NEX Installation and Upgrade Guide for an ECP/EDX-endpoint in NEM

Version: 4.14.0.4

Date: 14.04.2025

1 Document History

Version	Date	Changes
4.12.0.0	10/1-2024	- Updated document to support ECP v4.12 and EDX v1.13
4.12.0.1	14/3-2024	- Updated document with properties to support monitoring++ in chapter 9.1.1 and chapter 13.1. Made minor clarifications elsewhere.
4.12.0.2	25/4-2024	- Minor (but critical) fix for the ecp.properties (Prometheus -> prometheus)
4.12.0.3	26/9-2024	- Updated ConnectivityCheck-chapter (14.1) and links to troubleshoot-document (chapter 14.3)
4.14.0.0	8/11-2024	- First version to support ECP v4.14 and EDX v1.14, with Artemis. A new chapter has been introduced for Artemis configuration setup.
4.14.0.1	13/11-2024	- Review of Docker-related chapters
4.14.0.2	20/11-2024	- Review by NEX completed
4.14.0.3	10/01-2025	- Minor change in coloring of the AMQP-configuration
4.14.0.4	14/04-2025	- Fixed a missing xml-header in bootstrap.xml (chapter 8.1.1) - Added some properties in ecp.properties (chapter 9.1.1) and edx.properties (chapter 13.1)

2 Terminology

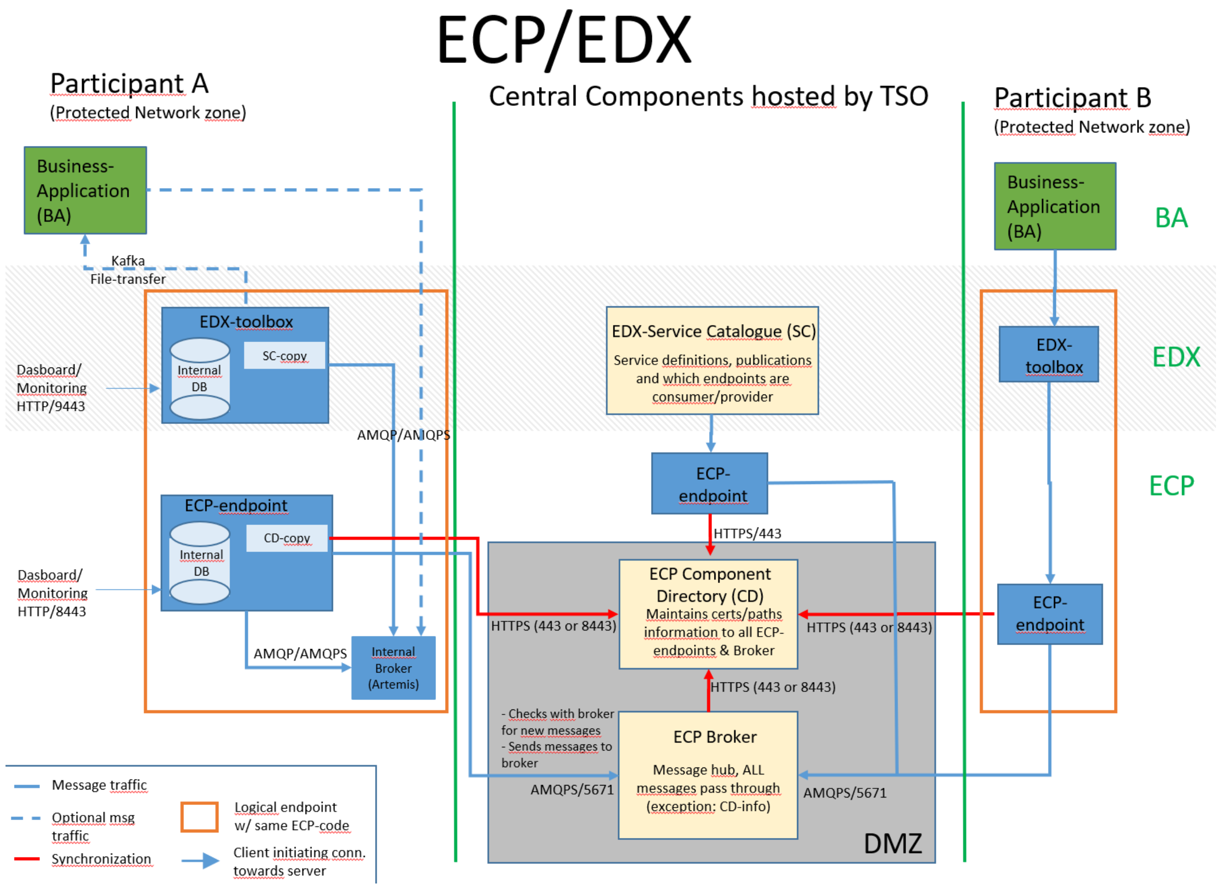
Read later - this is a reference for acronyms/names used in this installation guide. Defined words are in bold font in the "Long" column.

Short	Expanded	Long
AG	ECP/EDX Administration Guide	The ECP/EDX administration guide for ECP made by ENTSO-e/Unicorn
AMQP	Advanced Message Queuing Protocol	A protocol/standard developed in 2014 by OASIS for reliable (persisted) message communication.
BA	Business Application	A "normal" application outside **ECP**, communicating through ECP with other **BAs**. The **BA** must connect with an **EDX-toolbox** to send/receive messages from the network.
Broker	Central Broker	The central broker in an **ECP**-network is directly reachable for all **ECP-endpoints** and all messages in the network are sent to and retrieved from this broker. It supports **AMQP**(S). In addition to the central broker, each **ECP-endpoint** and **EDX-toolbox** also have an "internal broker" (same type of broker) for message handling.
CD	Component Directory	An **ECP**-component/server, maintaining information about all **ECP-endpoints**, **Broker** and **SC**. This is like the phone book of an **ECP**-network.
EC	Endpoint Code	These codes are provided by the TSO and identifies your particular **Endpoint** and is stored in the **CD**. The code is an EIC-code of type V (https://www.entsoe.eu/data/energy-identification-codes-eic/)
ECP	Energy Communication Platform	A platform developed for **ENTSO-E**, by Unicorn, according to **MADES** 1.1/2.x specification – intended to provide secure and reliable messaging between the actors (**TSOs** and others) in the energy sector. The platform consists of **EDX-toolbox**, **ECP-endpoints**, **Broker**, **Component Directory (CD)** and **Service Catalogue (SC)**.
ECP-endpoint	ECP-endpoint	A specific component/server in the **ECP**-network, responsible for sending/receiving messages to/from the central **Broker**.
EDX-toolbox	EDX-toolbox	A "front" to the **ECP-endpoint**, logically a part of the same endpoint. EDX offers a richer set of interfaces for a **BA** to connect to. EDX has a Service concept which allows for more advanced routing of messages and addressing of endpoints.
Endpoint	Endpoint	An endpoint is the "logical endpoint" – a combination of both the **ECP-endpoint** and the **EDX-toolbox**.
EO	Endpoint Operator	The party that operates an **Endpoint** and has the technical communication with the **ECP/EDX**-network and operational communication with the **TSO**.
ENTSO-E	European Network of TSOs	An organization of **TSOs**
HA	High Availability	A term used for a database-setup, with multiple databases in a cluster. This is not part of the regular setup of **ECP**-endpoints, but it is possible to use MySQL or MSSQL for such a setup.
Hawtio	Hawtio	A monitoring software – access it on **ECP-endpoint** and **EDX-toolbox** on /hawtio on whichever port your webserver is running. You can browse your internal broker queues.
IG	ECP/EDX Installation Guide	The ECP/EDX installation guide for ECP made by ENTSO-e/Unicorn
MA	Market Actor	The actor utilizing the data offered through and ECP/EDX-connection. In simple terms: "the business partner". A Market Actor can consist of several legal entities.
MADES	Market Data Exchange Standard	A specification developed by **ENTSO-E** describing a communication system between actors in the energy sector.
NEM	Nordic ECP-network for Market Actors	NEM is the combined ECP network of Statnett, SvK, Fingrid and Energinet, using Internet as the carrier.
NEX	Nordic ECP/EDX Group	The governing group of ECP/EDX in the Nordics, with representatives from Statnett, SvK, Fingrid and Energinet

SC	Service Catalogue	An **ECP**-component/server which keeps information about which endpoints consume/provides certain services. Without registration here, an **EDX-toolbox** cannot access services.
TSO	Transmission System Operator	Responsible for the distribution of energy (electricity or natural gas), in an area/country.
UG	ECP/EDX Upgrade Guide	The ECP/EDX upgrade guide for ECP made by ENTSO-e/Unicorn

3 The big picture

The goal of ECP is to provide secure and reliable messaging between participants in the energy sector, from one Business Applications (BA) to another. The big picture is shown below:


ECP/EDX
Participant A
(Protected Network zone)
Central Components hosted by TSO
Participant B
(Protected Network zone)
Business-Application (BA)
Kafka
File-transfer
EDX-toolbox
Internal DB
SC-copy
Dasboard/ Monitoring HTTP/9443
AMQP/AMQPS
ECP-endpoint
CD-copy
Dasboard/ Monitoring HTTP/8443
AMQP/AMQPS
Internal Broker (Artemis)
EDX-Service Catalogue (SC)
Service definitions, publications and which endpoints are consumer/provider
ECP-endpoint
HTTPS/443
ECP Component Directory (CD)
Maintains certs/paths information to all ECP-endpoints & Broker
HTTPS (443 or 8443)
HTTPS (443 or 8443)
HTTPS (443 or 8443)
- Checks with broker for new messages
- Sends messages to broker
ECP Broker
Message hub, ALL messages pass through (exception: CD-info)
AMQPS/5671
AMQPS/5671
DMZ
BA
EDX
ECP
EDX-toolbox
ECP-endpoint
Message traffic
Optional msg traffic
Synchronization
Logical endpoint w/ same ECP-code
Client initiating conn. towards server

The drawing aims at answering the following questions you might have:

- What components exists in the ECP and which must a particpant in the network install?
- Which ports must be opened in which direction and in which firewall?
- What is the general purpose of the various components in ECP?
- How does the messages flow in the system and how is the logical flow?

Allthough a picture says more than a thousand words, a little explanation might still be in order:

- The messages flow from one BA to the other through a number of steps, follow the blue line for "message traffic)
- All componets are connected to the CD, to retrieve information about rest of the network
- An endpoint should be installed in a protected network zone, no firewall openings into the endpoint is necessary (all traffic is outbound)

4 Preparations

4.1 Download software & binaries

- Go to https://ediel.org/nordic-ecp-edx-group-nex/market-actor-onboarding/ and find links to software and documentation
    - Download "Installation Package" for ECP v4.14.0 (4.15 is not yet approved)
    - Download "Installation Package" for EDX v1.14.0 (1.15 is not yet approved)
    - Download "Docker Binary" if you need these for container-based installation Within these 2 downloads you'll find official documentation, but the documentation is not tailored for NEM Market Actors and can be voluminous (if you ever thought this document was long). You will need the following 9 documents:
        - ECP/EDX Installation Guide (IG) – will be referenced later in this doc
        - ECP/EDX Upgrade Guide (UG) – will be referenced later in this doc
        - ECP/EDX Administration Guide (AG) – advanced configuration, may not be necessary
        - ECP/EDX Release Notes (RN)
        - EDX User Guide (USG) – how BA connects to EDX
    - There are more resources on ENTSO-e website, especially lot's of training videos found here: https://www.entsoe.eu/ecco-sp/training-videos/. These videos are not necessary to watch, but are a repository for various question you may come across later.

4.2 Retrieve Endpoint Code (EC) and Registration Keystore

4.2.1 Statnett

- Go to https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/ and find links to "terms of use". Scan and sign it, one per company (not one per endpoint).
- Contact mailto:ecp@statnett.no with this information:
    - Company name
    - Signed "terms of use"
    - Which network you want to connect to (test or production)
- In return you'll get information you need later on:
    - Registration keystore (jks-file)
    - Endpoint code (EC)

4.2.2 Energinet

Go to https://en.energinet.dk/Electricity/Electricity-market/How-to-get-started-with-ECP/ to get information about the process.

4.2.3 SvK

Follow instructions, provided by SvK, for delivery of Component Codes and Registration keystore. In case of problem send an email to ecp@svk.se.

4.2.4 Fingrid

Request the Endpoint code (EC) for your ECP endpoint from lio@fingrid.fi.

4.3 Software requirements

- OS must be Windows Server 2019/2022, RHEL 8/9 or Oracle Linux 8/9.

- We recommend running a single EDX & ECP on a standalone[1] server to avoid port-conflicts.

4.4 Hardware requirements (see ECP/EDX IG for more information)

	NEX Recommendation (1000*X msg/h)	ENTSO-e Rec. (1000s msg/h)	
	ECP + EDX (same host)	ECP	EDX
CPU	X Cores	4 Cores	2 Cores
Memory	12+X GB	8GB	4GB
Disk	100+10*X GB	40 GB	100 GB

4.5 Firewall configuration

Look at "the big picture" (chapter 3) to identify which ports you need to open. The arrow on the traffic denotes from where the traffic is initiated. The table below summarizes the information and provide specific hosts which are essential for the operation of your endpoint.

Client	Server	Port	Protocol	Doing what?
Operator	EDX-toolbox	9443	HTTPS	Monitoring/Dashboard (GUI)
BA	Artemis Internal Broker	5672/5671	AMQP/AMQPS	Message transport, **if** you decide to use AMQP(S) to connect from BA -> EDX
Operator	ECP-endpoint	8443	HTTPS	Monitoring/Dashboard (GUI)
ECP-endpoint – open **only** to your TSO CD	CD (prod): **ecp4prod.statnett.no** **iecp.prod.energinet.dk** **ecp4.svk.se** **ecp.fingrid.fi*** CD (test): **ecp4.statnett.no** **iecp.preprod.energinet.dk** **ecp4-test.svk.se** **ecp-test.fingrid.fi***	443 for Statnett 8443 for other TSOs	HTTPS	Synch of ECP-network information/certificates. Essential for ECP to work
ECP-endpoint – open to **all** TSO	Broker (prod): **ecp4prod.statnett.no** **iecp.prod.energinet.dk** **ecp4.svk.se** **ecp.fingrid.fi*** Broker (test): **ecp4.statnett.no** **iecp.preprod.energinet.dk** **ecp4-test.svk.se** **ecp-test.fingrid.fi***2	5671	AMQPS	Message transport. Essential for ECP to work.

4.6 Test environment is mandatory

NEX requires a test-endpoint which will be connected to NEM TEST. **NB!** It is **not possible** to connect from test to production (or vice versa) in NEM. The Endpoint Code (EC) is decided by the TSOs and will differ between test and production.

4.7 Endpoint policy

NEM will enforce some rules regarding the number of endpoints you can connect:

[1] An OS dedicated to run ECP-endpoint and EDX-toolbox and no other servers/processes. It doesn't matter if it's a virtual or physical server.

* Fingrid has whitelisting of IPs, only Nordic IPs are allowed plus IPs from Azure-WestEurope (Netherlands). If your endpoint is located in an "exotic" location then you need to contact Fingrid.

- The rule of thumb is one Endpoint pr Market Actor (MA). The Endpoint may be operated or even owned by an Endpoint Operator (EO). See Terminology chapter for definition of terms.
- An MA may run more than one Endpoint if it is necessary or beneficial. A reason could be that the MO uses various EOs, or that it will reduce risk of service interrupts to distribute the traffic to more Endpoints. The TSO may require the MA to add extra endpoints.
- An Endpoint must not be used by more than one MA.
- If the EO is not the same entity as the MA, then it is expected that information about the operation which goes from the TSO to the EO will be distributed to the MA by the EO.
- In NEM-TEST less strict rules can be applied. Thus, for example System Integrators (typically those that take on the EO role) can have endpoints as well as MAs.

5 Installation or Upgrade of ECP-endpoint on Windows

5.1 Java

Version 17.0.10 is the recommended version – although also later versions of Java 17 will most likely work fine (but is not tested). To get Java in correct state execute the following chapters:

- For installation: ECP Installation Guide (IG) chapter 5.1
- For updating: ECP Upgrade Guide (UG) Chapter 5.1.1 - 5.1.3.

NB! Some tips to get a successful installation – perform this check **AFTER** java-installation/upgrade and **BEFORE** you install the ECP-package.

- Make sure that you have JRE (not JDK!) installed
- Make sure the environmental variable JRE_HOME is defined and points to the folder where JRE is installed
- Make sure the environmental variable JAVA_HOME is removed
- Do not have any MMC-consoles opened
- Do not have any Services-windows (services.msc) open – services might not be created correctly if you "occupy" that particular window
- Do not have any explorer-window or command-window opened within the file locations of ECP – files may not be deleted correctly if you are "in the way".

5.2 Installation – skip if not applicable

Execute ECP IG chapter 7.2 and then ECP IG Chapter 8.2.

5.3 Upgrade – skip if not applicable

You may read the ECP Release Notes to get more information. Then execute ECP UG chapter 5.1.4, 5.1.5 and 5.1.6 (5.1.7 and 5.1.8 are not strictly necessary at this point). Warning: A re-installation could be easier than upgrade.

6 Installation or Upgrade of ECP-endpoint on Linux

6.1 Installation

If you do a first-time installation, then execute ECP Installation Guide (IG) chapter 5.1 (Java installation).

Then execute ECP IG chapter 7.4 and chapter 8.4

6.2 Upgrade

You may read the ECP Release Notes to get more information (see chapter 4.1). Then Execute the ECP Upgrade Guide (UG) chapter 5.2.

7 Installation or upgrade of ECP endpoint using Docker image

Docker support is limited, and it is expected that users have more understanding and experience than those that install on Linux or Windows. Still, there are a few resources that may provide the necessary support to get you through this:

- This installation guide you're reading. It intends to tie together the other resources and also provide some extra information where it seems to be lacking
- For the images themselves, please see chapter 4.1.
- The standard ECP Download package contains examples of setup
    - Ecp-docker-test-env.zip (a "bare-bone" setup – see chapter 7.1.1)
    - Ecp-endpoint-kubernetes.zip (useful for "cloud" setup – see chapter 7.1.2)
- The ECP Installation Guide chapter 15.

Warning 1: If you do not use local disks (typical in cloud setup), you may run into problem with hangup/freeze in some of the lower layer network stack. To avoid problems as much as possible, do make sure to have fast disks rather close to the host running the container.

Warning 2: You may believe that using containers as runtime makes ECP scalable, but that is only true if you setup a so-called High Availability (HA) configuration of ECP and this is not covered in this document because NEX has low confidence that HA solves more problems than it creates. Thus, the best reason for going with containers is that your entire environment is container-based.

This chapter is not focused on the whole configuration of ECP, but rather on the specific parts related to Docker. Therefore, in order to configure ECP, please read the relevant section for Linux (see chapter 8).

7.1 Installation

7.1.1 Bare-bone Docker Host setup

You may use the docker-compose.yml in the zip-files provided in the download. Such a setup is convenient for testing/development.

7.1.2 Cloud setup

The log folder and the activemq temporary data folder should not be mounted on “slow” network attached storage. This may cause log events to be missed or delays in the AMQP message flows. If you see problems of this kind, please consider faster or "more local" storage.

7.1.2.1 Database

In general, it is recommended to use an external database when deploying containers. Storing database files on the container host itself is considered bad practice, because it locks in the container to that specific host.

If the storage used for the local disk in the container is physically located elsewhere than on the container host itself (e.g., probably in any cloud environment), please configure the endpoint to use an external database, instead of the default embedded Derby database. Running a Derby database with its data files stored on network attached storage has proven to result in all sorts of weird issues with the endpoint.

Read IG chapter 13 (External Databases) for details on configuration for use of an external database.

7.1.2.2 Storage for ECP network keystore files
Please mount the /var/lib/ecp-endpoint folder to a folder outside of the image. As the keystore files are stored in the database, and copied to the file storage on each startup, there is no need to use persistent storage for this folder.

Not doing as described above prevents use of different keystore passwords than the default, because the software will report password failure for the authKeystore.jks file during startup, which will result in the software running in a half-broken state, where one sees a 404 when the dashboard interface is being accessed.

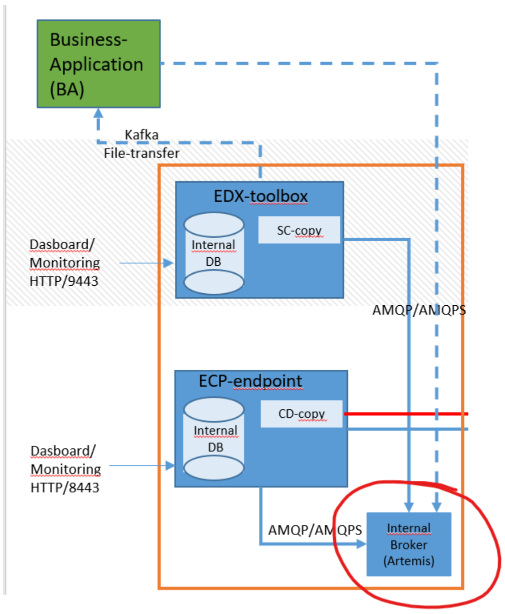
Checking the keystore file in the folder after startup, the file appears to have the correct password. But due to what seems like a timing issue, the software is reading the keystore file provided by the image, before the file has been updated from the content in the database, hence throwing a keystore password error.

7.2 Upgrade
The official documentation from Unicorn is not comprehensive when it comes to Docker Installation. However, it should suffice to follow the advice for Linux (see previous chapter) to the best of your abilities. At least the part about configuration change will apply also for Docker. Also read ECP Upgrade Guide (UG) chapter 8 and ECP Release Notes to get information on what has changed.

Extract the config files from the new version of the docker image and compare these to the config files in the current container. Be sure to also check for changed or new Java VM variables and update accordingly.

8 InternalBroker/Artemis Setup (for all OS installations)

In ECP v4.14+ and EDX v1.14+ the "InternalBroker" (IB), previously a part of the ECP and EDX, is now a totally separate system. The drawing below (taken from chapter 3) highlights the component. This component is the Apache Artemis ActiveMQ broker, with some configuration to make it suited for its role in the ECP/EDX-endpoint.


Business-Application (BA)
Kafka
File-transfer
EDX-toolbox
Internal DB
SC-copy
Dasboard/ Monitoring HTTP/9443
AMQP/AMQPS
ECP-endpoint
Internal DB
CD-copy
Dasboard/ Monitoring HTTP/8443
AMQP/AMQPS
Internal Broker (Artemis)

As seen from the drawing, there is one connection made from EDX-toolbox towards the IB, and another from the ECP-endpoint. In this section we will set up the IB configuration and at the same time explain how to configure the connections from EDX and ECP. This is done to show how the various configuration these components relates to one another. At this point in the installation guide you might not have installed EDX yet, but once you have – you can come back to this section to check how to configure the AMQP/S connection to the IB.

8.1 InternalBroker GUI/API/metrics

Before we configure the AMQP/S part, let's focus on the GUI of the IB. This interface is available on port 8161. The interface offers something resembling Hawtio (for those that used that in older versions of ECP/EDX). But the interface also offers Prometheus metrics and an API that tools like ekit.jar (see https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/) can extract information and perform operations.

8.1.1 Bootstrap.xml

We suggest the following as a template for bootstrap.xml. The brackets with bold font content is supposed to be replaced by you.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<broker xmlns="http://activemq.apache.org/schema">
   <jaas-security domain="activemq"/>
   <web path="web" rootRedirectLocation="console">
       <binding name="artemis" uri="https://0.0.0.0:8161"
            keyStorePath="file:[your-jks-file]"
            keyStorePassword="[password]"
            trustStorePath="file:[your-jks-file]"
            trustStorePassword="[password]"
            includedTLSProtocols="TLSv1.3"
            includedCipherSuites="TLS_AES_256_GCM_SHA384,TLS_CHACHA20_POLY1305_SHA256,TLS_AES_128_GCM_SHA256"
            sniHostCheck="false">
           <app name="branding" url="activemq-branding" war="[path]/eccosp-artemis/web/activemq-branding.war"/>
           <app name="plugin" url="artemis-plugin" war="[path]/eccosp-artemis/web/artemis-plugin.war"/>
           <app name="console" url="console" war="[path]/eccosp-artemis/web/console.war"/>
           <app url="metrics" war="metrics.war"/>
       </binding>
   </web>
</broker>
```

8.1.2 Jolokia-access.xml

The jolokia-access-file restrict access to the API for tools like previously mentioned ekit.jar. **If** you want to be able to use ekit.jar, then this is a simple solution to allow remote access:

```
<?xml version="1.0" encoding="utf-8"?>
<restrict>
    <cors>
        <allow-origin>*</allow-origin>
    </cors>
</restrict>
```

8.2 AMQPS/AMQP-configuration

One important choice to make is whether the IB will support AMQPS or AMQP.

The reasons for AMQPS are:

- EDX and ECP runs on different hosts (typically container environment)
- A BA is connected and uses AMQPS
- It's the default setup – best practice

The reasons for AMQP are

- A BA is connected and uses AMQP – and you want to avoid changing the BA
- EDX and ECP runs on same host (typically Windows/Linux installation) and no BA is connected
- Avoid the PKI involved in distributing/updating the AMQPS certs every year, especially if a BA is involved

8.2.1 Artemis configuration files

Keep in mind the following before looking at the configuration files:

- The following shows configuration for AMQPS.
- In this example we've chosen to re-use the AUTH-certificate (referred to as "ecp_module_auth") from ECP, since it's readily available and is renewed every so often. This is a simple option to get started. The example shows that we use the exact same file, but the important thing is to have a jks-file with the appropriate certificate in it. The mature solution would be to create your own host-certificates and renew them every so often.
- Only the relevant part of each configuration file is shown, not the entire files.
- Each property that is related to another property in another property-file is shown with a color.
- Some newlines/whitespace is added to make it readable, remove those if copying
- The broker is listening on 0.0.0.0, which is to say: available for remote connections.

broker.xml:

```
<!--security-enabled>false</security-enabled-->
<acceptors>
   <acceptor name="amqps-internal">
      tcp://0.0.0.0:5672?
         sslEnabled=true;
         keyStorePath=/var/lib/ecp-endpoint/authKeystore.jks;
         keyStorePassword=password;
         keyStoreAlias=ecp_module_auth;
         trustStorePath=/var/lib/ecp-endpoint/authKeystore.jks;
         trustStorePassword=password;
         needClientAuth=false;
         protocols=AMQP;
         enabledProtocols=TLSv1.3;
         enabledCipherSuites=TLS_AES_256_GCM_SHA384,
                             TLS_CHACHA20_POLY1305_SHA256,
                             TLS_AES_128_GCM_SHA256
   </acceptor>
</acceptors>
```

artemis-users.properties:

```
endpoint = password
toolbox = password
```

artemis-roles.properties:

```
amq = endpoint,toolbox
```

ecp.properties:

```
internalBroker.host=127.0.0.1
internalBroker.amqp.port=5672
internalBroker.useAuthentication=true
internalBroker.keystore.location=/var/lib/ecp-endpoint/authKeystore.jks
internalBroker.keystore.password=password
internalBroker.keystore.authAlias=ecp_module_auth
internalBroker.auth.user=endpoint
internalBroker.auth.password=password
internalBroker.parameters=jms.prefetchPolicy.queuePrefetch=10
```

edx.properties

```
internalBroker.amqp.host=127.0.0.1
internalBroker.amqp.port=5672
internalBroker.useAuthentication=true
internalBroker.keystore.location=/var/lib/ecp-endpoint/authKeystore.jks
internalBroker.keystore.password=password
internalBroker.keystore.authAlias=ecp_module_auth
internalBroker.auth.user=toolbox
internalBroker.auth.password=password

ecpBroker.amqp.host=127.0.0.1
ecpBroker.amqp.port=5672
ecpBroker.useAuthentication=true
ecpBroker.keystore.location=/var/lib/ecp-endpoint/authKeystore.jks
ecpBroker.keystore.password=password
ecpBroker.keystore.authAlias=ecp_module_auth
ecpBroker.auth.user=toolbox
ecpBroker.auth.password=password
ecp.broker.url=amqps://${ecpBroker.amqp.host}:${ecpBroker.amqp.port}
```

Import notes

- If you decide to follow this example to the letter, make sure that your Artemis and EDX process have read-access to the authKeystore.jks which is owned by ECP process.
- If you want to run AMQP, then change the orange-colored fields (uncomment the first, set the next four orange to "false" and change to "amqp" in the last orange field). The blue/turquoise-colored fields are no longer used

9 ECP Setup (for all OS installations)

9.1 ECP-endpoint configuration

9.1.1 Configuration of ecp.properties

Make sure the properties in the table are configured. Some of them might be in place, others might need to be changed. In addition, there will be properties already defined, but not mentioned here – that is as expected and presents no problem.

NB! Only the application and root/administrator should have access to this configuration file.

Property	Description

`ecp.endpoint.amqpApiEnabled = true`
`ecp.endpoint.sendHandler[0].beanName=amqpApiSendHandler`
`ecp.endpoint.sendHandler[0].typeName=*`
	You **MUST ADD** these properties, otherwise messages cannot be sent from ECP to EDX.

`spring.profiles.active= ecp-nonha`	ECP will run in non-HA mode

`ecp.directory.client.synchronization.directorySynchronizationInterval=30 * * * * *`
`ecp.directory.client.statistics.directorySynchronizationInterval=15 * * * * *`
`ecp.directory.client.synchronization.messagePathSynchronizationInterval=45 * * * * *`
	You may have changed them according to Unicorn Guide, but this is a **better setup for our network. Use this one.** With this setup each type of synchronization with the Component Directory will run on different second (15, 30 and 45) in every minute.

`ecp.endpoint.connectivityCheckAttemptTimeout=6000`
`ecp.endpoint.connectivityCheckAttemptsCount=10`
	These are optional settings to allow for 60s (6s*10) timeout on a connectivity-check. The default is 10s. See chapter 14.1 for more info.

`ecp.db.compressionJobEnabled=false`
`ecp.db.messageCompressionJobEnabled=false`
	The database may grow large, thus a regular compression job is wanted. But the job can lead to problems in a high-traffic endpoint, since it will conflict will on-going logging to the database. Since compression is not strictly necessary, it's better to turn off completely. It's possible to run compression safe manually, when the endpoint is stopped.

`ecp.csrf.secret=password`
 	
`management.endpoint.health.show-details=ALWAYS`
`endpoints.prometheus.sensitive=false`
`management.endpoints.web.exposure.include=info,health,readiness,prometheus`
	The csrf-secret is used to generate CSRF-tokens when logging in to ECP GUI. Should perhaps not be 'password'…
	
	The next block are properties related to monitoring. You can read about them in ECP AG. With this setup you will get all possible monitoring turned on, and you can access the following URLs:
	/ECP_MODULE/actuator/prometheus
	/ECP_MODULE/actuator/info
	/ ECP_MODULE/actuator/health
	/ ECP_MODULE/actuator/readiness

`internalBroker.parameters=jms.prefetchPolicy.queuePrefetch=10`	Change prefetch limit from 1000 to 10. Prefetch will cause messages to stick to a consumer, and if that consumer gets "stuck", the messages can be lost.

NB! Do not forget to add the properties listed for ecp.properties in chapter 8.2.1.

9.1.2 Configuration of ecp-users.properties

NB! Only the application and the root/administrator should have access to this file.

The configuration file defines users, roles and passwords needed to log on to the ECP GUI. Here is a simple example of the two types of users available:

```
ecp.endpoint.users[0].login=admin
ecp.endpoint.users[0].password=supersecret
ecp.endpoint.users[0].role=admin

ecp.endpoint.users[1].login=user
ecp.endpoint.users[1].password=secret
ecp.endpoint.users[1].role=user
```

To add more users, simply follow the examples and increase the index. It is advisable to change the passwords. It is possible to hash/encrypt these passwords (see ECP IG or ECP AG, search for 'hash'), but since no one but root/administrator should be able to access this file – it does not improve security much.

9.2 Start ECP-endpoint

For Windows use Services to start/stop or use the command-window with the appropriate privileges to run "**SC start/stop ecp-endpoint**". For Linux use the command "**systemctl start/stop ecp-endpoint.service**". You may start or restart the endpoint now.

9.3 Installation verification

Check the application log files for more information about its status. The catalina-log is useful for telling about the tomcat-startup of the application, while the ecp-log is giving information about what happens thereafter.

Check the ECP-Dashboard-URL:

https://localhost:8443/ECP_MODULE (change "localhost" to the IP-address of the host if needed)

Most browsers will not show this page without giving a security warning. This is because of the usage of self-signed certificate and the fact the hostname of the certificate usually does not match the URL. To properly fix this you must create a TLS-certificate for this host and change some settings in server.xml in Tomcat. The details of TLS-certificate creation are not explained in this document, one needs to check with other resources – it is beside the ECP-domain, and strictly a webserver/browser-issue.

In case login is required (due to settings in ecp.properties and ecp-user.properties), the **default login is admin/password**. See previous chapter for configuration.

9.4 ECP-endpoint Registration – skip this one in Upgrade

9.4.1 Registration process

- Open ECP-Dashboard-URL in a browser
- Select the registration keystore provided by your TSO (see 4.2) and enter the password. You will receive the password from the TSO, or else try "password". Click “Continue” to proceed to the next step.

- Enter the appropriate CD URL and CD Code (see tables below), then click on the "check connectivity"-button. In this step you must know which TSO you're connecting to – you must **only select one** CD Code and CD URL from the tables below. Then click "Continue".

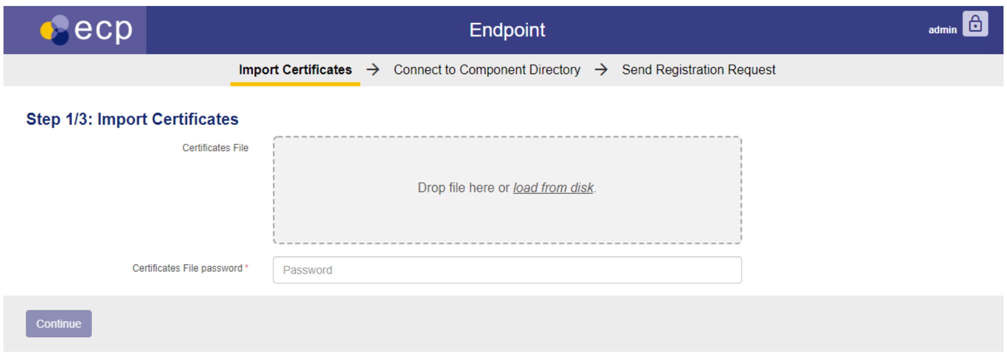
NEM TEST/PREPROD

TSO	CD Code	CD URL
Fingrid	44V000000000017H	https://ecp-test.fingrid.fi:8443/ECP_MODULE
Energinet	45V000000000057R	https://iecp.preprod.energinet.dk:8443/ECP_MODULE
SvK	46V0000000000012	https://ecp4-test.svk.se:8443/ECP_MODULE
Statnett	50V000000000111W	https://ecp4.statnett.no/ECP_MODULE

NEM PROD

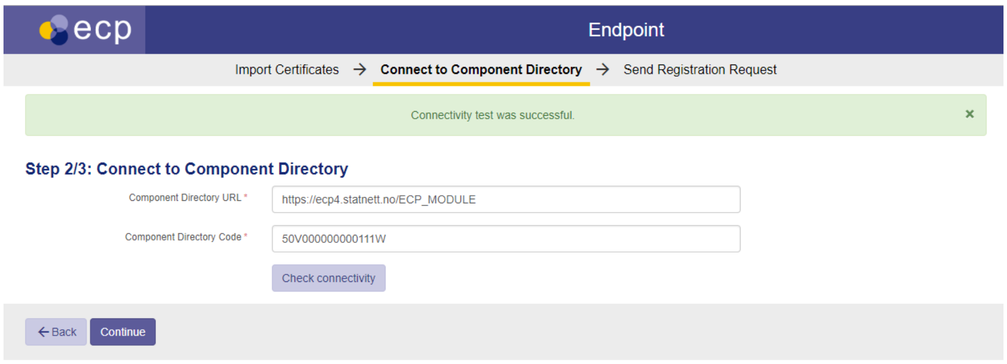
TSO	CD Code	CD URL
Fingrid	44V000000000006M	https://ecp.fingrid.fi:8443/ECP_MODULE
SvK	46V000000000019K	https://ecp4.svk.se:8443/ECP_MODULE
Energinet	45V000000000053Z	https://iecp.prod.energinet.dk:8443/ECP_MODULE
Statnett	50V000000000118I	https://ecp4prod.statnett.no/ECP_MODULE

- Next, fill in your own Endpoint Code (EC) which you received from the TSO (see chapter 4.2)
- Next, fill in the name of the Market Actor (MA) (see chapter 4.7) – not the System Operator (SO). If MA is a group of several "sister companies", then combine them into one name. Example: The sister companies Power AS, Power AB and Power Oy is to be written as "Power AS/AB/Oy" or something to that effect.
- For contact email in NEM PROD, use a **monitored**[3] **email address** of the SO. A personal email-address will in general not be approved, because this email-address will be used to send information about upgrades and issues. In NEM TEST personal email address is allowed.
- Phone number is not as important, it's a secondary option if email contact fails.
- Environment/Project-fields are new (not shown in the screenshot below). Their main purpose of the fields is to be printed in the GUI menu/tab so that you have a way to separate the various GUIs. See last screenshot below.


ecp
Endpoint
admin
Import Certificates
Connect to Component Directory
Send Registration Request
Step 1/3: Import Certificates
Certificates File
Drop file here or load from disk.
Certificates File password *
Password
Continue

Register – password is usually "password" unless your TSO has decided otherwise.

[3] Monitored means that some organization will be responsible for processing the email at least daily.


ecp
Endpoint
Import Certificates → Connect to Component Directory → Send Registration Request

Connectivity test was successful.

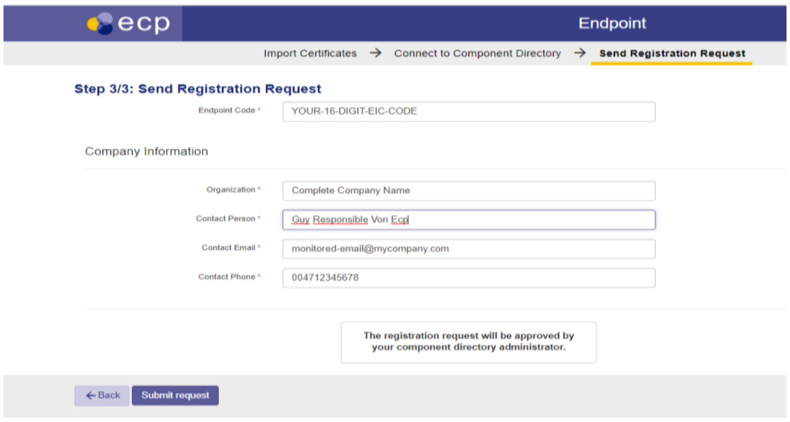
Step 2/3: Connect to Component Directory

Component Directory URL *: https://ecp4.statnett.no/ECP_MODULE

Component Directory Code *: 50V000000000111W

Check connectivity
← Back
Continue

Enter the CD Code and CD URL – not your own Endpoint Code (EC)!


ecp
Endpoint
Import Certificates → Connect to Component Directory → Send Registration Request

Step 3/3: Send Registration Request

Endpoint Code *: YOUR-16-DIGIT-EIC-CODE

Company Information

Organization *: Complete Company Name

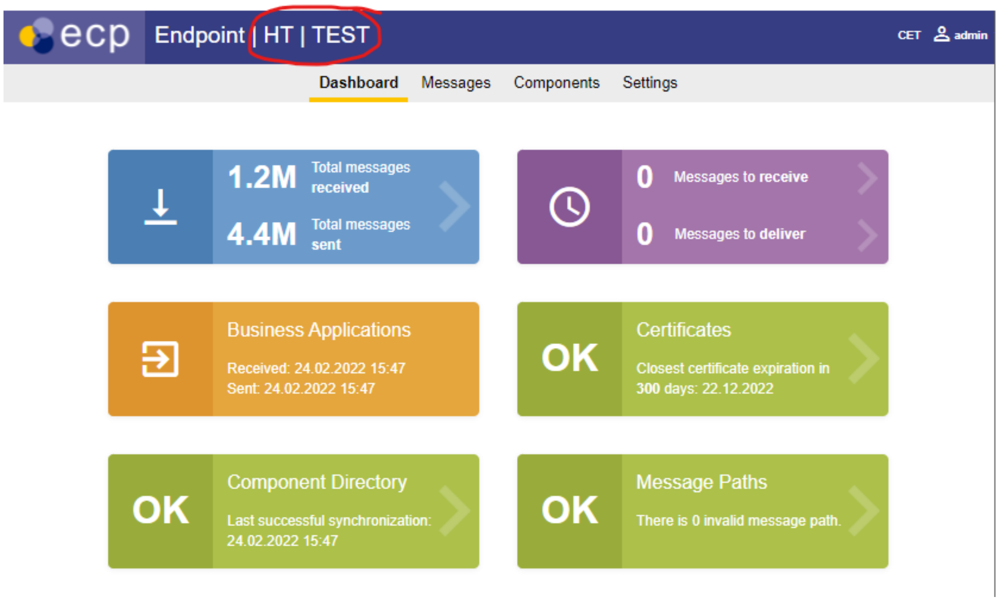
Contact Person *: Guy Responsible Von Ecp

Contact Email *: monitored-email@mycompany.com

Contact Phone *: 004712345678

The registration request will be approved by your component directory administrator.

Here you add your own Endpoint Code (EC), the Organization of the Market Actor (MA), and the contact information of the System Operator (SO). This screenshot is missing a couple of fields (Project/Environment), see effect of these fields in next screenshot. You can change Project/Environment-settings later on the Settings-page.



1.2M Total messages received

4.4M Total messages sent

0 Messages to receive

0 Messages to deliver

Business Applications

Received: 24.02.2022 15:47

Sent: 24.02.2022 15:47

OK Certificates

Closest certificate expiration in 300 days: 22.12.2022

OK Component Directory

Last successful synchronization: 24.02.2022 15:47

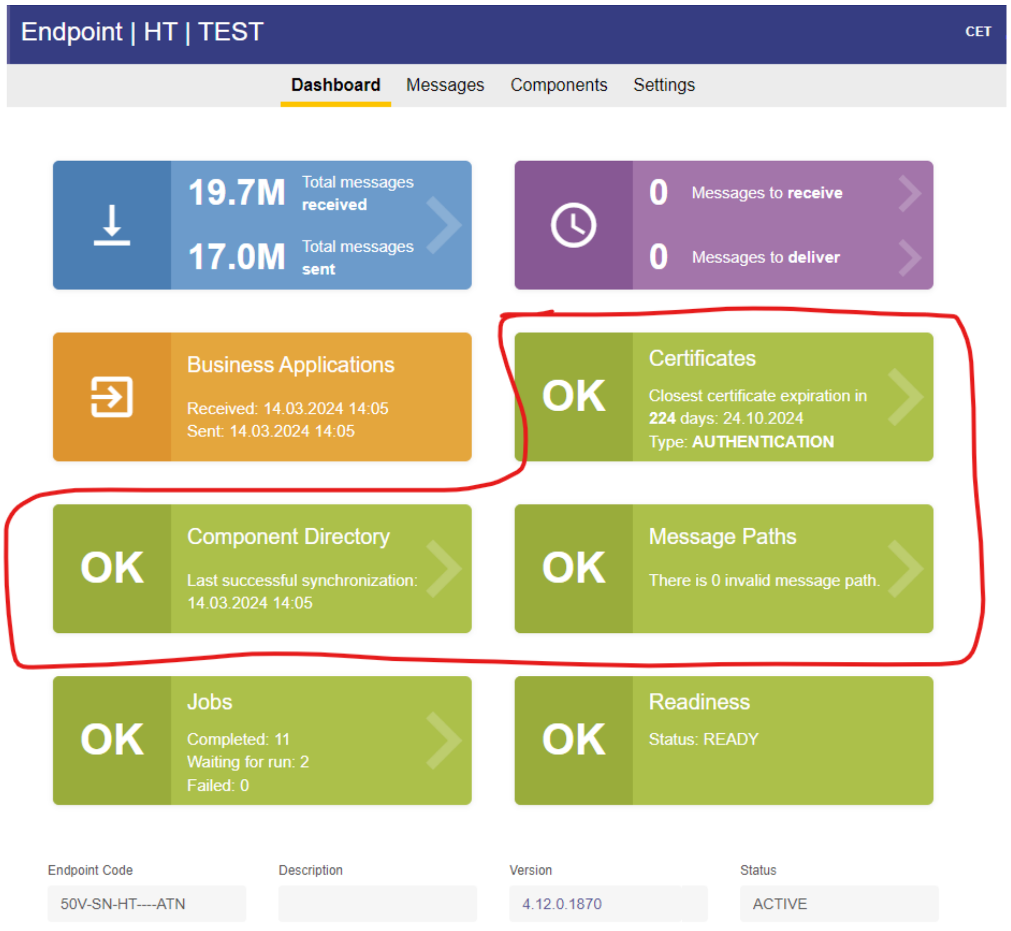
OK Message Paths

There is 0 invalid message path.

See HT|TEST in the top menu, this denotes the fields "Project" and "Environment" mentioned earlier.

9.4.2 Send email to TSO and wait for approval

At this point, you must wait until the CD administrator approves your registration request. **You must send an email to ecp@statnett.no, ecp@svk.se, ecp@energinet.dk or ecp.support@fingrid.fi to notify of the registration request, otherwise no one will detect that a request has been sent**. In the ECP Dashboard you can monitor whether your endpoint has been approved or not – the "Component Directory" and "Certificates" tile should be green when this happens. The dashboard will look like this after a couple of minutes after the approval by the TSO.



19.7M Total messages **received**
17.0M Total messages **sent**

0 Messages to **receive**
0 Messages to **deliver**

Business Applications
Received: 14.03.2024 14:05
Sent: 14.03.2024 14:05

OK Certificates
Closest certificate expiration in **224** days: 24.10.2024
Type: **AUTHENTICATION**

OK Component Directory
Last successful synchronization: 14.03.2024 14:05

OK Message Paths
There is 0 invalid message path.

OK Jobs
Completed: 11
Waiting for run: 2
Failed: 0

OK Readiness
Status: READY

Endpoint Code	Description	Version	Status
50V-SN-HT----ATN		4.12.0.1870	ACTIVE

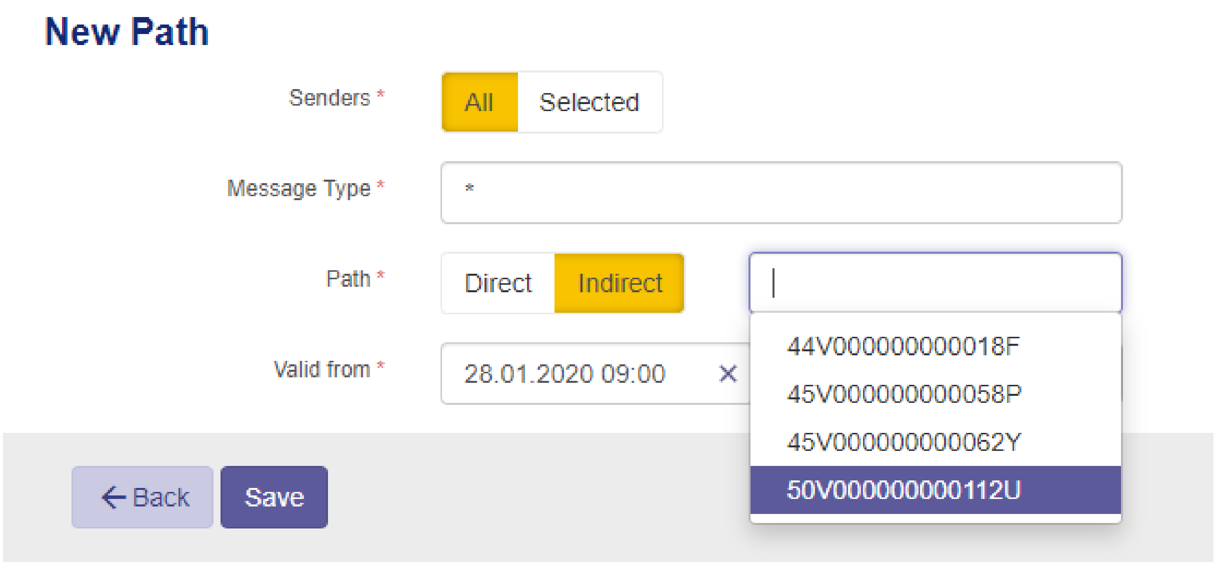
Once the request is approved your ECP-endpoint is connected to the CD and can exchange information about the network.

9.5 Message Path

You must define a Message Path to tell how message are supposed to be routed **to** your endpoint. If you forget this step you will **not receive any messages** and the dashboard will show a red tile.

- Choose Settings tab in the ECP Dashboard
- Choose button "+ New Path"

- Set Message Type to "*", Path to "Indirect". In the drop-down, choose the broker which starts with 44V(Fingrid), 45V(Energinet), 46V(SvK) or 50V(Statnett) depending on which TSO you're registered with.
- Set "Valid from" back in time since the endpoint is running UTC-time as default. There is no danger in setting yesterday as "valid from".
- Do not set "Valid to"-field – the message path should be valid forever
- Press "Save" – your message path will be known to everyone in the national ECP-network within 2 minutes and within 12 minutes in the cross-Nordic ECP-network.


New Path

Senders *: [✓ All] Selected

Message Type *: *

Path *: Direct [✓ Indirect]

Valid from *: 28.01.2020 09:00

44V000000000018F
45V000000000058P
45V000000000062Y
[✓ 50V000000000112U]

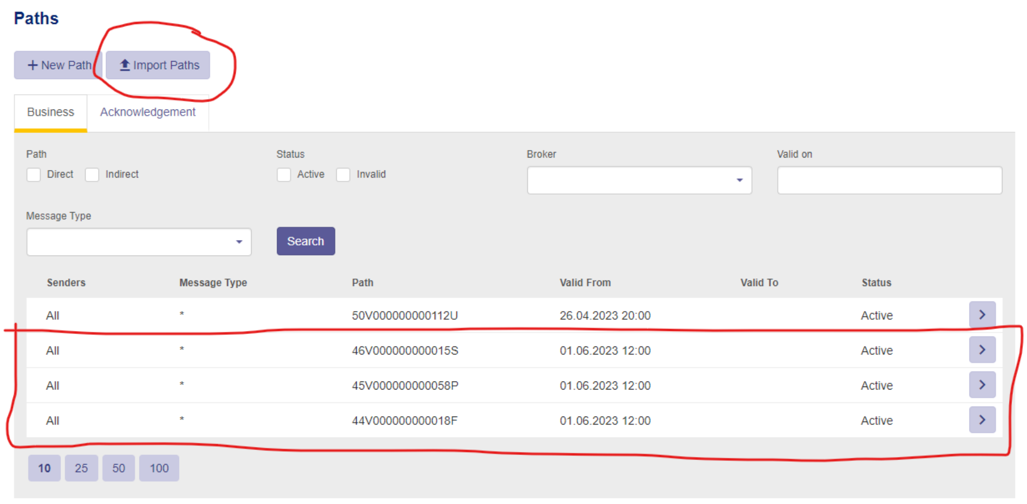
9.5.1 Failover Message Path is now mandatory

To achieve high uptime on the communication to the ECP-network, add additional (AKA failover) message paths to your endpoint. The effect is that if one central broker is down (e.g., if Statnett's broker is down), your endpoint will be able to retrieve messages via other brokers. The other TSOs brokers will be used for failover, thus they're in constant use. Therefore, one can expect that the failover paths will actually work once they are needed. By following the procedure below, you will add 3 failover brokers (for each of the other TSOs in the Nordics). BUT! This will not work unless you open the firewall to allow traffic to these brokers. See chapter 4.5.

Go to one of these pages, depending on which TSO CD your endpoint is connected to:

https://ediel.org/nordic-ecp-edx-group-nex/energinet/
https://ediel.org/nordic-ecp-edx-group-nex/fingrid/
https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/
https://ediel.org/nordic-ecp-edx-group-nex/svenska-kraftnat/

Based on whether your endpoint is connected to NEM-TEST/PREPROD or NEM-PROD, then download the appropriate MessagePath-file (TEST-MP or PROD-MP). This is a JSON-file, but suffix is CSV. Next Go the ECP Settings page and click on "Import Paths" (next to "+ New Path"). The choose the file you've downloaded. The result should be something similar to this (at least in TEST/PREPROD):

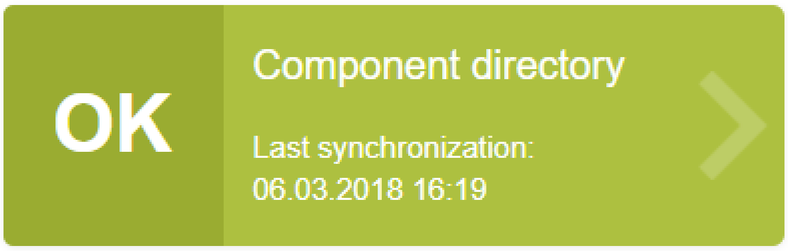


Senders	Message Type	Path	Valid From	Valid To	Status
All	*	50V000000000112U	26.04.2023 20:00		Active
All	*	46V000000000015S	01.06.2023 12:00		Active
All	*	45V000000000058P	01.06.2023 12:00		Active
All	*	44V000000000018F	01.06.2023 12:00		Active

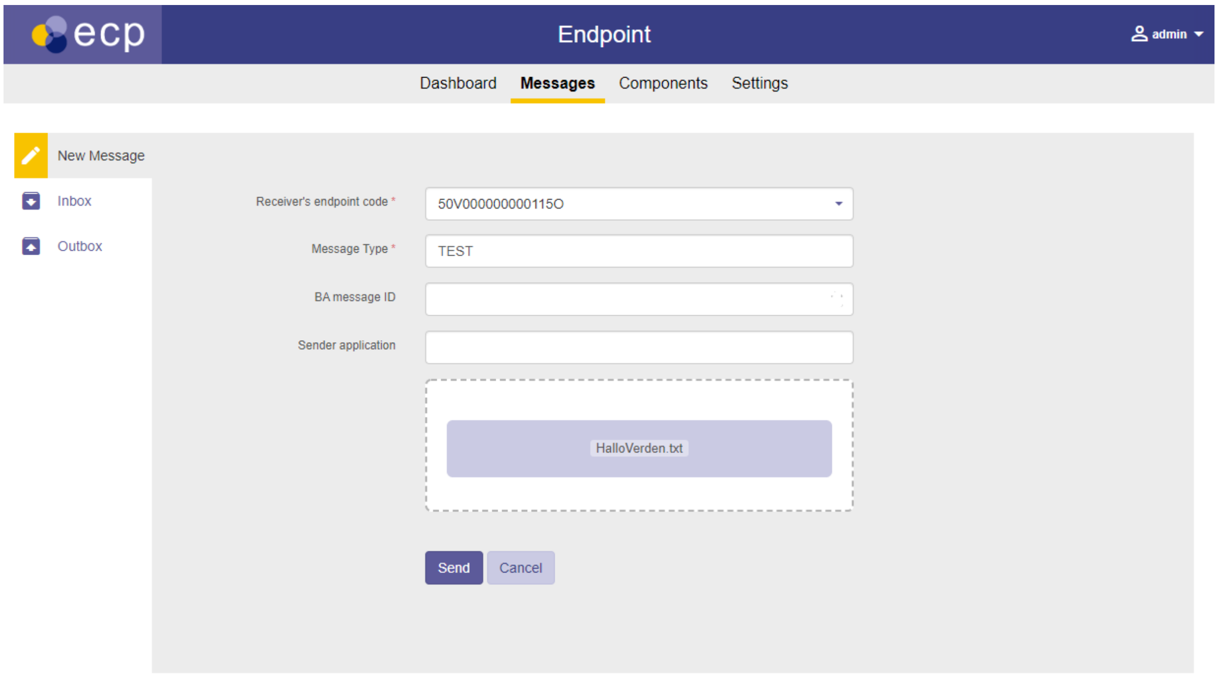
The message paths highlighted are the extra message paths you've just added. The list of brokers in the "Path" column should all be different ECP-codes.

9.6 Verify the installation

- Open ECP-Dashboard
- Check that the box indicating synchronization is green



- Make a "New Message" and send to a TSO-endpoint. You can find which endpoint belongs to which organization on the Components-page. The MESSAGE TYPE should be set to "TEST" and the file you send should be a small text file. Press the Send-button, wait a few seconds and then refresh the page. If the message (check Outbox) gets the status "Received" the test is successful. See screenshots below:



Endpoint

admin

Dashboard **Messages** Components Settings

New Message

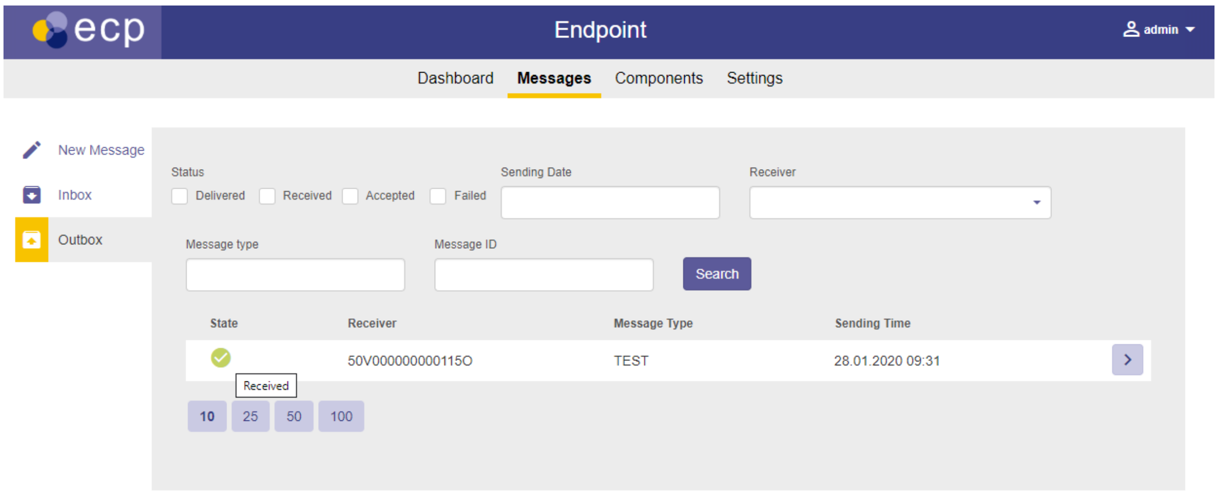
Inbox

Outbox

Receiver's endpoint code *	50V000000000115O
Message Type *	TEST
BA message ID	
Sender application	

HalloVerden.txt

Send Cancel



Endpoint

admin

Dashboard **Messages** Components Settings

New Message

Inbox

Outbox

Status: Delivered, Received, Accepted, Failed

Sending Date

Receiver

Message type

Message ID

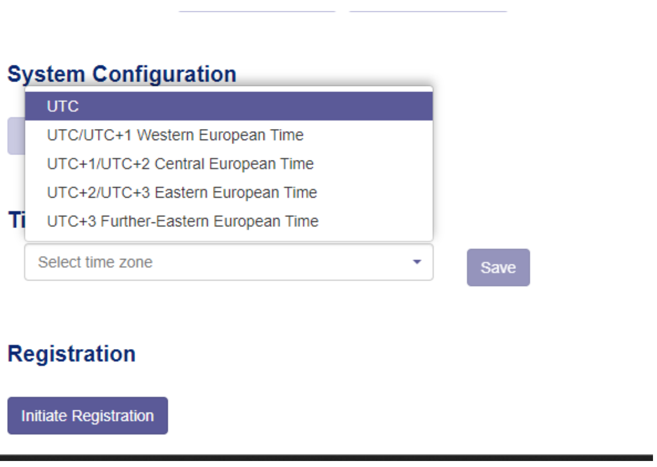
Search

State	Receiver	Message Type	Sending Time
✓ Received	50V000000000115O	TEST	28.01.2020 09:31

10 25 50 100

9.7 Change timezone of the endpoint

Change the timezone of the endpoint on the Settings page, so the Dashboard will reflect correct timestamps. The file logs will continue to be in UTC-time.



System Configuration

- UTC
- UTC/UTC+1 Western European Time
- UTC+1/UTC+2 Central European Time
- UTC+2/UTC+3 Eastern European Time
- UTC+3 Further-Eastern European Time

Select time zone

Save

Registration

Initiate Registration

9.8 Troubleshooting

9.8.1 ECP-Endpoint does not respond on expected port

- Check catalina-logs – it should log "BindException: Address already in use". To find which process this is, run (in console as admin) "netstat -a -n". You can then identify the PID of the process LISTENING on port 8443 and find that process in Task Manager. If you want to, you can change the ports in tomcat\conf\server.xml and restart the service to restart ECP.

9.8.2 Message status "Failed"

- The broker or the receiving endpoint might not have gotten the information about your endpoint's certificate – thus rejecting the message. The problem is then related to synchronization with the CD, either between your own endpoint and the CD, or between the receiving endpoint/broker and the CD. Synchronization of new certificates should take at maximum 12 minutes. If you are certain that your own endpoint is synchronized (check this status on the Settings page – there is a "connectivity check" for your CD), then you must contact the CD administrator.

9.8.3 Message status "Accepted"

- Possibly, your firewall does not allow outgoing traffic to the Central Broker (see "Big Picture" in chapter 3)

9.8.4 Component Directory is not synchronized

- You're not approved yet – wait a little or remind the administrator of the ECP-network
- Your firewall does not allow outgoing traffic to the CD (see "Big Picture" in chapter 3)

9.8.5 You're able to send, but do not receive any messages

- Your Message Path is not defined in "Settings" (see chapter 9.5)

10 Installation or Upgrade of EDX-toolbox on Windows

10.1 Java

In all likelihood you've already installed Java on this host, if you follow the advice of this guide. If you insist on installing EDX on a separate host, then you must of course install Java again. Please follow the advice given for Java-installation on ECP (se chapter 5.1).

10.2 Installation – skip if not applicable

Execute EDX IG chapter 6.2.

10.3 Upgrade – skip if not applicable

Execute EDX UG chapter 5.1

10.4 Service properties (ports used)

Following this guide, the EDX is installed on the same OS as ECP. Please go through the following chapters to make sure that there are no port conflicts (there should not be any, if the installation script worked as it should): EDX IG, chapter 6.7.2-6.7.3 (or possibly 6.7.4 depending on how EDX-toolbox was installed).

11 Installation or Upgrade of EDX-toolbox on Linux

11.1 Installation

If you do a first-time installation then execute EDX Installation Guide (IG) chapter 5.1 (Java installation).

Then execute EDX IG chapter 6.4 + 6.7.1.

11.2 Upgrade

Execute the EDX Upgrade Guide (UG) chapter 5.2

12 Installation or Upgrade of EDX-toolbox using Docker image

Docker support is limited, and it is expected that users have more understanding and experience than those that install on Linux or Windows. Still, there are a few resources that may provide the necessary support to get you through this:

- This installation guide you're reading. It intends to tie together the other resources and also provide some extra information where it seems to be lacking
- For the images themselves, please see chapter 4.1.
- The standard EDX Download package examples of setup in the EDX Other Deliverables folder
  - edx-docker-test-env.zip (a "bare-bone" setup – see chapter12.1.1)
  - edx-endpoint-kubernetes.zip (useful for "cloud" setup – see chapter12.1.2)
- The EDX Installation Guide chapter 11.

This chapter is not focused on the whole configuration of EDX, but rather on the specific parts related to Docker. Therefore, in order to configure EDX, please read the relevant section for Linux (see chapter 6).

12.1 Installation

12.1.1 Bare-bone Docker Host setup

You may use the docker-compose.yml in the zip-files provided in the download. Such a setup is convenient for testing/development.

12.1.2 Cloud setup

The log folder and the activemq temporary data folder should not be mounted on “slow” network attached storage. This may cause log events to be missed or delays in the AMQP message flows. If you see problems of this kind, please consider faster or "more local" storage.

12.1.2.1 Database

In general, it is recommended to use an external database when deploying containers. Storing database files on the container host itself is considered bad practice, because it locks in the container to that specific host.

If the storage used for the local disk in the container is physically located elsewhere than on the container host itself (e.g., probably in any cloud environment), please configure the endpoint to use an external database, instead of the default embedded Derby database. Running a Derby database with its data files stored on network attached storage has proven to result in all sorts of weird issues with the endpoint.

Read IG chapter 10 (External Databases) for details on configuration for use of an external database.

12.1.2.2 EDX DMS cache

Make sure that the EDX DMS cache storage is placed on persistent storage.

If located on non-persistent storage, there is a risk of the EDX-toolbox getting into a bad state after a restart, in case of the ECP-endpoint being down while the EDX-toolbox is stopping. Getting the EDX-toolbox to run properly again, requires fiddling with entries in the database.

It is recommended to use high-performance storage (e.g., backed by SSD drives) – if the storage is too slow, message throughput will be impacted.

12.2 Upgrade

The official documentation from Unicorn is not comprehensive when it comes to Docker Installation. However, it should suffice to follow the advice for Linux (see previous chapter) to the best of your abilities. At least the part about configuration change will apply also for Docker. Also read EDX Upgrade Guide (UG) chapter 5 and ECP Release Notes to get information on what has changed.

Extract the config files from the new version of the docker image and compare these to the config files in the current container. Be sure to also check for changed or new Java VM variables and update accordingly.

13 EDX Setup (for all OS installations)

13.1 Configuration of edx.properties

The edx.properties configuration file comes with many configuration parameters (all of them are commented briefly in the edx.properties file itself). Most of the configuration parameters use default values, but the following parameters must be configured for each installation:

Parameter	**Description**
edx.toolbox.code=<Endpoint-Code>	ECP-endpoint code assigned to this Toolbox. This is the same Endpoint Code (EC) you received in chapter 4.2 and which you specified in the last step of chapter 9.4.1. Remove the angle brackets.
edx.serviceCatalogue.code=<ServiceCatalogue-Code>	Choose the TSO which you're connected to and enter the code of the ServiceCatalogue: NEM Test-network: • Statnett: **50V000000000113S** • Fingrid: **44V000000000023M** • Energinet: **45V000000000059N** • SvK: **46V000000000016Q** NEM Production-network: • Statnett: **50V000000000120V** • Fingrid**: 44V000000000024K** • Energinet**: 45V000000000055V** • SvK**: 46V000000000021X**
spring.profiles.active= edx-nonha	With the recommended setting you require authenticated access to dashboard and webservices. For other options, see EDX User Guide.
edx.toolbox.deleting.dms.deleteOlderThan=72	The default value of this parameter is 168 which says that a copy of all messages going through EDX is stored in 168h. This might require some extra disk space if you have many messages passing through. If you shorten this time period you'll need a little bit less disk space, but at the same time you risk failed delivery if some problem in EDX (or a lack of ACK) lasts for more than this time period.

edx.csrf.secret=password
 	
endpoints.prometheus.sensitive=false
management.endpoints.web.exposure.include=info,health,readiness,prometheus
	This a host of properties related to monitoring. You can read about them in ECP AG. With this setup you will get all possible monitoring turned on, and you can access the following URLs:
	/actuator/prometheus
	/actuator/info
	/actuator/health
	/actuator/readiness

edx.amqp.client.prefetch=10	Change prefetch limit from 1000 to 10. Prefetch will cause messages to stick to a consumer, and if that consumer gets "stuck", the messages can be lost.

edx.toolbox.ecp4.redeliveryAttempts=3
edx.toolbox.ecp4.redeliveryDelay=5000
	There are certain situations in EDX where redelivery of a message will be attempted. The default redelivery is 100 * 10000 ms = 1000s, which we lower to 15s in this case, to avoid a situation where the EDX is stuck doing nothing for 1000s.

NB! Do not forget to add the properties listed for edx.properties in chapter 8.2.1.

13.2 Configuration of edx.yml

The configuration in the edx.yml deals with the various interfaces available to access EDX from the BA. Default settings (allow Web Service interface) is ok to begin with. You can use the default edx.yml configuration, move on to next chapter, and later change the edx.yml to suit your needs. The configuration is explained in chapter 13.6.

NB! Make sure to have only **one** yml-file in the config-directory – since EDX will read all files with yml-suffix.

13.3 Configuration of edx-users.properties

To enable authentication, set the spring.profiles.active-parameter as explained in the edx.properties configuration above.

NB! Only the application and the root-administatrator should have read-access to this file. The same goes for write-access. The file defines the user, role and passwords. Here is a simple example of the two types of users available:

```
edx.toolbox.users[0].login=admin
edx.toolbox.users[0].password=supersecret
edx.toolbox.users[0].role=serviceManager

edx.toolbox.users[1].login=user
edx.toolbox.users[1].password=secret
edx.toolbox.users[1].role=user
```

To add more users, simply add new lines and increase the index.

13.4 Starting and stopping EDX-toolbox

For Windows use Services to start/stop or use the command-window with the appropriate privileges to run "**SC start/stop edx-toolbox**". For Linux use the command "**systemctl start/stop edx-toolbox.service**". You may start the toolbox now.

13.5 Installation verification

The application should be installed in the installation path folder. This installation folder should contain configuration files, tomcat folder and uninstaller. After the application is successfully started, it creates additional folders for data and log files. You could browse through edx-toolbox.log or edx.log to see if any ERROR-entries occur – there should be none. Catalina-logs will tell you if there is a port conflict (if so go back where you set up the ports for the toolbox). The status of the service, if installed, can be checked either via command line: "**SC query edx-toolbox"** or in the Windows Services tool.

13.5.1 EDX GUI verification

Check the EDX-Dashboard-URL:

https://localhost:9443/ (change "localhost" to the IP-address of the host if needed)

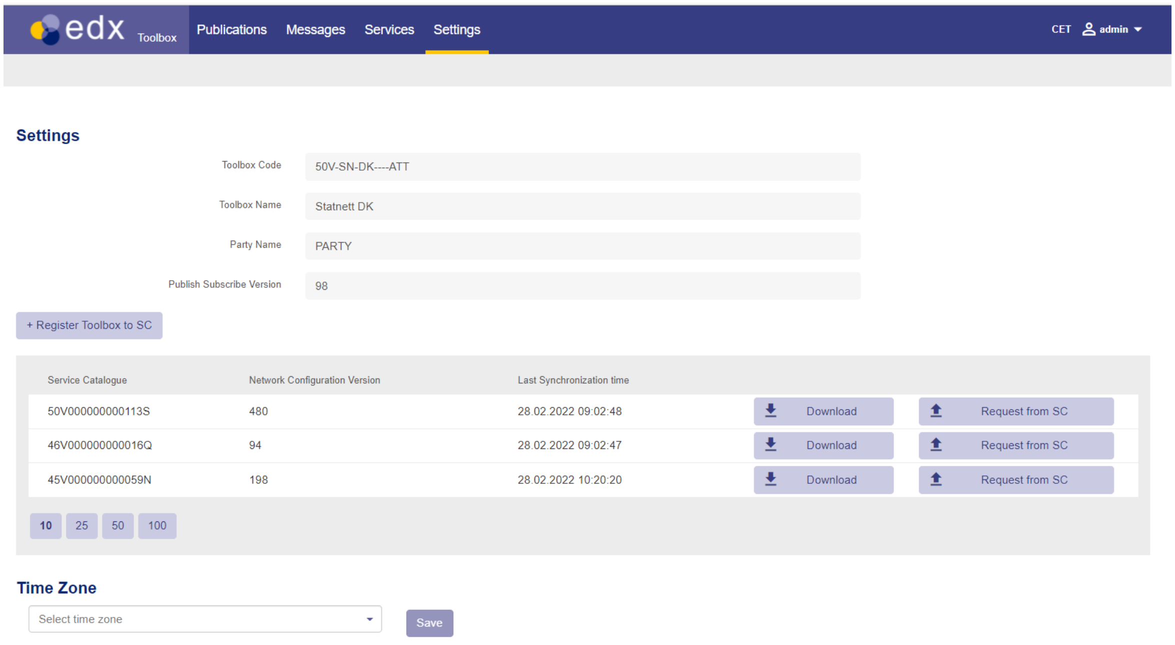
Most browsers will not show this page without giving a security warning. This is because of the usage of self-signed certificate and the fact the hostname of the certificate usually does not match the URL. To properly fix this you must create a TLS-certificate for this host and change some settings in server.xml in Tomcat. The details of TLS-certificate creation are not explained in this document, one needs to check with other resources.

In case login is required (due to settings in edx.properties and edx-user.properties), the **default login is admin/password**. See previous chapters for configuration.

13.5.2 Service Catalogue verification – this is the perfect verification of ECP/EDX!

Open the Settings page to check if your copy of the Service Catalogue has arrived. Look at screenshot below, which shows several SCs – the minimum requirement is that you have received the SC that you've specified as in edx.properties. The timestamp should be recent! If it hasn't arrived you can first check your ECP-endpoint to see that a message called EDX-INTERNAL-CONFIGURATION-REQUEST was sent (see Outbox of your ECP-endpoint) shortly after you've started the EDX Toolbox. If the TSO has added your toolbox to the Service Catalogue, then you will see in ECP Inbox that you receive an EDX-INTERNAL-CONFIGURATION-MESSAGE message. If you do not receive such a file, check that you have a proper Message Path in your ECP (chapter 9.5). If Message Path is ok, then notify the TSO – they might have forgotten to add you to the Service Catalogue, since it is a manual process to add the toolbox to the Service Catalogue.

Finally, when the file arrives in the Inbox of ECP, the file should then be consumed by EDX Toolbox, but you will not see this file in the Messages-GUI of EDX. The file/SC-copy will be shown in the Settings-page of EDX. The screenshot below shows an EDX Toolbox which is added in multiple Service Catalogues, but you need at least one! If your ECP received the EDX-CONFIGURATION file, but the Settings page does not show the Service Catalogue, please check EDX logs.



Settings

Toolbox Code	50V-SN-DK----ATT
Toolbox Name	Statnett DK
Party Name	PARTY
Publish Subscribe Version	98

+ Register Toolbox to SC

Service Catalogue	Network Configuration Version	Last Synchronization time		
50V000000000113S	480	28.02.2022 09:02:48	Download	Request from SC
46V000000000016Q	94	28.02.2022 09:02:47	Download	Request from SC
45V000000000059N	198	28.02.2022 10:20:20	Download	Request from SC

10 25 50 100

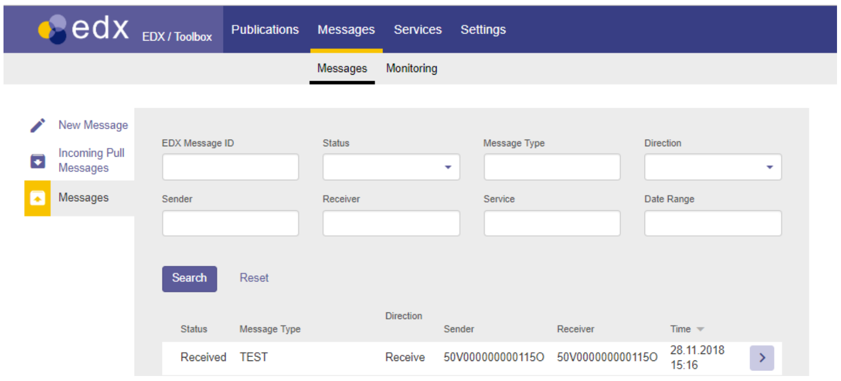
Time Zone

Select time zone

Save

13.5.3 Send test messages

Open a web browser and navigate to the dashboard URL of your toolbox. You should see this, except you won't have any messages in the list:

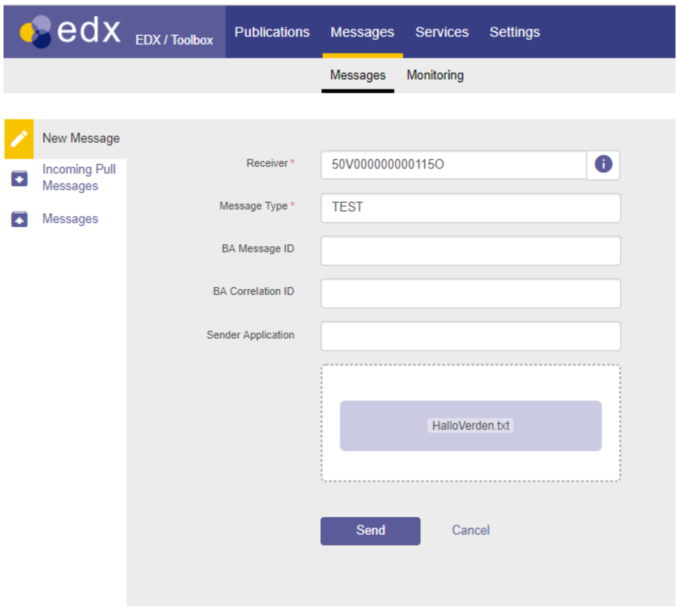


Status	Message Type	Direction	Sender	Receiver	Time
Received	TEST	Receive	50V000000000115O	50V000000000115O	28.11.2018 15:16

Make a "New Message" and send to a TSO-endpoint:

- NEM Test-network:
  - Statnett: **50V000000000115O**
  - Fingrid: **44V000000000019D**
  - Energinet: **45V0000000000601**
  - SvK: **46V000000000017O**
- NEM Production-network:
  - Statnett: **50V000000000188Y**
  - Fingrid: **44V000000000010V**
  - Energinet: **45V000000000056T**
  - SvK: **46V000000000021X**

The MESSAGE TYPE should be set to "TEST" and the file you send should be a small text file. Press the Send-button, wait a few seconds and then refresh the page. If the message get the status "Received" the test is successful. See screenshot below:



Receiver *	50V000000000115O
Message Type *	TEST
BA Message ID	
BA Correlation ID	
Sender Application	
	HalloVerden.txt

13.6 Configuring the edx.yml file (see EDX User Guide chp 5/6 for more information)

NB! Make sure to have only **one** yml-file in the config-directory – since EDX will read all yml-files.

This is an example of a relatively complete edx.yml, carefully crafted to show a number of features. It should hopefully provide enough examples to help you configure your own edx.yml. The file will be explained in detail below. Make sure not to introduce any tabs in this file, only spaces are allowed. Also, be very careful about the number of spaces used for indentation – otherwise it will not be parsed correctly. If you accidently miss a comma, EDX might not warn you about it. Some line breaks are introduced in the example below for readability of very long lines; remove them! Make sure to read the edx.log and catalina.log carefully after startup of EDX, it should show if the file was parsed as expected. Especially – look for lines with "RoutesConfig" in edx.log – there should be one such entry at startup for every route you've specified. Test your edx.yml in an online YAML-parser (http://www.yamllint.com/), but beware of sharing password/secrets.

```
integrationChannels:
  amqpEndpoints:
   - {direction: in,  code: amqp-ba1-outbox, queueName: edx.endpoint.outbox.ba1,  redeliveryAttempts: 1, replyQueueName: edx.endpoint.reply.ba1}
   - {direction: out, code: amqp-ba1-inbox,  queueName: edx.endpoint.inbox.ba1,   redeliveryAttempts: 1}
  fssfEndpoints:
   - {direction: in,  code: fssf-ba2-outbox, directory: /ba2/outbox, redeliveryAttempts: 1, replyDirectory: /ba2/reply}
   - {direction: out, code: fssf-ba2-inbox,  directory: /ba2/inbox,  redeliveryAttempts: 1}
   - {direction: out, code: edx-default,     directory: /edx-default, redeliveryAttempts: 1}
   - {direction: out, code: edx-errors,      directory: /edx-errors, redeliveryAttempts: 1}
  ftpEndpoints:
   - {direction: in,  code: sftp-ba3-outbox, directory: ba3/outbox,  redeliveryAttempts: 1, replyDirectory: ba3/reply, protocol: sftp, hostname: sftp.host.org,
     port: 22, username: user, password: pass, connectionParams: {stepwise: true, separator: UNIX, knownHostsFile: /home/edx-toolbox/.ssh/known_hosts }}
   - {direction: out, code: sftp-ba3-inbox,  directory: ba3/inbox,   redeliveryAttempts: 1, tempPrefix: ../tmp/,       protocol: sftp, hostname: sftp.host.org,
     port: 22, username: user, password: pass, connectionParams: {stepwise: true, separator: UNIX, knownHostsFile: /home/edx-toolbox/.ssh/known_hosts }}
  kafkaEndpoints:
   - {direction: out, code: kafka-publish,   topicName: publish,     redeliveryAttempts: 1, connectionURI: "k1.host.org:9092,k2.host.org:9092",
     partitionKeyMadesHeaders: [businessType, sender], options: "compressionCodec=gzip&maxRequestSize=31457280&
     valueSerializer=org.apache.kafka.common.serialization.BytesSerializer"}
   - {direction: in,  code: kafka-inbox, topicName: inbox, replyTopicName: inbox_reply, connectionURI: " k1.host.org:9092,k2.host.org:9092", options:
"groupId=edxGroup&sslTruststoreLocation=keystore.jks&sslTruststorePassword=password&sslKeystoreLocation=keystore.jks&sslKeystorePassword=password&sslKeystoreType=JKS
&sslTruststoreType=JKS&securityProtocol=SSL"}

components:
  validations: []
  transformations: []
  externalProcessing: []
routing:
  routes:
   - {code: R-sftp-ba3, start: toolbox-gateway, end: sftp-ba3-inbox,                    messageType: EXT-EI-MAGASINDATA }
   - {code: R-amqp-ba1, start: toolbox-gateway, end: amqp-ba1-inbox,                    service: {serviceCode: FASIT,   domainCode: DEFAULT_DOMAIN,
serviceCatalogueCode: 50V000000000113S } }
   - {code: R-fssf-ba2, start: toolbox-gateway, end: fssf-ba2-inbox,                    service: {serviceCode: MMS,     domainCode: DEFAULT_DOMAIN,
serviceCatalogueCode: 50V000000000113S } }
   - {code: R-ba1-ba2,  start: toolbox-gateway, end: [amqp-ba1-inbox, fssf-ba2-inbox],  service: {serviceCode: DK-MNA,  domainCode: DEFAULT_DOMAIN,
serviceCatalogueCode: 50V000000000113S } }
   - {code: R-kafka-ba4,start: toolbox-gateway, end: kafka-publish,                     service: {serviceCode: NO-NUCS, domainCode: DEFAULT_DOMAIN,
serviceCatalogueCode: 50V000000000113S } }

  sendProcessDefaultRoute:    {start: "*",             end: toolbox-gateway, fail: ecp-endpoint, steps: [] }
  receiveProcessDefaultRoute: {start: toolbox-gateway, end: edx-default,     fail: edx-errors,   steps: [] }
```

There are three sections in this file: **integrationChannels**, **components** and **routing:**

13.6.1 Components

We are not interested in **components** – this section has no configuration, [ ] simply means an empty array. The lack of interest in **components** configuration is deliberate: We don't want to introduce validations and transformations in the EDX, even though it works quite nice. The point is that from the moment EDX takes on the responsibility of validating and transforming the messages, it becomes more than a simple messenger – it becomes part of the business logic and fault handling. It is a NEX recommendation to deliver the message unaltered from one BA to another. This will ensure less trouble in the transport-layer and more flexibility for the BA. The cost is that each Business Application must handle validation/transformation for themselves.

13.6.2 IntegrationChannels

IntegrationChannels define a set of "endpoints" which specifies where BA and EDX can place or pick up a message. These "endpoints" are not the same kind explained in chapter 2, so please do not confuse them. There are five types of channel endpoints:

- AMQP (Advanced Message Queue Protocol)
- FSSF (File System Shared Folders)
- FTP (File Transfer Protocol)
- Kafka (Statnett use this for internal publish/subscribe)
- WS (Web Service) – the default endpoint, not specifically configured

Each channel endpoint is placed in its own section. The channel endpoint must specify a **direction** and a **code**. The **direction** can be

- "in": Location where BA place a message and EDX picks it up and delivers it to the receiver
- "out": Location where EDX place a message coming from a sender and BA then picks it up

The **code** must be a **unique** identifier of this channel endpoint. The codes in "out"-endpoints will be used in routes, because routes define where to place messages when they're received from the network (other endpoints).

Further important notes:

- You may create as many channel endpoints as you wish
- If you don't want any channel endpoints, simply type "[]" after the colon. (ex: sftpEndpoints: [])
- We're using the naming convention inbox/outbox as seen from the BA's point of view, which may cause some confusion with the **direction** (seen from EDX' point of view)
- Default redelivery-attempts is 10 in EDX, but we override this in our example to 1. The reason is that redelivery seldom solves any issue, it just creates a lot of "noise" in the logs and it lowers the message throughput.
- You should specify one in-endpoint for each BA. The reason is that you will have access to a reply-endpoint for each BA. The reply-message (=technical ACK) can tell if the message has safely arrived to the receiver's EDX. It can also tell you if you've used a non-existent EDX address (receiverCode). However, the only way to know if the receiving BA has picked up the message from the receiving EDX is to listen for a regular message from the other BA with "business acknowledgement".
- The BA or some other consumer must consume the reply-endpoint. If not, then the reply-queue will eventually fill up and the toolbox will stop working.
- When EDX receives a message from the network, it may fail to deliver it to the correct out-endpoint. The reason could be that you have specified validation or that the message is to big (can happen with Kafka) or something else. In those cases, the failed message will be placed on a shared folder defined at the very last line in the config: fail: edx-errors (which in turn points to the fssfEndpoint with code "edx-errors"). It will fall to the manager of the EDX to resolve such issues.

13.6.2.1 AMQP-endpoint

The configuration suggested in the example shows 2 channel endpoints (total 3 queues) for a BA named "ba1". One queue is for messages from ba1 to other recipients (outbox.ba1) with the corresponding reply-queue (reply.ba1) mentioned above. Another for messages to ba1 from other BAs in the network (inbox.ba1).

Queues are automatically created by EDX.

13.6.2.2 FSSF-endpoint

The configuration suggested in the example show 2 folders for a BA named "ba2", following the same pattern as for AMQP. A reply folder is also specified in the same manner as for AMQP.

You must create this folder yourself and assign read/write/execute-privileges to EDX-Toolbox process for these folders.

13.6.2.3 FTP-endpoint
The configuration suggested in the example shows the same setup as for AMQP and FSSF, now for BA "ba3". What happens here is that EDX has an FTP/SFTP-client which connects to an FTP/SFTP-server. The connectionParams are optional, but useful. The parameters sent directly the underlying apache-component and are documented here:

https://camel.apache.org/components/latest/file-component.html

https://camel.apache.org/components/latest/ftp-component.html

By setting the tempPrefix-attribute to "../tmp" you ensure that the file is not moved into the correct folder until it's completely written. The "../tmp"-folder must created (as ba3/tmp) and given proper privileges.

13.6.2.4 Kafka-endpoint
EDX can both consume and produce from a Kafka topic, both with and without SSL. We've provided both examples in the configuration.

Again, as for the FTP-endpoints, there are optional attributes which can be specified. These attributes are specified here:

https://camel.apache.org/components/latest/kafka-component.html

NEX recommends "compressionCodec" and "maxRequestSize", because Kafka is usually not suited for big messages (maxRequestSize is default 1MB). However, with compression, many text-messages can be compressed up till 90%. Specify the maxRequestSize so that all messages are attempted to be sent to Kafka and not rejected before EDX has tried to compress it.

13.6.3 Routes
The routes listed in the example show how each BA listens to a particular type of message, determined by the filter (service or messagetype) and the end-attribute specification.

- Ba1 listens to messages with EDX-service code = FASIT
- Ba2 listens to messages with EDX-service code = MMS
- Ba1 and Ba2 listens to messages with EDX-service code = DK-MNA
- The serviceCatalogueCode must be the same SC code as in edx.properties.
- The serviceCode contains a Country-prefix if it is a cross-border service (may be provided by a toolbox in another country than where your toolbox reside).
- Ba3 listens to messages with MessageType = EXT-EI-MAGASINDATA
- Ba4 listens to messages (through Kafka) with EDX-service code = NUCS

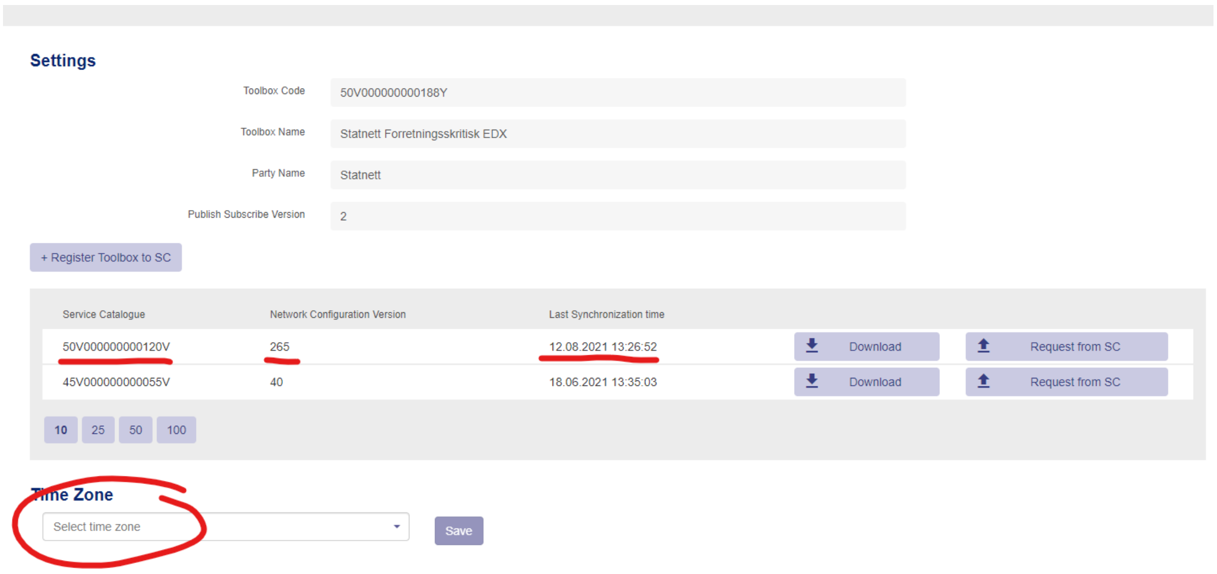
A few notes about this:

- We advise you to keep the routing as simple as possible.
- We advise you to avoid using MessageType in the routing. This is because the MessageType is specified by the BA and it's better for the routing to be independent of changes in the BA.
- EDX-Service is used here as something like a "system-to-system" channel. This is defined solely within EDX and makes it well suited to perform routing (the BA may change, but the routing stays the same).
- You may have multiple filters, both Service, MessageType and even Sender. EDX will use the routing rule which matches the message and is most specific. NEX advise against such rules, it will be hard to maintain.

At the end of the routing section you'll find the two default send/receive-routes. If the routes above do not match anything, the message will go to the default channel endpoint, which we set to "edx-default" which is a fileshare. If a problem occurs with the message routing (example: message to big to send to Kafka), then EDX will send the message to the error channel endpoint defined in the default-route. The error channel endpoint should be an FSSF-endpoint.

13.7 Change timezone of the toolbox

Change the timezone to CET on the Settings page, so the GUI will reflect correct timestamps. The file logs will continue to be in UTC-time.



Settings

Toolbox Code	50V000000000188Y
Toolbox Name	Statnett Forretningsskritisk EDX
Party Name	Statnett
Publish Subscribe Version	2

Service Catalogue	Network Configuration Version	Last Synchronization time		
50V000000000120V	265	12.08.2021 13:26:52	Download	Request from SC
45V000000000055V	40	18.06.2021 13:35:03	Download	Request from SC

13.8 Troubleshooting

13.8.1 You have not received Service Catalogue (AKA network configuration)

Check settings on your EDX Dashboard (see screenshot above). You should see that the network configuration from your TSO (see chapter 13.1), although other might also be present. If not, you cannot send/receive messages from EDX. The reason may be because you cannot receive files – see chapter 9.8.5. Another reason may be that your TSO has not updated the Service Catalogue with your endpoint (this is a manual process at the TSO). Please notify your TSO.

13.8.2 You cannot send/receive on a particular service

If you can send messages unrelated to a specific service (ex-address: the endpoint-code for Service Catalogue in chapter 13.1), but cannot send to the service you're supposed to be a part of (ex-address: SERVICE-FASIT), then the error may be that the Service Catalogue has not been updated properly (this is a manual process in the TSO). Please notify your TSO if you suspect this to be the case. Also, please check the Services->Consumed menu on the EDX Dashboard: A list of services should appear to show which services your endpoint may "consume".

14 Appendix

14.1 Connectivity Check

To properly monitor your endpoint send a "technical" test-message to a TSO-endpoint. This is because you likely will communicate with a TSO on a NEM network. You should send such a message every 5 minutes, but not more frequently. The result of this connectivity check should be picked up by your alarm/monitoring system.

14.1.1 How-to I

The simplest way to do this is by using ekit, a tool developed in Java by Statnett.

- Download the tool from here: https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/
- Run ekit on your endpoint-host (other options are possible)
- Execute command like the one below (change ECP-code, specify correct user/pw)

```
java -jar ekit.jar CC <ECP-code> https://user:pw@localhost:8443
```

The output will something like this:

```
{"statustime": "2024-09-26 09:59:23", "timestamp": 1727337563, "connectivity_check_status": "OK", "hostname": "A_HOST_NAME", "endpoint_code_checked": "ECP_CODE"}
```

To see other options for the CC tool of ekit, run:

```
java -jar ekit.jar CC
```

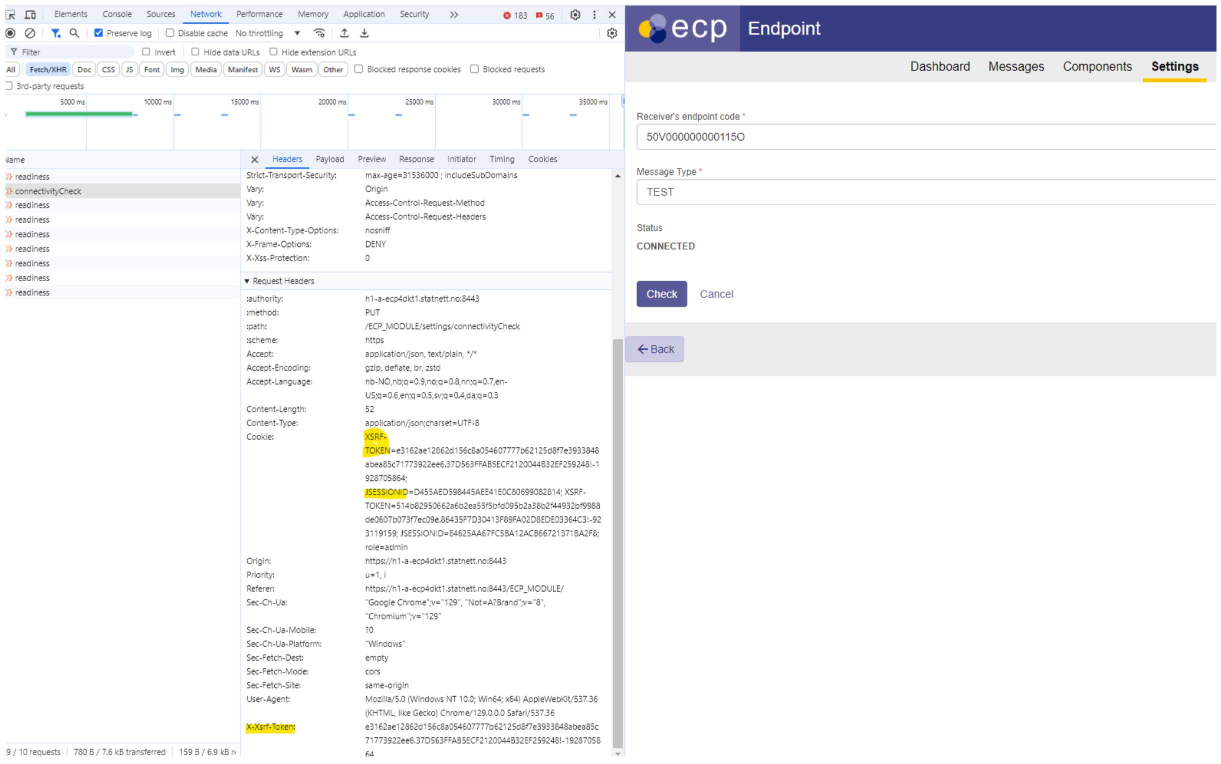
14.1.2 How-to II

Another approach is to make some script/program to access the HTTP-API yourself. The underlying API is this:

```
PUT http://<host>:8443/ECP_MODULE/settings/connectivityCheck
```

Set HTTP header for "Content-Type" to "application/json"
Set HTTP body to **{"receiver":"<ECP-code>","messageType":"TEST"}**

However - your endpoint most likely a require login! Therefore, you must login first, then retrieve xsrf-tokens and pass them back into the HTTP-URL shown above. You can study this by using a web browser and look at how the ECP GUI is using the HTTP-API:



```
Strict-Transport-Security:   max-age=31536000 ; includeSubDomains
Vary:                        Origin
Vary:                        Access-Control-Request-Method
Vary:                        Access-Control-Request-Headers
X-Content-Type-Options:      nosniff
X-Frame-Options:             DENY
X-Xss-Protection:            0

▼ Request Headers
:authority:                  h1-a-ecp4dkt1.statnett.no:8443
:method:                     PUT
:path:                       /ECP_MODULE/settings/connectivityCheck
:scheme:                     https
Accept:                      application/json, text/plain, */*
Accept-Encoding:             gzip, deflate, br, zstd
Accept-Language:             nb-NO,nb;q=0.9,no;q=0.8,nn;q=0.7,en-
                             US;q=0.6,en;q=0.5,sv;q=0.4,da;q=0.3
Content-Length:              52
Content-Type:                application/json;charset=UTF-8
Cookie:                      XSRF-
                             TOKEN=e3162ae12862d156c8a054607777b62125d8f7e3933848
                             abea85c71773922ee6.37D563FFAB5ECF2120044B32EF259248!-1
                             928705864;
                             JSESSIONID=D455AED598445AEE41E0C80699082814; XSRF-
                             TOKEN=514b82950662a6b2ea55f5bfd095b2a38b2f44932bf9988
                             de0607b073f7ec09e.86435F7D30413F89FA02D8EDE03364C3!-92
                             3119159; JSESSIONID=E4625AA67FC5BA12ACB66721371BA2F8;
                             role=admin
Origin:                      https://h1-a-ecp4dkt1.statnett.no:8443
Priority:                    u=1, i
Referer:                     https://h1-a-ecp4dkt1.statnett.no:8443/ECP_MODULE/
Sec-Ch-Ua:                   "Google Chrome";v="129", "Not=A?Brand";v="8",
                             "Chromium";v="129"
Sec-Ch-Ua-Mobile:            ?0
Sec-Ch-Ua-Platform:          "Windows"
Sec-Fetch-Dest:              empty
Sec-Fetch-Mode:              cors
Sec-Fetch-Site:              same-origin
User-Agent:                  Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
                             (KHTML, like Gecko) Chrome/129.0.0.0 Safari/537.36
X-Xsrf-Token:                e3162ae12862d156c8a054607777b62125d8f7e3933848abea85c
                             71773922ee6.37D563FFAB5ECF2120044B32EF259248!-19287058
                             64
```

Receiver's endpoint code *
50V000000000115O

Message Type *
TEST

Status
CONNECTED

14.2 Administration of ECP-endpoint

The ECP-server offers a Settings-page which has some important features:

- Message Path: Check that you have a path defined for message type "*" (chapter 9.5), otherwise you cannot receive any messages.
- Certificates: You can delete old certificates (Preferred = No, Valid to < Today). Old certificates may make noise in your logs.
- Message Connectivity: Connectivity check is the simplest way to test if another ECP-Endpoint is reachable. If no endpoint is reachable, the conclusion will usually be that your own endpoint has lost connection to the (central) Broker. In that case see chapter 4.5 for hostname and port number and try to telnet directly from the endpoint and from another location to determine whether the problem is on Statnett's end or your own.
- Component Directory: Connectivity check to see if the CD (over port 443) is available. Data is exchanged here every minute. If CD is offline for a long time (usually many hours or days) you will not be allowed to send messages any more.

The ECP-server offers a Dashboard which shows

- Component Directory synchronization is ok (or not)
- Certificates are valid (or not)
- Messages are delivered (or not)

14.3 Monitoring & more advanced troubleshooting

Instead of expanding the installation doc more, a "troubleshoot/management" document has been written and placed here: https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/

This document contains detailed information about the core concepts of ECP and how to troubleshoot the most frequent issues. If you aspire to understand ECP and be able to do more complicated fixes on ECP than a simple restart, you should read this.