EUROPEAN COMMISSION

DIRECTORATE-GENERAL INFORMATICS

eDelivery Pilot for BRIS

Quick Start Guide

Author(s) : CEF
Reviewed by :

Approved by :

Version : 1.00

Date: 5/11/2015

CONTENTS

1. INTRODUCTION

CEF e-Delivery provides a set of components to exchange messages over the internet using B2B protocols. See the document concerning the "Introduction to the Connecting Europe Facility eDelivery building block" also included in this package for more information.

In this particular 'static' deployment context of Business Registers Interconnection System (BRIS), the full set of components (like dynamic discovery, connector) is not required. Business Registers cannot communicate directly with one another. Business Registers must always communicate through the European Central Platform. The required B2B protocol is AS4 (no need of AS2).

Therefore, this specific release (cipa-edelivery-distribution-3.1.0-as4-jboss) provides only an AS4 gateway (CEF e-Delivery component called domibus) running on a JBoss application server and using MySQL database to persist the data.

2. PURPOSE OF THIS GUIDE

In this document, you will find instructions to cover the deployment scenario as illustrated in the figure below. In other words, we will guide you to setup 2 JBoss standalone instances connected on two separate machines to exchange B2B documents securely over AS4 by:

- Deploying and configuring both JBoss instances (A and B)
- Configuring processing modes files for both AS4 gateways
- Using provided AS4 gateways certificates
- Setup the instances A and B for running test cases (Cf. Testing section)

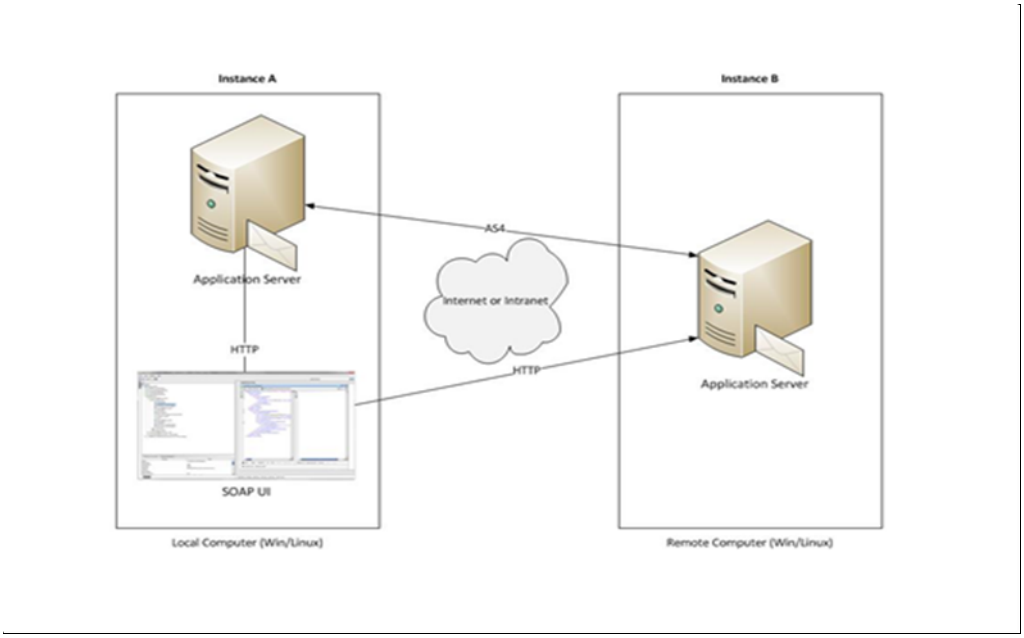


Figure 1 Installation on 2 different machines

Remarks:

- The same procedure can be extended to a third (or more) instance(s).
- This guide doesn't cover the preliminary network configuration allowing communication between separate networks (i.e. infrastructure firewall/Proxy setup).

3. PREREQUISITES

- Java runtime environment (JRE), version 7:
  http://www.oracle.com/technetwork/java/javase/downloads/index.html

- JCE Unlimited Strength Policy files, for JRE7:
  http://www.oracle.com/technetwork/java/javase/downloads/jce-7-download-432124.html
  Copy the jar files from the extracted zip to <JRE_HOME>\lib\security.

- MySQL database server listening on the default port 3306:
  http://dev.mysql.com/downloads/windows/installer/5.6.html

Please install the above software on your host machine. For further information and installation details, we kindly advise you to refer to the manufacturers' websites.

4. CONFIGURE YOUR ENVIRONMENT

4.1. Package Overview

Download the CIPA eDelivery Distribution from the shared drive:

u:\COMMON\CIPASHARE\eDelivery\BRIS pilot\November release\cipa-edelivery-distribution-3.1.0-as4-jboss.zip

This package contains the following structure:

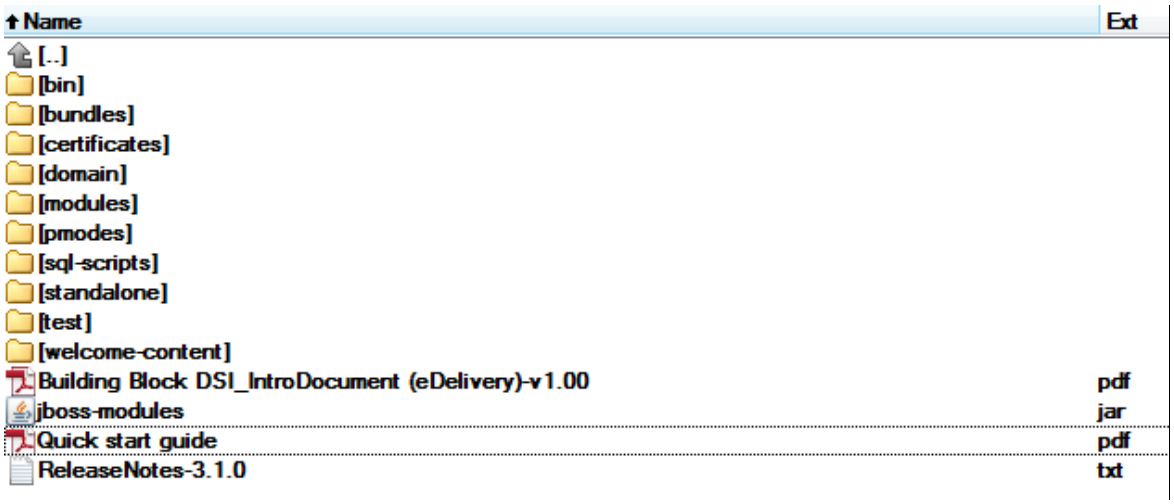


Figure 2 Package content

- **<CEF-eDelivery path>/bin** contains the executable batch file (windows) and shell script (linux) required to launch the JBoss instance.
- **<CEF-eDelivery path>/certificates** contains a keystore (including private keys of instance A and instance B) and a trustore (including public keys of Instance A and instance B) that can be used by both instances. For this test release, each instance uses self-signed certificates. Please refer to Annex 5 for more information about AS4 security.
- **<CEF-eDelivery path>/modules/eu/europa/ec/cipa/configuration/main/domibus/** contains domibus configuration files.
- **<CEF-eDelivery path>/pmodes** contains an AS4 processing mode (pMode-configuration.xml) pre-configured to use compression, payload encryption, message signing and non-repudiation. The provided PMode file must be updated for both instances.
- **<CEF-eDelivery path>/sql-scripts** contains the required application sql code that needs to be executed on MySQL database.
- **<CEF-eDelivery path>/test** contains a SOAP UI test project.

4.2. JBoss Standalone Instance

As described in the purpose of this guide, we need to configure two instances running on two separate machines. Therefore, the procedure below would need to be applied on both machines 'Hostname A' (local machine) and 'Hostname B' (remote machine). Please note that an extra step is only required for 'Hostname B'.

1. Extract the zip file containing the installation package of the CIPA eDelivery to a location on your physical machine, which we will refer to in this document as your "< eDelivery installation path >".

2. Open a command prompt and navigate to this directory:

   < eDelivery installation path >\sql-scripts.

3. Execute the following commands in the command prompt :

```
mysql -h localhost -u root --password=root -e "drop schema if exists edelivery;create schema edelivery; create user edelivery identified by 'edelivery';grant all on edelivery.* to edelivery;
```

```
mysql -h localhost -u root --password=root edelivery < create-mysql.sql
```

Note: if you are using Windows, make sure to have mysql.exe added to your PATH variable.

4. Update default properties of my.ini (Windows) or my.cnf (Linux)

   a. 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
```

   b. 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
```

5. Restart MySQL service:

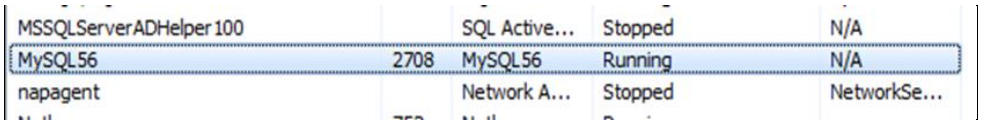


Figure 3 MySQL service

6. This step is only required for the **Hostname B**

Edit **<CEF-eDelivery path>/modules/eu/europa/ec/cipa/configuration/main/domibus/domibus-configuration.xml** and replace "instanceA" with "instanceB" as indicated below:

```
<!-- The default keystore alias to use, if none is specified. -->
<prop key="org.apache.ws.security.crypto.merlin.keystore.alias">instanceB</prop>
```

7. You can now start the JBoss standalone instance on your computer.

Execute:

a. bin/standalone.sh (for Linux)

b. bin/standalone.bat (for windows)

Expected result:

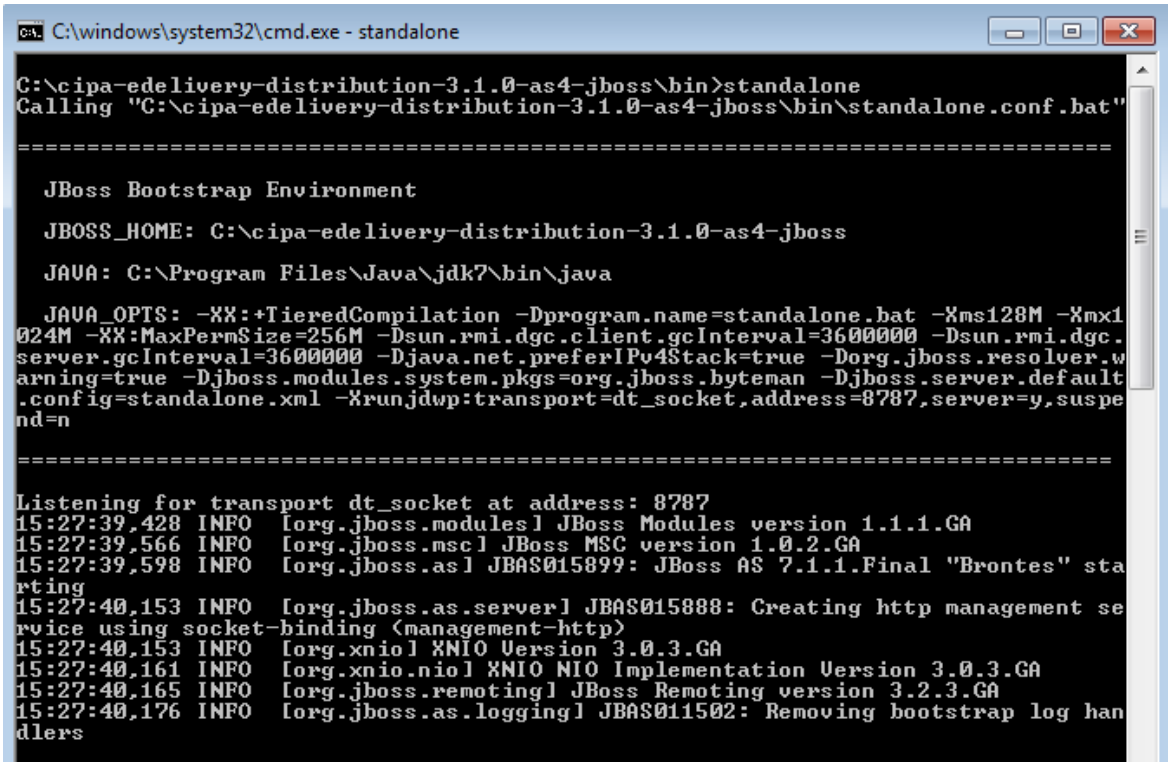


Figure 4 JBoss instance up and running

Remark:

If the application server doesn't start properly, more details about the encountered errors can be found in the log files. Refer to ***<CEF-eDelivery path>/standalone/log/***

8. Once the application server is started, you can ensure that this server is operational by displaying the administration dashboard (*http://localhost:8080/domibus/home*) in your browser as below:

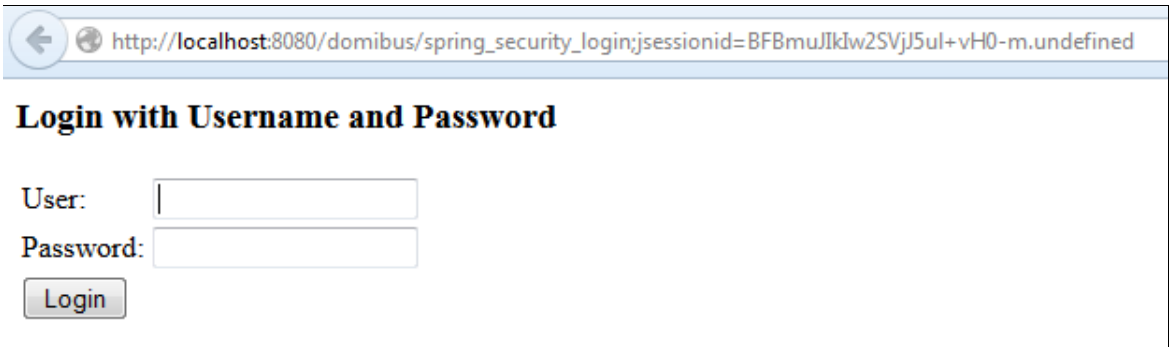


Figure 5 Domibus administration page

Remarks:

- *To allow the remote application to send a message to this machine, you would need to create a dedicated rule (to allow this port) from your local firewall (cf. annex "Firewall Settings")*

- *In the default configuration, the JVM allocation memory is set to 1024. This parameter has to be reduced if the system cannot reserve enough space during the initializing of the application server (edit the parameter Xmx in the file "bin/standalone.conf.bat"):*

```
rem # JVM memory allocation pool parameters - modify as appropriate.
set "JAVA_OPTS=-Xms128M -Xmx1024M -XX:MaxPermSize=256M"
```

- *If you intend to install both instances on the same server, you will need to change instance B ports to avoid conflicts and database schema 'edelivery' before starting the server.*

9. Edit ***< eDelivery installation path >\pmodes\pMode-configuration.xml*** and replace 'UndefinedHostnameA' and 'UndefinedHostnameB' with their real names as indicated below:

```
<parties>
    <partyIdTypes>
        <partyIdType name="exampleType" value="http://www.domibus.eu/exampleType"/>
    </partyIdTypes>
    <party name="instanceA"
           endpoint="http://UndefinedHostnameA:8080/domibus/services/msh"

        <identifier partyId="instanceAId1" partyIdType="exampleType"/>
        <identifier partyId="instanceAId2" partyIdType="exampleType"/>
    </party>
    <party name="instanceB"
           endpoint="http://UndefinedHostnameB:8080/domibus/services/msh"

        <identifier partyId="instanceBId1" partyIdType="exampleType"/>
        <identifier partyId="instanceBId2" partyIdType="exampleType"/>
    </party>
</parties>
```

Figure 6 PMode view

For more details about the provided PMode, please see Annex 4.

10. Upload the PMode file on both instances:

a. To upload a PMode xml file, connect to the administration dashboard using your credentials (by default: login = admin; password = 123456) to http://localhost:8080/domibus/home

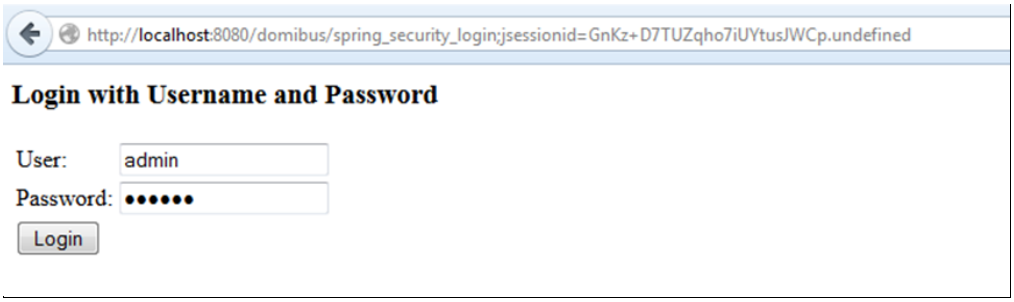


Figure 7 Login to administration dashboard

b. Click on 'Upload PMode' tab

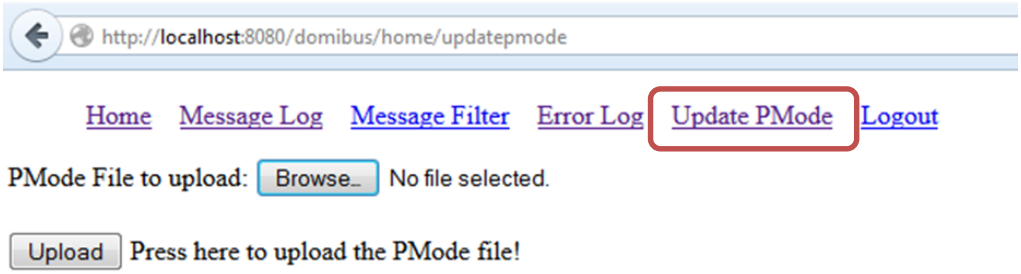


Figure 8 PMode update

c. Select your PMode from "< eDelivery installation path >\pmodes" and hit "Upload"

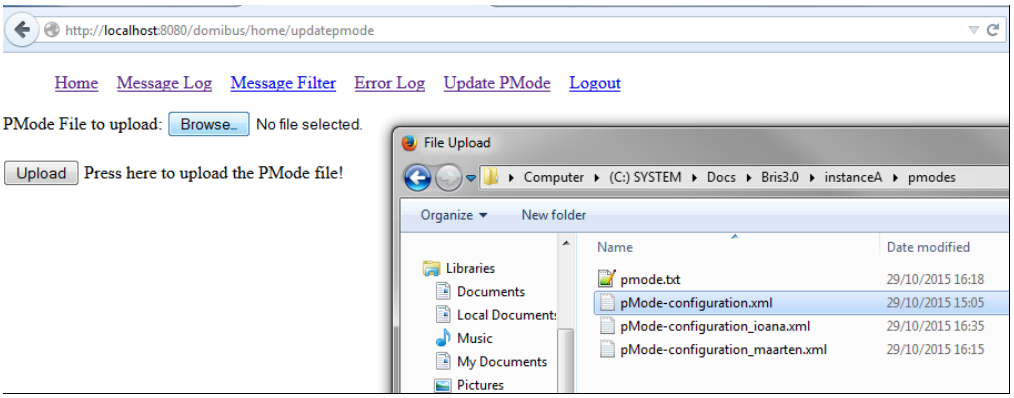


Figure 9 PMode uploading

Now your JBoss instances are running and ready to send or receive messages.

5. TESTING

In this release, we have implemented a "service façade" where the participant (business registers or ECP) will connect. This "service façade" has an interface(s) which can easily interact with the AS4 gateway (domibus). For testing purposes, this release implements a backend Web Service that provides an example of usage of this "service façade" (sendMessage). This interface is a Web Service entry point (SOAP HTTP binding).

This interface is a Web Service entry point (SOAP HTTP binding).

Notification mechanisms rely on an abstract interface that participants will need to implement themselves. This release does not implement the dynamic routing. Therefore all messages will be pushed to the same pre-configured endpoint defined in the ***<domibus.config.location>/backend-common-configuration.xml*** *under the "domibusProperties"* tag.

For testing purposes we have added by default a Web Service Mocking running on port 8088 (make sure this port is not used) as the final recipient's endpoint .

```
<util:properties id="domibusProperties">
…
<prop
key="domibus.recipient.wsdl">http://localhost:8088/mockDomibusServiceWSI
mplServiceSoapBinding?WSDL</prop>
<prop
key="domibus.recipient.namespaceURI">eu:europa:ec:cipa:domibus:services<
/prop>
<prop
key="domibus.recipient.localPart">DomibusServiceWSImplService</prop>
</util:properties>
```

For full instructions on how to send messages, please see the "Test guide.pdf", in the test folder.

Remark:

> *If you encounter connection timeouts on the test you should increase the Socket Timeout setting of SoapUI. This can be done in the following File -> Preferences.*

ANNEXES

ANNEX 1 PARAMETERS

Parameters	Local instance (Instance A)	Remote instance (Instance B)
Host Name	Host Name A	Host Name B
Database	MySQL database	MySQL database
Administrator Page	Username: admin Password: 123456 http://localhost:8080/domibus/home	Username: admin Password: 123456 http://localhost:8080/domibus/home
Database Schema	edelivery	edelivery
Database connector	jdbc:mysql:// "UndefinedHostnameA":3306/edeliver	jdbc:mysql:// "UndefinedHostnameB":3306/edelivery
DB username/password	edelivery/edelivery	edelivery/edelivery
PModes XML files	pmodes/pMode-configuration.xml	pmodes/pMode-configuration.xml
Keystore location	certificates/keystore.jks	certificates/keystore.jks
Keystore Alias Name	"instanceA"	"instanceB" "to be edited in "domibus-configuration.xml"
JBoss Admin console url	http://localhost:9990/console/index.html	http://localhost:9990/console/index.html

ANNEX 2 FIREWALL SETTINGS

Depending on your configuration, the firewall settings might prevent you from exchanging messages from your local JBoss instance and the other one remotely deployed.

To test if a port is blocked or opened, you can use the tool "telnet" and run the command "telnet <server_ip> <port>". If the port is blocked then you need to open it.

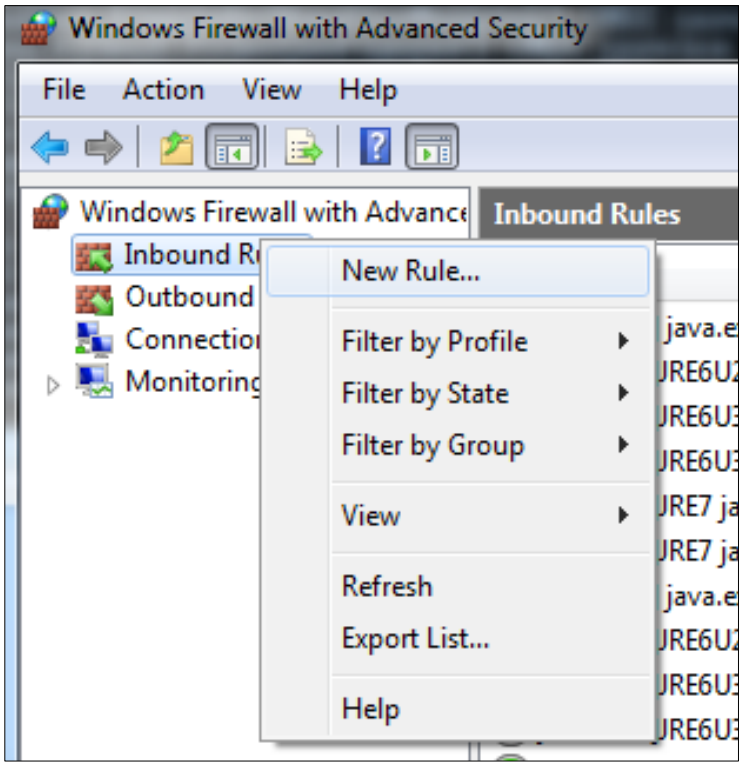
The following ports must be opened on both machines A and B (TCP protocol):

- 8080 (HTTP port)
- 5445 (JMS messaging)
- 5455 (JMS messaging)
- 3306 (MySQL port)
- 9990 (JBoss admin console)

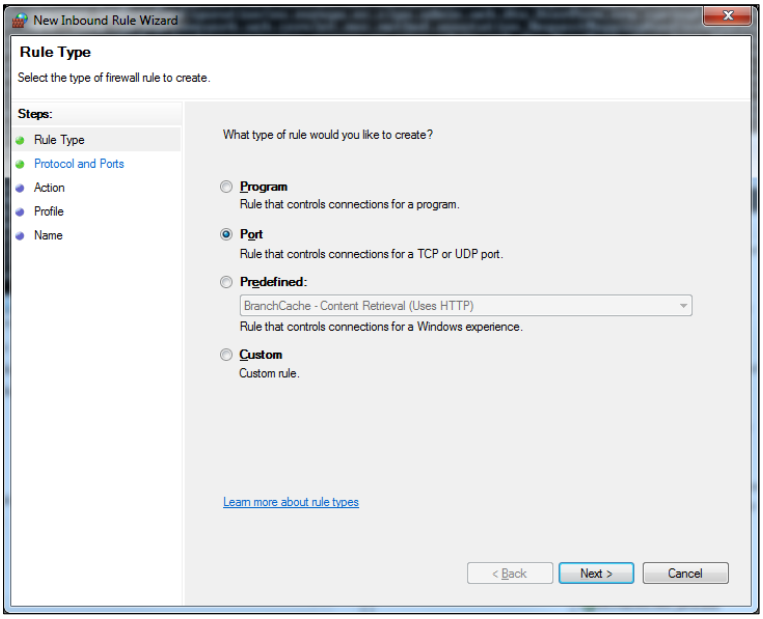
If your computer is protected by the Windows firewall, this is how you can open a port:

1. Open the "Windows Firewall with Advanced Security window": click on Start > Control Panel > System and Security > Windows Firewall and then click on "Advanced Settings".

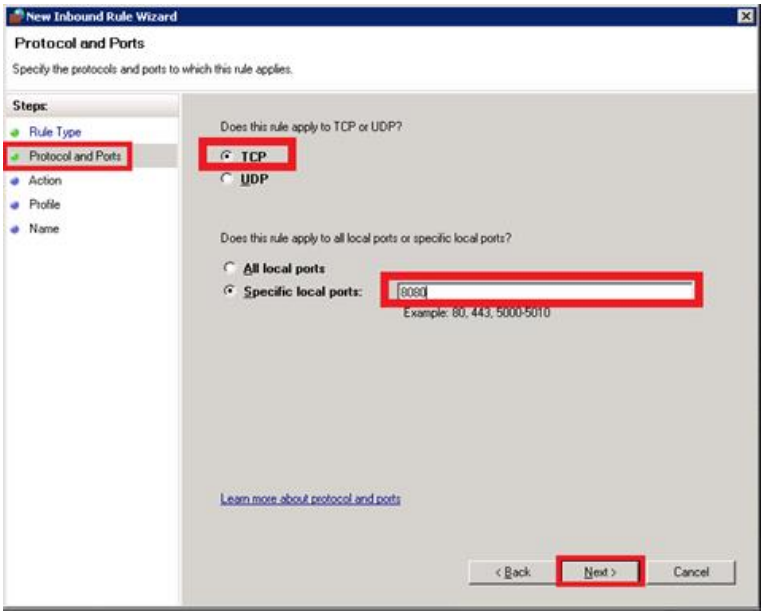
2. Right click on "Inbound Rules New Rule":



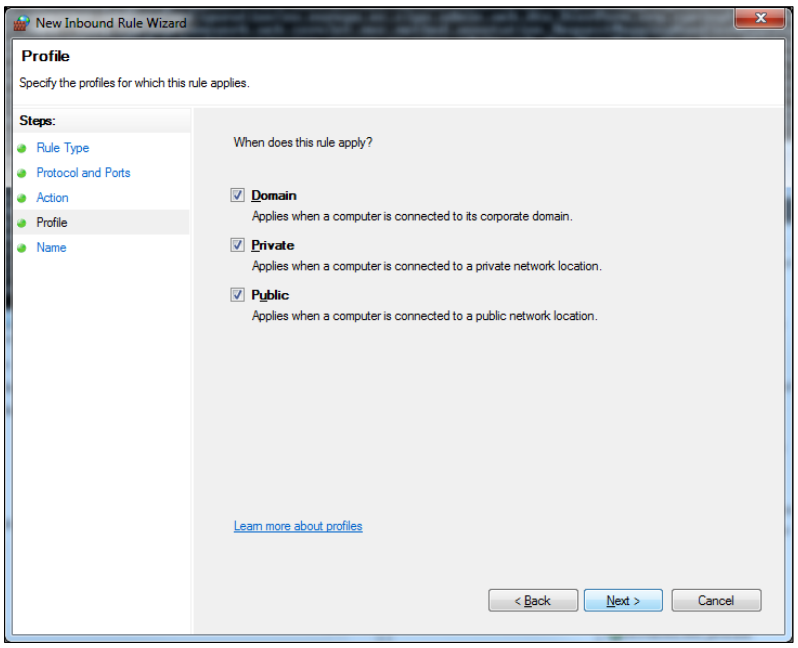
3. Select "Port" and click on "Next":



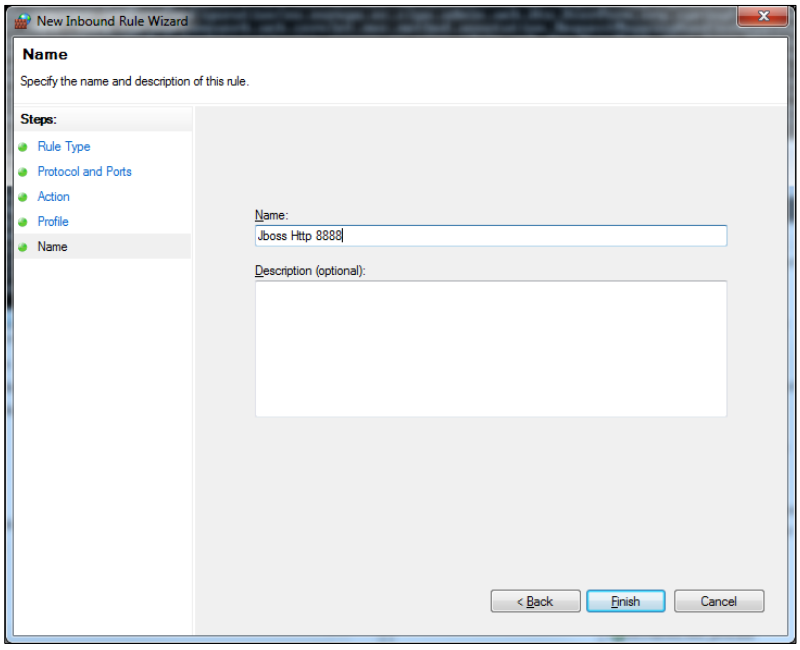
4. Enter a specific local port (for example 8080) and click on "Next":



5. Click on "Next":



6. Name the rule and click on "Finish":



ANNEX 3 DEFAULT PORTS CONFIGURATION

The default package includes an embedded JBoss application server with default ports (as illustrated in the table below).

Default parameter	Value
Server listening HTTP port	8080
MySQL Port	3306
JBoss remote port	4447
JBoss admin Console	9990

ANNEX 4 PROCESSING MODE

Processing modes (PModes) describe how messages are exchanged between AS4 partners (Instance A and Instance B). These files contain the identifiers of each AS4 gateway (identified as parties in the PMode file below).

InstanceAId1, instanceAId2, instanceBId1, instanceBId2 represent the clients' backend connected to their associated AS4 gateway respectively (instanceA and instanceB). Therefore, adding, modifying or deleting a participant imply modifying those PMode files.

In a production environment, you will have an XML file for each instance generated by a plugin (External plugin for Eclipse). This XML file is updated every time a new partner is added or modified. For testing purposes, you will simply need to edit the PMode file dedicated to Instance B.

Here is an example of the content of a PMode XML file:

```
<?xml version="1.0" encoding="UTF-8"?>
<db:configuration xmlns:db="http://domibus.eu/configuration"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://domibus.eu/configuration
file:/C:/development/git-repos/domibus/Domibus-MSH/domibus-
configuration.xsd" party="instanceA">
      <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="60"/>
      </mpcs>
      <businessProcesses>
            <roles>
                  <role name="default"
                              value="http://docs.oasis-
open.org/ebxml-msg/ebms/v3.0/ns/core/200704/defaultRole"/>
                  <role name="exampleMessageProducer"
                              value="exampleMessageProducer"/>
                  <role name="exampleMessageReceiver"
                              value="exampleMessageReceiver"/>
```

```
            </roles>
            <parties>
                  <partyIdTypes>
                        <partyIdType name="exampleType"
value="http://www.domibus.eu/exampleType"/>
                  </partyIdTypes>
                  <party name="instanceA"

      endpoint="http://UndefinedHostnameA:8080/domibus/services/msh"
>
                        <identifier partyId="instanceAId1"
partyIdType="exampleType"/>
                        <identifier partyId="instanceAId2"
partyIdType="exampleType"/>
                  </party>
                  <party name="instanceB"

      endpoint="http://UndefinedHostnameB:8080/domibus/services/msh"
>
                        <identifier partyId="instanceBId1"
partyIdType="exampleType"/>
                        <identifier partyId="instanceBId2"
partyIdType="exampleType"/>
                  </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="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="listPropertySet">
                        <propertyRef
property="finalRecipientProperty"/>
                        <propertyRef
property="originalSenderProperty"/>
                  </propertySet>
            </properties>
            <payloadProfiles>
                  <payload name="businessContentPayload"
                              cid="BRISPayload"
                              required="true"
                              mimeType="text/xml"/>
                  <payload name="businessContentAttachment"
                              cid="BRISAttachment"
                              required="false"
                              mimeType="application/octet-stream"/>
                  <payloadProfile name="BRISMessageProfile"
                              maxSize="40894464">
                        <attachment name="businessContentPayload"/>
                        <attachment
name="businessContentAttachment"/>
                  </payloadProfile>
            </payloadProfiles>
            <securities>
                  <security name="signAndEncrypt"
                              policy="signEncrypt.xml"
                              signatureMethod="RSA_SHA256" />
            </securities>
            <errorHandlings>
                  <errorHandling name="BRISErrorHandling"
                              errorAsResponse="true"
                              businessErrorNotifyProducer="true"
                              businessErrorNotifyConsumer="true"
```

```
                              deliveryFailureNotifyProducer="true"/>
            </errorHandlings>
            <agreements>
                  <agreement name="exampleAgreement"
value="http://domibus.eu/agreement" type=""/>
            </agreements>
            <services>
                  <service name="as4Service" value="AS4"
type="exampleService"/>
            </services>
            <actions>
                  <action name="sendMessage" value="SendMessage"/>
            </actions>
            <as4>
                  <receptionAwareness
name="exampleReceptionAwarenessRetryThreeDuplicateDetectionTrue"
retry="1;6;CONSTANT" duplicateDetection="true"/>
                  <reliability
name="exampleReliabilityNonrepudiationTrueReplypatternResponse"
nonRepudiation="true" replyPattern="response"/>
            </as4>
            <legConfigurations>
                  <legConfiguration name="examplePushLegOne"
                              service="as4Service"
                              action="sendMessage"
                              payloadProfile="BRISMessageProfile"
                              defaultMpc="defaultMpc"

      reliability="exampleReliabilityNonrepudiationTrueReplypatternRe
sponse"
                              security="signAndEncrypt"

      receptionAwareness="exampleReceptionAwarenessRetryThreeDuplicat
eDetectionTrue"
                              errorHandling="BRISErrorHandling"
                              compressPayloads="true">
                              </legConfiguration>
            </legConfigurations>
            <process name="as4exampleProcess"
                        agreement="exampleAgreement"
```

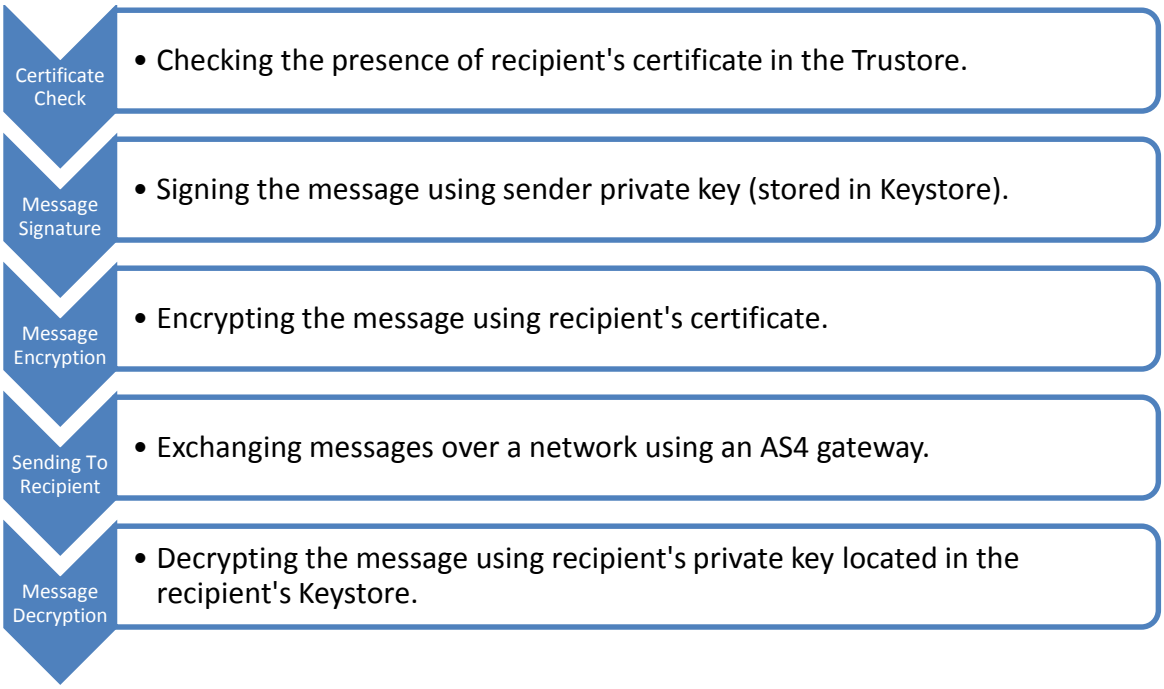
```
                        mep="oneway"
                        binding="push"
                        initiatorRole="exampleMessageProducer"
                        responderRole="exampleMessageReceiver">
                  <initiatorParties>
                        <initiatorParty name="instanceA"/>
                        <initiatorParty name="instanceB"/>
                  </initiatorParties>
                  <responderParties>
                        <responderParty name="instanceA"/>
                        <responderParty name="instanceB"/>
                  </responderParties>
                  <legs>
                        <leg name="examplePushLegOne"/>
                  </legs>
            </process>
      </businessProcesses>
</db:configuration>
```

Remarks:

- *In this setup we have allowed each party (instanceA or instanceB) to initiate the process. If only instanceA is supposed to send messages, we need to put only instanceA in <initiatorParties> and instanceB in <responderParties>.*
- *SSL mutual authentication is only required if we use HTTPS for endpoint. In that case the* ***<CEF-eDelivery path> /modules/eu/europa/ec/cipa/configuration/main/domibus/clientauthentication.xml*** *file is mandatory.*
- *The parameter maxSize (in green) represents the maximum size allowed for a message and its value can be edited according to the user's need.*

ANNEX 5 INTRODUCTION TO AS4 SECURITY

To secure the exchanges between instances A and B (Instance A is sending a message to Instance B in this example), it is necessary to set up each instance's keystore and trustore accordingly. The diagram below provides a short explanation on the main steps of this process:



In order to allow B2B messages and documents between instances A and B, it is necessary to check the following:

For Instance A	For Instance B
Check that Instance B certificate (public key of B) is in trustore.jks of A, if not add it.	Check that Instance A certificate (public key of A) is in trustore.jks of B, if not add it.
Check that the name of B private key is in the keystore.jks, if not add it.	Check that the name of A private key is in the keystore.jks, if not add it.
In "domibus-configuration.xml": the keystore alias should be "instanceA", you might edit the keystore password (by default "test"), and the path to keystore.jks (if you change it).	In "domibus-configuration.xml" edit: the alias property to "instanceB", the keystore password (by default "test") if you need to, and the path to keystore.jks (if you change it).

In a production environment, each participant would need a certificate delivered by a certification authority and remote exchanges between business partners would be managed by each partner's PMode (that should be uploaded on each instance).

For testing purposes, this package provides a pre-configured PMode xml file, a trustore.jks file and a keystore.jks file to be added to instances A and B as described in the Annex 4 .

Remark: If instance A or instance B are behind a local firewall, it is necessary to open the required ports on it. As an example, port 8080 is not opened by default on Windows; we would need to create a dedicated rule on Windows firewall to open TCP 8080 port. See annex "Firewall Settings".