

domibusConnectorPlugin - Installation Guide

domibusConnectorPlugin-4.5.0

Table of Content

1. Preconditions	2
2. First Steps	3
3. Installing the plugin on domibus	4
3.1. Jar File	4
3.2. Properties	4
3.3. Configuring certificates	4
3.4. Pull Plugin Configuration	4
Configuring Database (Only needed for PULL-MODE)	4
3.5. Push Plugin Configuration	4
Configuring Properties	4
Configuring Queues	7
4. Checks	8
4.1. Compatibility-Matrix	8
4.2. Test-Matrix	9
4.2.1. Current Test-Matrix	9
4.2.2. Historic Test-Matrix	9



The pull plugin does not work in combination with push and pull plugin.

Chapter 1. Preconditions

- Domibus Gateway (5.x) setup is complete and domibus is running
- Domibus UI is reachable under \$UI-URL
- Domibus is reachable under \$DOMIBUS-URL
- Domibus-config folder is located under \$DOMIBUS-CONFIG-LOCATION



Consult the [domibus administration guide](#) for details

Chapter 2. First Steps

Download the most recent domibusConnectorDomibusPlugin via <https://secure.e-codex.eu/nexus/content/groups/public/eu/domibus/connector/plugin/domibus-connector-plugin-Distribution/>

Extract the downloaded distribution package to any location, this location will from now on be referenced as \$PLUGIN-DISTRIBUTION

Also make sure that you use the installation manual of this specific version. This installation manual is for version: **4.5.0**.

Chapter 3. Installing the plugin on domibus

3.1. Jar File

The plugin comes packaged as a jar file. This jar file has to be put into the domibus plugin lib folder, so domibus will find it during startup:

```
cp $PLUGIN-DISTRIBUTION/plugins/lib/domibus-connector-[push|pull]-plugin.jar $DOMIBUS-CONFIG-LOCATION/plugins/lib/domibus-connector-[push|pull]-plugin.jar
```

3.2. Properties

Create a property file under `$DOMIBUS-CONFIG-LOCATION/plugins/config/dc-[push|pull]-plugin.properties`

And overwrite the default properties in this file.

3.3. Configuring certificates

3.4. Pull Plugin Configuration

Configuring Database (Only needed for PULL-MODE)

Please create the following table for PULL-Mode. You have to set the Column `ID_PK` to either `AUTOINCREMENT` or create an `SEQUENCE` for it. The column must be incremented by the DBMS!

```
-- MySQL example:
-- The column ID_PK must be incremented (no order required, just make sure it is a PK)
by the DBMS
CREATE TABLE DC_PLUGIN_TB_MESSAGE_LOG (
    ID_PK INTEGER PRIMARY KEY AUTO_INCREMENT,
    MESSAGE_ID VARCHAR(255),
    RECEIVED TIMESTAMP(6),
    DOMAIN_CODE VARCHAR(255)
)
```

3.5. Push Plugin Configuration

Configuring Properties

The plugin can be configured with numerous parameters. The following **default properties** contains all available properties. The **example properties** contains a list of properties you should change!

default properties

```

#
#
# the location of the trust-store
# by default ${domibus.config.location}/keystores/
connector.delivery.trust-store.file=file:///${domibus.config.location}/keystores/gw-
gwlink-keystore.p12
# the password of the trust-store
# by default 12345
connector.delivery.trust-store.password=12345
# default store type: JKS
connector.delivery.trust-store.type=PKCS12

# the location of the key-store, the key-store holds
# the private key for decrypting received messages
# and signing sent messages
connector.delivery.key-store.file=file:///${domibus.config.location}/keystores/gw-
gwlink-keystore.p12
# the default key-store password
connector.delivery.key-store.password=12345
# the keystore type, by default JKS (java key store)
connector.delivery.key-store.type=PKCS12
# the private key alias name, which is used to decrypt/sign messages
connector.delivery.private-key.alias=gw
# the default password
connector.delivery.private-key.password=12345
# the certificate alias name which is used to encrypt the message
# by default connector
connector.delivery.encrypt-alias=gw
# required if push plugin is used
# by default not set
connector.delivery.service.address=
#which ws-security policy is activated.
# by default the backend.policy.xml is used, which will enforce signing + encryption
connector.delivery.service.service.security-policy=classpath:/wsdl/backend.policy.xml
# this will only enforce signing
#connector.delivery.service.service.security-
policy=classpath:/wsdl/signature.policy.xml
# this will do nothing, you have to secure the webservice call via TLS
#connector.delivery.service.service.security-policy=classpath:/wsdl/no-
security.policy.xml

# should the soap-messages be logged
# due performance reasons, this is by default false
# you also have to enable the org.apache.cxf logger itself to log level
org.apache.cxf=INFO
# in logback.xml config
#connector.delivery.service.service.logging-feature.enabled=false

#the default source for username: DEFAULT (the dcplugin.auth.username will be used,
# ALIAS the alias of the connector.delivery.trust-store will be used,

```

```
# CN the CN of the provided certificate will be used)
# anyway only with a valid cert the request will pass, if the security policy requires
it
dcplugin.auth.use-username-from=DEFAULT
# username which will be set
dcplugin.auth.username=dcplugin
# the role the username will get
dcplugin.auth.roles=ROLE_ADMIN
#

#the default publish url will result to /services/dcpushplugin
dcplugin.push.publish.url=/dcpushplugin
#the default publish url will result to /services/dcpullplugin
dcplugin.pull.publish.url=/dcpullplugin

# how many message should be listed in pull mode? zero will return all messages
connector.delivery.pull.messages.pending.list.max=100

# Queue Name...
dcplugin.push.notifications.queue=domibus.dcplugin.push.notifications.queue
```

example properties


```

# Example Properties to perform integration tests
#
#

connector.delivery.trust-store.file=file:///${domibus.config.location}/keystores/gw-
gwlink-truststore.p12
# the password of the trust-store
# by default 12345
connector.delivery.trust-store.password=12345
# default store type:
connector.delivery.trust-store.type=PKCS12

# the location of the key-store, the key-store holds
# the private key for decrypting received messages
# and signing sent messages
#connector.delivery.key-store.file=file:///C:/Entwicklung/test.p12
connector.delivery.key-store.file=file:///${domibus.config.location}/keystores/gw-
gwlink-keystore.p12
# the default key-store password
connector.delivery.key-store.password=12345
# the keystore type, by default JKS (java key store)
connector.delivery.key-store.type=PKCS12
# the private key alias name, which is used to decrypt/sign messages
connector.delivery.private-key.alias=gw
# the default password
connector.delivery.private-key.password=12345
# the certificate alias name which is used to encrypt the message
# by default connector
connector.delivery.encrypt-alias=connector

```

Configuring Queues

The domibus-connector-plugin requires an additional queue. You have to configure it at your Message-Broker. If you are running the gateway with the embedded Message-Broker you have to add the queue definition into the activemq.xml file. The file is located at \$DOMIBUS-CONFIG-LOCATION/internal/activemq.xml

Add the following lines next to the other queue definitions (this will create the queue):

```

<!-- DCPlugin queues -->
<queue id="dcPushPluginNotificationsQueue"
physicalName="${dcplugin.push.notifications.queue}" />

```

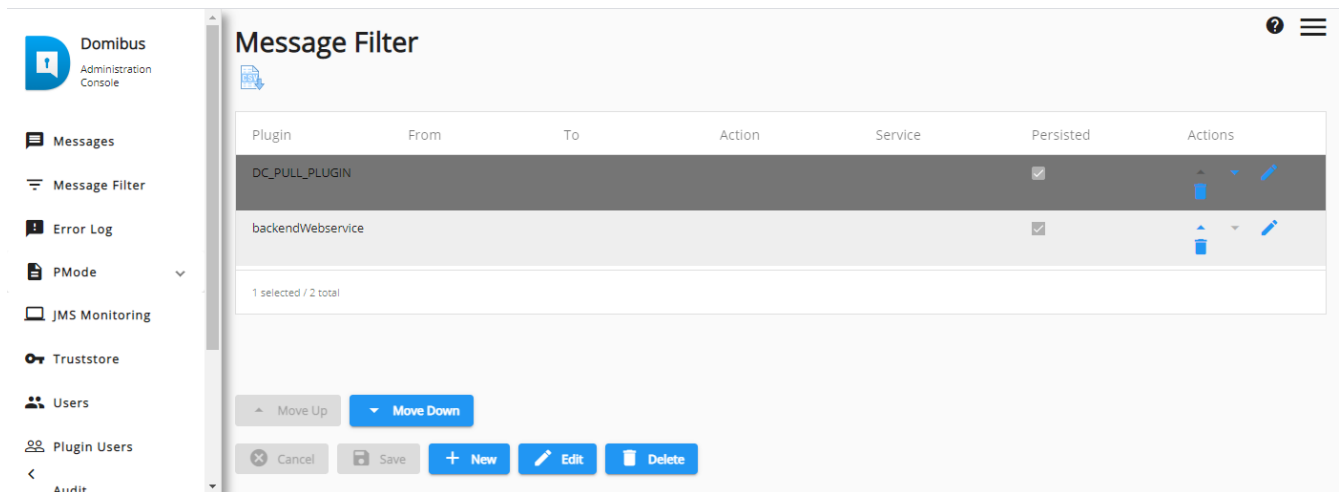
Chapter 4. Checks

Restart the domibus GW. Open the domibus UI and login with the admin user. On the left side open the 'Message Filter' there should now be a **DC_PULL_PLUGIN** and/or **DC_PUSH_PLUGIN** available (depends on copied plugins).



Consult the [domibus administration guide](#) (chapter 10.4 Message Filtering) to learn more about message filtering.

Message Filter View



Also check if the Plugin is available under

- push-plugin: \$DOMIBUS-URL/services/dcpushplugin?wsdl
- pull-plugin: \$DOMIBUS-URL/services/dcpullplugin?wsdl



Please make sure, that **ONLY** the MSHWebservice is available from the internet. Otherwise you are exposing internal webservices to the outside world.

4.1. Compatibility-Matrix

The 4.4.x plugin should work with the following component versions: For a more details consult the [test-matrix](#).

Gateway Plugin-API	Connector API
5.0	4.5.x



Gateway Plugin-API does not directly refer to the gateway version. Instead it declares by the gateway supported plugin API. So as long as a future gateway version supports the used plugin-API version it should work. (Gateway 4.2.0 has deprecated the 4.1 gateway plugin API, but a plugin built against 4.1 plugin API would still run on a gateway 4.2.2.)

4.2. Test-Matrix

The current plugin has been tested with the following components:



This matrix ist a list of tested combinations. If you have a up and running installation please report back so this list can be extended.

4.2.1. Current Test-Matrix

Plugin Version	Java	GW-Webserver	Gateway	Connector (standalone)	Thanks To
4.5.0 / PUSH	Java 8, AdoptJDK	Tomcat 9	5.0	4.4.5	
4.4.0 / PUSH		Tomcat 9	5.x	4.4.x	Spain

4.2.2. Historic Test-Matrix

Older Plugin versions are still listed in this matrix.

Plugin Version	Java	GW-Webserver	Gateway	Connector (standalone)	Mode
4.2.0	Java 8, AdoptJDK	Tomcat 9	4.2.1	4.3.0	Mode-Pull
4.2.0	Java 8, AdoptJDK	Tomcat 9	4.2.1	4.3.0	Mode-Push
4.2.0	Java 8, AdoptJDK	Tomcat 9	4.2.6	4.4.4	Mode-Pull
4.2.0	Java 8, AdoptJDK	Tomcat 9	4.2.6	4.4.4	Mode-Push