

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

Version: 4.10.0.4

Date: 19.07.2023

<u>1</u>	<u>DOCUMENT HISTORY</u>	<u>3</u>
<u>2</u>	<u>TERMINOLOGY</u>	<u>4</u>
<u>3</u>	<u>THE BIG PICTURE</u>	<u>5</u>
<u>4</u>	<u>PREPARATIONS</u>	<u>6</u>
<u>5</u>	<u>INSTALLATION OR UPGRADE OF ECP-ENDPOINT ON WINDOWS</u>	<u>9</u>
<u>6</u>	<u>INSTALLATION OR UPGRADE OF ECP-ENDPOINT ON LINUX</u>	<u>10</u>
<u>7</u>	<u>INSTALLATION OR UPGRADE OF ECP ENDPOINT USING DOCKER IMAGE</u>	<u>11</u>
<u>8</u>	<u>ECP SETUP (FOR ALL OS INSTALLATIONS)</u>	<u>14</u>
<u>9</u>	<u>INSTALLATION OR UPGRADE OF EDX-TOOLBOX ON WINDOWS</u>	<u>23</u>
<u>10</u>	<u>INSTALLATION OR UPGRADE OF EDX-TOOLBOX ON LINUX</u>	<u>24</u>
<u>11</u>	<u>INSTALLATION OR UPGRADE OF EDX-TOOLBOX USING DOCKER IMAGE</u>	<u>25</u>
<u>12</u>	<u>EDX SETUP (FOR ALL OS INSTALLATIONS)</u>	<u>27</u>
<u>13</u>	<u>APPENDIX</u>	<u>35</u>

1 Document History

Version	Date	Changes
4.8.2	01/03-2022	<ul style="list-style-type: none"> - Updated to version 4.8.2 of ECP and 1.9.2 of EDX - Document made by Statnett and made primarily for Statnett connected parties, but adapted to be possible to use by all/many actors in NEM, on behalf of NEX
4.8.2.1	08/03-2022	<ul style="list-style-type: none"> - Updated screenshots for Service properties (both ECP and EDX)
4.8.2.2	08/04-2022	<ul style="list-style-type: none"> - Completely harmonized in the NEX group to suit the needs of all TSOs in the Nordic. Major re-organizing of chapters.
4.8.2.4	09/05-2022	<ul style="list-style-type: none"> - Improved Docker installation/upgrade for ECP. Reviewed by NEX until chapter 12.5.2, except chapter 11
4.8.2.5	16/05-2022	<ul style="list-style-type: none"> - Reviewed until 12.5.3, except chapter 11
4.8.2.6	23/05-2022	<ul style="list-style-type: none"> - Almost complete review, except the Docker-chapters – they are still unfinished.
4.8.2.7	28/05-2022	<ul style="list-style-type: none"> - Updated Docker-chapters, but the cloud-install part of Docker is still not reviewed properly - Also added some text in the upgrade section of Windows (ECP/EDX) to suggest the best way to handle configuration files that you may have touched or that might be subject to change.
4.9.0.1	12/09-2022	<ul style="list-style-type: none"> - This guide has a crucial update for how to install java correctly on Windows (namely – specify JRE_HOME) - This guide will most likely work with ECP 4.9.0 and EDX 1.10.0 – these versions are considered stable and usable. There is no configuration change from 4.8.2/1.9.2
4.9.0.2	27/09-2022	<ul style="list-style-type: none"> - Removed information about Fingrid-addresses – contact Fingrid to get these
4.9.0.3	14/12-2022	<ul style="list-style-type: none"> - Cropped a couple of screenshots to avoid showing any URL
4.9.0.4	23/12-2022	<ul style="list-style-type: none"> - Now referencing correct document to find EDX profiles-property setting
4.10.0.0	3/2-2023	<ul style="list-style-type: none"> - Updated to support v4.10 of ECP and v1.11 of EDX. Simplified some parts of the document and updated others to be in line with current ideas/guidelines. If you perform an upgrade, this guide assumes you start from 4.8.2 or higher. If you want to upgrade from lower version (4.6 or 4.7) then please use the old version of this document (4.9.0.4).
4.10.0.1	7/2-2023	<ul style="list-style-type: none"> - Minor corrections
4.10.0.2	27/3-2023	<ul style="list-style-type: none"> - Small change to be more precise about port changes necessary for EDX and ECP on same host in the Linux case.
4.10.0.3	16/6-2023	<ul style="list-style-type: none"> - Reinserted the Fingrid addresses (removed in 4.9.0.2) - Offer new failover messagepaths for all endpoints (see chapter 8.5.1)
4.10.0.4	19/7-2023	<ul style="list-style-type: none"> - Corrected "internalBroker.port" to "internalBroker.amqp.port" in chapter 8.1.1. This property default nonetheless to 5671 – as suggested, so even if not set – the port number will be 5671.

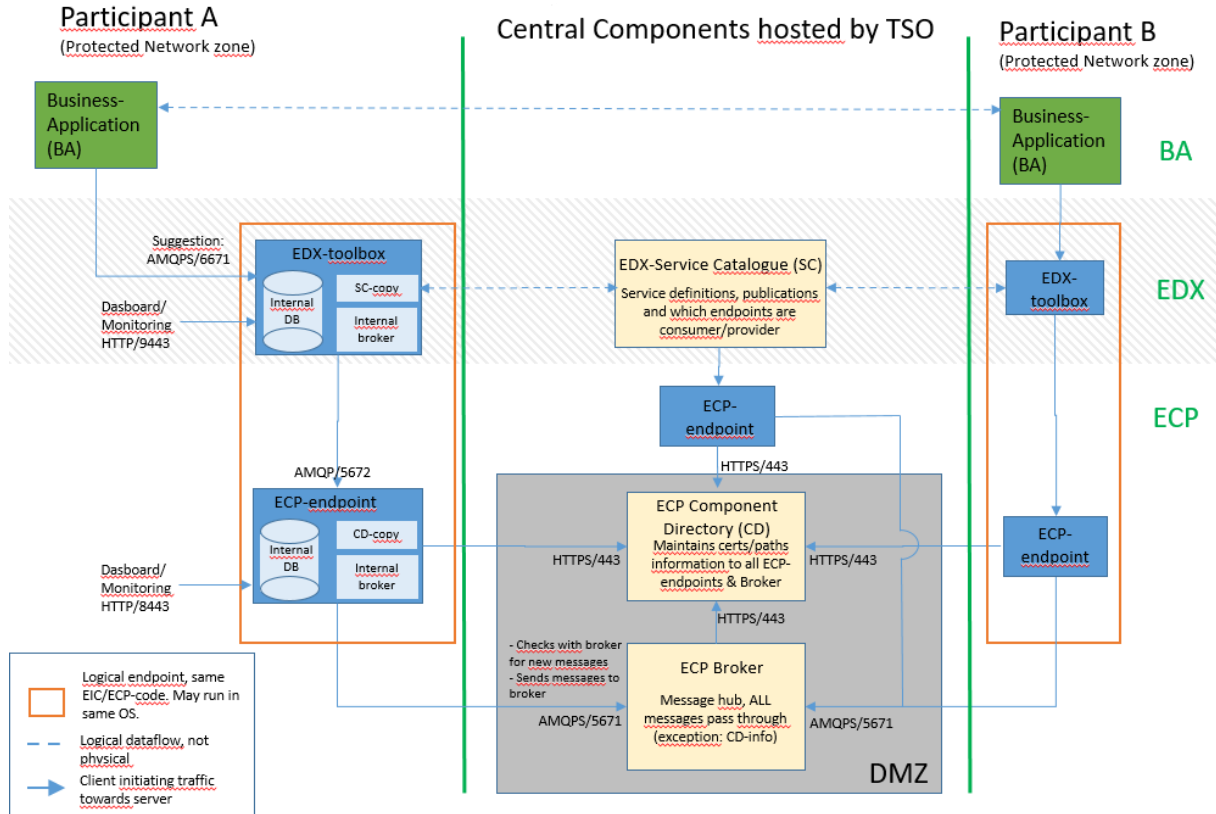
2 Terminology

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
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.
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.

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:



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

- What components exist in the ECP and which must a participant 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 do the messages flow in the system and how is the logical flow?

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

- ECP-endpoint and EDX-toolbox should be in the same OS-runtime.
- During the installation phase you will need to connect in the Component Directory.
- After the approval by the TSO, you may send messages from one ECP-endpoint to another.
- By adding the EDX "on top", you will gain the service-concept which among many things will allow you to address a service provider without knowing a specific address (example of destination: "SERVICE-NO-AFFRCAP").
- You can therefore test ECP-to-ECP traffic (using the ECP-Dashboard) before you test EDX-to-EDX traffic (using the EDX-Dashboard)
- EDX-traffic will not work unless the Service Catalogue is updated by the TSO with information about your endpoint and which services you will consume.
- When EDX-to-EDX traffic works you may test BA-to-BA traffic.

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 package for ECP v4.10.1
 - Download package for EDX v1.11.0
 - In these 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) – a reference for advanced configuration, may not be necessary
 - ECP/EDX Release Notes (RN)
 - EDX User Guide (USG) – a reference for how BA connects to EDX
 - Download images for Docker/Kubernetes if you need this. You will need special access to retrieve these images from your TSO (contact email is listed on the web page)

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 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 Key. 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 2016 or 2019, RHEL 7/8 or CentOS 7 (but not 8!). In v4.11 (coming summer 2023) RHEL 8 will be dropped, and support for RHEL 9 and Oracle Linux 9 will be added.

- We recommend running a single EDX & ECP on a standalone¹ 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.

Client	Server	Port	Protocol	Doing what?
Operator	EDX-toolbox	9443	HTTPS	Monitoring/Dashboard
BA	EDX-toolbox	6672/6671	AMQP/AMQPS	Message transport
Operator	ECP-endpoint	8443	HTTPS	Monitoring/Dashboard
ECP-endpoint – open only to your TSO CD	CD (prod): ecp4prod.statnett.no iecp.prod.energinet.dk ecp4.svk.se ecp.fingrid.fi	443 for Statnett 8443 for other TSOs	HTTPS	Synch of ECP-network information/certificates
	CD (test): ecp4.statnett.no iecp.preprod.energinet.dk ecp4-test.svk.se ecp-test.fingrid.fi			
ECP-endpoint – open to all TSO	Broker (prod): ecp4prod.statnett.no iecp.prod.energinet.dk ecp4.svk.se ecp.fingrid.fi	5671	AMQPS	Message transport
	Broker (test): ecp4.statnett.no iecp.preprod.energinet.dk ecp4-test.svk.se ecp-test.fingrid.fi			

Special note about the firewall openings to all brokers (but only one opening to the CD). The reason behind this is twofold.

- 1) You might need to access a service located at another TSO than your primary TSO
- 2) NEX can in the future utilize all brokers as failover brokers – thus gaining higher uptime for the network.

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.

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.

4.7 Endpoint policy

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

- The rule of thumb is one Endpoint pr Market Actor (MA). The Endpoint may be operated or even owned by an Endpoint Operator (EO).
- 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 can have endpoints as well as MAs.

4.8 Uptime & Performance

Go to <https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/> to locate the document "Ideal maintenance". This document explains NEX recommendations and reasoning regarding uptime and setup.

5 Installation or Upgrade of ECP-endpoint on Windows

5.1 Java

Make sure to have an updated JRE (Java Runtime Environment) installed. Currently the recommended version is 11.0.15. 11.0.16 is not safe to use, newer versions **may** work fine. Read more about installation of Java in ECP Installation Guide (IG) chapter 5.1.

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 JRE_HOME is defined and points to the folder where JRE is installed. JAVA_HOME should not be present, no matter what the guides says otherwise.
- 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

Execute ECP IG chapter 7.2. Skip "Upgrade" chapter below and continue in this document.

5.3 Upgrade

You may read the ECP Release Notes to get more information (see chapter 4.1). Then execute ECP UG chapter 5.1.

5.4 Service properties

When ECP-endpoint is registered as a Windows service, it is possible to change its system properties using the Tomcat application for management of Windows services:

- `cd <install_path>\tomcat\bin`
- `tomcat9w.exe //ES//ecp-endpoint` (see footnote for syntax explanation ²)

When the property dialogue opens, change tab to Java. Consider changing the following settings:

- Make sure "Use default" is on/true – important **if you upgraded Java/JRE and for future upgrades**
- Make sure you have data in the Startup-tab – if this tab is void of information, the service will not work. If it happens, please uninstall and try again – following each step very closely.

Continue to chapter 8.

² <https://tomcat.apache.org/tomcat-9.0-doc/windows-service-howto.html>

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 (but 7.4.7 and 7.4.8 is not necessary)

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 you need to go to the NEX website (ediel.org) and follow the instructions.
- 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 14. This chapter is a general guide to Docker and a (little bit too high-level) description on how to setup an entire ECP-network with Broker and Component Directory.

A general warning: 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 make sure to have fast disks and rather close to the host running the container.

Another warning: 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 is not covered in this document because NEX has zero 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 6).

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, but you only need the part provided for the endpoint itself. Here we show a slight variation from the one provided in the zip-file:

```
version: '3'
services:
  ecp-endpoint1:
    image: entsoe/ecp-endpoint:4.8.2.1475
    container_name: ecp-endpoint1
    tty: true
    environment:
      # Due to a bug we have to specify the entire CATALINA_OPTS to specify the Xmx-argument. We run with a fairly low
      # memory requirement here. This may not be necessary in v4.10 of ECP
      - CATALINA_OPTS=-Xms64M -Xmx512M -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/var/log/ecp-endpoint/ecp-
      dump.hprof -Dspring.config.additional-location=file:/etc/ecp-endpoint/ecp.properties,file:/etc/ecp-endpoint/ecp-
      users.properties -Dbroker.internal.auth.settings.location=/etc/ecp-endpoint -Decp.endpoint.jms.directory=/etc/ecp-
      endpoint/jms -Decp.password.location=/etc/ecp-endpoint/ecp-password.properties -
      Dcom.sun.management.config.file=/etc/ecp-endpoint/jmxremote.properties -Djava.security.egd=file:/dev/urandom
      # We specify each property file we want to touch here, instead of the entire config folder.
      # We also specify hawtio, in order to avoid having to install it every time inside the container
    volumes:
      - ./ecp-endpoint1/log:/var/log/ecp-endpoint:Z
      - ./ecp-endpoint1/data:/var/lib/ecp-endpoint:Z
      - ./ecp-endpoint1/config/ecp.properties:/etc/ecp-endpoint/ecp.properties:Z
      - ./ecp-endpoint1/config/ecp-users.properties:/etc/ecp-endpoint/ecp-users.properties:Z
      - /opt/hawtio/ecp-hawtio-web-1.5.11.war:/usr/share/ecp-endpoint/webapps/hawtio.war:Z
    ports:
      - 25003:8443
```

```
networks:
  ecp-test-network:
    ipv4_address: 172.32.3.20

networks:
  ecp-test-network:
    driver: bridge
    ipam:
      driver: default
      config:
        - subnet: 172.32.3.0/24
```

Such a setup is meant for testing/development, and maybe not for production. However, it will suffice for many who have a low-traffic endpoint.

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-cribled 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 very good 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 (IG) 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 ECP Setup (for all OS installations)

8.1 ECP-endpoint configuration

8.1.1 Configuration of ecp.properties

Make sure the following properties 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.

The internalBroker-properties are where we expect most problems, since the values you're using here **must** match those you use for "ecpBroker"-properties in edx.properties! See chapter 5.5.3 in ECP Administration Guide for more information about internalBroker-properties.

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

Property	Description
<pre>ecp.endpoint.amqpApiEnabled = true ecp.endpoint.sendHandler[0].beanName=amqpApiSendHandler ecp.endpoint.sendHandler[0].typeName=*</pre>	<p>You MUST ADD these properties, otherwise messages cannot be sent from ECP to EDX.</p>
<pre>spring.profiles.active= ecp-nonha</pre>	<p>With this setting you require authenticated access to dashboard and webservice. For other options, see ECP Administration Guide.</p>
<pre>ecp.directory.client.synchronization.directorySynchronizationInterval=30 * * * * * ecp.directory.client.statistics.directorySynchronizationInterval=15 * * * * * ecp.directory.client.synchronization.messagePathSynchronizationInterval=45 * * * * *</pre>	<p>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.</p>
<pre>internalBroker.host=127.0.0.1 internalBroker.amqp.port=5672 internalBroker.useAuthentication=false # Parameters below are not necessary, unless # useAuthentication=true internalBroker.keystore.location=\${dataDirectory}/authKeystore.jks internalBroker.keystore.password=password internalBroker.keystore.authAlias=ecp_module_auth internalBroker.auth.user=endpoint internalBroker.auth.password=password</pre>	<p>These settings define how ECPs broker will make itself available to EDX and assumes you install EDX on the same host as ECP. If you install EDX on another host, set host-property to '0.0.0.0'.</p> <p>If you want to change from using AMQP to AMQPS between EDX and ECP, then set useAuthentication=true. Furthermore, note that the authKeystore.jks must be available in EDX-setup ("ecpBroker"-properties in edx.properties) to allow EDX to trust ECP.</p> <p>Also note that ECP will itself log on to this broker using the auth-user/password parameters. These parameters must match a user found in users.properties (which differs from ecp-users.properties). With default values this should be ok and no changed is needed.</p>

8.1.2 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. It does **not** deal with how ECP itself connects to the internal broker (that is handled by ecp.properties and user.properties). 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.

8.2 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 the endpoint now.

8.3 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.

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.

8.4 ECP-endpoint Registration – skip this one in Upgrade

8.4.1 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 some³. 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	45V0000000000057R	https://iecp.preprod.energinet.dk:8443/ECP_MODULE

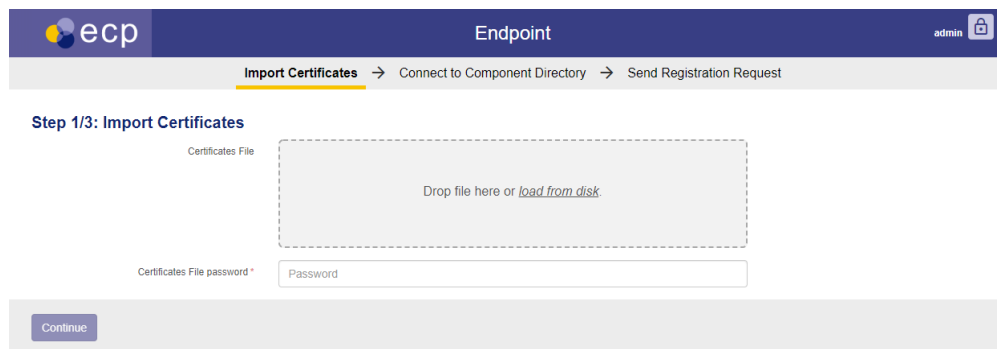
³ Usually not a hard guess. But cannot be written here.

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 **monitored⁴ 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.



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

⁴ Monitored means that some organization will be responsible for processing the email at least daily.

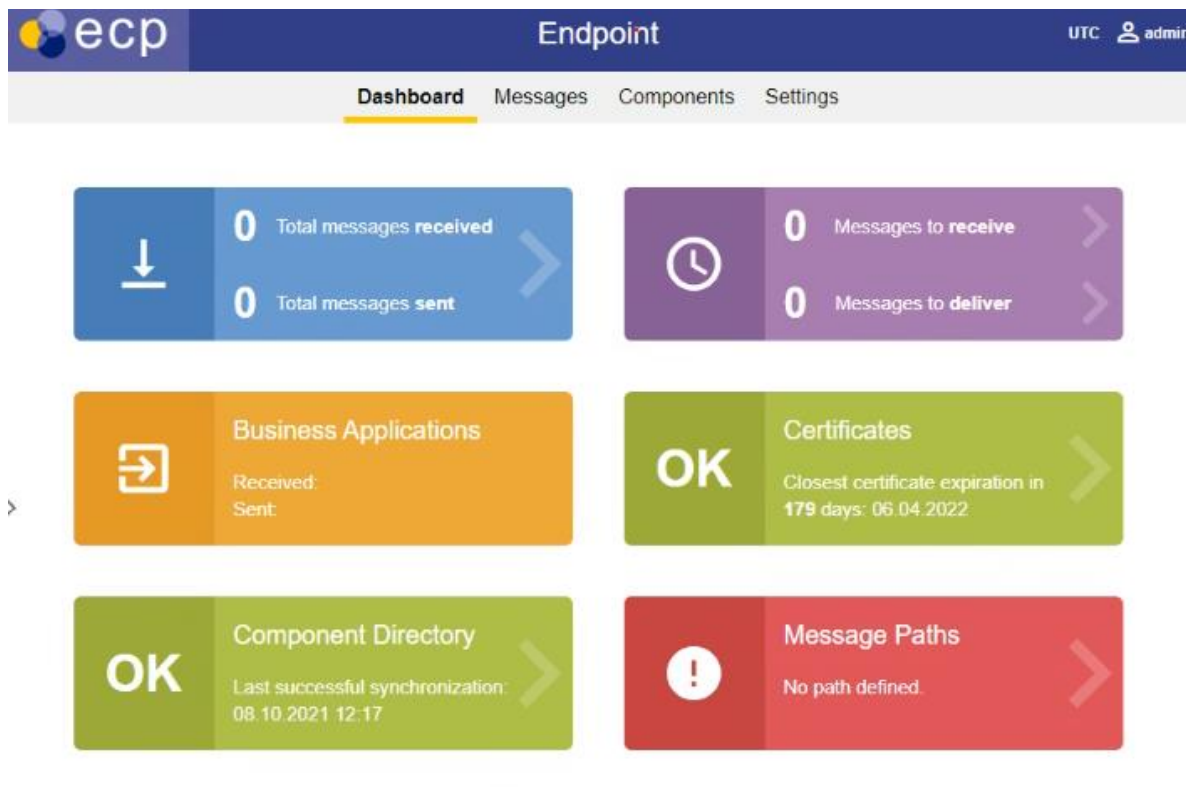
Enter the CD Code and CD URL – not your own Endpoint Code (EC)!

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

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

8.4.2 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.



The screenshot shows the ECP Endpoint Dashboard with the following tiles:

- Total messages received:** 0
- Total messages sent:** 0
- Messages to receive:** 0
- Messages to deliver:** 0
- Business Applications:** Received: 0, Sent: 0
- Certificates:** OK, Closest certificate expiration in 179 days: 06.04.2022
- Component Directory:** OK, Last successful synchronization: 08.10.2021 12:17
- Message Paths:** !, No path defined.

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

8.5 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 *

44V000000000018F

45V0000000000058P

45V0000000000062Y

50V000000000112U

8.5.1 Failover Message Path

If you want higher uptime on the communication to the ECP-network, you can now 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):

Paths

+ New Path **Import Paths**

Business Acknowledgement

Path Direct Indirect Status Active Invalid Broker Valid on

Message Type **Search**

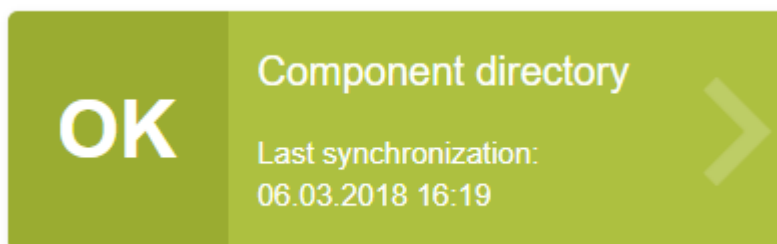
Senders	Message Type	Path	Valid From	Valid To	Status
All	*	50V000000000112U	26.04.2023 20:00		Active
All	*	46V000000000015S	01.06.2023 12:00		Active
All	*	45V000000000058P	01.06.2023 12:00		Active
All	*	44V000000000018F	01.06.2023 12:00		Active

10 25 50 100

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.

8.6 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:

State	Receiver	Message Type	Sending Time
Received	50V0000000001150	TEST	28.01.2020 09:31

8.7 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

Registration

8.8 Troubleshooting

8.8.1 ECP-Endpoint does not respond on expected port

- You've upgraded to ECP 4.7.2/EDX v1.8.2 or above, and default ports are now SSL-port 8443. 8080 is by default disabled.
- Another possible reason is that you have other processes listening on the same 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 -p". You can then identify the PID of the process listening on port 8080 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.

8.8.2 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.

8.8.3 Message status "Accepted"

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

8.8.4 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)

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

- Your Message Path is not defined in "Settings" (see chapter 8.5)

9 Installation or Upgrade of EDX-toolbox on Windows

9.1 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).

9.2 Installation

Execute EDX IG chapter 6.2 and 6.3. Skip "Upgrade" chapter below and continue in this document.

9.3 Upgrade

Execute EDX UG chapter 5.1

9.4 Service properties

When EDX-toolbox is registered as a Windows service, it is possible to change its system properties using the Tomcat application for management of Windows services:

- `cd <install_path>\tomcat\bin`
- `tomcat9w.exe //ES//edx-toolbox` (see footnote for syntax explanation⁵)
- On the Java-tab: Select "Use default" checkbox – important if you ever upgrade JRE on your system
- If EDX runs in same OS as ECP, you need to change ports to avoid conflict. That should be covered in the EDX Installation Guide (chapter 6.3.3).
- Make sure you have data in the Startup-tab – if this tab is void of information, the service will not work. If it happens, please uninstall and try again – following each step very closely.

⁵ <https://tomcat.apache.org/tomcat-9.0-doc/windows-service-howto.html>

10 Installation or Upgrade of EDX-toolbox on Linux

10.1 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. Chapter 6.4 does not cover well how to change the ports if you run EDX on the same host as ECP. You will have to look up the file **/etc/systemd/system/edx-toolbox.service.d/env.conf** and change port numbers there, in the same manner as covered for the Windows installation (chapter 6.3 in EDX IG).

10.2 Upgrade

Execute the EDX Upgrade Guide (UG) chapter 5.2 (5.2.11 is not important)

11 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 you need to go to the NEX website (ediel.org) and follow the instructions.
- The standard EDX Download package examples of setup
 - edx-docker-test-env.zip (a "bare-bone" setup – see chapter 7.1.1)
 - edx-endpoint-kubernetes.zip (useful for "cloud" setup – see chapter 7.1.2)
- 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).

11.1 Installation

11.1.1 Bare-bone Docker Host setup

You may use the docker-compose.yml in the zip-files provided in the download, but you only need the part provided for the toolbox itself. Here we show a slight variation from the one provided in the zip-file:

```
version: '3'
services:
  edx-endpoint1:
    image: entsoe/edx-toolbox:1.9.2.1165
    container_name: edx-endpoint1
    tty: true
    environment:
      # We have to specify the whole CATALINA_OPTS in order to specify the Xmx-argument. This will probably be fixed
      # in version 1.11 of EDX
      - CATALINA_OPTS=-Xms64M -Xmx512M -XX:+HeapDumpOnOutOfMemoryError -XX:HeapDumpPath=/var/log/edx-toolbox/edx-
dump.hprof -Dspring.config.additional-location=file:/etc/edx-toolbox/edx.properties,file:/etc/edx-toolbox/edx-
users.properties -Dedx.toolbox.yaml.directory=/etc/edx-toolbox -Dedx.toolbox.jms.directory=/etc/edx-toolbox/jms -
Dbroker.internal.auth.settings.location=/etc/edx-toolbox -Dedx.toolbox.tomcat.port.shutdown=8005 -
Dedx.toolbox.tomcat.port.http=8080 -Dedx.toolbox.tomcat.port.ajp=8009 -Dedx.toolbox.tomcat.port.ajp.redirect=8443 -
Dedx.toolbox.tomcat.port.http.redirect=8443 -Dedx.password.location=/etc/edx-toolbox/edx-password.properties -
Dcom.sun.management.config.file=/etc/edx-toolbox/jmxremote.properties -Djava.security.egd=file:/dev/urandom
    volumes:
      # We specify those particular property files we want to touch and hawtio as well
      - ./edx-endpoint1/log:/var/log/edx-toolbox:Z
      - ./edx-endpoint1/data:/var/lib/edx-toolbox:Z
      - ./edx-endpoint1/config/edx.properties:/etc/edx-toolbox/edx.properties:Z
      - ./edx-endpoint1/config/edx-users.properties:/etc/edx-toolbox/edx-users.properties:Z
      - ./edx-endpoint1/config/edx.yml:/etc/edx-toolbox/edx.yml:Z
      - /opt/hawtio/edx-hawtio-web-1.5.11.war:/usr/share/edx-toolbox/webapps/hawtio.war:Z
    ports:
      - 25013:8443
      - 25002:5672
    networks:
      ecp-test-network:
        ipv4_address: 172.32.3.21

networks:
  ecp-test-network:
    driver: bridge
    ipam:
      driver: default
      config:
        - subnet: 172.32.3.0/24
```

Such a setup is meant for testing/development, and maybe not for production. However, it will suffice for many who have a low-traffic endpoint.

11.1.2 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.

11.1.2.1 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.

11.1.2.2 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.

11.2 Upgrade

The official documentation from Unicorn is not very good 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 (IG) 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.

12 EDX Setup (for all OS installations)

12.1 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 8.4.1. 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: <ul style="list-style-type: none"> • Statnett: 50V000000000113S • Fingrid: 44V000000000023M • Energinet: 45V000000000059N • SvK: 46V000000000016Q NEM Production-network: <ul style="list-style-type: none"> • Statnett: 50V000000000120V • Fingrid: 44V000000000024K • Energinet: 45V000000000055V • SvK: 46V000000000021X
ecpBroker.amqp.port=5672 ecpBroker.amqp.host=127.0.0.1 ecp.broker.url= amqp://\${ecpBroker.amqp.host}:\${ecpBroker.amqp.port} # These properties are only necessary if # internalBroker.useAuthentication=true # in ecp.properties (not this file!!) ecpBroker.keystore.location=authKeystore.jks ecpBroker.keystore.password=password ecpBroker.keystore.authAlias=ecp_module_auth ecpBroker.auth.user=toolbox ecpBroker.auth.password=password	Port number of your ECP AMQP Broker, most likely 5672 . The property must be the same as specified in ecp.properties (chapter 8.1.1) in the internalBroker.port property. Host is 127.0.0.1 if you install EDX in same OS as ECP. Otherwise, use the IP/hostname of you ECP-endpoint. Read chapter 8.1.1 to about internalBroker.host which is the property that must fit with this property. Optional: Use AMQPS between EDX and ECP: The keystore can be a copy or the same as keystore used in ecp.properties for internalBroker.keystore.location , or a keystore which contains the same root-certificate to trust. The user/password must be found in ECPs users.properties file. The ecp.broker.url must start with " amqps://.. "
internalBroker.amqp.port=6671 internalBroker.amqp.host=0.0.0.0 internalBroker.useAuthentication=true internalBroker.keystore.location=authKeystore.jks internalBroker.keystore.password=password internalBroker.keystore.authAlias=ecp_module_auth internalBroker.auth.user=toolbox internalBroker.auth.password=password edx.broker.url= amqps:// \${internalBroker.amqp.host:127.0.0.1}: \${internalBroker.amqp.port:5672}	We recommend useAuthentication=true – it results in AMQPS (AMQP+SSL), but of course then your BA must also handle this. Portnumber is recommended 6671 for AMQPS and 6672 for AMQP. Also note that EDX will itself log on to this broker using the auth-user/password parameters. These parameters must match a user found in users.properties (which differs from edx-users.properties). With default values this should be ok and no changed is needed. Make sure edx.broker.url starts with "amqps" if AMQPS is used, "amqp" otherwise.

spring.profiles.active= edx-nonha	With the recommended setting you require authenticated access to dashboard and webservice. 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.

12.2 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 12.6. NB! Make sure to have only **one** yml-file in the config-directory – since EDX will read all files with yml-suffix.

12.3 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-administrator 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.

12.4 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.

12.5 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.

12.5.1 EDX GUI verification

Check the EDX-Dashboard-URL:

https://localhost:9443/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.

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.

12.5.2 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 8.5). 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.

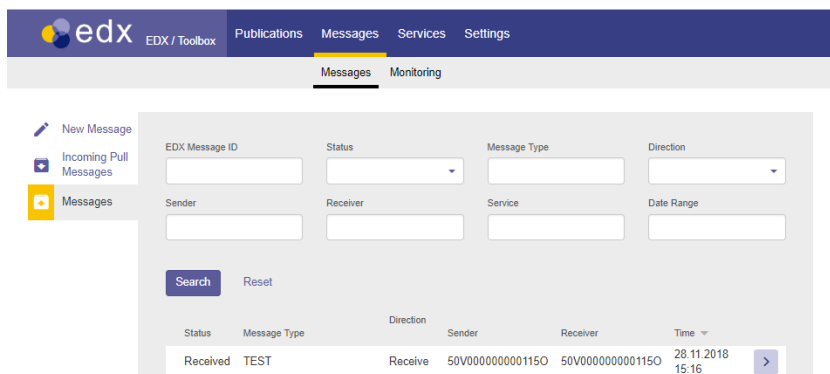
Finally, when the file arrives in the Inbox of ECP, the file should then be consumed by EDX Toolbox, but you will not see this file in the Messages-GUI of EDX. The file/SC-copy will be shown in the Settings-page of EDX. The screenshot below shows an EDX Toolbox which is added in multiple Service Catalogues, 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.

The screenshot shows the EDX Toolbox Settings page. At the top, there is a navigation bar with 'edX Toolbox' and tabs for 'Publications', 'Messages', 'Services', and 'Settings'. The user is logged in as 'admin'. Below the navigation bar, the 'Settings' section is active, showing fields for 'Toolbox Code' (50V-SN-DK---ATT), 'Toolbox Name' (Statnett DK), 'Party Name' (PARTY), and 'Publish Subscribe Version' (98). A '+ Register Toolbox to SC' button is visible. Below this is a table of Service Catalogues with columns for 'Service Catalogue', 'Network Configuration Version', and 'Last Synchronization time'. Each row has 'Download' and 'Request from SC' buttons. At the bottom, there is a 'Time Zone' dropdown menu and a 'Save' button.

Service Catalogue	Network Configuration Version	Last Synchronization time	Download	Request from SC
50V00000000113S	480	28.02.2022 09:02:48	Download	Request from SC
46V000000000016Q	94	28.02.2022 09:02:47	Download	Request from SC
45V0000000000059N	198	28.02.2022 10:20:20	Download	Request from SC

12.5.3 Send test messages

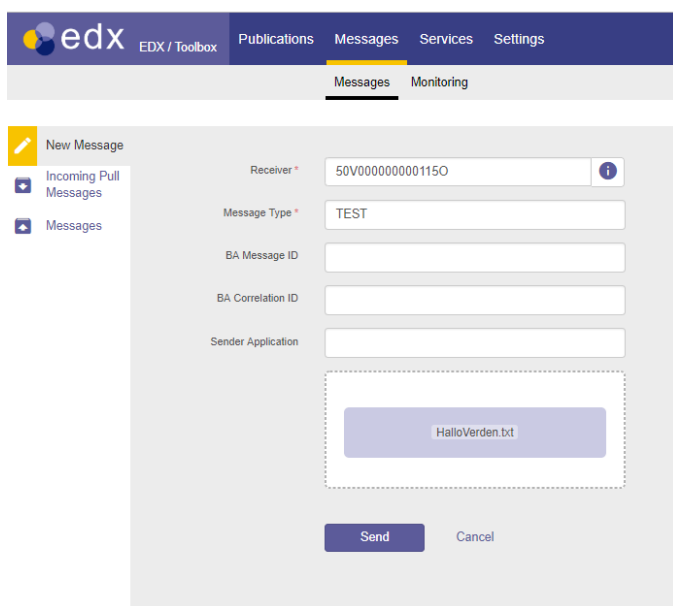
Open a web browser and navigate to the dashboard URL of your toolbox. You should see this, except you won't have any messages in the list:



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

- NEM Test-network:
 - Statnett: **50V000000000115O**
 - Fingrid: **44V00000000019D**
 - Energinet: **45V0000000000601**
 - SvK: **46V000000000017O**
- NEM Production-network:
 - Statnett: **50V000000000188Y**
 - Fingrid: **44V00000000010V**
 - 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:



12.6 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 accidentally 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&valueSerializer=org.apache.kafka.common.serialization.BytesSerializer"}
  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: 50V00000000113S }}
      - {code: R-fssf-ba2, start: toolbox-gateway, end: fssf-ba2-inbox, service: {serviceCode: MMS, domainCode: DEFAULT_DOMAIN, serviceCatalogueCode: 50V00000000113S }}
      - {code: R-ba1-ba2, start: toolbox-gateway, end: [amqp-ba1-inbox, fssf-ba2-inbox], service: {serviceCode: DK-MVA, domainCode: DEFAULT_DOMAIN, serviceCatalogueCode: 50V00000000113S }}
      - {code: R-kafka-ba4, start: toolbox-gateway, end: kafka-publish, service: {serviceCode: NO-NUCS, domainCode: DEFAULT_DOMAIN, serviceCatalogueCode: 50V00000000113S }}
    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**:

12.6.1 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.

12.6.2 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 too 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.

[12.6.2.1 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.

[12.6.2.2 FSSF-endpoint](#)

The configuration suggested in the example shows 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.

12.6.2.3 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 be created (as ba3/tmp) and given proper privileges.

12.6.2.4 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.

12.6.3 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.

12.7 Change timezone of the toolbox

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

The screenshot shows the 'Settings' page for a toolbox. It includes fields for Toolbox Code, Name, Party Name, and Publish/Subscribe Version. Below these is a table of service catalogues with columns for Service Catalogue, Network Configuration Version, and Last Synchronization time. At the bottom, there is a 'Time Zone' dropdown menu which is circled in red, and a 'Save' button.

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

12.8 Troubleshooting

12.8.1 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 12.1), 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 8.8.5. 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.

12.8.2 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 12.1), 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".

13 Appendix

13.1 Hawtio

In `edx.properties` and `ecp.properties` specify the following property (or change the property if it exists) and restart the applications:

- `spring.jmx.enabled=true`

You may use Hawtio to monitor queues – it can be very useful. To install it, download Hawtio from [here](#)

<https://repo1.maven.org/maven2/io/hawt/hawtio-web/1.5.11/hawtio-web-1.5.11.war>

and copy it in the `webapps`-folder of the ECP/EDX-application with the filename **hawtio.war**. It should auto-deploy and you can access it on the following URL (admin/password):

- <http://<YOUR-ECP-ENDPOINT>:8443/hawtio/>
- <http://<YOUR-EDX-TOOLBOX>:9443/hawtio/>

You can, among other things, browse queues and see if messages are picked up, and even purge queues if something is stuck. Very useful.

13.2 Connectivity Check

Send a "technical" test-message to a TSO-endpoint. This is because you in all likelihood will communicate with a TSO on a NEM network, but connectivity to other endpoints might also be necessary or useful to monitor.

The simplest approach is to run an HTTP-request like this:

```
PUT http://<Your-ECP-endpoint>:<port>/ECP_MODULE/settings/connectivityCheck
```

Set HTTP header for "Content-Type" to "application/json"

Set HTTP body to `{"receiver": "<ECP-code>", "messageType": "TEST"}`

The more advanced approach is when your endpoint require login: Then you need a token from login to perform the connectivity-check. We've made a python3-script, which is now updated to support ECP 4.8.1, using port 8443 in this example. Some long lines are wrapped (shown with 2 space indentation):

```
import requests
import json
import time
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
requests.packages.urllib3.disable_warnings()

host = 'hostname-of-your-ecp-endpoint'
endpoint_code_to_check = 'VALID-ECP-CODE'
username = 'admin'
password = 'password'

session = requests.Session();
response = session.get('https://%s:8443/ECP_MODULE/' % host, timeout=30, verify=False).content
token = (session.cookies.get_dict())
```

```

xxtoken = token['XSRF-TOKEN']
headers = {'X-XSRF-TOKEN': xxtoken, 'Content-Type': 'application/json;charset=UTF-8'}
response_conncheck = session.put('https://s:8443/ECP_MODULE/settings/connectivityCheck' % host,
auth=(username, password), verify=False, data={'receiver':"%s","messageType":"TEST"}' %
endpoint_code_to_check, headers=headers).text

response_status_json = json.loads(response_conncheck)

```

The response JSON can be parsed and dumped to file like this:

```

status_check = response_status_json['status']
output_json = {}
statustime = time.strftime("%Y-%m-%d %H:%M:%S")
timestamp = int(time.time())
output_json['statustime'] = statustime
output_json['timestamp'] = timestamp
output_json['connectivity_check_status'] = status_check
output_json['hostname'] = '%s' % host
output_json['endpoint_code_checked'] = '%s' % endpoint_code_to_check

print (json.dumps(output_json))

```

And it will print something like this to your output_json file:

```

{"timestamp": 1599561602, "statustime": "2020-09-08 12:40:02", "connectivity_check_status": "OK",
"hostname": "your-own-ecp-endpoint-host", "endpoint_code_checked": "50V000000000121T"}

```

The connectivity-check will give up after 5 seconds, which is can be a bit too soon. We recommend to change this to 30 seconds, using these parameters settings in the ecp.properties file:

```

ecp.endpoint.connectivityCheckAttemptTimeout=3000
ecp.endpoint.connectivityCheckAttemptsCount=10

```

When the monitoring alerts you of a problem, most likely the problem is your own ECP-endpoint. Do not contact the TSO (if you test connectivity towards it) to alert about issues, unless you are very sure that your own endpoint is fine and that "some time" has passed.

13.3 Queue monitoring and queue cleaning

The ECP/EDX does not clean queues, especially the ActiveMQ.DLQ and reply-queues can sometime run full and be the cause of a full stop of the EDX or ECP. There fore you need to run some tool which can help you fix the problem.

13.3.1 Hawtio solution

Sometimes it can be valuable to monitor the queues in EDX. For that to work you need to install Hawtio (see previous chapter in Appendix). The you can run a simple curl-command, the queue-name is marked in red font.

```

curl 'https://<EDX-toolbox>:<port>/hawtio/jolokia/' -H 'Content-Type: text/json' -
-data-binary
'["type": "read", "mbean": "org.apache.activemq:type=Broker,brokerName=localhost,des
tinationType=Queue,destinationName=edx.endpoint.inbox", "config": {}]' --compressed
--insecure

```

If you need to login to run the command, you'll need to adapt the script in the previous chapter.

13.3.2 QueueCleaner solution

Statnett has developed a simple tool to monitor and clean the queues of ECP/EDX using the AMQP-interface. This tool can be found here: <https://ediel.org/nordic-ecp-edx-group-nex/nex-statnett/>

Download it and run it using this command (requires Java 11):

```
java -jar QueueCleaner.jar
```

It should then display all the options and many examples. No guarantee is offered to the reliability of the tool and does not answer to any malfunction the tool might inadvertently cause. The tool can print both to stdout and JSON-output to a file – so log-monitoring like Splunk can pick up the results.

13.4 ECPerfAnalyzer

Statnett has developed a simple tool to monitor message activity across ECP and EDX. It can be found in the same area as QueueCleaner (see above). Download and run it using this command (requires Java 11):

```
java -jar ECPerfAnalyzer.jar
```

No guarantee is offered to the reliability of the tool and does not answer to any malfunction the tool might inadvertently cause.

13.5 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 8.5), 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)

13.6 Create an endpoint with 99.9% uptime & 100K messages per day

Read the 999-document published in same location as this document (ediel.org)