zenon
by COPA-DATA

zenon manual

Docker

v.8.20

Contents

1 Welcome to COPA-DATA help

ZENON VIDEO TUTORIALS

You can find practical examples for project configuration with zenon in our YouTube channel (https://www.copadata.com/tutorial_menu). The tutorials are grouped according to topics and give an initial insight into working with different zenon modules. All tutorials are available in English.

GENERAL HELP

If you cannot find any information you require in this help chapter or can think of anything that you would like added, please send an email to documentation@copadata.com.

PROJECT SUPPORT

You can receive support for any real project you may have from our customer service team, which you can contact via email at support@copadata.com.

LICENSES AND MODULES

If you find that you need other modules or licenses, our staff will be happy to help you. Email sales@copadata.com.

2 Docker

Docker is the brand name of a free software for the isolation of applications using container virtualization. COPA-DATA enables zenon Runtimes to be used in a Docker container. For this, you need Docker for Windows.

3 Basics

Docker simplifies the deployment of applications. In contrast to a virtual machine, a Docker container does not have its own operating system. Instead, it directly uses the operating system of the host.

All dependencies of an application are encapsulated in a Docker **Image**. This contains all the necessary packages and can be easily transported and installed as a file. Containers ensure the isolation and management of the resources used on a computer. They are executed locally on a computer. The containers you create remain stored locally after the application is exited and are not removed. If you wish to remove them afterwards, the containers can already be set with a parameter for automatic deletion when you call the container.

Docker can be used with Linux and Windows. Docker for Windows is required for the zenon Runtime.

DOCKER ENGINE

The Docker Engine provides access to the kernel of the host operating system and is able to create, start and stop containers. Since Docker uses the kernel of the host operating system, a Docker image is much smaller than a virtual machine. Docker containers can be run on any computer on which a Docker Engine has been installed.

4 Terminology

Terms that are used in this manual:

Term	Description
Container:	Active instance of an image. The **Container** is thus currently running and busy. The container will be closed automatically once it is no longer running a program or it has completed its task.
Container-ID:	Unique ID of a container. A part of the ID is enough to address a container, provided that this part clearly identifies a specific container.
Docker Engine:	Provides access to the kernel of the host operating system and is able to create, start and stop containers. It is also known as the **Docker Daemon**.
Docker Hub:	Online service that contains a **Registry** for Docker images and **Repositories**.

Term	Description
Image:	A Docker image is a file, comprised of multiple layers, that is used to execute code in a Docker container. An **Image** consists of multiple layers which are read-only and thus cannot be changed. An image is portable, can be saved in **Repositories** and can be shared with other users. Several containers can always be started from one **Image**.
Registry:	Used for the administration of repositories. For example, **Docker Hub**. The **Registry** is divided into a public and a private part. In the public part, each user can upload images he or she created and thus make them available to other users. In the meantime, there are also official images, for example, of Linux distributors. In the private part, users can upload their Docker images and thereby distribute them easily, for example, in house, without thus making them publicly accessible.
Repository:	Set of images of the same name with different tags, mostly versions.
Tag:	Used to identify different versions of an image. This definition only applies in connection with Docker. It is not used for terms in connection with variables or parameters of batch control.

5 Docker for Windows

Docker for Windows enables zenon Runtimes to be used in a Docker image.

5.1 Installation and configuration

In order to run zenon in a Docker Container, you need both zenon as well as Hyper-V and Docker. COPA-DATA can provide no support for the installation of Hyper-V and Docker.

You can find information on Docker for Windows, for example, at:

- General info: https://docs.docker.com/docker-for-windows/ (https://docs.docker.com/docker-for-windows/)

- Network: https://docs.docker.com/network/ (http://www.)
- Network features: https://docs.docker.com/docker-for-windows/networking/ (https://docs.docker.com/docker-for-windows/networking/)

REQUIREMENTS

In order to use Docker with zenon, you will need:

- appropriate operating system
  - Windows 10 Pro or higher
  - Windows Server 2016 or higher
- active virtualization environment
- a valid license
  Note: zenon cannot be run in demo mode in a Docker container.

For the configuration, you need:

- Docker Desktop
- available zenon installation
- PowerShell

INSTALL AND CONFIGURE DOCKER

To install Docker for Windows:

1. Load Docker for Windows from the website
   https://docs.docker.com/docker-for-windows/install/ (http://www.)
   You require Version 2.0.3 or higher.
2. Install the package.
   Please note:
   - The **Use Windows containers instead of Linux containers Windows-Container** option must be activated on the configuration page.
   - When setup is finished, you will be logged out.
   - Once you log in again, check if Hyper-V is activated.
     If this is not the case, activate Hyper-V.
3. Right click on the Docker icon in the system tray of the taskbar.
4. In the context menu, select **Switch to Windows container...**.
   If this entry is not shown, Docker Desktop is already configured properly.
5. Check if Docker is installed correctly.
   To do this:

- Open a Powershell in Windows.
- Enter: **docker --version**
  The version information will be displayed.

EXTERNAL HYPER-V SWITCH

A NAT switch is used by default for Docker containers. However, you will address your zenon Runtime from other clients via the Docker container hostname.

To release your Docker container for the DHCP and DNS server:

1. Create an external switch in Hyper-V in the **Virtual Switch Manager**.
2. Restart Docker Desktop.
   In this way, the virtual switch is recognized.
3. Check if the new switch is available.
   To do this, enter **docker network ls** in PowerShell

You can find further information on zenon networks with Docker in the **Runtime Docker with name resolution** (on page 12) section.

5.2 Obtain Docker Image forzenon Runtime from Docker Hub

You can find the image for the zenon Runtime at https://hub.docker.com/u/copadata (https://hub.docker.com/u/copadata).

To load the image from Docker Hub:

1. Open PowerShell.
2. Select the desired Runtime version.
3. Download the image with the **Docker Pull Command** displayed.
4. You will receive information on the fingerprint and download status.

5.3 Create a container with zenon Runtime

Now create a container with a zenon Runtime. The container contains a reference to the zenon files. Log files, Runtime files and licenses are stored there.

To create the container:

1. On your host, create a folder in which the Runtime files of your Runtime container are stored. Settings information and projects are stored there. This folder is called **data path** in the

documentation. You can find the structure in the **zenonDataTemplate.zip** file on the installation medium.

2. Copy the Runtimes files into the **data-path\projects** subfolder.
   If this folder is empty, the files will be created in the container the first time the Runtime is started.
3. Configure the zenon installation. To do this, edit the files in **data path**.
   You can find detailed information on this in the **zenon Configure Runtime for Docker** (on page 9) chapter.
4. Change the values in the **zenon6.ini** (**data path\system\zenon6.ini**):
   - **VBF30**: path to the Runtime project.
     For example: **C:\zenondata\projects[workspace]**
   - **DEFANWENDUNG30**: Name of the Runtime project.
5. Change the **settings.xml** (**data-path\settings.xml**).
   Here you can configure the license server, the primary DNS name and the processes to be started.
6. Open PowerShell.
7. Enter the following command: **docker run -d -v [data-path]:C:\zenondata --name [Container-Name] copadata/zenon-runtime820-windows**
   - **data-path**: path to the folder which should contain the saved data. The path you created in the first step.
   - **Container name**: name of the newly created container, for example: zenon

   Example: **docker run -d -v C:\zenondata:C:\zenondata --name zenon copadata/zenon-runtime820-windows**

   In this way, the files are created in the [**data-path**] folder.
   - Stop the container.
     To do this, enter: **docker stop [Container-Name]**
     Container name: Name of the container which was created before with **docker run**.

5.4 Configure zenon Runtime for Docker

REQUIREMENTS

To be able to use zenon in a Docker container, the required services must be available. These are automatically activated during installation.
This makes it possible:

- to transfer Runtime files by remote transport.
- to read back Runtime files by remote transport.

- to automatically start, reload and close Runtime.

FOLDERS AND FILES

For the start, all the necessary subfolders and files must be available in the **data path**:

- **System\K5LICENSE.INI**: license for zenon Logic.
- **System\License.ini**: license for zenon.
- **System\zenon6.ini**: settings for zenon
- **settings.xml**: Settings for license server, container start behavior and Primary DNS-Suffix.
- **Projects**: collects project files.
- **LOG**: collects log files.

The configured license server is checked during the start.
If a valid network license is available, Runtime is licensed. If there is no valid license available, Runtime cannot be started.

LOGGING

If a functionality is used in the Runtime which is not supported by Docker, an entry is made in the log file.

Locations:

- zenon log files: ...\ *[data path]\LOG*
  Can be viewed with Diagnosis Viewer.
- Docker log files: *Windows Logs\Application* ...
  Can be viewed with Windows Event Viewer.

CONFIGURATION

To configure zenon for use in a Docker image, the files **settingx.xml**, **license.ini** and **zenon6.ini** must be modified using a text editor.

SETTINGS.XML

In the settings.xml file, you can configure:

- CodeMeter license server: This entry is needed by **license.ini**.
- Primary DNS suffix: For network forwarding.

To configure settings.xml:

1. Go to the ...\ path *[data path]*\.

2. Open the **settings.xml** file.
3. Go to the **<CodeMeterLicenseServer>** section.
   - Enter the name of the computer on which the license is stored between the Tags **<Address>** and **</Address>**.
4. Repeat this step if necessary for an additional server entry.
5. Switch to the **<Network>** section.
   - Enter the primary DNS suffix for your network between the Tags **<PrimaryDnsSuffix>** and **<PrimaryDnsSuffix>**.
     This is the part of your DNS entry without the hostname.
     If this entry remains empty, the default settings of the container will be used.
6. If necessary, you can configure the autostart of the Runtime in the container.
   To do this, enter the following service and program information between tags **<Autostart>** and **</Autostart>**.
   - **<Program>**: initiates the section for the start of the service.
   - **<Active></Active>**: activates the autostart for the service.
     Active: *true*
     Inactive: *false*
     For example: **<Active>***true***</Active>**.
   - **<Path></Path>**: path to the service.
     For example, **<Path>C:\zenonSetup\64\zenSysSrv.exe</Path>**.
   - **</Program>**: closes the section.
   - **<Program>**: initiates the section for the start of the program.
   - **<Active></Active>**: activates the autostart for the program similar to that of the service.
     For example: **<Active>***true***</Active>**.
   - **<Path></Path>**: path to the program.
     For example, **<Path>C:\zenonSetup\64\zenrt32.exe</Path>**.
   - **</Program>**: closes the section.
7. Save the file.

LICENSE.INI ENTER LICENSE

To be able to use zenon in a Docker container, you must provide a valid license and define the location where the license is saved.

Sources of information for:

- License: license certificate or license manager
- License storage location: Codemeter Web Admin

To license zenon for Docker:

1. Go to the ...\ path *[data path]\system*.
2. Open the **License.ini** file.
3. Enter the license information according to your license certificate:
   - **SERIAL0 =** : Serial number of your license.
     This entry is mandatory.
   - **SERIAL0_DESC =**: optional description of the use of the license.
   - **SERIAL0_LOCATION =**: License server address.
     This entry is mandatory.
4. Save the file.
5. Stop and restart the Docker container.
   Enter: **docker stop [container name]** and **docker start [container name]**

A licensed zenon is available to you in the Docker container.

ZENON6.INI

To specify the Runtime project:

1. Go to the ...\ path *[data path]\system*.
2. Open the **zenon6.ini**.
3. Edit the following entries:

   [PATH]

   VBF30= Enter the path to the workspace here.
   Example: **C:\zenondata\projects\Workspace1**

   [DEFAULT]

   DEFANWENDUNG30= Enter the project name here.

5.4.1 Runtime Docker with name resolution

For the zenon Runtime server to run and be accessible, a Docker container is needed which can be accessed from the outside via the container hostname. Should access take place across domains, the fully qualified hostname (FQDN) must be used.

Note: According to Windows conventions, hostnames may not contain more than 15 characters.

REQUIREMENTS

You need:

- a previously loaded zenon Runtime Docker image.
- a zenon Runtime that has already been configured (on page 9) for Docker.
  First and foremost, **settings.xml** and **license.ini** must be configured.

CONFIGURE A VIRTUAL EXTERNAL SWITCH IN HYPER-V

You need an external network switch in your Hyper-V server. Check this in the **Virtual Switch Manager** setting on the Hyper-V server. If no external switch is available, you need to create one. This external switch is used by the Docker container.

SET THE PRIMARY DNS SUFFIX IN THE DOCKER CONTAINER

If you would like to use a server name with a DNS suffix, you must communicate the **Primary DNS Suffix** to the Docker container. For example: **zenRT001.domain.internal**. If the **Primary DNS Suffix** is not set, the zenon Runtime does not recognize that it acts as the server. This is also the case if the Docker container can be pinged from the outside.
You can configure the suffix in the **Settings.xml** (on page 9) in the Tag **<Settings> <Network> <PrimaryDnsSuffix>**.

CREATE A CONTAINER WITH AN EXTERNAL SWITCH

You can create a container with an external switch using the following command:

docker run -d -v [data path]c:\zenondata --network [name of external switch] --hostname [unique host name] --name [container name] copadata/zenon-runtime820-windows

EXAMPLE

A container should be created that meets the following conditions:

- uses the virtual Hyper-V switch **zrtExtSwitch**
- sets the hostname to **zenRT001**
- can manage the container with the name **zenon**

Command: **PS> docker run -d -v C:\zenondata:C:\zenondata --network zrtExtSwitch --hostname zenRT001 --name zenon copadata/zenon-runtime820-windows**

SET UP THE ZENON SERVER

To set up the zenon server :

- use the previously configured hostname **zenRT001.domain.internal** as the server name in your zenon network project.
- You can use the same hostname for the zenon remote transport.
- After the Runtime files have been transferred and the zenon Runtime in the Docker container has been started, zenon clients can access the server via **zenRT001.domain.internal**.

TROUBLESHOOTING

In the case of errors:

- Make sure that the zenon Runtime recognizes itself as the server.
  To do this, check the log files.
- Check the network connection between the client and the server.

REMOTE TRANSPORT: CONTAINER RESTART PREVENTS CONNECTION

When a Docker container is restarted, it receives a new IP address. However, the DNS cache for the hostname of the container is not updated at the same time. Connections with the container by hostname are thus no longer possible.

Solutions:

1. In the context of the same domain: Execute the **ipconfig /flushdns** command line.
2. In general: Wait until the DNS cache has been updated after the defined timeout.

5.4.2 zenon-Standard-Ports

Ports used by zenon by default:

Application	Standard port
Network Service	*1100*
Transport Service	*1101*
WEB Service Classic	*1102*
DB Service	*1103*
SQL Browser Service, (for distributed engineering in the Editor)	*1434*
zenAdminSrv.exe	*50777*
zenLicTransfer (License Transfer Service)	*50784*

Application	Standard port
Logging Service	*50780*
SNMP Trap Service	*50782*
WEB Service Tunneling	*8080*

5.5 A selection of commands for Docker

You need a number of commands for the configuration of zenon with Docker. You enter these via Windows PowerShell. The following table contains frequently needed commands.

You can also find information on this at https://docs.docker.com/engine/reference/commandline/docker/ (http://www.)

Command	Description
docker --version	Displays the Docker version information.
docker network ls	Lists all networks that the engine daemon recognizes. This includes networks that spread across several hosts in a cluster.
docker images	Lists all loaded Images.
docker run -d -v [data path]:c:\zenondata --name [container name] [image name]	Creates a container from a loaded Image.
docker run -d -v [data path]:c:\zenondata --network [name of switch] --hostname [host name] --name [container name] [image name]	Creates a container from a loaded Image including the forwarding of ports for the zenon network.
docker ps -a	Lists all containers including the stopped ones.
docker start [container name]	Starts a stopped container.
docker stop [container name]	Stops a started container. **Tip:** To stop all containers, use the command **$(docker ps -a -q) as [container name]**.
docker inspect [container name]	Displays container details including hostname and network configuration.
docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' [container name]	Outputs the static IP address of the container.

Command	Description
docker rm [container name]	Removes a container. **Tip:** To remove all containers, use the command **$(docker ps -a -q) as [container name]**.
docker exec -it [container name] powershell	Enables PowerShell commands to be executed in the Docker container.
docker system prune -a	This command removes: ▸ all stopped containers ▸ all networks that are not used by at least one container ▸ all images that are not linked to at least one container ▸ all build caches