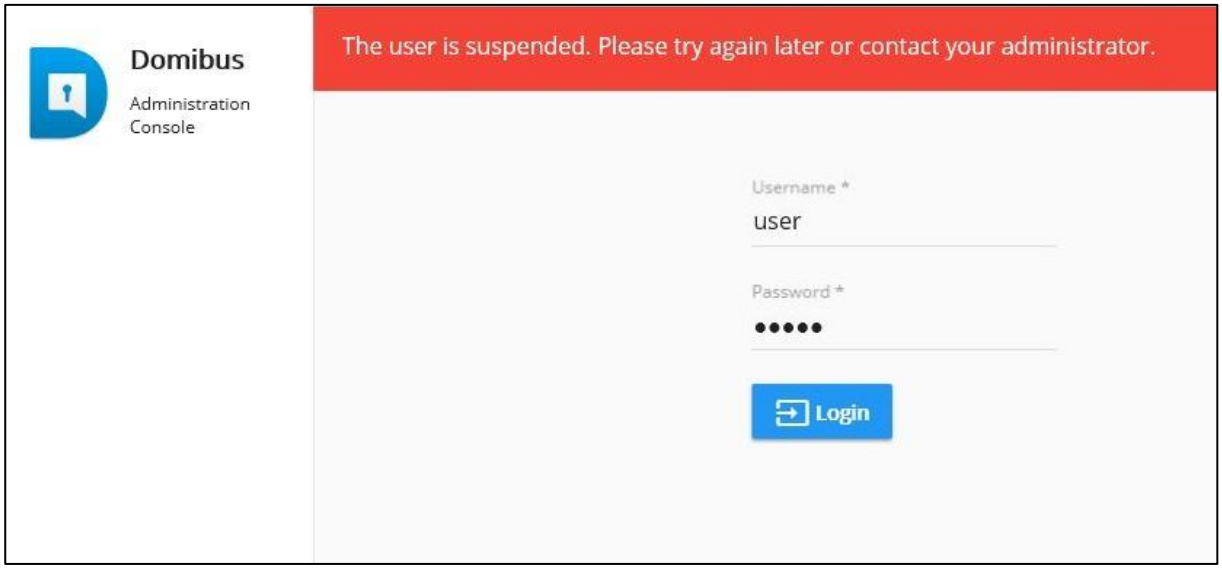


6. When done, either click on **Save,** to save the new password or **Cancel** to leave the password unchanged.



9.10.3. User Account Lockout Policy

A user account lockout policy has been implemented on Domibus Admin Console. By default, if a user tries to log to the Admin Console with a wrong password 5 times in a row, his account will be suspended (locked):

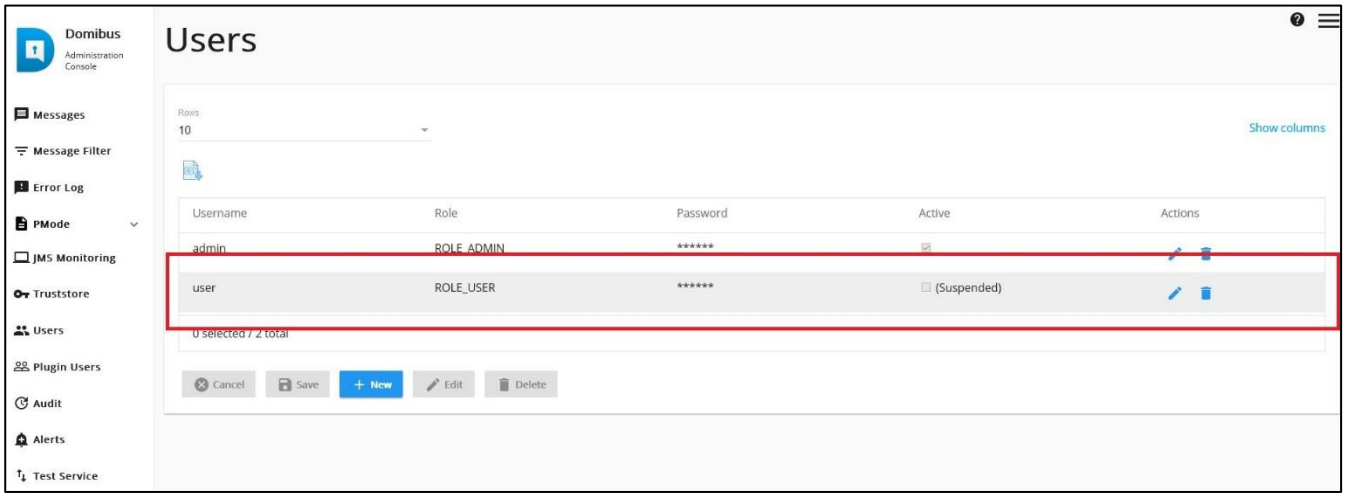


You can define in domibus.properties the number of failed attempts after which a user's account will be locked (see also §5.2- "*Domibus Properties*").

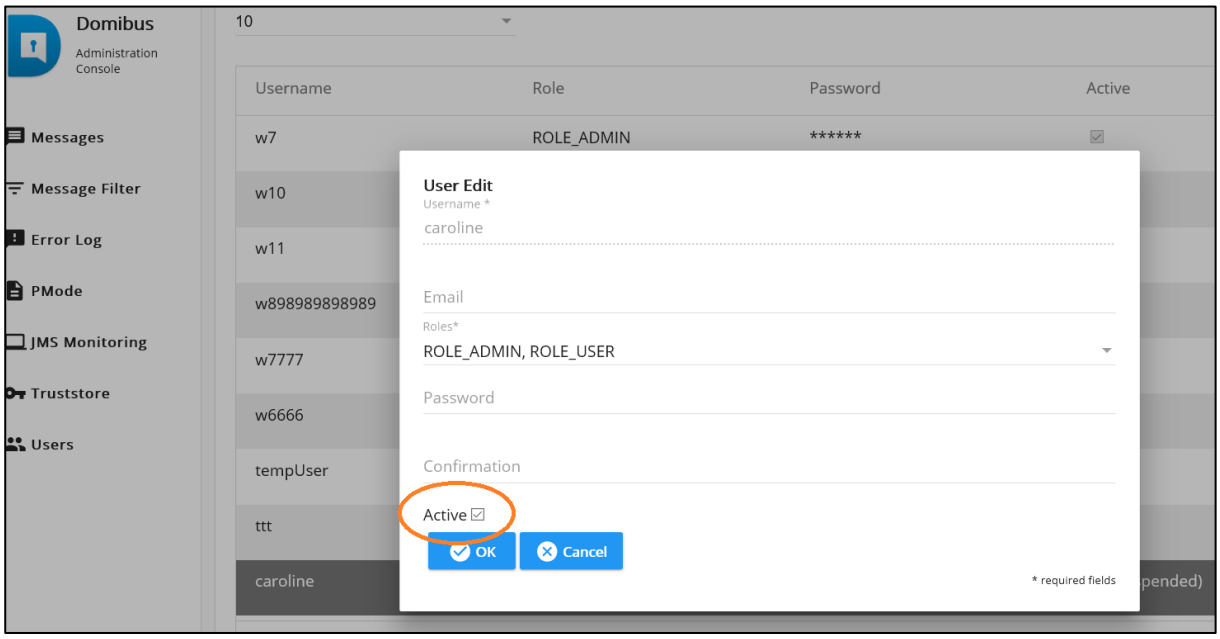
By default, a user remains suspended during one hour before his account is automatically unlocked and the user can try to log again.

If the user wants his account to be unlocked without waiting the default one hour, he can ask his administrator to unlock the account. To unlock the account, the administrator must change the user's status on the Admin Console from "Suspended" to "Active".

Select the suspended user and click on "Edit":



Re-activate the user (unlock it) by checking the "Active" status and confirming with OK:



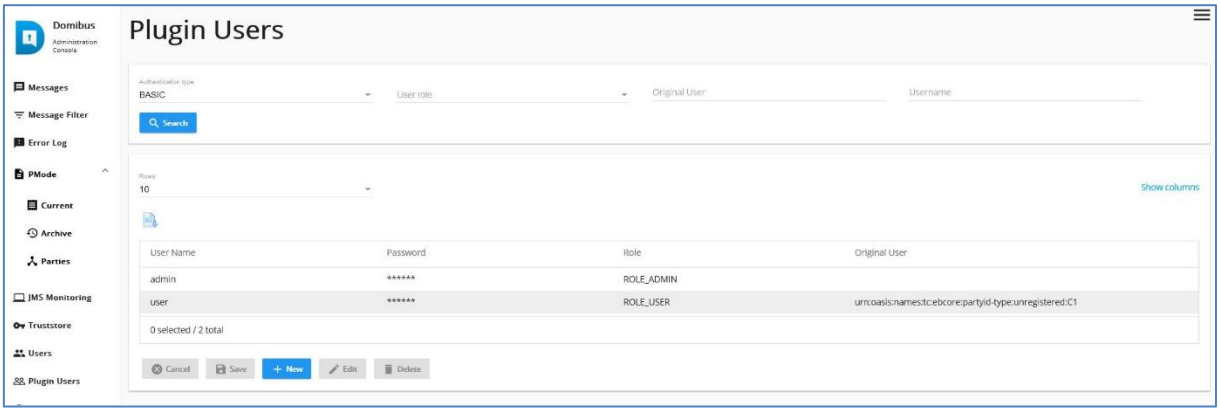
Do not forget to click on **Save** on the next window and then on **Yes** to confirm the change.

9.11. Plugin Users

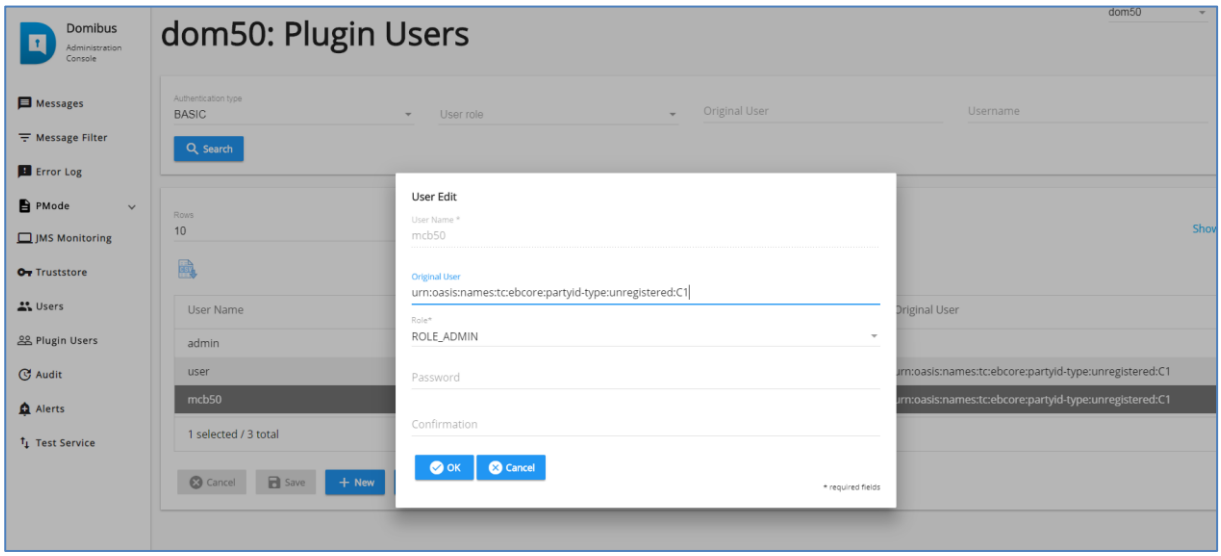
In Multitenancy mode the plugins security is activated by default, no matter if value configured in domibus.properties for the **domibus.auth.unsecureLoginAllowed** property.

This is needed in order to identify the request performed by the user and associate it to a specific domain. As a result, every request sent to Domibus needs to be authenticated.

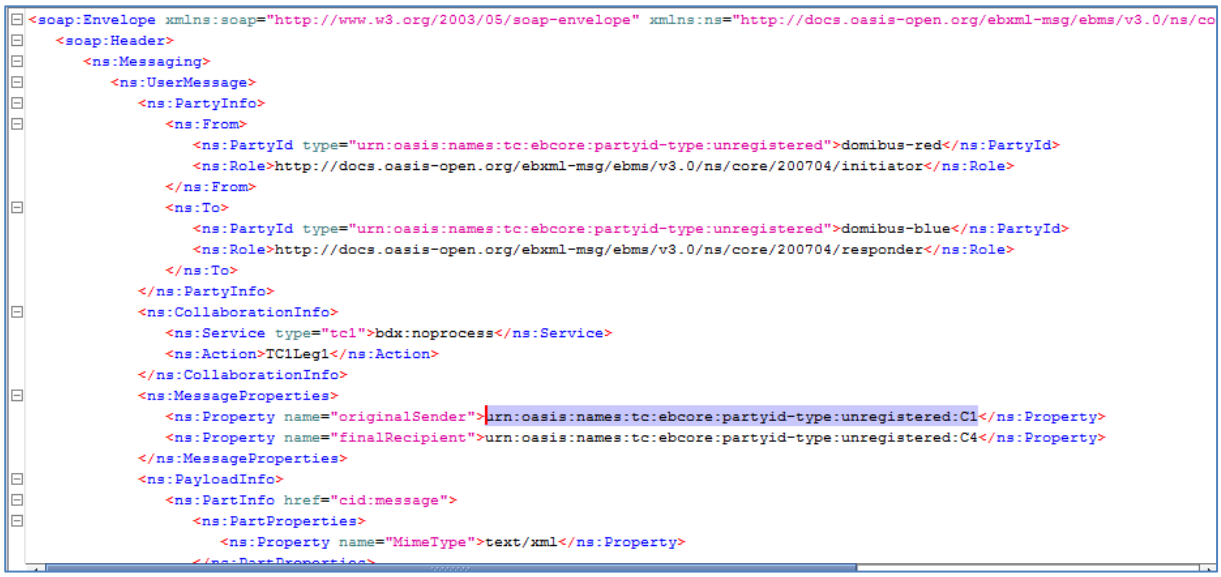
A plugin must use a configured plugin user associated to a specific domain in order to authenticate every request sent to Domibus. The management of the plugin users is implemented in the **Plugin Users** page:



The example below shows a **plugin user** that has been added:



Note that the Original user ID can be obtained from the **orginalSender** Property in **the SoapUI** project as shown here:



Do not forget to click on **Save** on the next window and then on **Yes** to confirm the change.

9.12. Audit

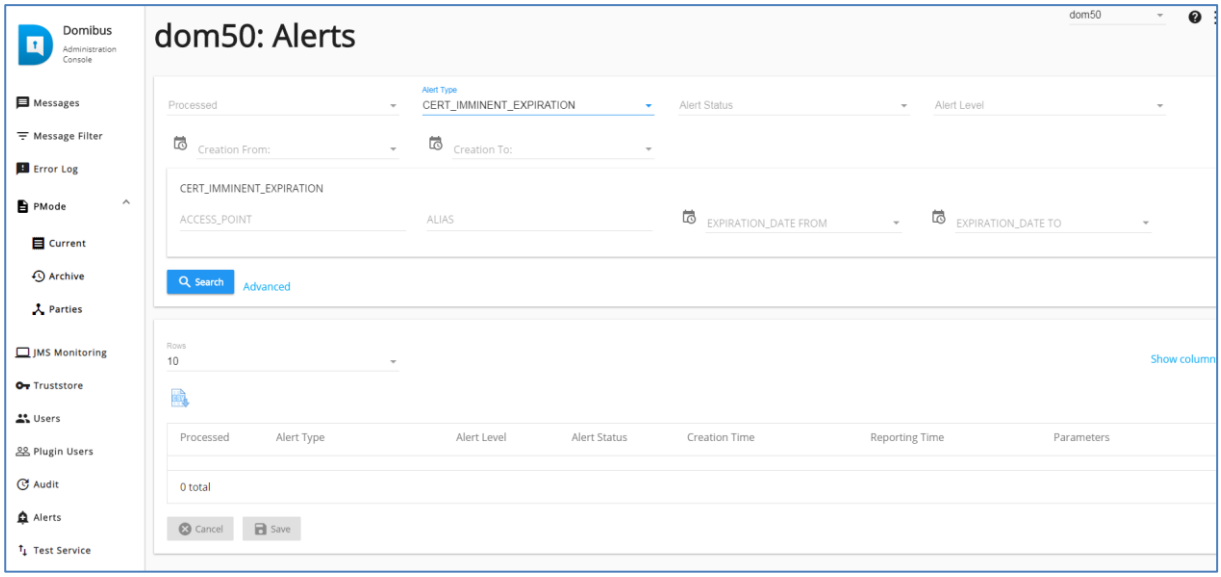
Audit support: Domibus keeps track of changes performed in the PMode, Parties, Message Filter and Users pages.

9.13. Alerts

Users can configure the alert feature as described in §18 – *"Alerts"*.
The purpose of the alert feature is to use different available media to notify the Domibus administrator in case of unusual behaviour. Currently alerts can be sent via mail.

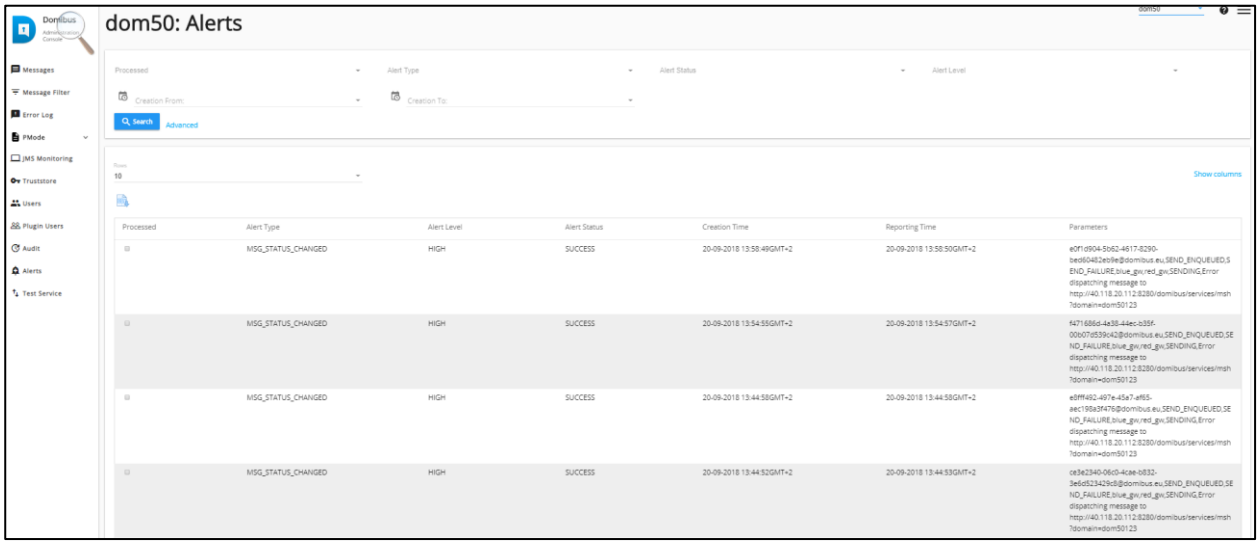
There are three types of alerts that can be configured: Message status change, Authentication issues and Certificate expiration.

Example: If the **CERT_IMMINENT_EXPIRATION** alert is chose, the following screen is presented:



The generated alerts can be checked in the **Alerts** page of the Administration console.

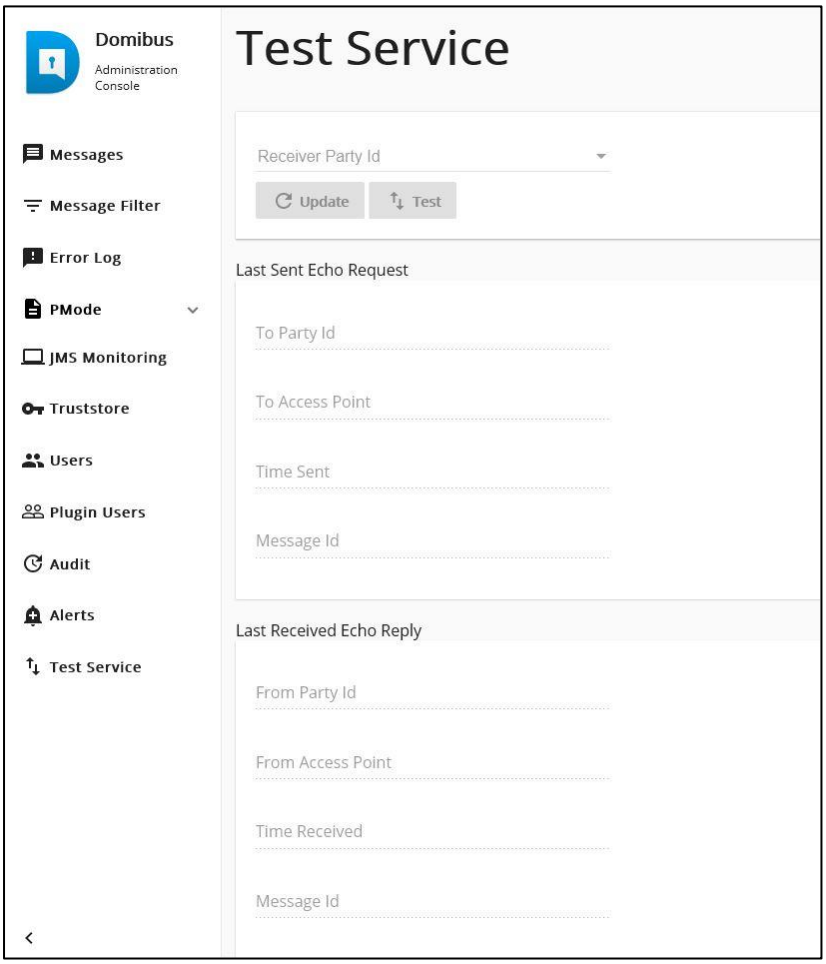
9.13.1. Example: Alerts on SEND_FAILURE



9.14. Test Service

The test service allows communication partners to perform a basic test of the communication configuration (including security at network, transport and message layer, and reliability) in any environment, including the production environment.

On the Test Service page of the administration console you can inspect and send Test messages to other Access Points. Information on both messages are available. Every time “Receiver Party Id” changes, you are able to check the updated information of Last Sent and Last Received Test Messages.



10. LARGE FILES SUPPORT

Domibus supports transfers between Access Points of files up to 2 GB using Java 8. In order to compute the message signature, Domibus loads the whole message into memory using a byte array. In Java, byte arrays can hold a maximum of 2 GB hence the Domibus limitation of 2 GB.

In order to optimize the sending of such large files, HTTP chunking is activated by default in the connection with the receiver Access Points. As chunked encoding is useful when sending larger amounts of data but decreases the performance on smaller amounts, Domibus uses a threshold to activate the chunking when appropriate only.

The following properties are used to configure chunking: **domibus.dispatcher.allowChunking** and **domibus.dispatcher.chunkingThreshold.** For more information about these properties, please refer to §5.2-*"Domibus Properties"*.

11. Data Archiving

11.1. What's archiving?

Data archiving consists of moving messages that have been processed successfully or unsuccessfully by the access point to an external storage location for long-term retention.
Archived data consists of older data that have been processed at the communication level by the access points that are still significant to the business and may be needed for future reference. They may also be retained for legal constraints.
Data archives are indexed and searchable to allow easy retrieval.
It is not recommended to use Domibus as an archiving solution. Nevertheless, if the data really needs to be stored for long periods, then it is possible to set the Data Retention Policy to allow it to be extracted from the database through the webservices or through an external archiving tool.

11.2. Data Retention Policy

A data retention policy is a procedure established by the business for continuous information storage for operational, legal or compliance reasons.
The data retention policy needs to be defined based on the business needs and constraints.
In Domibus, the data retention policy can be found in the PMode file:

```
<mpcs>
    <mpc name="defaultMpc"
         qualifiedName="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/defaultMPC"
         enabled="true"
         default="true"
         retention_downloaded="0"
         retention_undownloaded="14400"/>
</mpcs>
```

In the above extract of the sample PMode configuration of Domibus, the data retention policy is set to **14400 minutes** (10 days) if the message is not downloaded. This means that if the message is not downloaded, it will be deleted and then only the metadata containing the information of the receiver and the acknowledgement will be retained.

The data retention policy is set by default to **0 minutes** if the message is downloaded. This means that the message will be instantaneously deleted as soon as it is downloaded. These two parameters, retention_downloaded and retention_undownloaded, can therefore be modified to meet the needs of the business.

11.3. Data Extraction

In order to keep the metadata and the payload of the message for a longer period than the one set, in the PMode, it is recommended to extract it to an external storage. As long as the retention worker does not delete it, data can be extracted through the webservices or through an external archiving tool.

For more information, please refer to the Data Model provided in the "Domibus Software Architecture Document" that can be found on the CEF Digital single web portal [REF6].

12. NON REPUDIATION

In order to guarantee non-repudiation, the sending Access Point (C2) stores the full **SignalMessage**, including the **MessageInfo,** the Receipt (that contains the **NonRepudiationInformation** for each part) and the signature of the receipt by the receiver Access Point (C3).

This will guarantee that the receiver Access Point (C3) cannot deny having received a message from the sender Access Point (C2) during the sending process. However; if the initial sender (C1) wants to be sure that the final recipient (C4) cannot deny having received a specific content inside this message, then the sender must be able to show the specific content that was used to produce the receiver Access Point (C3) signature.

Domibus, as a sending Access Point (C2), keeps track of the metadata of the sent messages but does not store the actual message payloads. Therefore; it is recommended that the initial sender (C1) stores the message payloads safely for the time needed to guarantee non-repudiation of the sent messages.

In order to guarantee non-repudiation, the receiving Access Point (C3) stores the full UserMessage and the associated signature of the sender (C2).

This will guarantee that the sender Access Point (C2) cannot deny having sent a message to the receiver during the sending process. However; if the final recipient (C4) wants to be sure that the sender cannot deny having sent a specific content inside this message, then the final recipient (C4) must be able to show the specific content that was used to produce the sender Access Point signature (C2).

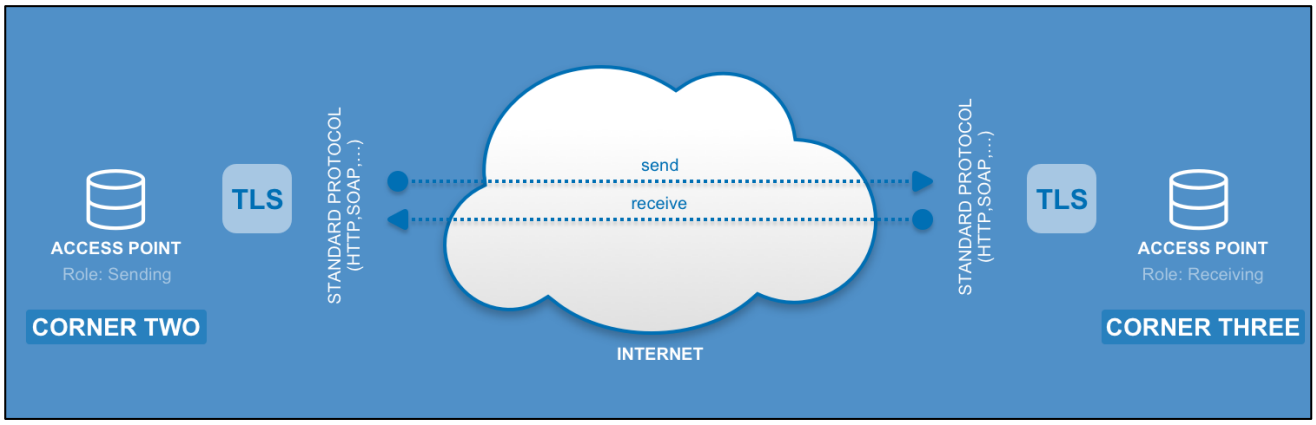
Domibus, as a receiving Access Point (C3), keeps track of the metadata of the received messages and will store the message payloads, only for the (limited) duration configured in the retention period (specified in the PMode). Therefore, it is recommended that the final recipient (C4) either stores the message payloads safely or aligns the retention period on the receiving Access Point (C3) with the time needed to guarantee non-repudiation of the received messages.

13. TLS CONFIGURATION

13.1. TLS Configuration

13.1.1. Transport Layer Security in Domibus

In addition to the message level security, Domibus may be configured to exchange messages using TLS (HTTPS). The use of TLS is mandatory according to the eDelivery AS4 profile. However, you can choose to configure it in the Access Point itself or delegate it to another appropriate network component.



13.1.2. Client Side Configuration

The implementation of the Domibus MSH is based on the CXF framework. According to CXF documentation, when using an "https" URL, CXF will, by default, use the certs and keystores that are part of the JDK. For many HTTPs applications, that is enough and no configuration is necessary. However, when using custom client certificates or self signed server certificates or similar, you may need to specifically configure in the keystores and trust managers and such to establish the SSL connection.

Apache provides full description of all possible configuration of the **tlsClientParameters** [see http://cxf.apache.org/docs/client-http-transport-including-ssl-support.html#ClientHTTPTransport(includingSSLsupport)-ConfiguringSSLSupport].

In Domibus, the TLS configuration is read from the file *cef_edelivery_path***/conf/domibus/clientauthentication.xml** and it applies to all the domains when Domibus is configured in multi tenancy mode.

Below example presents two possible configurations, One-Way SSL and Two-Way SSL:

clientauthentication.xml – One-Way SSL

```
<http-conf:tlsClientParameters disableCNCheck="true" secureSocketProtocol="TLSv1.2"
      xmlns:http-conf="http://cxf.apache.org/transports/http/configuration"
      xmlns:security="http://cxf.apache.org/configuration/security">

<security:trustManagers>
      <security:keyStore type="JKS" password="your_trustore_password"
                  file="${domibus.config.location}/keystores/your_trustore_ssl.jks"/>
</security:trustManagers>
```

```
</http-conf:tlsClientParameters>
```

In One-Way SSL, the sender validates the signature of the receiver using the public certificate of the receiver, provided in *your_trustore_ssl*.jks.

clientauthentication.xml – Two-Way SSL

```
<http-conf:tlsClientParameters disableCNCheck="true" secureSocketProtocol="TLSv1.2"
      xmlns:http-conf="http://cxf.apache.org/transports/http/configuration"
      xmlns:security="http://cxf.apache.org/configuration/security">
   <security:trustManagers>
      <security:keyStore type="JKS" password="your_trustore_password"
           file="${domibus.config.location}/keystores/your_trustore_ssl.jks"/>
   </security:trustManagers>
     <security:keyManagers keyPassword="your_keystore_password">
      <security:keyStore type="JKS" password="your_keystore_password"
           file="${domibus.config.location}/keystores/your_keystore_ssl.jks"/>
   </security:keyManagers>
</http-conf:tlsClientParameters>
```

In Two-Way SSL, both the sender and the receiver sign the request and validate the trust of the other party. In addition to the public certificate of the receiver (*your_trustore_ssl*.jks), the private certificate of the sender is also configured (*your_keystore_ssl*.jks).

Remark:

TLSv1.2 is mandatory for eDelivery AS4 Profile.

When self-signed certificates are used, the CN check must be disabled: **disableCNCheck="true"**.

The attribute **disableCNCheck** specifies whether JSSE should omit checking if the host name specified in the URL matches the host name specified in the Common Name (CN) of the server's certificate. The attribute is "false" by default and must not be set to "true" during production use (cf.**[REF7]**).

13.1.3. Server side configuration

13.1.3.1. Tomcat 8

In Server.xml, add a new connector with the **SSLEnabled** attribute set to "true":

```
<Connector SSLEnabled="true"
        protocol="org.apache.coyote.http11.Http11NioProtocol"
        port="8443" maxThreads="200"
        scheme="https" secure="true"
        keystoreFile="${domibus.config.location}/keystores/your_keystore_ssl.jks"
keystorePass="your_keystore_password"
        clientAuth="false" sslProtocol="TLS" />
```

The keystore jks location and password must be specified, otherwise the default ones will be taken into account.

TLS version can also be specified.

The above connector has **clientAuth="false"**, which means that only the server has to authenticate itself (One Way SSL). To configure "Two Way SSL", which is optional in the eDelivery *AS4* Profile, set

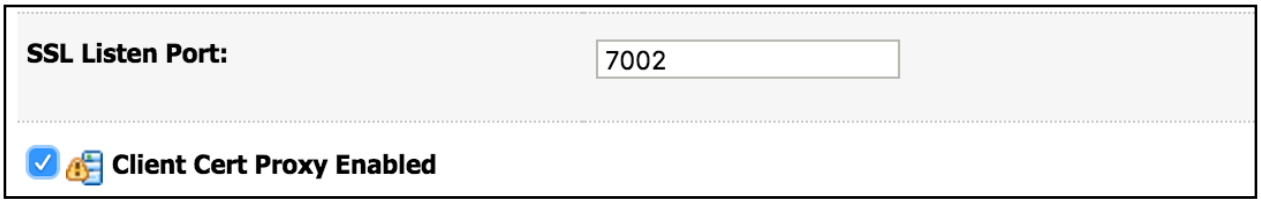
clientAuth="true" in Server.xml and provide the location of the *your_truststore_ssl***.jks** file so that the server can verify the client:

```
<Connector SSLEnabled="true"
        protocol="org.apache.coyote.http11.Http11NioProtocol"
        port="8443" maxThreads="200"
        scheme="https" secure="true"
        keystoreFile="${domibus.config.location}/keystores/your_keystore_ssl.jks"
keystorePass="your_keystore_password"
        truststoreFile="${domibus.config.location}/keystores/your_truststore_ssl.jks"
truststorePass="your_trustore_password"
        clientAuth="true" sslProtocol="TLS" />
```

13.1.3.2. WebLogic

1. Specify the use of SSL on default port 7002

Go to Servers ➔ select Server Name ➔ Configuration ➔ General then **click** on **Client Cert Proxy Enabled**:



2. Add keystore and truststore:

Go to Servers ➔ select Server Name ➔ Configuration ➔ Keystores and SSL tabs and use **Custom Identity and Custom Trust** then set keystore and trustore jks.

To disable basic authentication at WebLogic level:

By default WebLogic performs its own basic authentication checks before passing the request to Domibus. As we want basic authentication to be performed by Domibus, we need to disable it at the application server level.

To do so, in **DOMAIN_HOME/config/config.xml,** add:

```
<enforce-valid-basic-auth-credentials>false</enforce-valid-basic-auth-credentials>
```

13.1.3.3. WildFly 9

In the *cef_edelivery_path***/domibus/standalone/configuration/standalone-full.xml file***:*

- add the keystore and trustore jks file names to the ApplicationRealm:

```
<security-realm name="ApplicationRealm">
 <server-identities>
  <ssl>
```

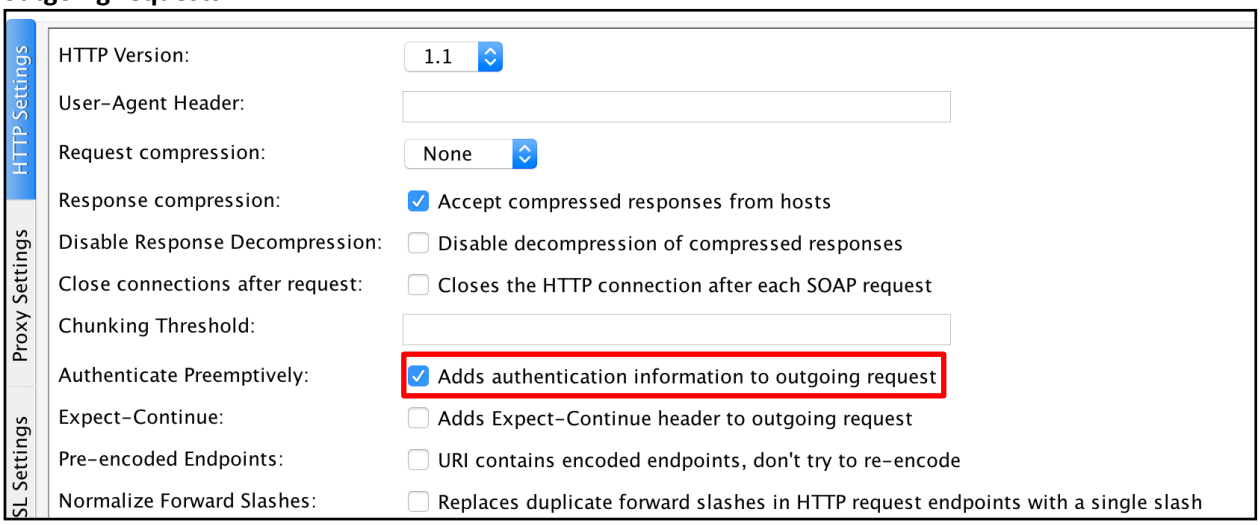
```
    <keystore path="../conf/domibus/keystores/gateway_keystore.jks" relative-
to="jboss.server.base.dir" keystore-password="test123" alias="blue_gw" key-
password="test123"/>
   </ssl>
  </server-identities>
  <authentication>
    <truststore path="../conf/domibus/keystores/gateway_truststore.jks" relative-
to="jboss.server.base.dir" keystore-password="test123" />
...
</authentication>
```

- add https-listener to default-server:

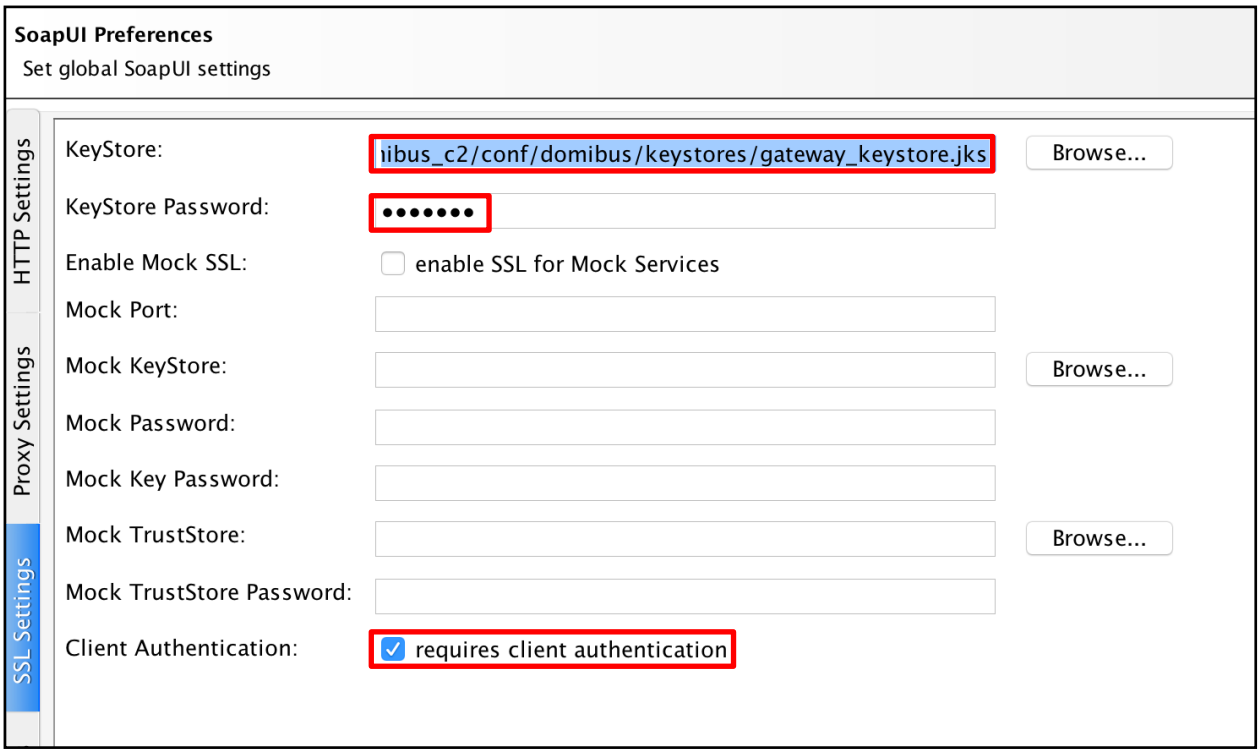
```
<subsystem xmlns="urn:jboss:domain:undertow:2.0">
        <buffer-cache name="default"/>
        <server name="default-server">
          <http-listener name="default" socket-binding="http" redirect-socket="https"/>
          <https-listener name="default_https" socket-binding="https" security-
realm="ApplicationRealm" verify-client="REQUIRED"/>
```

13.1.3.4. Configure Basic and Certificates authentication in SoapUI

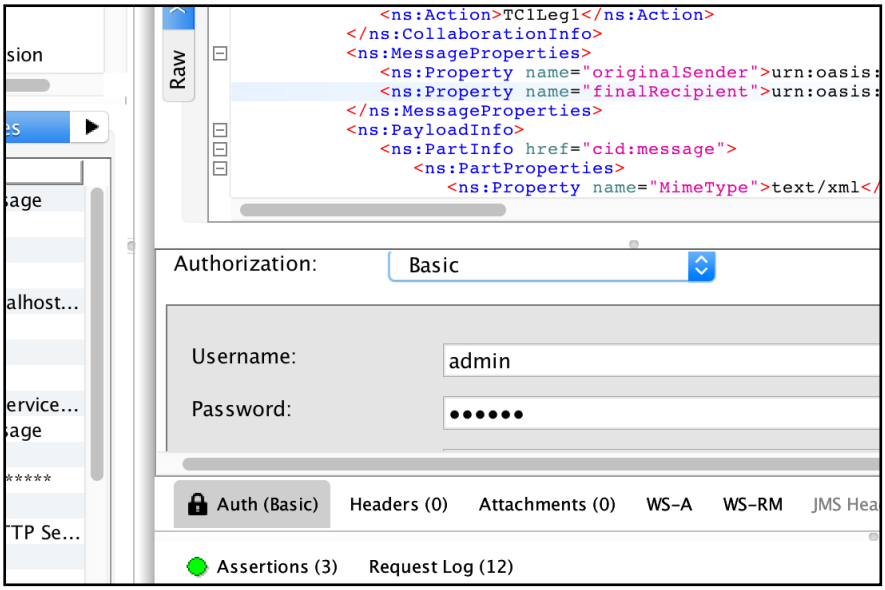
Go to File ➔ Preferences ➔ HTTP Settings and check the option **Adds authentication information to outgoing requests**:



Go to File ➔ Preferences ➔ SSL Settings, add the **KeyStore** and **KeyStore Password** and check the option **requires client authentication**:



To allow Basic Authentication, select the Auth tab, click Add New Authorization and select Basic. Enter user and password (e.g: Username = ***admin***; Password = ***123456***)



13.1.3.5. PMode update

If you enable HTTPS then your PMode Configuration Manager needs to make sure that all other endpoint PModes are modified accordingly.

With the SSL connector configured as above, the MSH endpoint is now: **https://***your_domibus_host***:*8443*/domibus/services/msh**.

After the updates, upload the PModes via the Admin Console:

Example:

```
<party name="party_id_name1"
endpoint=
"https:// party_id_name1_hostname:8443/domibus/services/msh">
```

14. DYNAMIC DISCOVERY OF UNKNOWN PARTICIPANTS

14.1. Overview

In a dynamic discovery setup, the sender and/or the receiver parties and their capabilities are not configured in advance.

The sending Access Point will dynamically retrieve the necessary information for setting up an interoperability process from the Service Metadata Publisher (SMP). The SMP stores the interoperability metadata which is a set of information on the recipient or end entity (its identifier, supported business documents and processes) and AP (metadata which includes technical configuration information on the receiving endpoint, such as the transport protocol and its address) cf.[REF8].

The receiving AP registers its metadata in the SMP and configures the PMode to be able to accept messages from trusted senders that are not previously configured in the PMode. The receiving AP will have to configure one process in its PMode for each SMP entry.

The mapping between the PMode process and the SMP entry is defined for PEPPOL in "§14.3 – *PMode configuration for PEPPOL"* and for OASIS in "§14.8 - *PMode configuration for OASIS*".

Please note that the sender does not have to be registered in the SMP and the receiver merely extracts its identifier from the received message.

The following sections describe how to configure Domibus AP in order to use Dynamic Discovery (§14.3 – *"PMode configuration for PEPPOL"*, §14.3.3 – "*Sender and Receiver PMode*

Dynamic discovery configuration when the Access Point acts as both sender and receiver would look like this:

```
...
<services>
    <service name="testService1"
        value="urn:www.cenbii.eu:profile:bii05:ver2.0"
        type="urn:fdc:peppol.eu:2017:identifiers:proc-id"/>
</services>
<actions>
    <action name="tc1Action"
        value=" busdox-docid-qns:: urn:oasis:names:specification:ubl:schema:xsd:CreditNote-
2::CreditNote##urn:www.cenbii...."/>
</actions>
<securities>
    <security name="eDeliveryAS4Policy_BST"
        policy="eDeliveryAS4Policy_BST.xml"
        signatureMethod="RSA_SHA256"/>
</securities>
<legConfigurations>
    <legConfiguration name="pushTestcase1tc1Action"
    service="testService1"
    action="tc1Action"
    defaultMpc="defaultMpc"
    reliability="AS4Reliability"
    security="eDeliveryAS4Policy_BST"
    receptionAwareness="receptionAwareness"
```

```
    propertySet="eDeliveryPropertySet"
    payloadProfile="MessageProfile"
    errorHandling="demoErrorHandling"
    compressPayloads="true"/>
</legConfigurations>
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <initiatorParties>
        <initiatorParty name="senderalias"/>
    </initiatorParties>
    <!-- no responderParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
<process name="tc2Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <responderParties>
        <responderParty name="receiveralias"/>
    </responderParties>
    <!-- no initiatorParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
```

Policy and certificates for PEPPOL", §14.8 – "*PMode configuration for OASIS",* §14.9 – "*Policy and certificates for OASIS"*).

14.2. Domibus configuration for PEPPOL

To enable the integration with the SMP/SML components, Domibus requires some changes in the domibus.properties configuration file which include:

1. Adding the following properties to enable the usage of the PEPPOL dynamic discovery client:

```
domibus.dynamicdiscovery.client.specification">PEPPOL
```

2. Setting the dynamic discovery client to use certificates to access the SMP. These certificates are different in TEST and PRODUCTION, therefore we need to specify the Mode used by the dynamic discovery client by setting the following property:

   domibus.dynamicdiscovery.peppolclient.mode">TEST

3. Setting the "**domibus.smlzone**" property.

14.3. PMode configuration for PEPPOL

14.3.1. Sender PMode

In a dynamic discovery process, the receiver of the messages is not known beforehand and therefore the **PMode.Responder** parameter SHOULD NOT be set.

The dynamic discovery process must include a leg which maps the configured entry (action, service and service type – see section. 14.5 – "*Message format for PEPPOL*") of the Receiver in the SMP.

The security policy to be used in the leg is the policy that embeds the Binary Security Token into the security header (see section §5.1.1 – *"Security Policies"* for more information):

```
security="eDeliveryAS4Policy_BST"
```

Sample Sender PMODE configuration extract:

```
...
<services>
    <service name="testService1"
        value="urn:www.cenbii.eu:profile:bii05:ver2.0"
        type="urn:fdc:peppol.eu:2017:identifiers:proc-id"/>
</services>
<actions>
    <action name="tc1Action"
        value=" busdox-docid-qns:: urn:oasis:names:specification:ubl:schema:xsd:CreditNote-
2::CreditNote##urn:www.cenbii...."/>
</actions>
<securities>
    <security name="eDeliveryAS4Policy_BST"
        policy="eDeliveryAS4Policy_BST.xml"
        signatureMethod="RSA_SHA256"/>
</securities>
<legConfigurations>
    <legConfiguration name="pushTestcase1tc1Action"
    service="testService1"
    action="tc1Action"
    defaultMpc="defaultMpc"
    reliability="AS4Reliability"
    security="eDeliveryAS4Policy_BST"
    receptionAwareness="receptionAwareness"
    propertySet="eDeliveryPropertySet"
    payloadProfile="MessageProfile"
    errorHandling="demoErrorHandling"
    compressPayloads="true"/>
</legConfigurations>
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
```

```
    <initiatorParties>
        <initiatorParty name="senderalias"/>
    </initiatorParties>
    <!-- no responderParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
…
```

14.3.2. Receiver PMode

Dynamic discovery configuration of the receiver is similar to the configuration of the sender, except that the roles are swapped: the sender of the messages is not known beforehand. As a consequence the ***PMode.Initiator*** parameter SHOULD NOT be set.

```
…
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <responderParties>
        <responderParty name="receiveralias"/>
    </responderParties>
    <!-- no initiatorParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
…
```

14.3.3. Sender and Receiver PMode

Dynamic discovery configuration when the Access Point acts as both sender and receiver would look like this:

```
…
<services>
    <service name="testService1"
        value="urn:www.cenbii.eu:profile:bii05:ver2.0"
        type="urn:fdc:peppol.eu:2017:identifiers:proc-id"/>
</services>
<actions>
    <action name="tc1Action"
        value=" busdox-docid-qns:: urn:oasis:names:specification:ubl:schema:xsd:CreditNote-
2::CreditNote##urn:www.cenbii...."/>
</actions>
<securities>
    <security name="eDeliveryAS4Policy_BST"
        policy="eDeliveryAS4Policy_BST.xml"
```

```
        signatureMethod="RSA_SHA256"/>
</securities>
<legConfigurations>
    <legConfiguration name="pushTestcase1tc1Action"
    service="testService1"
    action="tc1Action"
    defaultMpc="defaultMpc"
    reliability="AS4Reliability"
    security="eDeliveryAS4Policy_BST"
    receptionAwareness="receptionAwareness"
    propertySet="eDeliveryPropertySet"
    payloadProfile="MessageProfile"
    errorHandling="demoErrorHandling"
    compressPayloads="true"/>
</legConfigurations>
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <initiatorParties>
        <initiatorParty name="senderalias"/>
    </initiatorParties>
    <!-- no responderParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
<process name="tc2Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <responderParties>
        <responderParty name="receiveralias"/>
    </responderParties>
    <!-- no initiatorParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
```

14.4. Policy and certificates for PEPPOL

The receiver must include the certificate of the trusted authority(ies) in its truststore. It will only accept messages that were signed with certificates issued by the trusted authority(ies) (cf. §21 – "Annex 1 - Usage of certificates in PEPPOL and OASIS" for more information).

14.5. Message format for PEPPOL

When dynamic discovery is used, the "to" field should not be statically configured in the PMode (the "to" field may even be omitted in the message). The lookup is performed by C2 based on the **finalRecipient** message property.

Note: In Peppol, the service@type has a fixed value while the service@value is made of ProcessIdentifier@Scheme::ProcessIdentifier

Example of a message using the **finalRecipient** for dynamic discovery:

```
<ns:UserMessage>
 <ns:PartyInfo>
  <ns:From>
   <ns:PartyId type="urn:fdc:peppol.eu:2017:identifiers:ap">senderalias</ns:PartyId>
   <ns:Role> http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/initiator</ns:Role>
  </ns:From>
  <ns:To>
  </ns:To>
 </ns:PartyInfo>
 <ns:CollaborationInfo>
     <ns:Service type=" urn:fdc:peppol.eu:2017:identifiers:proc-
id">urn:www.cenbii.eu:profile:bii05:ver2.0</ns:Service>
<ns:Action>busdox-docid-qns:: urn:oasis:names:specification:ubl:schema:xsd:CreditNote-
2::CreditNote##urn:www.cenbii.eu:transaction:biitrns014:ver2.0:extended:urn:www.peppol.eu:bis:pep
pol5a:ver2.0::2.1</ns:Action>
 </ns:CollaborationInfo>
  <ns:MessageProperties>
   <ns:Property name="originalSender">urn:oasis:names:tc:ebcore:partyid-
type:unregistered:C1</ns:Property>
    <ns:Property name="finalRecipient" type="iso6523-actorid-
upis">0007:9340033829test1</ns:Property>
  </ns:MessageProperties>
</ns:UserMessage>
```

14.6. SMP entry

The following table describes the mapping between the PMode static configuration and the dynamic SMP records structure:

SMP Endpoint registration record	PMode attributes
ServiceMetadata/ServiceInformation/ProcessIdentifier	PMode[1].BusinessInfo.Service
ServiceMetadata/ServiceInformation/DocumentIdentifier	Pmode[1].BusinessInfo.Action
ServiceInformation/Processlist/Process/ServiceEndpointList/Endpoint/EndpointReference/Address	Pmode[].Protocol.Address

Table 4 - SMP Entry Mapping

The Service Metadata Record also provides the receiving end's certificate. This certificate can be used to encrypt the message to be sent to the receiver. The certificate can also provide the name of the Access Point for this PMode by using the Certificate's CNAME as the PMode identifier (cf.[REF9]).

14.7. Domibus configuration for OASIS

To enable the integration with the SMP/SML components, Domibus requires some changes in the domibus.properties configuration file:

1. Add the following properties to enable the usage of the OASIS dynamic discovery client:

   domibus.dynamicdiscovery.client.specification"> OASIS

   Note: this property is not mandatory as it defaults to the above value.
2. Set the property "**domibus.smlzone**", e.g. "ehealth.acc.edelivery.tech.ec.europa.eu"

14.8. PMode configuration for OASIS

14.8.1. Sender PMode

In a dynamic discovery process, the receiver of the messages is not known beforehand and therefore the **PMode.Responder** parameter SHOULD NOT be set.

The dynamic discovery process must include a leg which maps the configured entry (action, service and service type – cf. 14.10 – "Message format for PEPPOL") of the Receiver in the SMP.

The security policy to be used in the leg is the policy that embeds the Binary Security Token into the security header (see section §5.1.1 – *"Security Policies"* for more information):

```
security="eDeliveryAS4Policy_BST"
```

Sample Sender PMODE configuration extract:

```
...
<services>
    <service name="testService1"
        value="urn:www.cenbii.eu:profile:bii05:ver2.0"
        type="cenbii-procid-ubl"/>
</services>
<actions>
    <action name="tc1Action"
        value="'your-schema-name'::urn:oasis:names:specification:ubl:schema:xsd:CreditNote-
2::CreditNote##urn:www.cenbii...."/>
</actions>
<securities>
    <security name="eDeliveryAS4Policy_BST"
        policy="eDeliveryAS4Policy_BST.xml"
        signatureMethod="RSA_SHA256"/>
</securities>
<legConfigurations>
    <legConfiguration name="pushTestcase1tc1Action"
    service="testService1"
    action="tc1Action"
    defaultMpc="defaultMpc"
    reliability="AS4Reliability"
    security="eDeliveryAS4Policy_BST"
    receptionAwareness="receptionAwareness"
    propertySet="eDeliveryPropertySet"
    payloadProfile="MessageProfile"
    errorHandling="demoErrorHandling"
    compressPayloads="true"/>
</legConfigurations>
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <initiatorParties>
        <initiatorParty name="senderalias"/>
    </initiatorParties>
    <!-- no responderParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
...
```

Remark:

Schema name should be added to action value. E.g: ***ehealth-actorid-qns::urn:oasis:names:specification:ubl:schema:xsd:CreditNote-2::CreditNote##urn:www.cenbii...***

14.8.2. Receiver PMode

The dynamic discovery configuration of the receiver is similar to the configuration of the sender, except that the roles are swapped: the sender of the messages is not known beforehand. As a consequence, the **PMode.Initiator** parameter SHOULD NOT be set.

```
...
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    inding="push"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <responderParties>
        <responderParty name="receiveralias"/>
    </responderParties>
    <!-- no initiatorParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
...
```

14.9. Policy and certificates for OASIS

The receiver must include the certificate of the trusted authority(ies) in its truststore. It will only accept messages that were signed with certificates issued by the trusted authority(ies).

The sender truststore must include the SMP public certificate. This certificate is used by the AP to validate the identity of the used SMP (cf. §21 –Annex 1 - Usage of certificates in PEPPOL and OASIS for more information).

14.10. Message format for OASIS

When dynamic discovery is used, the "to" field should not be statically configured in the PMode (the "to" field may even be omitted in the message). The lookup is performed by C2 based on the **finalRecipient** message property.

Note 1: For OASIS clients: in the PMode "action" value, the document scheme must be included with the document ID (for PEPPOL client, busdox-docid-qns:: should be preappended to the document ID).

Note 2: For OASIS clients: the value of the “service@type” must be set to the “processIdentifier@scheme”.

Example of message using the **finalRecipient** for dynamic discovery:

```
<ns:UserMessage>
 <ns:PartyInfo>
  <ns:From>
   <ns:PartyId type="urn:oasis:names:tc:ebcore:partyid-type:unregistered">senderalias</ns:PartyId>
   <ns:Role>http://docs.oasis-open.org/ebxml-msg/ebms/v3.0/ns/core/200704/initiator</ns:Role>
```

```
   </ns:From>
   <ns:To>
   </ns:To>
  </ns:PartyInfo>
  <ns:CollaborationInfo>
   <ns:Service type="cenbii-procid-ubl">urn:www.cenbii.eu:profile:bii05:ver2.0</ns:Service>

          <ns:Action>'your_schema_name'::urn:oasis:names:specification:ubl:schema:xsd:CreditNote-
2::CreditNote##urn:www.cenbii.eu:transaction:biitrns014:ver2.0:extended:urn:www.peppol.eu:bis:pep
pol5a:ver2.0::2.1</ns:Action>
  </ns:CollaborationInfo>
   <ns:MessageProperties>
    <ns:Property name="originalSender">urn:oasis:names:tc:ebcore:partyid-
type:unregistered:C1</ns:Property>
    <ns:Property name="finalRecipient" type="iso6523-actorid-
upis">0007:9340033829test1</ns:Property>
   </ns:MessageProperties>
</ns:UserMessage>
```

15. MESSAGE PULLING

15.1. Setup

In order to configure message pulling the process section should be configured with **mep** set to "oneway" and binding set to "pull" as shown in the following example:

```
<process name="tc1Process"
    agreement="agreementEmpty"
    mep="oneway"
    binding="pull"
    initiatorRole="defaultInitiatorRole"
    responderRole="defaultResponderRole">
    <initiatorParties>
        <initiatorParty name="initiatoralias"/>
    </initiatorParties >
    <responderParties>
        <responderParty name="receiveralias"/>
    </responderParties>
    <!-- no initiatorParties element -->
    <legs>
        <leg name="pushTestcase1tc1Action"/>
    </legs>
</process>
```

In the case of a pull process, the **initiatorParties** section contains the party that initiate the pull request. The **responderParties** section contains the parties that can be pulled from.

In domibus.properties configuration file adapt the following properties to your needs. Note that domibus.msh.pull.cron and domibus.pull.queue.concurency are mandatory.

```
# ------------------------------- Pulling-------------------------------------

#Cron expression used for configuring the message puller scheduling.
domibus.msh.pull.cron=0 0 0/1 * * ?

# Number of threads used to parallelize the pull requests.
domibus.pull.queue.concurency=1-1

#Number or pull requests executed every cron cycle
#domibus.pull.request.send.per.job.cycle=1
```

15.2. Configuration restriction

A correctly configured **one-way pull process** should only contain one party configured in the **initiatorParties** section.

Different **legConfiguration** with the same **defaultMpc** (highlighted in red in the following configuration) should not be configured in the same pull process or across different pull processes.

If those restrictions are not respected, the message will not be exchanged and a warning message will detail the configuration problem.

```
<legConfiguration name="pushTestcase1tc2Action"
                                                    service="testService1"
                                                    action="tc2Action"
                                                    defaultMpc="defaultMpc"
                                                    reliability="AS4Reliability"
                                                    security="eDeliveryAs4Policy"
receptionAwareness="receptionAwareness"
                                                    propertySet="eDeliveryPropertySet"
                                                    payloadProfile="MessageProfile"
errorHandling="demoErrorHandling"
                                                    compressPayloads="true"/>
```

16. MULTITENANCY

Domibus supports multiple domains (configurations) configured in one Domibus instance. This means that each domain has its own configuration (PMode, keystore, truststore and Domibus properties, etc). These multiple configurations allow one Domibus instance to process messages from multiple domains simultaneously.

Domibus uses **Schema per tenant** strategy to implement Multitenancy, meaning that the data associated to a domain will be saved in a database schema dedicated to the domain.

16.1. Configuration

By default, Multitenancy is not activated. In order to activate Multitenancy, the following property that defines the database general schema needs to be configured in **domibus.properties.**

For Weblogic, this step can only be done after changing the Schema username and password as described in Section §16.1.4

```
domibus.database.general_schema=general_schema
```

Where *general_schema* is the database schema in which the association between users and domains is stored. The *general_schema* is not associated to any domain.

16.1.1. Database general schema

The *general_schema* needs to be initialized using the distributed database script **mysql5innoDb-x.y.z-multitenancy.ddl** for MySQL or **oracle10g-x.y.z-multitenancy.ddl** for Oracle**.**

Please find below the steps needed to create the *general_schema* for MySQL and Oracle.

16.1.1.1. MySQL

1. Unzip **domibus-distribution-X.Y.Z-sql-scripts.zip** in *cef_edelivery_path*/**sql-scripts**
2. Open a command prompt and navigate to this directory: *cef_edelivery_path*/**sql-scripts**
3. Execute the following MySQL commands at the command prompt:

```
mysql -h localhost -u root_user --password=root_password -e "drop schema if exists general_schema;create schema general_schema;alter database general_schema charset=utf8 collate=utf8_bin; create user edelivery_user@localhost identified by 'edelivery_password'; grant all on general_schema.* to edelivery_user@localhost;"
```

The above script creates a schema (*general_schema*) and a user (*edelivery_user*) that has all the privileges on the *general_schema*.

Remark:

The edelivery_user creation can be skipped if the user already exists.

You need to make sure the user edelivery_user is granted full rights on all schemas used for all the domains.

mysql -h *localhost* -u *root_user* **--password**=*root_password* general_*schema* **< mysql5innoDb-x.y.z-multitenancy.ddl**

The above command creates the required objects in ***general_schema***.

16.1.1.2. Oracle

1. Unzip **domibus-distribution-X.Y.Z-sql-scripts.zip** in *cef_edelivery_path*/**sql-scripts**

2. Open a command prompt and navigate to the following directory: *cef_edelivery_path*/**sql-scripts**

3. Execute the following commands at the command prompt:

sqlplus sys as sysdba (password should be the one assigned during the Oracle installation)
==========
Once logged in Oracle:

CREATE USER <edelivery_general_user> IDENTIFIED BY <edelivery_general_password>
DEFAULT TABLESPACE <tablespace>
QUOTA UNLIMITED ON <tablespace>;
GRANT CREATE SESSION TO <edelivery_general_user>;
GRANT CREATE TABLE TO <edelivery_general_user>;
GRANT CREATE SEQUENCE TO <edelivery_general_user>;
GRANT EXECUTE ON DBMS_XA TO <edelivery_general_user>;
GRANT SELECT ON PENDING_TRANS$ TO <edelivery_general_user>;
GRANT SELECT ON DBA_2PC_PENDING TO <edelivery_general_user>;
GRANT SELECT ON DBA_PENDING_TRANSACTIONS TO <edelivery_general_user>;

CONNECT **<** edelivery_general_user >
SHOW USER**;** (should return : edelivery_general_user)
@ oracle10g-x.y.z-multitenancy.ddl

EXIT
==========

Remarks:

1. Replace <edelivery_general_user> and <edelivery_general_password> with the corresponding values.

2. <tablespace> is created and assigned by your DBA; for local/test installations just replace it with users tablespace. The quota could be limited to a certain size.

3. DDL/SQL scripts must be run with the @ sign from the location of the scripts.

16.1.2. Creating new domains

A new domain can be created by adding a domain specific configuration file under the *cef_edelivery_path***/conf/domibus** directory. The domain configuration file name must start with the new domain name (**domain_name**) using the following convention:

domain_name-domibus.properties

The ***domain_name*** value is case-sensitive. It is a 50-character sequence of Unicode letters, digits or underscores characters. It must start with a letter and the subsequent characters may be letters, digits or underscore characters.

Each domain uses its own dedicated schema which is configured in the domain configuration file and has its own keystore, Truststore configured.

The domain database schema must be initialized using the distributed database script **mysql5innoDb-x.y.z.ddl** or **oracle10g-x.y.z.ddl**. For more information on how to execute these scripts, go to §4.1-"*Database Configuration"*.

Remarks:

*The **mysql5innoDb-x.y.z-data.ddl** or **oracle10g-x.y.z-data.ddl** database scripts **must not be executed in Multitenancy mode**.*

The database user used to connect to the ***general_schema*** schema must have the necessary privileges to access the database schemas for all the **configured domains**. Please follow the steps bellow for each Database type:

16.1.2.1. MySQL

Execute the following MySQL commands at the command prompt:

If the user ***edelivery_general_user*** is the one having rights on general schema for a particular domain schema just run:

mysql -h *localhost* -u *root_user* **--password=***root_password* **-e "grant all on** *domain_schema.** **to** *edelivery_general_user@localhost***;"**

Repeat this command for all the other domains.

16.1.2.2. Oracle

1. Unzip **domibus-distribution-X.Y.Z-sql-scripts.zip** in *cef_edelivery_path***/sql-scripts**
2. Open a command prompt and navigate to this directory: *cef_edelivery_path***/sql-scripts**.
3. Open a command line session, log in and execute the following commands to connect to current domain schema :

sqlplus s<domain_user>/<domain_password>@host:port/service
==========
Once logged in Oracle:
@oracle10g-4.0-multitenancy-rights.sql
==========

Before running this script, edit it and just replace domain_schema and general_schema values with the desired values. Repeat this command for each domain of the Multitenancy installation.

This script needs to be run after completing a migration of domain Domibus schema (new objects - table, view, sequence - could added in current domain schema).

Once Multitenancy is activated and with no other additional configuration, Domibus will use the domain named **default** for the incoming and outgoing messages. The domain **default** is configured in **domibus.properties.**

More information on how Multitenancy is implemented can be found in the **Domibus Software Architecture Document (c.f. [REF11]).**

16.1.3. Tomcat

The Domibus database in Tomcat is configured in the **domibus.properties** file. Running Domibus in **Multitenancy** mode requires that the some related database properties are adapted as shown in the following example:

```
domibus.database.general.schema=general_schema

# optional property used in case the default domain is used
domibus.database.schema=domibus

domibus.datasource.xa.property.url=jdbc:mysql://${domibus.database.serverName}:${domibus.database.port}/${domibus.database.general.schema}?pinGlobalTxToPhysicalConnection=true
# the user that has access to general_schema
domibus.datasource.xa.property.user=edelivery_user
domibus.datasource.xa.property.password=edelivery_password

domibus.datasource.url=jdbc:mysql://${domibus.database.serverName}:${domibus.database.port}/${domibus.database.general.schema}?useSSL=false
# the user that has access to general_schema
domibus.datasource.user=edelivery_user
domibus.datasource.password=edelivery_password
```

16.1.3.1. domain_name-domibus.properties configuration

Within the domain_name-domius.properties file, the domain_name field must be replaced by the actual name of the domain as shown in the following sample of the **dom50-domibus.properties** example, where **dom50** is the domain name created:

```
# ---------------------------------- GUI -------------------------
#The title shown in the Tab of Admin Console
#dom50.domibus.UI.title.name=windowTitle
#The name of the domain
#dom50.domain.title=domainTitle
#Number of console login attempt before the user is deactivated (default 5)
#dom50.domibus.console.login.maximum.attempt=5
#Time in seconds for a suspended user to be reactivated. (1 hour per default if property is not set, if 0 the user will not be reactivated)
#dom50.domibus.console.login.suspension.time=3600
#Max rows for CSV export
#dom50.domibus.ui.csv.max.rows=10000
```

```
# ---------------------------------- Keystore/Truststore ----------------------
#The location of the keystore
dom50.domibus.security.keystore.location=${domibus.config.location}/keystores/dom1_keystore.jks
#The type of the used keystore
dom50.domibus.security.keystore.type=jks
#The password used to load the keystore
dom50.domibus.security.keystore.password=test123
#Private key
#The alias from the keystore of the private key
dom50.domibus.security.key.private.alias=blue_gw
#etc...
```

16.1.4. WebLogic and WildFly

Most of the database configuration for WebLogic and WildFly is done in the application server. The datasources configured in the application server need to be configured with the user and password that has access to the *general_schema* schema and to all the domain schemas. At runtime the database schema will be changed based on the current domain.

16.1.5. WebLogic specific configuration

Activate the Multitenancy by configuring the following property in **domibus.properties**:

```
domibus.database.general_schema=general_schema
```

Disable basic authentication at the WebLogic level by setting the following property in **DOMAIN_HOME/config/config.xml** (End of the <security-configuration> tag):

```
<enforce-valid-basic-auth-credentials>false</enforce-valid-basic-auth-credentials>
```

Example:

```
<security-configuration>
….
<node-manager-password-
encrypted>{AES}hFKbHz7XZ19urplEtWmafYeUm9mr2yXEwyNC9ZpqJHY=</node-manager-password-
encrypted>
<enforce-valid-basic-auth-credentials>false</enforce-valid-basic-auth-credentials>
 </security-configuration>
```

Remark:

Weblogic might not start properly if **domibus.database.general_schema** is set before the **general_schema** username and password have been specified in the Weblogic console. This can be resolved using the following procedure:

1. Comment out (with a #) the domibus.database.general_schema=general_schema line
2. Start the Weblogic server and configure the weblogic server with the the username and password of the general_schema in both the XA and NonXA datasources
3. Remove the comment in the domibus.database.general_schema=general_schema
4. Restart the Weblogic server

16.2. PMode

When C2 wants to send messages to a C3 running in Multitenancy mode, the endpoint URL of C3 configured in the C2 PMode can contain the domain name at the end, configured as an HTTP parameter to indicate the domain that will receive the message.

Example:

Let us suppose that C3 exposes the MSH endpoint under the URL: http://localhost:8080/domibus/service/msh. If C2 wants to send messages to C3 to the domain DIGIT, it will call the following MSH C3 endpoint URL:

http://localhost:8080/domibus/service/msh?domain=DIGIT

In case C2 does not specify the domain in the endpoint URL, the message will be sent to the C3 **default** domain.

16.3. Domain Properties

The following properties listed in the table are used to configure a domain. Some properties are defaulting to the properties defined in domibus.properties if they are not defined in the domain properties file. All the properties defined in a domain property file (e.g. ***domain_name*-domibus.properties**) need to be prefixed by the domain name and override the properties from the **domibus.properties** file.

Example:

1. If the domain name is **DIGIT,** the property file **DIGIT-domibus.properties** is used to configure the **DIGIT** domain.

2. Defining a property named **DIGIT.domibus.msh.messageid.suffix** will override the property **domibus.msh.messageid.suffix** defined in **domibus.properties**.

Domain configuration Property	Defaults to domibus.properties if not defined
domain_name.domibus.database.schema	no
*domain_name.*domibus.UI.title.name	yes
*domain_name.domibus.*ui.csv.max.rows	yes

domain_name.domibus.ui.replication.enabled	yes
domain_name.domibus.msh.messageid.suffix	yes
domain_name.domibus.msh.retry.cron	yes
*domain_name.*domibus.dynamicdiscovery.useDynamicDiscovery	yes
domain_name.domibus.smlzone	yes
domain_name.domibus.dynamicdiscovery.client.specification	yes
domain_name.domibus.dynamicdiscovery.peppolclient.mode	yes
domain_name.domibus.dynamicdiscovery.oasisclient.regexCertificateSubjectValidation	yes
domain_name.domibus.dynamicdiscovery.partyid.responder.role	yes
domain_name.domibus.dynamicdiscovery.partyid.type	yes
domain_name.domibus.dispatcher.allowChunking	yes
domain_name.domibus.dispatcher.chunkingThreshold	yes
domain_name.domibus.dispatcher.concurency	yes
domain_name.domibus.dispatcher.connectionTimeout	yes
domain_name.domibus.dispatcher.receiveTimeout	yes
domain_name.domibus.dispatcher.cacheable	yes
domain_name.domibus.msh.pull.cron	yes

*domain_name.*domibus.pull.queue.concurency	yes
*domain_name.*domibus.pull.request.send.per.job.cycle	yes
*domain_name.*domibus.pull.retry.cron	yes
domain_name.domibus.retentionWorker.cronExpression	yes
domain_name.message.retention.downloaded.max.delete	yes
domain_name.message.retention.not_downloaded.max.delete	yes
domain_name.domibus.sendMessage.messageIdPattern	no
*domain_name.*domibus.attachment.storage.location	no
domain_name.domibus.msh.retry.tolerance	yes
domain_name.domibus.security.keystore.location	no
domain_name.domibus.security.keystore.type	no
domain_name.domibus.security.keystore.password	no
domain_name.domibus.security.key.private.alias	no
domain_name.domibus.security.key.private.password	no
domain_name.domibus.security.truststore.location	no
domain_name.domibus.security.truststore.type	no
domain_name.domibus.security.truststore.password	no
domain_name.domibus.receiver.certificate.validation.onsending	yes

domain_name.domibus.sender.certificate.validation.onsending	yes
domain_name.domibus.sender.certificate.validation.onreceiving	yes
domain_name.domibus.sender.trust.validation.onreceiving	Yes
domain_name.domibus.alert.retry.cron	yes
domain_name.domibus.alert.cleaner.cron	yes
domain_name .domibus.alert.sender.email	
domain_name .domibus.alert.receiver.email	
domain_name.domibus.alert.cleaner.cron	0 0 0/1 * * ?
domain_name.domibus.alert.cleaner.alert.lifetime	20
domain_name.domibus.alert.active	TRUE
domain_name.domibus.alert.mail.sending.active	FALSE
domain_name.domibus.alert.mail.smtp.timeout	5000
domain_name.domibus.alert.queue.concurrency	1
domain_name.domibus.alert.retry.cron	0 0/1 * * * ?
domain_name.domibus.alert.retry.time	1
domain_name.domibus.alert.retry.max_attempts	2
domain_name.domibus.alert.msg.communication_failure.active	TRUE
domain_name.domibus.alert.msg.communication_failure.states	SEND_FAILURE

domain_name.domibus.alert.msg.communication_failure.level	HIGH
domain_name.domibus.alert.msg.communication_failure.mail.subject	Message status change
domain_name.domibus.alert.user.login_failure.active	TRUE
domain_name.domibus.alert.user.login_failure.level	LOW
domain_name.domibus.alert.user.login_failure.mail.subject	Login failure
domain_name.domibus.alert.user.account_disabled.active	TRUE
domain_name.domibus.alert.user.account_disabled.level	HIGH
domain_name.domibus.alert.user.account_disabled.moment	WHEN_BLOCKED
domain_name.domibus.alert.user.account_disabled.subject	Account disabled
domain_name.domibus.alert.cert.imminent_expiration.active	TRUE
domain_name.domibus.alert.cert.imminent_expiration.frequency_days	14
domain_name.domibus.alert.cert.imminent_expiration.level	HIGH
domain_name.domibus.alert.cert.imminent_expiration.mail.subject	Certificate imminent expiration
domain_name.domibus.alert.cert.expired.active	TRUE
domain_name.domibus.alert.cert.expired.frequency_days	7
domain_name.domibus.alert.cert.expired.duration_days	90
domain_name.domibus.alert.cert.expired.level	HIGH
domain_name.domibus.alert.cert.expired.mail.subject	Certificate expired

domain_name.domibus.dynamicdiscovery.transportprofileas4	yes

Remark:

A domain property is mandatory to be defined if it does not default to ***domain.properties.***

16.4. Logging

Domibus generates logs in 3 log files when running in non Multitenancy mode (**domibus.log, domibus-business.log and domibus-security.log**), that are configured in the **logback.xml** file. More information about what is being logged into those files can be found in §9.5 – "*Application Logging*".

In Multinenancy mode, the following should be expected:

- main files - **domibus.log**, **business.log** and **security.log** will contain only general logging information and not domain specific;

- 'per domain' files, e.g. domain1-domibus.log, domain1-business.log and domain1-security.log will contain logging entries only for the specific domain 'domain1';

- it is mandatory to add a **domain logback.xml** for each domain including the 'default' one. Attention, if such file does not exist, the logging information to be lost for that domain.

When running in Multitenancy mode, the Domibus log configuration file **logback.xml** has to be modified as followed:

a. uncomment all the sections marked like this one:

```
<!-- multitenancy: uncomment this
<filter class="eu.domibus.logging.DomibusLoggerDomainFilter">
  <domain></domain>
  <OnMismatch>DENY</OnMismatch>
</filter>
-->
```

b. edit the file in order to include the log configuration for each domain. This is necessary to segregate the log statements per domain, each domain having its own set of the 3 logs files mentioned above:

```
<!-- multitenancy: start include domains config files here -->
<!--<include optional="true" file="${catalina.home}/conf/domibus/domain_name-logback.xml"/>-->
<!-- multitenancy: end include domains config files here -->
```

c. add a domain config file for the 'default' domain.

In order to configure the logs per domain please follow the steps:

1. Customize the **domain_name-logback.xml** file distributed in each server configuration archive.

a. Rename the **domain_name-logback.xml** file according to the domain name. Eg: if the domain name is **domain1**, the file should be renamed to **domain1-logback.xml.**

b. Adapt the value of the **domainName** variable defined in the domain logback configuration file. The value should correspond to the name of the configured domain.

```
<included>
  <property name="domainName" value="domain1" scope="local" />
```

2. Include the domain configuration file into the main **logback.xml** file:

```
<configuration>
  <!-- start include domains config files here -->
  <include optional="true" file="${catalina.home}/conf/domibus/domain1-logback.xml"/>
```

In order to add some particular logging information per domain, the user must add in the **logback.xml** file the following section (example for 'domain1' domain):

```
    <logger name="eu.domibus" level="DEBUG" additivity="false">
     <appender-ref ref="domain1-file"/>
     <appender-ref ref="stdout"/>
    </logger>
```

In the example above, 'eu.domibus' is the name of the package for setting DEBUG level, 'domain1-file' is the appender of 'domain1'.

The line with 'stdout' is optional and it will print the DEBUG info on the server console.

16.5. Users

In Multitenancy mode there is a new user named **super** with role ROLE_AP_ADMIN which has the privileges to access all the available domains. The default password for the **super** user is **123456**.

The first time a new domain is created, the **super** user creates a new user in the **Domibus Administration Console** with role ROLE_ADMIN associated to the newly created domain. All normal users (ROLE_ADMIN, ROLE_USER) can be associated to only and only one domain. More details how to create users can be found in the help page of the **Users** page.

Afterwards the **super** user sends the credentials to the domain admin user. The domain admin logs into the **Domibus Administration Console** using the received credentials and has to change its password in the **Users** page. The domain admin has only access to his domain and he has the privileges to create only new users that are associated to his domain.

Remark:

Please note that user names need to be unique amongst existing domains.

16.6. Plugins

When running in Multitenancy mode the plugins security is activated by default, no matter if the property **domibus.auth.unsecureLoginAllowed** in the domibus.properties files is set to true or not. This is needed in order to identify the request performed by the user and associate it to a specific domain. As a result, every request sent to Domibus needs to be authenticated.

Remark:

Please note that the **Default JMS Plugin** *requires the creation of additional JMS queues. More information on which queues need to be created can be found in the* JMS Plugin Interface Control Document (ICD) *(see* [REF12]).

More information on this topic can be found in the Domibus Software Architecture Document (SAD) (c.f. [REF11]).

16.6.1. Plugin Users

In Multitenancy mode, a plugin must use a configured plugin user associated to a specific domain in order to authenticate every request sent to Domibus. The management of the plugin users is implemented in the **Plugin Users** page of **Domibus Administration Console**. More details on how to manage the plugin users can be found in the help page of the **Plugin Users** page (see also §9 – "*Administration Tools"*).

The **Default JMS Plugin** and the **Default FS Plugin** implement only authentication mechanism. The two previously mentioned plugins must use any configured plugin user to send requests to Domibus, no matter the role: ROLE_ADMIN or ROLE_USER. The request will be sent to the domain associated to the plugin user used for authentication.

The **Default WS Plugin** implements authentication and authorization mechanism.

For authentication the **Default WS Plugin** must use a configured plugin user to send requests to Domibus, the configuration is the same as for the **Default JMS Plugin** and the **Default FS Plugin**.

More details on how the authorization is implemented in the **Default WS Plugin** can be found in §6.1.2 *"WS Plugin"* and in the plugin cookbook document (cf.[REF6]).

Remark:

Please note that user names need to be unique amongst existing domains.

16.7. Switching from non Multitenancy to Multitenancy mode

When switching an existing installation of Domibus to Multitenancy mode, the instructions described in §16.1 – *"Configuration"* have to be executed.

After the switch to Multitenancy mode is finished, the schema that was previously used in non Multitenancy mode will be used by a specific domain. Additionally the **super** user must select the migrated domain in Domibus Administration console and re-create the existing users present in the **Users** and **Plugin Users**. This step is required because in Multitenancy mode there is an automatic synchronization of domain users into the general schema. More info about the synchronization of domain users can be found in the Domibus Software Architecture Document (SAD) (c.f. [REF11]).

17. TEST SERVICE

This section describes how to configure and test the communication between two Access Points. This feature allows communication partners to perform a basic test of the communication configuration (including security at network, transport and message layer, and reliability) in any environment, including the production environment.

Domibus offers a page in the Administration Console which gives the possibility to inspect and send Test messages to other Access Points.

The information about both messages are displayed in the **Test Service** screen and, every time "Receiver Party Id" changes, you are able to check the updated information of Last Sent and Last Received Test Messages.

17.1. Test Service Configuration

In order to configure the **Test Service**, the PMode needs to contain the correct configuration for sending Test messages.

Note: the Domibus sample PModes contain already the configuration for the **Test Service:**

```
<services>
        ...
        <service name="testService" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/service"/>
</services>
<actions>
        ...
        <action name="testAction" value="http://docs.oasis-open.org/ebxml-
msg/ebms/v3.0/ns/core/200704/test"/>
</actions>

<legConfigurations>
        ...
        <legConfiguration name="testServiceCase"
                                service="testService"
                                action="testAction"
                                defaultMpc="defaultMpc"
                                reliability="AS4Reliability"
                                security="eDeliveryAS4Policy"
                                receptionAwareness="receptionAwareness"
                                propertySet="eDeliveryPropertySet"
                                payloadProfile="MessageProfile"
                                errorHandling="demoErrorHandling"
                                compressPayloads="true"/>
</legConfigurations>
```

18. ALERTS

18.1. Description

The purpose of the alert feature is to use different available media to notify the Domibus administrator in case of unusual behaviour. Those notifications are presented to the Domibus administrator under the form of configurable alerts. The alerts can be browsed in the **Domibus Admin Console** in the Alerts section and can be sent by **email**.

Currently, only email notification channel is available, but other communication media will be added in future releases.

Three topics are available for monitoring:

- Message status change
- Authentication issues
- Certificate expiration.

18.2. Main configuration

The properties, described below, can be configured in the domibus.properties configuration file.

By default, alerts are not activated. A single property can activate or deactivate the entire alert concept. In order to activate it, the following property should be set to true.

```
# ---------------------------------- Alert management ----------------------------------
#enable/disable the entire alert module. Pay attention to the fact that if the module is activated, all properties
#under the mandatory section should be configured.
domibus.alert.active=true
```

Once the alerts are activated, the SMTP server needs also to be configured. In that case the following properties are mandatory:

```
# ------------------------------------Mandatory configuration start (if domibus.alert.active=true) -----------

#Smtp sever url for sending alert
#domibus.alert.sender.smtp.url=

#Smtp sever port
#domibus.alert.sender.smtp.port=

#Smtp sever user
#domibus.alert.sender.smtp.user=

#Smtp sever user password
#domibus.alert.sender.smtp.password=
```

```
#Alert sender email
#domibus.alert.sender.email=

#Alert email receiver.
#domibus.alert.receiver.email=
```

The first four properties are used to configure respectively the URL, the port, the user and the password to authenticate to the SMTP server.

The last two properties are needed to respectively set the emails of the alert sender and the alert receiver.

The following properties are already preconfigured with default values and therefore are not mandatory to be configured:

```
#The following properties can stay commented if no modifications to the default values are needed.

#Cron configuration for cleaning alerts.
#domibus.alert.cleaner.cron=0 0 0/1 * * ?

# Alerts lifetime in days of before cleaning.
#domibus.alert.cleaner.alert.lifetime=20

#Concurency to process the alerts.
#domibus.alert.queue.concurrency=1

#Frequency of failed alerts retry.
#domibus.alert.retry.cron=0 0/1 * * * ?

#Elapsed time in minute between alert retry.
#domibus.alert.retry.time=1

#Number of retry for failed alerts.
#domibus.alert.retry.max_attempts=2
```

By default, Domibus will check every hour for expired alerts. The default lifetime for an alert is 20 days after which the alert is deleted from the system.

The concurrency property allows processing multiple alerts in parallel. Alerts can be configured with a retry in case of dispatch failure. By default Domibus will wait one minute between two alert dispatch attempts, and it will retry twice.

Multitenancy

In multinancy mode, the four SMTP properties should be configured in the main domibus.properties. Indeed only one SMTP server can be configured for all the tenants.

On the other hand, the sender and receiver properties must be configured in each domain configuration file.

Multitenancy also introduces the existence of a super user. Authentication alerts can be configured for it. Some specific global properties have been created for the super user. The following properties are documented with their default value. They can be overrided in domibus.properties file:

```
# ---------------------------------- Super user Alert management ----------------------------------

#Cron configuration for cleaning alerts.
#domibus.alert.super.cleaner.cron=0 0 0/1 * * ?

#Lifetime in days of alerts before cleaning.
#domibus.alert.super.cleaner.alert.lifetime=20

#Enable/disable the entire alert module.
#domibus.alert.super.active=true

#Allow to disable alert mail sending.
#domibus.alert.super.mail.sending.active=false

#Frequency of failed alerts retry.
#domibus.alert.super.retry.cron=0 0/1 * * * ?

#Elapsed time in minutes between alert retry.
#domibus.alert.super.retry.time=1

#Maximum number of attempts for failed alerts
#domibus.alert.super.retry.max_attempts=2
```

18.3. Message status change alerts

Domibus is able to track Message status changes. All status changes can be tracked but it is advised not to track the status of frequently changing statuses (e.g.: From SEND_ENQUEUED to ACKNOWLEDGE) to avoid being spammed.

Each alert topic (Message status change, authentication and certificate expiration) can be activated or deactivated independently from each other. Attention, in order for the alert feature to work, the main alert module must always be activated (see § 18.2-*"Main configuration"*).

By default, message status change alerts are not activated. In order to activate them, the following property should be set to true:

```
# ---------------------------------- Alert management: messaging module -----------------
#enable/disable the messaging alert module.
domibus.alert.msg.communication_failure.active=true
```

The following properties are already preconfigured with default values and therefore are not mandatory to be configured:

```
#Message status change that should be notified by the messaging alert module. Comma separated.
#domibus.alert.msg.communication_failure.states=SEND_FAILURE

#Alert levels corresponding to message status defined in previous
property(domibus.alert.msg.communication_failure.states) . Should be (HIGH, MEDIUM or LOW)
#domibus.alert.msg.communication_failure.level=HIGH

#Messaging alert module mail subject.
#domibus.alert.msg.communication_failure.mail.subject=Message status change
```

Per default, Domibus will only track message status change to SEND_FAILURE. The level of the alert that will be triggered is HIGH. The last property allows configuring the subject of the mail sent.

If there is a need to track another message status change, a comma separated list can be configured:

```
Eg: domibus.alert.msg.communication_failure.states=SEND_FAILURE,ACKNOWLEDGED
```

If there is a need to set an alert level per status change it can also be done with a comma separated list:

```
domibus.alert.msg.communication_failure.level=HIGH,LOW
```

In the example above, an alert for a message being set in send_failure status will have a high level and an alert for a message being set to acknowledged status will have a low level.

18.4. Authentication Alerts

Domibus is able to track admin console login failure and user account disabling. The login failure alert will occur for each unsuccessful attempt. Note that if the username encoded is unknown to the system, no alert will be created. Only known user with invalid password will be tracked. The account disabled alert will occur either because the user did too many invalid login attempts or because an administrator disabled the account.

By default, login failure alerts are not activated. In order to activate them, the following property should be set to true:

```
# ---------------------------------- Alert management: Authentication module -----------------

#Enable/disable the login failure alert of the authentication module.
domibus.alert.user.login_failure.active=true
```

The following properties are already preconfigured with default values and therefore are not mandatory to configure:

```
#Alert level for login failure.
#domibus.alert.user.login_failure.level=LOW

#Login failure mail subject.
#domibus.alert.user.login_failure.mail.subject=Login failure
```

Per default, the alert level for a login failure is low. The last property allows configuring the subject of the mail sent.

By default, account disabled alerts are not activated. In order to activate them, the following property should be set to true:

```
#Enable/disable the account disable alert of the authentication module.
domibus.alert.user.account_disabled.active=true
```

The following properties are already preconfigured with default values and therefore are not mandatory to configure:

```
#Alert level for account disabled.
#domibus.alert.user.account_disabled.level=HIGH

#When should the account disabled alert be triggered.
# 2 possible values:
# AT_LOGON: An alert will be triggered each time a user tries to login to a disabled account.
# WHEN_BLOCKED: An alert will be triggered once when the account got disabled.
#domibus.alert.user.account_disabled.moment=WHEN_BLOCKED

#Account disabled mail subject.
#domibus.alert.user.account_disabled.subject=Account disabled
```

Per default, the alert level for an account disabled is high. The next property specifies when an account_disabled alert should be triggered. It can be only at disabling time or at every new login attempt after the account has been disabled. The default value WHEN_BLOCKED will therefore create only one alert when the account is disabled.

The last property allows configuring the subject of the mail sent.

Multitenancy

The following super user authentication alerts properties are documented with their default value. They can be overrided in the domibus.properties file:

```
# ---------------------------------- Super user alert management:Authentication module ----------------

#Enable/disable the login failure alert of the authentication module.
#domibus.alert.super.user.login_failure.active=true

#Alert level for login failure.
#domibus.alert.super.user.login_failure.level=LOW

#Login failure mail subject.
#domibus.alert.super.user.login_failure.mail.subject=Super user login failure

#Enable/disable the account disable alert of the authentication module.
#domibus.alert.super.user.account_disabled.active=true

#Alert level for account disabled.
#domibus.alert.super.user.account_disabled.level=HIGH

#When should the account disabled alert be triggered.
# 2 possible values:
# AT_LOGON: An alert will be triggered each a time user tries to login to a disabled account.
# WHEN_BLOCKED: An alert will be triggered once when the account got disabled.
#domibus.alert.super.user.account_disabled.moment=WHEN_BLOCKED

#Account disabled mail subject.
#domibus.alert.super.user.account_disabled.subject=Super user account disabled
```

18.5. Certificate scanner alerts

Domibus is able to track certificate expiration and imminent expiration. Obviously the certificate expired alert occurs when a certificate expires. The number of days the alert should be triggered after the expiration is configurable. The imminent expiration alert occurs a certain time before the certificate expiration. The number of days the alert should be triggered before expiration is configurable. The alert frequency for both trackers can be configured.

By default, imminent certificate expiration alerts are not activated. In order to activate them, the following property should be set to true:

```
# ---------------------------------- Alert management: Certificate scanner ----------------

#Enable/disable the imminent certificate expiration alert of certificate scanner module.
domibus.alert.cert.imminent_expiration.active=true
```

The following properties are already preconfigured with default values and therefore are not mandatory to configure:

```
#Number of days before revocation as from when the system should start sending alerts.
#domibus.alert.cert.imminent_expiration.delay_days=61

#Frequency in days between alerts.
#domibus.alert.cert.imminent_expiration.frequency_days=14
```

```
#Certificate imminent expiration alert level.
#domibus.alert.cert.imminent_expiration.level=HIGH

#Certificate imminent expiration mail subject.
#domibus.alert.cert.imminent_expiration.mail.subject=Certificate imminent expiration
```

By default, Domibus will send certificate imminent expiration alerts 61 days before the expiration. It will send alerts at a pace of one alert every 14 days. The level of the alert will be HIGH. The last property allows configuring the subject of the mail sent.

By default, certificate expired alerts are not activated. In order to activate them, the following property should be set to true:

```
#Enable/disable the certificate expired alert of certificate scanner module.
domibus.alert.cert.expired.active=true
```

The following properties are already preconfigured with default values and therefore are not mandatory to configure:

```
#Frequency in days between alerts.
#domibus.alert.cert.expired.frequency_days=7

#How long(in days) after the revocation should the system trigger alert for the expired certificate.
#domibus.alert.cert.expired.duration_days=92

#Certificate expired alert level.
#domibus.alert.cert.expired.level=HIGH

#Certificate expired mail subject.
#domibus.alert.cert.expired.mail.subject=Certificate expired
```

By default, Domibus will send certificate expired alerts during 92 days after the expiration. It will send alerts at a pace of one alert every 7 days. The level of the alert will be HIGH. The last property allows configuring the subject of the mail sent.

18.6. Configuration example

18.6.1. Example: domibus.properties

Below is shown only the section relevant to the alerts configuration in the **domibus.properties** configuration file, when the SMTP server is running in the same host as domibus (localhost):

```
…
…
# ---------------------------------- Alert management ----------------------------------
#Enable/disable the entire alert module. Pay attention to the fact that if the module is activated, all
properties
#under the mandatory section should be configured.
domibus.alert.active=true
#Allow to disable alert mail sending.
domibus.alert.mail.sending.active=true
domibus.alert.mail.smtp.starttls.enable=false
```

```
domibus.alert.mail.smtp.auth=false
#domibus.alert.mail.smtp.timeout=10000
# ----------------------------------Mandatory configuration start (if domibus.alert.active=true) ----------
#Smtp server url for sending alert.
domibus.alert.sender.smtp.url=localhost
#Smtp server port.
domibus.alert.sender.smtp.port=25
#Smtp server user.
#domibus.alert.sender.smtp.user=
#Smtp server user password
#domibus.alert.sender.smtp.password=
#Alert sender email.
domibus.alert.sender.email=sender@exemple.com
#Alert email receiver.
domibus.alert.receiver.email=mcb@gmail.com
# ---------------------------------Mandatory configuration end---------------------------------
#The following properties can stay commented if no modifications to the default values are needed.
#Cron configuration for cleaning alerts.
domibus.alert.cleaner.cron=0 0/1 * * * ?
#Lifetime in days of alerts before cleaning.
domibus.alert.cleaner.alert.lifetime=1
#Concurency to process the alerts.
#domibus.alert.queue.concurrency=1

#Frequency of failed alerts retry.
#domibus.alert.retry.cron=0 0/1 * * * ?
#Elapsed time in minutes between alert retry.
#domibus.alert.retry.time=1
#Maximum number of attempts for failed alerts
#domibus.alert.retry.max_attempts=2
# --------------------------------- Alert management:messaging module -----------------
#Enable/disable the messaging alert module.
#domibus.alert.msg.communication_failure.active=true
#Message status change that should be notified by the messaging alert module. Comma separated.
domibus.alert.msg.communication_failure.states=SEND_FAILURE,WAITING_FOR_RETRY
#Alert levels corresponding to message status defined in previous
property(domibus.alert.msg.communication_failure.states) .
#Should be (HIGH, MEDIUM OR LOW)
#domibus.alert.msg.communication_failure.level=HIGH
#Messaging alert module mail subject.
domibus.alert.msg.communication_failure.mail.subject=Message status change MCB
# --------------------------------- Alert management:Authentication module -----------------
#Enable/disable the login failure alert of the authentication module.
domibus.alert.user.login_failure.active=true
#Alert level for login failure.
#domibus.alert.user.login_failure.level=LOW
#Login failure mail subject.
domibus.alert.user.login_failure.mail.subject=Login failure MCB
#Enable/disable the account disable alert of the authentication module.
#domibus.alert.user.account_disabled.active=true
#Alert level for account disabled.
#domibus.alert.user.account_disabled.level=HIGH
#When should the account disabled alert be triggered.
```

```
# 2 possible values:
# AT_LOGON: An alert will be triggered each time a user tries to login to a disabled account.
# WHEN_BLOCKED: An alert will be triggered once when the account got disabled.
domibus.alert.user.account_disabled.moment=WHEN_BLOCKED,AT_LOGON
#Account disabled mail subject.
domibus.alert.user.account_disabled.subject=Account disabled MCB
# ---------------------------------- Alert management:Certificate scanner -----------------
#Enable/disable the imminent certificate expiration alert of certificate scanner module.
domibus.alert.cert.imminent_expiration.active=false
#Number of days before revocation as from when the system should start sending alerts.
domibus.alert.cert.imminent_expiration.delay_days=20000
#Frequency in days between alerts.
#domibus.alert.cert.imminent_expiration.frequency_days=14
#Certificate imminent expiration alert level.
#domibus.alert.cert.imminent_expiration.level=HIGH
#Certificate imminent expiration mail subject.
domibus.alert.cert.imminent_expiration.mail.subject=Certificate imminent expiration MCB
#Enable/disable the certificate expired alert of certificate scanner module.
domibus.alert.cert.expired.active=false
#Frequency in days between alerts.
#domibus.alert.cert.expired.frequency_days=7
#How long(in days) after the revocation should the system trigger alert for the expired certificate.
#domibus.alert.cert.expired.duration_days=90
#Certificate expired alert level.
#domibus.alert.cert.expired.level=HIGH
#Certificate expired mail subject.
domibus.alert.cert.expired.mail.subject=Certificate expired MCB
# ---------------------------------- UI Replication ----------------------------------
….
```

18.6.2. *Example: domain_name-domibus.properties*

Below is shown only the section relevant to the alertes configuration in the **dom50-domibus.properties** configuration file, where dom50 is the name of a domain:

```
…
#Pull Retry Worker execution interval as a cron expression
#dom50.domibus.pull.retry.cron=0/10 * * * * ?
# ---------------------------------- Alert management ----------------------------------
#Enable/disable the entire alert module. Pay attention to the fact that if the module is activated, all
properties
#under the mandatory section should be configured.
dom50.domibus.alert.active=true
#Allow to disable alert mail sending.
dom50.domibus.alert.mail.sending.active=true
# ----------------------------------Mandatory configuration start (if domibus.alert.mail.sending.active=true) ---
#Alert sender email.
dom50.domibus.alert.sender.email=mcb@gmail.com
#Alert email receiver.
dom50.domibus.alert.receiver.email=mcb@gmail.com
# ----------------------------------Mandatory configuration end----------------------------------
#The following properties can stay commented if no modifications to the default values are needed
```

```
...
```

19. UIREPLICATION FEATURE

19.1. Description

This feature has been introduced in order to assure a faster search on Admin Console Messages page when the database contains more than 100k messages and the time to perform a search is longer.

It uses a flat table called TB_MESSAGE_UI instead of the 5-6 native message tables and a JMS queue to synchronize data between.

19.2. Configuration and first synchronization of data

By default the UIReplication is disabled. To enable it set the following property to true:

```
#enabled or disabled the UI Replication mechanism
domibus.ui.replication.enabled=true
```

Just **before** enabling there are several things to check:

Step 1:

- if there was a migration from 3.3.4 to 4.0 and the oracle/mysql*-3.3.4-to-4.0-migration.ddl script has been run then the data has been migrated from native tables into TB_MESSAGE_UI

So run this query and check if the result is greater than 0 (zero):

```
SELECT COUNT(*) FROM TB_MESSAGE_UI
```

If yes, go to Step 2 and if not go to Step 3.

- if there was no migration from 3.3.4 and just a fresh install then go to Step 2

Step 2:

- check if there is unsynchronized data between native tables and TB_MESSAGE_UI by running this query:

```
SELECT COUNT(*) FROM V_MESSAGE_UI_DIFF
```

If the query returns 0 (zero) then proceed to enable the UIReplication as above. If not, go to Step 3

Step 3:

UIReplication data could be synchronized for the first time in two ways:

- calling REST method /rest/uireplication/sync

- running the oracle10g/mysql5innoDb-xyz-uireplication-insert.sql script present in distribution packages in which case the TB_MSSAGE_UI should be empty before.

The second way is recommended for large volume of data, in general over 100K records.

Additionally, there is a cron job which has the purpose of checking any unsychronized data and is set to run daily at 2 AM. In order to change the time of day, edit or add the following property:

```
#Cron job that will check unsynchronized data between native tables and TB_UI_MESSAGE_UI
domibus.ui.replication.sync.cron=0 0 2 * * ?
```

Also for the cron job there is another property which defines the maximum number of messages to be synchronized and could be added / edited in domibus.properties file:

```
#max number of records that will be processed by cron job
domibus.ui.replication.sync.cron.max.rows=10000
```

If there are more messages than this limit there will be no synchronization done and a warning will be issued in the log file suggesting that the user should use the REST resource instead.

19.3. REST resources

There are two REST resources which could be useful for manual synchronization and works when UIReplication is enabled.

Assuming that http://<server>:<port>/domibus/ point to your Domibus installation please login first into Admin Console.

19.3.1. Count method

Enter this address on your browser:

```
http://<server>:<port>/domibus/rest/uireplication/count
```

This will return the number of unsynchronized records. If this count is equal to zero it means that the UI replication data is synchronized.

19.3.2. Sync method

Enter this address on your browser:

```
http://<server>:<port>/domibus/rest/uireplication/sync

or

http://<server>:<port>/domibus/rest/uireplication/sync?limit=x
```

Where:

- limit=x parameter could be optional and has a default value of 10.000

Calling this will perform a synchronization of the data for the first 'x' records.

19.4. Recommendations

1. As described in the previous paragraph 19.2 about first synchronization of data on large volume of records, generally over 100k running the query:

```
SELECT COUNT(*) FROM V_MESSAGE_UI_DIFF
```

Could take some time (several seconds) to complete.

So for Oracle we could use the PARALLEL hint:

```
SELECT COUNT(*) /*+ PARALLEL(8) */ FROM V_MESSAGE_UI_DIFF
```

Or use the REST method /rest/uireplication/count which internally has the same optimization.

2. Do not setup UIReplicationJob to run more than once per day as running too often could interfere with the internal mechanism of synchronization.

3. Searching into Admin Console Messages page could be slower for some criteria – in that case just analyze the queries and add more indexes on TB_MESSAGE_UI table.

By default Domibus is creating indexes on message type, message status, received date and message subtype and not on all combinations of the search criteria of Admin Console messages page because this is up to the user to know which ones are the most used.

For Oracle some tools like SQLDeveloper – Explain Plan and Tunning Advisor could be used to analyze the queries and add more indexes.

20. OPERATIONAL GUIDELINES

In this section you will find some recommendations on how to administer Domibus in an efficient way. The following topics are tackled: JMS Queue management, log management, capacity planning, database management and the monitoring of message life cycle.

20.1. JMS Queue Management

Domibus provides following out of the box features to manage the JMS Queues used in Domibus (see also §9.7- *"Queue Monitoring"*):

- Inspecting and filtering the messages from a queue based on the contents of Source, Period, JMS Type or Selector
- Move message from the DLQ (Dead Letter Queue) to the original Queue
- Delete stuck or pending message(s) from Queues

It is recommended to monitor the Queue size and number of messages in the different Queues. If some messages are stuck in any of the Queue then alerts must be sent to the Domibus Administrator.

Please pay special attention to the deadletter queue (DLQ). Messages stuck in this queue is a signal that there is some issue in Domibus that needs to be analysed and an alert should be sent to the Domibus Administrator.

Important:

The 'ListPendingMessages' operation on WS Plugin browses the JMS queue. Max count is limited to destination MaxBrowsePageSize which can be changed via the 'domibus.listPendingMessages.maxCount' Domibus property.

If your received messages are not returned by the webservice listPendingMessages method the you should:

1. increase the value of the 'domibus.listPendingMessages.maxCount'property;
2. delete the messages from the domibus.notification.webservice queue with selector NOTIFICATION_TYPE=MESSAGE_SEND_SUCCESS using JMX tools : http://activemq.apache.org/how-can-i-monitor-activemq.html .

20.2. Log Management

20.2.1. Log Level

It is recommended that the log level is correctly set in all the environments:

- The log level should be set to INFO/DEBUG in all the test environments for de-bugging purpose.
- The log level should be set to ERROR/WARN in production environment (keeping log level to INFO in production environment will degrade the performance of Domibus).

20.2.2. Log Rotation and Archiving

It is recommended that log rotation and archiving logic is implemented.

Domibus provides by default log rotation, but Domibus administrator should manage Domibus archiving logic.

20.2.3. Log Monitoring

It is recommended to monitor continuously Domibus logs. It can be done using an automated script which looks for keywords like "ERROR", WARNING", etc. and reports all the errors and warnings to the Domibus administrator.

20.3. Capacity Planning

20.3.1. JVM Memory Management

Hereafter some recommendations:

- the JVM memory parameters must first be tested in a test environment with the load expected in production
- the JVM parameters i.e. heap size must be monitored with the help of automated scripts and any abnormal hikes in heap size must be reported to the administrator.

20.3.2. CPU, IO operations and Disk Space Monitoring

CPU, IO operations and disk space must be continuously monitored using automated scripts. Any abnormal hikes must be reported to Domibus administrator and further investigated.

20.4. Database Management

20.4.1. Database Monitoring

It is important to monitor the database size.

The Payload of the message is deleted from the sending Access Point. Only the metadata of the message stays in the table. The Payload from the receiving Access Point is deleted based on the retention policy defined in the Pmode settings.

Domibus uses approximately 40 MB of table space to store the metadata of 1000 messages.

20.4.2. Database Archiving

Since the Database contains AS4 receipts that are used for non-repudiation purposes, they should be archived before purging the database.

The metadata of the database can be purged if it is no longer required.

20.4.3. Monitor Message Life Cycle

It is recommended to monitor the message status in the TB_MessageLog table. Automated scripts can be used to count different status in the table.

Please pay special attention to the following statuses:

- WAITING_FOR_RETRY: this means that there is some issue between C2 and C3 that must be resolved.
- SEND_FAILURE: this means that that there is some issue between C2 and C3 that must be resolved.
- SEND_ENQUEUED: this message status is part of the successful message life cycle, however abnormal increase in the count of messages with this status means that there is an issue. Further investigation is recommended.

21. Annex 1 - Usage of certificates in PEPPOL and OASIS

		C2		C3	
		Keystore	**Truststore**	**Keystore**	**Truststore**
PEPPOL	Certificate:	Sender's (issued by CA)	Empty	Receiver's	CA's
	Note:	C2 signs the message with its private key	C2 discover C3's public certificate from the SMP	C3 signs the receipt with its private key	The receiver trusts all senders who's certificate were issue dby these CA's
OASIS	Certificate:	Sender's (issued by CA)	SMP's	Receiver's	CA's
	Note:	C2 signs the message with its private key	C2 discover C3's public certificate from the SMP To trust the SMP, the sender needs its public certificate	C3 signs the receipt with its private key	The receiver trusts all senders who's certificate were issue dby these CA's

22. List of Figures

List of Tables

23. Contact Information

CEF Support Team

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

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