



meshIQ Platform

Installation Guide

Version 11

Document Number: MP11.000

Document Title: meshIQ Platform Installation Guide

Document Release Date: August 2024

Document Number: MP11.000

Published by:

Research & Development

meshIQ

88 Sunnyside Blvd, Suite 101

Plainview, NY 11803

Copyright © 2024. All rights reserved. No part of the contents of this document may be produced or transmitted in any form, or by any means without the written permission of meshIQ.

Confidentiality Statement: The information within this media is proprietary in nature and is the sole property of meshIQ. All products and information developed by meshIQ are intended for limited distribution to authorized meshIQ employees, licensed clients, and authorized users. This information (including software, electronic and printed media) is not to be copied or distributed in any form without the expressed written permission from meshIQ.

Acknowledgements: The following terms are trademarks of meshIQ in the United States or other countries or both: AutoPilot, AutoPilot M6, M6 Web Server, M6 Web Console, M6 for WMQ, MQControl, Navigator, XRay.

Docker and the Docker logo are trademarks or registered trademarks of Docker, Inc. in the United States and/or other countries. Docker, Inc. and other parties may also have trademark rights in other terms used herein.

The following terms are trademarks of the IBM Corporation in the United States or other countries or both: IBM, MQ, WebSphere MQ, WIN-OS/2, AS/400, OS/2, DB2, Informix, AIX, and z/OS.

Java, J2EE, and the Java Logos are trademarks of Sun Microsystems Inc. in the United States or other countries, or both.

InstallAnywhere is a trademark or registered trademark of Flexera Software, Inc.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>), including Derby Database Server. The Jakarta Project" and "Tomcat" and the associated logos are registered trademarks of the Apache Software Foundation.

Intel, Pentium and Intel486 are trademarks or registered trademarks of Intel Corporation in the United States, or other countries, or both.

Microsoft, Windows, Windows NT, Windows XP, the Windows logos, Microsoft SQL Server, and Microsoft Visual SourceSafe are registered trademarks of the Microsoft Corporation.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Mac, Mac OS, and Macintosh are trademarks of Apple Computer, Inc., registered in the U.S. and other countries.

"Linux" and the Linux Logos are registered trademarks of Linus Torvalds, the original author of the Linux kernel. All other titles, applications, products, and so forth are copyrighted and/or trademarked by their respective authors.

Oracle, Java, and MySQL are registered trademarks of Oracle and/or its affiliates.

Other company, product, and service names may be trademarks or service marks of others.

Table of Contents

CHAPTER 1: INTRODUCTION	1
1.1 HOW THIS GUIDE IS ORGANIZED	1
1.2 HISTORY OF THIS DOCUMENT	1
1.3 USER FEEDBACK	1
1.4 RELATED DOCUMENTS	1
1.5 RELEASE NOTES	2
1.6 INTENDED AUDIENCE	2
1.7 TECHNICAL SUPPORT	2
CHAPTER 2: ABOUT THE MESHIQ PLATFORM	3
CHAPTER 3: INSTALLING TYPICAL SERVERS FOR STREAMING	5
3.1 INSTALLING TYPICAL SERVERS FOR STREAMING USING DOCKER SWARM	5
3.1.1 <i>Download the compressed tar file</i>	5
3.1.2 <i>Docker Swarm initialization</i>	5
3.1.3 <i>Leave the Docker Swarm mode</i>	6
3.1.4 <i>Check information about services</i>	6
3.1.5 <i>Setting the number of replicas</i>	6
3.1.6 <i>Required Configuration Steps</i>	6
3.1.7 <i>Recommended Configuration Steps (Optional)</i>	6
3.1.8 <i>Optional Configuration (Changes are not Recommended Unless Needed)</i>	7
3.1.9 <i>meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)</i>	8
3.1.10 <i>meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)</i>	8
3.1.11 <i>Creating a meshIQ Platform Data Source</i>	10
3.2 INSTALLING TYPICAL SERVERS FOR STREAMING: HELM CHART FOR KUBERNETES	11
3.2.1 <i>Prerequisites</i>	11
3.2.2 <i>Pull the Helm Chart from the Repo</i>	11
3.2.3 <i>Set up and Run the meshIQ Platform on Your Kubernetes Cluster</i>	12
3.2.4 <i>Stop Services</i>	12
3.2.5 <i>Updating the Helm Chart</i>	12
3.2.6 <i>Log in to Track</i>	14
3.2.7 <i>Testing</i>	14
CHAPTER 4: MANAGING LARGER MESHIQ PLATFORM DEPLOYMENTS	15
4.1 INSTALLING A LARGER DEPLOYMENT USING DOCKER SWARM	15
4.1.1 <i>Download the compressed tar file</i>	15
4.1.2 <i>Docker Swarm initialization</i>	15
4.1.3 <i>Leave the Docker Swarm mode</i>	15
4.1.4 <i>Check information about services</i>	15
4.1.5 <i>Setting the number of replicas</i>	16
4.1.6 <i>Required Configuration Steps</i>	16
4.1.7 <i>Recommended Configuration Steps (Optional)</i>	16
4.1.8 <i>Optional Configuration (Changes are not Recommended Unless Needed)</i>	17
4.1.9 <i>meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)</i>	18
4.1.10 <i>meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)</i>	18
4.1.11 <i>Creating a meshIQ Platform Data Source</i>	20
4.2 INSTALLING A LARGER DEPLOYMENT: HELM CHART FOR KUBERNETES	21
4.2.1 <i>Prerequisites</i>	21
4.2.2 <i>Pull the Helm Chart from the Repo</i>	21

4.2.3	<i>Set up and Run the meshIQ Platform on Your Kubernetes Cluster</i>	22
4.2.4	<i>Stop Services</i>	22
4.2.5	<i>Updating the Helm Chart</i>	22
4.2.6	<i>Log in to Track</i>	24
4.2.7	<i>Testing</i>	24
CHAPTER 5: ADDING GRAFANA INTEGRATION TO AN EXISTING MESHIQ INSTALLATION (DSX) 25		
5.1	RUNNING GRAFANA INTEGRATION IN A DOCKER CONTAINER	25
5.2	ADDING GRAFANA INTEGRATION USING A DEPLOYMENT PACK	26
5.2.1	<i>Prerequisites</i>	26
5.2.2	<i>Installation steps</i>	27
5.2.3	<i>How to connect Grafana to the meshIQ DSX platform</i>	29
5.2.4	<i>Show, start, or stop meshIQ services</i>	29
5.2.5	<i>Setting the Installation Path (Optional)</i>	30
CHAPTER 6: INSTALLING A NEW GRAFANA INTEGRATION 31		
6.1	INSTALLING THE GRAFANA INTEGRATION IN A DOCKER CONTAINER	31
6.1.1	<i>Download the compressed tar file</i>	31
6.1.2	<i>Required Configuration Steps</i>	31
6.1.3	<i>Recommended Configuration Steps (Optional)</i>	31
6.1.4	<i>Optional Configuration (Changes are not Recommended Unless Needed)</i>	32
6.1.5	<i>meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)</i>	33
6.1.6	<i>meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)</i>	33
6.1.7	<i>Creating the meshIQ Platform Data Source</i>	35
6.2	INSTALLING THE GRAFANA INTEGRATION USING DOCKER SWARM	36
6.2.1	<i>Download the compressed tar file</i>	36
6.2.2	<i>Docker Swarm initialization</i>	36
6.2.3	<i>Leave the Docker Swarm mode</i>	36
6.2.4	<i>Check information about services</i>	36
6.2.5	<i>Setting the number of replicas</i>	36
6.3	INSTALLING THE GRAFANA INTEGRATION HELM CHART FOR KUBERNETES.....	37
6.3.1	<i>Prerequisites</i>	37
6.3.2	<i>Pull the Helm Chart from the Repo</i>	37
6.3.3	<i>Set up and Run the meshIQ Platform on Your Kubernetes Cluster</i>	38
6.3.4	<i>Stop Services</i>	38
6.3.5	<i>Updating the Helm Chart</i>	38
6.3.6	<i>Testing</i>	40
CHAPTER 7: INSTALLING THE MESHIQ PLATFORM FOR BASIC MANAGEMENT & MONITORING 41		
7.1	INSTALLING THE MESHIQ PLATFORM FOR BASIC MANAGEMENT AND MONITORING IN A DOCKER CONTAINER	41
7.1.1	<i>Download the compressed tar file</i>	41
7.1.2	<i>Required Configuration Steps</i>	41
7.1.3	<i>Recommended Configuration Steps (Optional)</i>	42
7.1.4	<i>Optional Configuration (Changes are not Recommended Unless Needed)</i>	42
7.1.5	<i>meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)</i>	43
7.1.6	<i>meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)</i>	43
7.1.7	<i>Creating a meshIQ Platform Data Source</i>	45

7.2	INSTALLING THE MESHIQ PLATFORM FOR BASIC MANAGEMENT & MONITORING USING DOCKER SWARM....	46
7.2.1	Download the compressed tar file	46
7.2.2	Docker Swarm initialization	46
7.2.3	Leave the Docker Swarm mode	46
7.2.4	Check information about services.....	46
7.2.5	Setting the number of replicas	47
7.3	INSTALLING THE MESHIQ PLATFORM FOR BASIC MANAGEMENT & MONITORING: HELM CHART FOR KUBERNETES	47
7.3.1	Prerequisites	47
7.3.2	Pull the Helm Chart from the Repo	47
7.3.3	Set up and Run the meshIQ Platform on Your Kubernetes Cluster.....	48
7.3.4	Stop Services	48
7.3.5	Updating the Helm Chart	48
7.3.6	Log in to Track	50
7.3.7	Testing.....	50
7.4	INSTALLING THE MESHIQ PLATFORM FOR BASIC MANAGEMENT AND MONITORING USING THE DEPLOYMENT PACK	51
7.4.1	Prerequisites for Installation	51
7.4.2	Setup steps	53
7.4.3	Setting the Installation Path (Optional).....	54
7.4.4	Access meshIQ Manage	55
7.4.5	Access meshIQ Track.....	55
CHAPTER 8: INSTALLING THE MANAGE COMPONENT.....		56
8.1	INSTALLING THE MANAGE COMPONENT IN A DOCKER CONTAINER.....	56
8.1.1	Download the compressed tar file	56
8.1.2	Required Configuration Steps.....	56
8.1.3	Recommended Configuration Steps (Optional)	56
8.1.4	Optional Configuration (Changes are not Recommended Unless Needed)	56
8.1.5	Wizard-driven Menu Script (Recommended)	57
8.1.6	Manual Lifecycle Management (Optional)	57
8.2	INSTALLING THE MANAGE COMPONENT USING DOCKER SWARM	59
8.2.1	Download the compressed tar file	59
8.2.2	Docker Swarm initialization	59
8.2.3	Leave the Docker Swarm mode	59
8.2.4	Check information about services.....	59
8.2.5	Setting the number of replicas	60
APPENDIX A: REFERENCES.....		61
A.1 MESHIQ DOCUMENTATION		61
APPENDIX B: TERMS AND PROJECT LAYOUTS.....		62
B.1 TERMS TO CONSIDER		62
B.2 PROJECT DIRECTORIES LAYOUTS: DOCKER DISTRIBUTIONS		63
B.3 PROJECT DIRECTORIES LAYOUTS: HELM CHART DISTRIBUTIONS		65
B.3.1 Helm Chart: Typical Servers for Streaming and Larger Deployments		65
B.3.2 Helm Chart: New Grafana Integration		70
B.3.3 Helm Chart: Basic Management and Monitoring		74

Chapter 1: Introduction

Welcome to the *meshIQ Platform Installation Guide*. This guide will introduce the user to the basic functionality of the platform and describe how to install its modules. Please review this guide carefully before installing the product.

The meshIQ Platform is available for installation in containers or through a deployment pack. Some modules are available to be installed using separate deployment packs.

1.1 How this Guide is Organized

- [Chapter 1:](#) Document information
- [Chapter 2:](#) Contains a brief functional description of the meshIQ Platform
- [Chapter 3:](#) Information on installation of typical servers for streaming
- [Chapter 4:](#) Information on installation of larger meshIQ Platform deployments
- [Chapter 5:](#) Information on installation of Grafana integration into an existing installation (DSX)
- [Chapter 6:](#) Information on installation of a new Grafana integration
- [Chapter 7:](#) Information on installation of the meshIQ Platform for Basic Management and Monitoring
- [Chapter 8:](#) Information on installation of the Manage component

1.2 History of this Document

Table 1.2-A. History of this Document			
Release Date	Document Number	Product Version	Summary
August 2024	MP11.000	11.x	Initial release

1.3 User Feedback

meshIQ encourages all users of the meshIQ Platform to submit comments, suggestions, corrections, and recommendations for improvement of all documentation. Please send your comments via email to: mysupport@meshIQ.com. You will receive a response, along with status of any proposed change, update, or correction.

1.4 Related Documents

The complete listing of related and referenced documents is listed in [Appendix A](#) of this guide.

1.5 Release Notes

Refer to the [meshIQ Platform release notes articles](#) in the Resource Center.

1.6 Intended Audience

This guide is intended for users of the meshIQ Platform. Capabilities of the platform include solutions for many problem-solving roles across an organization, including but not limited to middleware teams, application owners, application support, developers, DevOps, and shared services.

1.7 Technical Support

If you need additional technical support, you can contact meshIQ by telephone or by email. To contact meshIQ technical support by telephone, call **800-963-9822 ext. 1**, if you are calling from outside the United States, dial **001-516-801-2100**. To contact meshIQ technical support by email, send a message to mysupport@meshiq.com. To access the meshIQ mySupport Site (user ID and password required), go to <http://mysupport.meshiq.com/>. Contact your local meshIQ Platform administrator for further information.

Chapter 2: About the meshIQ Platform

The meshIQ platform is the core engine that supports observability, management, and tracking. A typical installation offers observability, management, and tracking. meshIQ's platform includes a suite of services, experts, a job scheduler, a Grafana integration, and the configuration capabilities to support visibility and control over your entire integration infrastructure.

meshIQ's Platform can be scaled to accommodate varying levels of complexity.

OBSERVABILITY

Many large banking, insurance, healthcare, and retail companies have long relied on technology from meshIQ to keep their systems operating efficiently and circumvent issues before they become customer problems.

At its most basic level, observability means you're provided with real-time access to all experts, facts, and policies data. meshIQ Manage workgroup server data can also be collected. Grafana integration provides both monitoring and alerting for your business environment. Write queries to gain insight into the collected data. Then configure and customize alerts for members of your organization.

Two services facilitate users' access to meshIQ analytic and tracking data, expert metrics, and administrative data through jKQL queries:

- a service URL (a REST endpoint is that is available as an expert)
- an autocomplete service URL. Autocomplete helps users complete jKQL queries by offering suggestions as they type.

Real-time, end-to-end monitoring enables users to proactively monitor system-wide performance, congestion, and latency using current and historical statistics. Maximize efficiency by diagnosing system and application bottlenecks.

meshIQ empowers your organization to “shift left,” detecting problems early by putting detailed health policies in place. Corrective actions—from commands to shell scripts, to executables—can be automated based on events.

MANAGEMENT

From legacy products to the newest PubSub and event-driven technologies, meshIQ supports configuration of multiple platforms. Simplify configuration management by easily comparing configurations and applying them consistently across objects.

The meshIQ Platform supports the principle of governance by providing you with granular access controls based on users' assigned roles. Well over one thousand specific rights control users' actions as well as the objects that they have permission to see and perform operations on.

When configuration changes turn out to be inaccurate or cause unexpected behavior, you can easily roll them back as needed, for remediation.

When you need to dig into the details to solve problems, introspection is key. With meshIQ Manage, advanced filtering allows you to narrow down lists of objects by a single attribute or multiple attributes by combining compare operations. You can even use variables so you can set values on the fly. Create filters for your own use, or share them with members of one or more groups.

In addition to attribute filters, meshIQ Manage offers comprehensive message search capability. Search by message properties, message header properties, or message data. When issues with message transmission are diagnosed, reroute messages to address integration slowdowns and bottlenecks.

With the meshIQ Platform you can take action by weaving integration in DevOps, speeding up integration deployments with scheduled actions.

TRACKING

Implement granular access controls by governing access to data. Two types of tokens are available for this purpose: streaming and access.

- Administrators can generate streaming tokens to grant others the right to stream data to a repository and to configure restrictions on streaming.
- Both administrators and other users can generate access tokens to grant access to a repository or organization and perform actions on specified data and objects. For each repository, the token options determine which items can be managed.

Once streaming access controls are in place, you can begin collecting data and monitoring logs.

Trace and track messages and message processing events by streaming data in real time. Diagnose issues by drilling down to problem areas in the data. Speed up root-cause analysis by diving into the data and detecting anomalies. Train models and forecast future values for data points with Machine Learning.

Gain intelligence about—and insight into—otherwise unrelated events and activities when they are stitched together based on overlapping correlators. Streamed data is analyzed based on stitching results, including Set mapping and Relative determinations.

Chapter 3: Installing Typical Servers for Streaming

In a typical configuration, data from the meshIQ Platform is streamed (creating a stream data source).

Streamed data and facts are stored as events and snapshots in the Solr repository. When a user asks for logs, activities, and so on, data is retrieved.

Since streaming requires more infrastructure, this configuration requires the following additional resources:

- Additional services.
- Three Solr instances. These support any additional data being streamed from external sources. An external data source is one that is not hosted in Solr but through some other interface. Three external data sources are available:
 1. meshIQ Manage workgroup server
 2. meshIQ Observe facts produced by experts
 3. meshIQ Observe Policies (Policies, sensors, sensor facts).

A typical meshIQ Platform distribution is available for Docker Swarm and Kubernetes.

3.1 Installing Typical Servers for Streaming using Docker Swarm



Swarm mode is required and must be active to run the meshIQ Platform in Docker Swarm.

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

3.1.1 Download the compressed tar file

1. Download the meshiq_swarm_typical_v11.1.X.docker.tgz file.
2. Extract it to a location of your choosing.

3.1.2 Docker Swarm initialization

First, Docker Swarm needs to be initialized. It has already been installed with Docker. However, to start Docker Swarm, run this command:

```
docker swarm init
```

To make sure that Docker Swarm is active, you can run this command:

```
docker info | grep -i swarm
```

3.1.3 Leave the Docker Swarm mode

To leave the Docker Swarm run this command:

```
docker swarm leave -f
```

3.1.4 Check information about services

To check for more information about the service, run this command:

```
docker service inspect --pretty SERVICE_NAME
```

Change SERVICE_NAME to your actual service name.

To see which nodes are running the service:

```
docker service ps SERVICE_NAME
```

To see all services:

```
docker service ls
```

3.1.5 Setting the number of replicas

Docker Swarm has a tremendous advantage over Docker Compose. Swarm is capable of managing the number of running instances and the container's resilience against faults. You can set the number of service instances in the set-env.sh script by changing the number of replicas.

For example:

```
export CM_KAFKA_SWARM_REPLICAS=1
```

The number of Kafka Connection Manager replicas is currently set to 1, although this setting can be adjusted as needed. Additionally, each configuration file specifies how many times the container will attempt to restart after encountering problems.

3.1.6 Required Configuration Steps

In the bin/set-env.sh shell script, set the DOCKER_HOST_IP variable to the external IP of the machine that is running docker.



The IP address can also be set when you create meshIQ components. If the 'DOCKER_HOST_IP' variable in 'set-env.sh' is left as 'localhost', then after you choose to create meshIQ components using the menu, you will have an opportunity to select an IP address from a list or to set a specific one.

3.1.7 Recommended Configuration Steps (Optional)

1. In the bin/set-env.sh shell script, pick the PROJECT_VERSION variable that matches the meshIQ version you need. It can be defined using a major.minor pattern (e.g.,

- 10.3 or 10.4), a major pattern (e.g., 10 or 11) or 'latest' (to have the latest Docker images used for meshIQ Components).
2. In the `config/meshiq/DB_USER` file, set the database user to be used by meshIQ components.
3. In the `config/meshiq/DB_USER_PWD` file, set the database user password to be used by meshIQ components.
4. In the `config/meshiq/SITE_KEY_PWD` file, set the SSL keystore password.

3.1.8 Optional Configuration (Changes are not Recommended Unless Needed)

1. The `config/license/meshiq.lic` file contains the license used by meshIQ WGS, Domain, and Track.
2. `config/cep-wgs/registry.xml` is a meshIQ WGS object registry configuration file.
3. `config/domain/registry.xml` is a meshIQ Domain service object registry configuration file.
4. `config/meshiq-db/meshiq.cnf` is a WGS bound database (Postgres) configuration file.
5. The `meshiq` folder contains docker compose service definition `*.yaml` files.

Configuration parts to consider changing are as follows:

- docker host machine port bindings (if your environment already used some of them, and you get port bind errors running the meshIQ Platform).
 - `manage-ui` environment variables `SSL_KEY_PATH`, `SSL_KEY_PWD`, and `SSL_KEY_ALIAS` (if you need Tomcat to enable HTTPS protocol support).
 - `grafana` environment variables `GF_SERVER_PROTOCOL`, `GF_SERVER_CERT_FILE` and `GF_SERVER_CERT_KEY` (if you need to enable HTTPS protocol support).
6. In `bin/set-env.sh` you can choose the DB initialization actions policy. (The default is demand.)
 - `skip` - skips all DB initialization actions.
 - `create` - performs DB (re)create action.

WARNING: this will destroy your existing database!

- `update` - performs only DB update action.



NOTE

This will fail if the database does not exist.

- `demand` - performs DB create/update depending on current environment demand.
7. In the `bin/set-env.sh` shell script, you can set following configurations to access an external database:
 - `USE_EXTERNAL_DB` - indicates whether an external database will be used (default is **false**).
 - `PROJECT_DB` - database URL (default is **jdbc:postgresql://meshiq-db:5432**).



NOTE

Set the external database user and password in the config/meshiq/DB_USER and config/meshiq/DB_USER_PWD files.

- PROJECT_DB_DRIVER - database driver (default is **org.postgresql.Driver**).
8. In bin/set-env.sh you can set meshIQ Domain information to access an external Domain:
 - AP_DOMAIN_HOST_IP - meshIQ Domain server host/ip (default is **domain**).
 - DOMAIN_SERVER_PORT - meshIQ Domain server port (default is **2323**).
 - DOMAIN_SERVER_NAME - meshIQ Domain server name (default is **DOMAIN_SERVER**).
 - DOMAIN_NAME - meshIQ Domain name (default is **DOMAIN**).
 9. meshIQ bound Solr database configuration:
 - In the ADMINISTRATOR_PWD file, set the Administrator user password for the external Domain.

3.1.9 meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)



NOTE

We strongly recommend that you **do not use** Git Bash, or Mintty in general, to run shell scripts, as this may produce unexpected results.

After setting up *set-env.sh*, you can use the wizard-like menu script *meshiq.sh* to manage the meshIQ Platform lifecycle:

```
./bin/meshiq.sh
```

Or to use a more advanced menu, use *meshiq_c.sh*:

```
./bin/meshiq_c.sh
```

3.1.10 meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)

3.1.10.1 Initial startup: Creates and runs application containers

1. To run the base meshIQ Platform component containers, run the *up_meshiq.sh* shell script.
2. To run any set of connection manager containers as needed, run these shell scripts:
 - *up_cm_mq.sh* - to run IBM MQ connection manager.
 - *up_cm_kafka.sh* - to run Kafka connection manager.
 - *up_cm_ems.sh* - to run EMS (Enterprise Message Service/JMS) connection manager.
 - *up_cm_ace-iib.sh* - to run IBM ACE/IIB connection manager.
 - *up_cm_solace.sh* - to run Solace connection manager.
 - *up_cm_rabbitmq.sh* - to run RabbitMQ connection manager.

3.1.10.2 Migrating existing volumes to new app version

If your meshIQ Platform version tag is defined using the major or major.minor pattern, the volume versions are migrated during the upgrade process. That is because Docker volume naming is bound to the `PROJECT_NAME` variable defined in the `set-env.sh` shell script, which is a combination of the project ID and version.

The dedicated `migrate-volumes.sh` shell script is available to migrate volumes from the previous app version to the new app version. This script runs automatically when you run the `up_meshiq.sh`, `start_all.sh` or `start_wgs.sh` shell scripts.

If your meshIQ Platform version tag is defined as `latest`, there is no need to migrate volume versions because the version tag value does not change.

3.1.10.3 Stopping running containers



These procedures stop running containers without destroying them. For instructions on how to destroy containers, see Permanent shutdown.

1. To stop running meshIQ Platform containers, run the `stop_all.sh` shell script.
2. To stop all running meshIQ Platform Connection manager containers, run the `stop_cms.sh` shell script.

3.1.10.4 Starting any available stopped application container

1. To start all meshIQ Platform containers that have been created, run the `start_all.sh` shell script.
2. To start the set of meshIQ Platform Connection manager containers that have been created, run the `start_cms.sh` shell script.

3.1.10.5 Restarting: Stops running application containers and starts them again

1. To restart running meshIQ Platform containers, run the `restart_all.sh` shell script.
2. To restart the set of running meshIQ Platform Connection manager containers, run the `restart_cms.sh` shell script.
3. To restart a running meshIQ Platform WGS service container, run the `restart_wgs.sh` shell script. For your convenience, it restarts the meshIQ WGS service container along with all running Connection manager containers.

3.1.10.6 Permanent shutdown

**IMPORTANT!**

These instructions permanently destroy application containers. To merely stop the containers without destroying them, see Stopping running containers.

1. To shut down any running meshIQ Platform Connection manager container, run the `down_cms.sh` shell script.
2. To shut down any running meshIQ Platform base component and Connection manager containers, run the `down_all.sh` shell script.
3. (Optional.) You can remove (destroy) persistent volumes used by the meshIQ Platform by running the `remove-volumes.sh` shell script.

3.1.11 Creating a meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
 - Username: admin
 - Password: admin
3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
 - **Name:** Choose a name for your data source.
 - **Service URL:** Set the URL to `http://ds-api:8080/ds-api/jkq1?` (default meshIQ Service URL).
`http://ds-api:8080/ds-api/jkq1?`
 - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.
`grafana-default`
 - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
 - **Autocomplete Service URL:** Set the URL to `http://cep-service:7580` (default Autocomplete Service URL).
`http://cep-service:7580`
8. Click **Save & Test** to validate the data source.

3.2 Installing Typical Servers for Streaming: Helm Chart for Kubernetes

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

3.2.1 Prerequisites

- [Docker](#)
- Kubernetes: [Docker Desktop bundled](#), [minikube](#), [Azure](#) or [AWS](#) Kubernetes service, [RH OpenShift](#)
- [Helm](#)

3.2.2 Pull the Helm Chart from the Repo

- Add meshIQ helm charts repo (**this is only required the first time**):
 - production repo

```
helm repo add meshiq-prod
https://containercomponents.s3.amazonaws.com/helm/charts
```
 - development repo

```
helm repo add meshiq-dev
https://containercomponents.s3.amazonaws.com/helm/charts/dev
```
- Update meshIQ helm repo index (we recommend that this be done before every pull to fetch new chart versions):

```
helm repo update
```
- Pull helm chart from repo:
 - Prod grade chart

```
helm pull meshiq-prod/meshiq-typical --untar
```
 - Dev grade chart

```
helm pull meshiq-dev/meshiq-typical --untar
```


3.2.3 Set up and Run the meshIQ Platform on Your Kubernetes Cluster

- Make these changes to the `values.yaml` file:
 - Replace `<YOUR-CLUSTER-PUBLIC-IP>` with your Kubernetes cluster public IP or your PC's IP address, if running Kubernetes cluster locally.



DO NOT USE `localhost` or `127.0.0.1`. On containers, it does not resolve to your PC's address!

NOTE

- After changing the IP address, run this command:

```
helm install meshiq meshiq-typical/
```
- You can also set the IP address when installing the helm chart by running this command:

```
helm install meshiq meshiq-typical/ --set global.service.external.ip=<YOUR-CLUSTER-PUBLIC-IP>
```

3.2.4 Stop Services

To stop services, run this command:

```
helm uninstall meshiq
```

3.2.5 Updating the Helm Chart

When a new helm chart version is released, you can upgrade it. The chart upgrade can be done like this:

1. Update helm repository index:

```
helm repo update
```
2. List all available chart versions:

```
helm search repo meshiq -l
```
3. Make a backup of your current `meshiq-typical` chart directory:

```
cp -r ./meshiq-typical meshiq-typical-back
```
4. Pull the new version of the helm chart:

```
helm pull <chart_name> --untar
```



You can add `--version <version>` argument to pull a specific version of the chart:

```
helm pull <chart_name> --untar --version <version>
```

NOTE

5. Restore executable mode for chart bundled shell scripts:

```
chmod +x ./meshiq-typical/scripts/*.sh
```

6. Merge the values.yaml of the new helm chart ./meshiq-typical/values.yaml with one on your backup ./meshiq-typical-back/values.yaml.
7. Update chart:

```
helm upgrade --install meshiq meshiq-typical/
```
8. See changed pods get terminated and started again.

```
kubectl get pod --watch
```

3.2.5.1 Creating the meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
 - Username: admin
 - Password: admin



If password was changed use the one set in the Values.yaml file as grafana.adminPassword.

3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
 - **Name:** Choose a name for your data source.
 - **Service URL:** Set the URL to `http://ds-api:8080/ds-api/jkql?` (default meshIQ Service URL).
`http://ds-api:8080/ds-api/jkql?`
 - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.
`grafana-default`
 - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
 - **Autocomplete Service URL:** Set the URL to `http://cep-service:7580` (default Autocomplete Service URL).
`http://cep-service:7580`
8. Click **Save & Test** to validate the data source.

3.2.5.2 Changing the Number of Shards

When the collection is too large for one node, it could be broken up and stored in sections by creating multiple shards. So, if you want to change the number of shards, you can do it in the `values.yaml` file.

- `solr.replicas` - number of Solr nodes (by default it is 3)
- `cep-service.solrShards` - number of Solr shards (by default it is 4)
- `cep-service.solrDbRep1` - number of Solr replications (by default it is 2)
- `cep-service.solrDbShardsPerNode` - number of Solr shards per node (by default it is 3)

3.2.5.3 Changing Solr Username and Password

If you would like to change a Solr user configuration, this change should be made in the [solr-secret.yaml](#) file.

- To change `username` - set a new value for `solr-user`. Write a new username encoded in base64.
- To change `password` - set a new value for `solr-pwd`. Write a new password encoded in base64.

3.2.6 Log in to Track

To access HTTP enabled Track in browser: `http://<YOUR-CLUSTER-PUBLIC-IP>:8081/track`.

To access HTTPS enabled Track in browser: `https://<YOUR-CLUSTER-PUBLIC-IP>:8441/track`.

3.2.7 Testing

To ensure the correctness of *Helm charts*, a set of tests has been included. To run the tests, execute the following command:

```
helm test meshiq
```

After running this command, you will see a series of messages regarding specific test outcomes. Information about the starting and completion times, as well as phases of each specific test, will be provided. If the phase is *successful*, you can be confident that everything is working as expected. In other cases, further investigation needs to be done.

meshiq is the release name in this case, so if you have written a different name, you should change it accordingly.

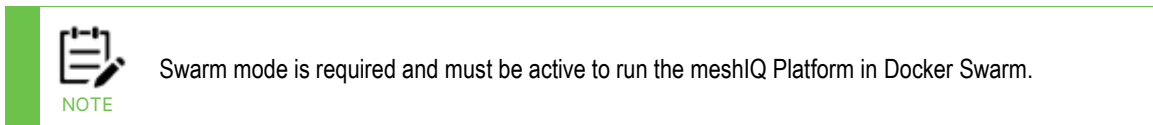
Chapter 4: Managing Larger meshIQ Platform Deployments

Large configurations are equipped for additional streaming and Machine Learning to detect anomalies and forecast trends.

At this level, an installation may require more than 3 Solr instances, as many as 12 services, and additional instances of Kafka and ActiveMQ.

A large meshIQ Platform distribution is available for Docker Swarm and Kubernetes.

4.1 Installing a Larger Deployment using Docker Swarm



If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

4.1.1 Download the compressed tar file

1. Download the `meshiq_swarm_big_v11.1.X.docker.tgz` file.
2. Extract it to a location of your choosing.

4.1.2 Docker Swarm initialization

First, Docker Swarm needs to be initialized. It has already been installed with Docker. However, to start Docker Swarm, run this command:

```
docker swarm init
```

To make sure that Docker Swarm is active, you can run this command:

```
docker info | grep -i swarm
```

4.1.3 Leave the Docker Swarm mode

To leave the Docker Swarm run this command:

```
docker swarm leave -f
```

4.1.4 Check information about services

To check for more information about the service, run this command:

```
docker service inspect --pretty SERVICE_NAME
```

Change `SERVICE_NAME` to your actual service name.

To see which nodes are running the service:

```
docker service ps SERVICE_NAME
```

To see all services:

```
docker service ls
```

4.1.5 Setting the number of replicas

Docker Swarm has a tremendous advantage over Docker Compose. Swarm is capable of managing the number of running instances and the container's resilience against faults. You can set the number of service instances in the `set-env.sh` script by changing the number of replicas.

For example:

```
export CM_KAFKA_SWARM_REPLICAS=1
```

The number of Kafka Connection Manager replicas is currently set to 1, although this setting can be adjusted as needed. Additionally, each configuration file specifies how many times the container will attempt to restart after encountering problems.

4.1.6 Required Configuration Steps

In the `bin/set-env.sh` shell script, set the `DOCKER_HOST_IP` variable to the external IP of the machine that is running docker.



NOTE

The IP address can also be set when you create meshIQ components. If the `'DOCKER_HOST_IP'` variable in `'set-env.sh'` is left as `'localhost'`, then after you choose to create meshIQ components using the menu, you will have an opportunity to select an IP address from a list or to set a specific one.

4.1.7 Recommended Configuration Steps (Optional)

1. In the `bin/set-env.sh` shell script, pick the `PROJECT_VERSION` variable that matches the meshIQ version you need. It can be defined using a major.minor pattern (e.g., 10.3 or 10.4), a major pattern (e.g., 10 or 11) or 'latest' (to have the latest Docker images used for meshIQ Components).
2. In the `config/meshiq/DB_USER` file, set the database user to be used by meshIQ components.
3. In the `config/meshiq/DB_USER_PWD` file, set the database user password to be used by meshIQ components.
4. In the `config/meshiq/SITE_KEY_PWD` file, set the SSL keystore password.

4.1.8 Optional Configuration (Changes are not Recommended Unless Needed)

1. The config/license/meshiq.lic file contains the license used by meshIQ WGS, Domain, and Track.
2. config/cep-wgs/registry.xml is a meshIQ WGS object registry configuration file.
3. config/domain/registry.xml is a meshIQ Domain service object registry configuration file.
4. config/meshiq-db/meshiq.cnf is a WGS bound database (Postgres) configuration file.
5. The meshiq folder contains docker compose service definition *.yaml files.

Configuration parts to consider changing are as follows:

- docker host machine port bindings (if your environment already used some of them, and you get port bind errors running the meshIQ Platform).
 - manage-ui environment variables SSL_KEY_PATH, SSL_KEY_PWD, and SSL_KEY_ALIAS (if you need Tomcat to enable HTTPS protocol support).
 - grafana environment variables GF_SERVER_PROTOCOL, GF_SERVER_CERT_FILE and GF_SERVER_CERT_KEY (if you need to enable HTTPS protocol support).
6. In bin/set-env.sh you can choose the DB initialization actions policy. (The default is demand.)
 - skip - skips all DB initialization actions.
 - create - performs DB (re)create action.

WARNING: this will destroy your existing database!

- update - performs only DB update action.



This will fail if the database does not exist.

NOTE

- demand - performs DB create/update depending on current environment demand.
7. In the bin/set-env.sh shell script, you can set following configurations to access an external database:
 - a. USE_EXTERNAL_DB - indicates whether an external database will be used (default is **false**).
 - b. PROJECT_DB - database URL (default is **jdbc:postgresql://meshiq-db:5432**).



Set the external database user and password in the config/meshiq/DB_USER and config/meshiq/DB_USER_PWD files.

NOTE

- c. PROJECT_DB_DRIVER - database driver (default is **org.postgresql.Driver**).
8. In bin/set-env.sh you can set meshIQ Domain information to access an external Domain:
 - a. AP_DOMAIN_HOST_IP - meshIQ Domain server host/ip (default is **domain**).
 - b. DOMAIN_SERVER_PORT - meshIQ Domain server port (default is **2323**).

- c. DOMAIN_SERVER_NAME - meshIQ Domain server name (default is **DOMAIN_SERVER**).
 - d. DOMAIN_NAME - meshIQ Domain name (default is **DOMAIN**).
9. meshIQ bound Solr database configuration:
 - a. In the ADMINISTRATOR_PWD file, set the Administrator user password for the external Domain.

4.1.9 meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)



We strongly recommend that you **do not use** Git Bash, or Mintty in general, to run shell scripts, as this may produce unexpected results.

After setting up *set-env.sh*, you can use the wizard-like menu script *meshiq.sh* to manage the meshIQ Platform lifecycle:

```
./bin/meshiq.sh
```

Or to use a more advanced menu, use *meshiq_c.sh*:

```
./bin/meshiq_c.sh
```

4.1.10 meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)

4.1.10.1 Initial startup: Creates and runs application containers

1. To run the base meshIQ Platform component containers, run the *up_meshiq.sh* shell script.
2. To run any set of connection manager containers as needed, run these shell scripts:
 - *up_cm_mq.sh* - to run IBM MQ connection manager.
 - *up_cm_kafka.sh* - to run Kafka connection manager.
 - *up_cm_ems.sh* - to run EMS (Enterprise Message Service/JMS) connection manager.
 - *up_cm_ace-iib.sh* - to run IBM ACE/IIB connection manager.
 - *up_cm_solace.sh* - to run Solace connection manager.
 - *up_cm_rabbitmq.sh* - to run RabbitMQ connection manager.

4.1.10.2 Migrating existing volumes to new app version

If your meshIQ Platform version tag is defined using the major or major.minor pattern, the volume versions are migrated during the upgrade process. That is because Docker volume

naming is bound to the `PROJECT_NAME` variable defined in the `set-env.sh` shell script, which is a combination of the project ID and version.

The dedicated `migrate-volumes.sh` shell script is available to migrate volumes from the previous app version to the new app version. This script runs automatically when you run the `up_meshiq.sh`, `start_all.sh` or `start_wgs.sh` shell scripts.

If your meshIQ Platform version tag is defined as `latest`, there is no need to migrate volume versions because the version tag value does not change.

4.1.10.3 Stopping running containers



These procedures stop running containers without destroying them. For instructions on how to destroy containers, see Permanent shutdown.

1. To stop running meshIQ Platform containers, run the `stop_all.sh` shell script.
2. To stop all running meshIQ Platform Connection manager containers, run the `stop_cms.sh` shell script.

4.1.10.4 Starting any available stopped application container

1. To start all meshIQ Platform containers that have been created, run the `start_all.sh` shell script.
2. To start the set of meshIQ Platform Connection manager containers that have been created, run the `start_cms.sh` shell script.

4.1.10.5 Restarting: Stops running application containers and starts them again

1. To restart running meshIQ Platform containers, run the `restart_all.sh` shell script.
2. To restart the set of running meshIQ Platform Connection manager containers, run the `restart_cms.sh` shell script.
3. To restart a running meshIQ Platform WGS service container, run the `restart_wgs.sh` shell script. For your convenience, it restarts the meshIQ WGS service container along with all running Connection manager containers.

4.1.10.6 Permanent shutdown



IMPORTANT!

These instructions permanently destroy application containers. To merely stop the containers without destroying them, see Stopping running containers.

1. To shut down any running meshIQ Platform Connection manager container, run the `down_cms.sh` shell script.
2. To shut down any running meshIQ Platform base component and Connection manager containers, run the `down_all.sh` shell script.
3. (Optional.) You can remove (destroy) persistent volumes used by the meshIQ Platform by running the `remove-volumes.sh` shell script.

4.1.11 Creating a meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
 - Username: admin
 - Password: admin
3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
 - **Name:** Choose a name for your data source.
 - **Service URL:** Set the URL to `http://ds-api:8080/ds-api/jkq1?` (default meshIQ Service URL).
`http://ds-api:8080/ds-api/jkq1?`
 - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.
`grafana-default`
 - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
 - **Autocomplete Service URL:** Set the URL to `http://cep-service:7580` (default Autocomplete Service URL).
`http://cep-service:7580`
8. Click **Save & Test** to validate the data source.

4.2 Installing a Larger Deployment: Helm Chart for Kubernetes

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

4.2.1 Prerequisites

- [Docker](#)
- Kubernetes: [Docker Desktop bundled](#), [minikube](#), [Azure](#) or [AWS](#) Kubernetes service, [RH OpenShift](#)
- [Helm](#)

4.2.2 Pull the Helm Chart from the Repo

- Add meshIQ helm charts repo (**this is only required the first time**):
 - production repo

```
helm repo add meshiq-prod
https://containercomponents.s3.amazonaws.com/helm/charts
```
 - development repo

```
helm repo add meshiq-dev
https://containercomponents.s3.amazonaws.com/helm/charts/dev
```
- Update meshIQ helm repo index (we recommend that you do this before every pull to fetch new chart versions):

```
helm repo update
```
- Pull helm chart from repo:
 - Prod grade chart

```
helm pull meshiq-prod/meshiq-big --untar
```
 - Dev grade chart

```
helm pull meshiq-dev/meshiq-big --untar
```

4.2.3 Set up and Run the meshIQ Platform on Your Kubernetes Cluster

- Make these changes to the `values.yaml` file:
 - Replace `<YOUR-CLUSTER-PUBLIC-IP>` with your Kubernetes cluster public IP or your PC's IP address if running Kubernetes cluster locally.



NOTE

DO NOT USE `localhost` or `127.0.0.1`. On containers, these do not resolve to your PC's address!

- After changing the IP address, run this command:

```
helm install meshiq meshiq-big/
```
- You can also set the IP address when installing the helm chart by running this command:

```
helm install meshiq meshiq-big/ --set global.service.external.ip=<YOUR-CLUSTER-PUBLIC-IP>
```

4.2.4 Stop Services

To stop services, run this command:

```
helm uninstall meshiq
```

4.2.5 Updating the Helm Chart

When a new helm chart version is released, you can upgrade it. The chart upgrade can be done like this:

1. Update helm repository index:

```
helm repo update
```
2. List all available chart versions:

```
helm search repo meshiq -l
```
3. Make a backup of your current `meshiq-big` chart directory:

```
cp -r ./meshiq-big meshiq-big-back
```
4. Pull the new version of the helm chart:

```
helm pull <chart_name> --untar
```



NOTE

You can add the `--version <version>` argument to pull a specific version of the chart:

```
helm pull <chart_name> --untar --version <version>
```

5. Restore executable mode for chart bundled shell scripts:

```
chmod +x ./meshiq-big/scripts/*.sh
```

6. Merge values.yaml of the new helm chart ./meshiq-big/values.yaml with one on your backup ./meshiq-big-back/values.yaml.
7. Update chart:

```
helm upgrade --install meshiq meshiq-big/
```
8. See changed pods get terminated and started again.

```
kubectl get pod --watch
```

4.2.5.1 Creating the meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
 - Username: admin
 - Password: admin



If password was changed use the one set in the Values.yaml file as grafana.adminPassword.

3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **"Create a data source"** button in the top right-hand corner.
7. Provide the following information:
 - **Name:** Choose a name for your data source.
 - **Service URL:** Set the URL to `http://ds-api:8080/ds-api/jkql?` (default meshIQ Service URL).
`http://ds-api:8080/ds-api/jkql?`
 - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.
`grafana-default`
 - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
 - **Autocomplete Service URL:** Set the URL to `http://cep-service:7580` (default Auto-complete Service URL).
`http://cep-service:7580`
8. Click **"Save & Test"** to validate the data source.

4.2.5.2 Changing the Number of Shards

When the collection is too large for one node, it could be broken up and stored in sections by creating multiple shards. So, if you want to change the number of shards, you can do it in the `values.yaml` file.

- `solr.replicas` - number of Solr nodes (by default it is 3)
- `cep-service.solrShards` - number of Solr shards (by default it is 4)
- `cep-service.solrDbRep1` - number of Solr replications (by default it is 2)
- `cep-service.solrDbShardsPerNode` - number of Solr shards per node (by default it is 3)

4.2.5.3 Changing the Solr Username and Password

If you would like to change a Solr user configuration, this change should be made in the [solr-secret.yaml](#) file.

- To change `username` - set a new value for `solr-user`. Write a new username encoded in base64.
- To change `password` - set a new value for `solr-pwd`. Write a new password encoded in base64.

4.2.6 Log in to Track

To access HTTP enabled Track in browser: `http://<YOUR-CLUSTER-PUBLIC-IP>:8081/track`.

To access HTTPS enabled Track in browser: `https://<YOUR-CLUSTER-PUBLIC-IP>:8441/track`.

4.2.7 Testing

To ensure the correctness of *Helm charts*, a set of tests has been included. To run the tests, execute the following command:

```
helm test meshiq
```

After running this command, you will see a series of messages regarding specific test outcomes. Information about the starting and completion times, as well as phases of each specific test, will be provided. If the phase is *successful*, you can be confident that everything is working as expected. In other cases, further investigation needs to be done.

meshiq is the release name in this case, so if you have written a different name, you should change it accordingly.


Chapter 5: Adding Grafana Integration to an Existing meshIQ Installation (DSX)

For our current customers who want to use Grafana Integration alongside their current implementation in their existing environment, meshIQ offers data services express, or DSX. This distribution is being made available because the version 11 upgrade does not include data services. DSX runs independently on a separate server.

It is composed of two services that facilitate users' access to meshIQ analytic and tracking data, expert metrics, and administrative data through jKQL queries. These two services are as follows:

- a service URL (a REST endpoint is that is available as an expert)
- an autocomplete service URL. Autocomplete helps users complete jKQL queries by offering suggestions as they type.

5.1 Running Grafana Integration in a docker container



IMPORTANT!

The default port for Grafana, port 3000, is also the port used by the domain server. So if you install Grafana locally on the same machine as the domain server, you need to change the port for either Grafana or the Domain server. The port for the Domain server can be changed in [AP_HOME]/naming/node.properties:

```
property server.user.url.port = 3000
```

You can run Grafana integration in a Docker container. Follow these steps:

1. Pull the Grafana image:

```
docker pull meshiq/meshiq-grafana
```

2. Run the Grafana container:

```
docker run --name meshiq-grafana -d -p 3000:3000 meshiq/meshiq-grafana
```


3. (Optional.) Add a volume for data persistence:

```
docker run --name meshiq-grafana -v grafana-data:/var/lib/grafana -d -p 3000:3000 meshiq/meshiq-grafana
```

4. Follow the instructions in [section 5.2.3](#), How to connect Grafana to the meshIQ DSX platform.

5.2 Adding Grafana Integration using a Deployment Pack

Follow the steps below to get started with the meshIQ DSX setup.




Before running this procedure, please verify the prerequisites below.

IMPORTANT!

5.2.1 Prerequisites

Linux System

- Operating System: Linux 64-bit OS, Ubuntu, RedHat (or equivalent, e.g. Fedora, CentOS, Rocky Linux, etc.).
- CPU: Minimum 4 virtual CPUs
- Memory: Minimum 16 GB RAM
- File System: Minimum 100 GB free space, SSD preferred
- Recommended installation directory: /opt/meshIQ



Some steps below require root access.

NOTE

1. Create a Linux User ID.

- Create the Linux user ID (UID) and group which will be used to start all meshIQ services.
- The default user and group are meshiq:meshiq. This can be changed to any preferred product owner user account.
- The UID and group must own all files and directories under the installation directory, \$APIN_HOME.

2. Disable Swap.

We highly recommended that you turn off Swap within the operating system on all virtual machine servers running meshIQ.

To disable swap, perform the following:

- a) Identify configured swap devices and files with the following command:
`cat /proc/swaps`
- b) Turn off all swap devices and files with the following command:
`/swapoff -a`
- c) Comment out any swap matching references found in /etc/fstab.


3. Modify Limits.

Modify /etc/security/limits.conf and add the lines below for your user. Replace meshiq with the user that will start meshIQ processes.

```
meshiq      soft  nofile 65536
meshiq      hard  nofile 65536

meshiq      soft  nproc  65536
meshiq      hard  nproc  65536
```

4. Network ports:



IMPORTANT! The default port for Grafana, port 3000, is also the port used by the domain server. So if you install Grafana locally on the same machine as the domain server, you need to change the port for either Grafana or the Domain server. The port for the Domain server can be changed in [AP_HOME]/naming/node.properties:
property server.user.url.port = 3000

The following list contains all processes network ports that are involved in meshIQ DSX deployment.

```
DOMAIN SERVER (ATPNAMES): ..... 2323, 3000 (TCP)
CEP SERVER (ATPNODE): ..... 3007 (TCP)
ZOOKEEPER: ..... 9983 (TCP)
SOLR: ..... 8983 (TCP)
DSX Service URL..... 8580 (TCP)
DSX Autocomplete service URL..... 7580 (TCP)
```


5.2.2 Installation steps

These steps assume local or remote installation of all related components.

The DSX node package is a self-contained package for meshIQ-Grafana integration. Log in to the [meshIQ software repository](#) to download the latest version of DSX_X.X.X_DPackVx.x.tar.Z. A username and password are required.

1. Install the DSX_X.X.X_DPackVx.x.tar.Z package to the installation path of your choice.

Shell command: tar -zxvf DSX_X.X.X_DPackVx.x.tar.Z



NOTE This package contains the directory "meshiq" as a root directory.

2. First-time setup of meshIQ Platform DSX.

Shell command: cd InstallationPATH/sbin/

Shell command (sets domain server location): ./config.sh dsconn



The first time you run these commands, you may be asked to set the Installation Path.

NOTE

Follow the onscreen instructions.

Press the "enter" key to set the install path automatically.

Then rerun:

Shell command: `./config.sh dsconn`

Continue to follow the onscreen instructions; the `config.sh` script will guide you through initial setup.

The final step is to start the DSX node:

Shell command: `./start.sh all`

3. Grafana Plugin Integration.

Using the packaged `meshIQ` Data-source Plug-in:

- a) Install Grafana (if you have not already). Follow the instructions provided in the [official Grafana documentation](#) for your specific operating system.
- b) Obtain the packaged `meshIQ` data-source plug-in for Grafana. You can find it in the following folder of the DSX tar file:

`APIN_HOME/misc/plugins/`

Unpack and place the plugin in the Grafana plugins directory (e.g., `/var/lib/grafana/plugins`).



Remember to adjust the paths and configurations according to your specific setup.

NOTE

Shell command: `tar -zxvf grafana-meshiq-plugin.tar.gz`

- c) Edit the `grafana.ini` configuration file (usually located at `/etc/grafana/grafana.ini`).

Add plugin ID to the `allow_loading_unsigned_plugins` configuration option:
`allow_loading_unsigned_plugins=meshiq-datasource`

- d) Save the configuration file and restart Grafana using the appropriate method:
 - a. If you are using Grafana from meshIQ, navigate to `/opt/meshiq/grafana-xx/bin` and run the following:

```
./grafana-server --config etc/grafana/grafana.ini
OR
```

- b. If Grafana was locally installed using yum install (or apt-get), run the following:

```
sudo systemctl start grafana-server
```

- e) Log into the Grafana UI.
- f) To set the admin password in Grafana, run the following command from the bin directory:

```
./grafana-cli admin reset-admin-password <new password>
```

Now you are ready to use the `meshIQ` data source plug-in in Grafana.

Continue by following the instructions in the section below.

5.2.3 How to connect Grafana to the meshIQ DSX platform

If step 3 in the Installation Steps section above has been completed, then proceed to Step 1: Create a meshIQ data source section in the online “Set up meshIQ Grafana integration” article at <https://customers.meshiq.com/hc/en-us/articles/18761296349203-Set-up-meshIQ-Grafana-integration>. Be sure to choose the correct Service URL, Access Token, and Autocomplete Service URL based on the edition of the meshIQ Platform that you are installing (DSX).

5.2.4 Show, start, or stop meshIQ services

Package scripts are located in the APIN_HOME/SBIN directory.

5.2.4.1 Show meshIQ services

You can verify whether the meshIQ services are running by using the following commands:

```
Shell command: cd APIN_HOME/sbin
```

```
Shell command: ./show.sh
```

5.2.4.2 Start meshIQ services

You can start all installed services by using the following commands:

```
Shell command: cd APIN_HOME/sbin
```

```
Shell command: ./start.sh all
```

You can also start a specific meshIQ service.

To see the list of valid parameters for starting a specific service, use the following command:

```
Shell command: ./start.sh
```

5.2.4.3 Stop meshIQ services

You can stop all installed services by using the following commands:

Shell command: `cd APIN_HOME/sbin`

Shell command: `./stop.sh all`

You can also stop a specific meshIQ service.

To see the list of valid parameters for stopping a specific service, use the following command:

Shell command: `./stop.sh`

5.2.5 Setting the Installation Path (Optional)

5.2.5.1 Setting the Installation Path as an Environment Variable

To run startup scripts from anywhere, we recommend that you set environment variables `APIN_HOME` and `PATH` in the user profile.

Copy and paste these two lines toward the end of the user profile file.

```
export APIN_HOME=/opt/meshiq
export PATH=$PATH:$APIN_HOME/sbin
```

Change `/opt/meshiq` to the path to the root directory of your meshIQ installation.

5.2.5.2 Setting the Installation Path Manually

To set the installation path manually, edit `InstallationPath/sbin/apin_env.sh`

Change the line:

```
export APIN_HOME=ExistingPath
```

Replace `ExistingPath` with `NewPath`.

Chapter 6: Installing a New Grafana Integration

If you are new to the meshIQ Platform and want to visualize current data from your infrastructure and use Grafana for alerting and monitoring, the Grafana Integration is designed for your needs. It combines meshIQ core services and a data services REST endpoint with a user interface such as Grafana or Splunk for monitoring.

6.1 Installing the Grafana Integration in a Docker Container

This section describes required, recommended, and optional configuration steps for the meshIQ Platform Grafana Integration. It also includes wizard-based and manual lifecycle management instructions, and procedures for creating the meshIQ Platform Data Source.

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

6.1.1 Download the compressed tar file

1. Download the `meshiq_thin_v11.X.docker.tgz` file.
2. Extract it to a location of your choosing.

6.1.2 Required Configuration Steps

In the `bin/set-env.sh` shell script, set the `DOCKER_HOST_IP` variable to the external IP of the machine that is running docker.



NOTE

The IP address can also be set when you create meshIQ components. If the `'DOCKER_HOST_IP'` variable in `'set-env.sh'` is left as `'localhost'`, then after you choose to create meshIQ components using the menu, you will have an opportunity to select an IP address from a list or to set a specific one.

6.1.3 Recommended Configuration Steps (Optional)

1. In the `bin/set-env.sh` shell script, pick the `PROJECT_VERSION` variable that matches the meshIQ version you need. It can be defined using a major.minor pattern (e.g., 10.3 or 10.4), a major pattern (e.g., 10 or 11) or 'latest' (to have the latest Docker images used for meshIQ components).
2. In the `config/meshiq/DB_USER` file, set the database user to be used by meshIQ components.
3. In the `config/meshiq/DB_USER_PWD` file, set the database user password to be used by meshIQ components.
4. In the `config/meshiq/SITE_KEY_PWD` file, set the SSL keystore password.

6.1.4 Optional Configuration (Changes are not Recommended Unless Needed)

1. The config/license/meshiq.lic file contains the license used by meshIQ WGS, Domain, and DSX.
2. config/cep-wgs/registry.xml is a meshIQ WGS object registry configuration file.
3. config/domain/registry.xml is a meshIQ Domain service object registry configuration file.
4. config/meshiq-db/meshiq.cnf is a WGS bound database (Postgres) configuration file.
5. The meshiq folder contains docker compose service definition *.yaml files.

Configuration parts to consider changing are as follows:

- docker host machine port bindings (if your environment already used some of them, and you get port bind errors running the meshIQ Platform).
 - manage-ui environment variables SSL_KEY_PATH, SSL_KEY_PWD, and SSL_KEY_ALIAS (if you need Tomcat to enable HTTPS protocol support).
 - grafana environment variables GF_SERVER_PROTOCOL, GF_SERVER_CERT_FILE and GF_SERVER_CERT_KEY (if you need to enable HTTPS protocol support).
6. In bin/set-env.sh you can choose the DB initialization actions policy (the default is demand):
 - skip - skips all DB initialization actions.
 - create - performs DB (re)create action.

WARNING: This will destroy your existing database!

- update - performs only DB update action.



NOTE

This will fail if the database does not exist.

- demand - performs DB create/update depending on current environment demand.
7. In the bin/set-env.sh shell script, you can set following configurations to access an external database:
 - USE_EXTERNAL_DB - indicates whether an external database will be used (default is false).
 - PROJECT_DB - database URL (default is **jdbc:postgresql://meshiq-db:5432**).
 - PROJECT_DB_DRIVER - database driver (default is **org.postgresql.Driver**).



NOTE

As described above in section [6.1.3, Recommended Configuration Steps \(Optional\)](#), set the external database user and password in the config/meshiq/DB_USER and config/meshiq/DB_USER_PWD files.

8. In bin/set-env.sh you can set meshIQ Domain information to access an external Domain:
 - AP_DOMAIN_HOST_IP - meshIQ Domain server host/ip (default is **domain**).
 - DOMAIN_SERVER_PORT - meshIQ Domain server port (default is 2323).
 - DOMAIN_SERVER_NAME - meshIQ Domain server name (default is DOMAIN_SERVER).

- DOMAIN_NAME - meshIQ Domain name (default is DOMAIN).
9. meshIQ bound Solr database configuration:
 - In the ADMINISTRATOR_PWD file, set the Administrator user password for the external Domain.

6.1.5 meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)



We strongly recommend that you **do not use** Git Bash, or Mintty in general, to run shell scripts, as this may produce unexpected results.

After setting up *set-env.sh*, you can use the wizard-like menu script *meshiq.sh* to manage the meshIQ Platform lifecycle:

```
./bin/meshiq.sh
```

Or to use a more advanced menu, use [meshiq_c.sh](#):

```
./bin/meshiq_c.sh
```

6.1.6 meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)

6.1.6.1 Initial Startup: Creates and Runs Application Containers

1. To run the base meshIQ Platform component containers, run the *up_meshiq.sh* shell script.
2. To run any set of connection manager containers as needed, run these shell scripts:
 - *up_cm_mq.sh* - to run IBM MQ connection manager.
 - *up_cm_kafka.sh* - to run Kafka connection manager.
 - *up_cm_ems.sh* - to run EMS (Enterprise Message Service/JMS) connection manager.
 - *up_cm_ace-iib.sh* - to run IBM ACE/IIB connection manager.
 - *up_cm_solace.sh* - to run Solace connection manager.
 - *up_cm_rabbitmq.sh* - to run RabbitMQ connection manager.

6.1.6.2 Migrating Existing Volumes to New App Version

If your meshIQ Platform version tag is defined using the major or major.minor pattern, the volume versions are migrated during the upgrade process. That is because Docker volume naming is bound to the PROJECT_NAME variable defined in the *set-env.sh* shell script, which is a combination of the project ID and version.

The dedicated `migrate-volumes.sh` shell script is available to migrate volumes from the previous app version to the new app version. This script runs automatically when you run the `up_meshiq.sh`, `start_all.sh` or `start_wgs.sh` shell scripts.

If your meshIQ Platform version tag is defined as `latest`, there is no need to migrate volume versions because the version tag value does not change.

6.1.6.3 Stopping Running Containers



NOTE

These procedures stop running containers without destroying them. For instructions on how to destroy containers, see [6.1.6.6, Permanent Shutdown](#).

1. To stop running meshIQ Platform containers, run the `stop_all.sh` shell script.
2. To stop all running meshIQ Platform Connection manager containers, run the `stop_cms.sh` shell script.

6.1.6.4 Starting Any Available Stopped Application Container

1. To start all meshIQ Platform containers that have been created, run the `start_all.sh` shell script.
2. To start the set of meshIQ Platform Connection manager containers that has been created, run the `start_cms.sh` shell script.

6.1.6.5 Restarting: Stops Running Application Containers and Starts Them Again

1. To restart running meshIQ Platform containers, run the `restart_all.sh` shell script.
2. To restart the set of running meshIQ Platform Connection manager containers, run the `restart_cms.sh` shell script.
3. To restart a running meshIQ Platform WGS service container, run the `restart_wgs.sh` shell script. For your convenience, it restarts the meshIQ WGS service container along with all running Connection manager containers.

6.1.6.6 Permanent Shutdown



IMPORTANT!

These instructions permanently destroy application containers. To merely stop the containers without destroying them, see [Stopping Running Containers](#)

1. To shut down any running meshIQ Platform Connection manager container, run the `down_cms.sh` shell script.

2. To shut down any running meshIQ Platform base component and Connection manager containers, run the `down_all.sh` shell script.
3. (Optional.) You can remove (destroy) persistent volumes used by the meshIQ Platform by running the `remove-volumes.sh` shell script.

6.1.7 Creating the meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
 - Username: admin
 - Password: admin
3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
 - **Name:** Choose a name for your data source.
 - **Service URL:** Set the URL to `http://cep-dsx:8580` (default meshIQ Service URL).
`http://cep-dsx:8580`
 - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.
`grafana-default`
 - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
 - **Autocomplete Service URL:** Set the URL to `http://cep-dsx:7580` (default Autocomplete Service URL).
`http://cep-dsx:7580`
8. Click **Save & Test** to validate the data source.

6.2 Installing the Grafana Integration using Docker Swarm

**NOTE**

Swarm mode is required and must be active to run the meshIQ Platform in Docker Swarm.

6.2.1 Download the compressed tar file

1. Download the meshiq_swarm_thin_v11.1.X.docker.tgz file.
2. Extract it to a location of your choosing.

6.2.2 Docker Swarm initialization

First, Docker Swarm needs to be initialized. It has already been installed with Docker. However, to start Docker Swarm, run this command:

```
docker swarm init
```

To make sure that Docker Swarm is active, you can run this command:

```
docker info | grep -i swarm
```

6.2.3 Leave the Docker Swarm mode

To leave the Docker Swarm run this command:

```
docker swarm leave -f
```

6.2.4 Check information about services

To check for more information about the service, run this command:

```
docker service inspect --pretty SERVICE_NAME
```

Change SERVICE_NAME to your actual service name.

To see which nodes are running the service:

```
docker service ps SERVICE_NAME
```

To see all services:

```
docker service ls
```

6.2.5 Setting the number of replicas

Docker Swarm has a tremendous advantage over Docker Compose. Swarm is capable of managing the number of running instances and the container's resilience against faults.

You can set the number of service instances in the `set-env.sh` script by changing the number of replicas.

For example:

```
export CM_KAFKA_SWARM_REPLICAS=1
```

The number of Kafka Connection Manager replicas is currently set to 1, although this setting can be adjusted as needed. Additionally, each configuration file specifies how many times the container will attempt to restart after encountering problems.

Continue with section [6.1.2: Required Configuration Steps of Installing the Grafana Integration in a Docker Container](#).

6.3 Installing the Grafana Integration Helm Chart for Kubernetes

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the `.tar` file.

6.3.1 Prerequisites

- [Docker](#)
- Kubernetes: [Docker Desktop bundled](#), [minikube](#), [Azure](#) or [AWS](#) Kubernetes service, [RH OpenShift](#)
- [Helm](#)

6.3.2 Pull the Helm Chart from the Repo

- Add meshIQ helm charts repo (**this is only required the first time**):
 - production repo

```
helm repo add meshiq-prod
https://containercomponents.s3.amazonaws.com/helm/charts
```
 - development repo

```
helm repo add meshiq-dev
https://containercomponents.s3.amazonaws.com/helm/charts/dev
```
- Update meshIQ helm repo index (we recommend that this be done before every pull to fetch new chart versions):

```
helm repo update
```
- Pull helm chart from repo:
 - Prod grade chart

- ```
helm pull meshiq-prod/meshiq-thin --untar
```
- Dev grade chart

```
helm pull meshiq-dev/meshiq-thin --untar
```

### 6.3.3 Set up and Run the meshIQ Platform on Your Kubernetes Cluster

- Make these changes to the `values.yaml` file:
  - Replace `<YOUR-CLUSTER-PUBLIC-IP>` with your Kubernetes cluster public IP or your PC's IP address if running Kubernetes cluster locally.



**DO NOT USE localhost or 127.0.0.1.** On containers, it does not resolve to your PC's address!

- After changing the IP address, run this command:

```
helm install meshiq meshiq-thin/
```
- You can also set the IP address when installing the helm chart by running this command:

```
helm install meshiq meshiq-thin/ --set global.service.external.ip=<YOUR-CLUSTER-PUBLIC-IP>
```

### 6.3.4 Stop Services

To stop services, run this command:

```
helm uninstall meshiq
```

### 6.3.5 Updating the Helm Chart

When a new helm chart version is released, you can upgrade it. The chart upgrade can be done like this:

1. Update helm repository index:

```
helm repo update
```
2. List all available chart versions:

```
helm search repo meshiq -l
```
3. Make a backup of your current `meshiq-thin` chart directory:

```
cp -r ./meshiq-thin meshiq-thin-back
```
4. Pull the new version of the helm chart:

```
helm pull <chart_name> --untar
```



You can add the `--version <version>` argument to pull a specific version of the chart:  
`helm pull <chart_name> --untar --version <version>`

5. Restore executable mode for chart bundled shell scripts:  

```
chmod +x ./meshiq-thin/scripts/*.sh
```
6. Merge values.yaml of the new helm chart `./meshiq-thin/values.yaml` with one on your backup `./meshiq-thin-back/values.yaml`.
7. Update chart:  

```
helm upgrade --install meshiq meshiq-thin/
```
8. See changed pods get terminated and started again.  

```
kubectl get pod --watch
```

### 6.3.5.1 Creating the meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
  - Username: admin
  - Password: admin
3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
  - **Name:** Choose a name for your data source.
  - **Service URL:** Set the URL to `http://cep-dsx:8580?` (default meshIQ Service URL).  

```
http://cep-dsx:8580?
```
  - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.  

```
grafana-default
```
  - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
  - **Autocomplete Service URL:** Set the URL to `http://cep-dsx:7580` (default Autocomplete Service URL).  

```
http://cep-dsx:7580
```
8. Click **Save & Test** to validate the data source.

### 6.3.5.2 Changing the Number of Shards

When the collection is too large for one node, it could be broken up and stored in sections by creating multiple shards. So, if you want to change the number of shards, you can do it in the `values.yaml` file.

- `solr.replicas` - number of Solr nodes (by default it is 1)
- `cep-service.solrShards` - number of Solr shards (by default it is 1)
- `cep-service.solrDbRepl` - number of Solr replications (by default it is 1)
- `cep-service.solrDbShardsPerNode` - number of Solr shards per node (by default it is 1)

### 6.3.5.3 Changing the Solr Username and Password

If you would like to change a Solr user configuration, this change should be made in the `solr-secret.yaml` file.

- To change `username` - set a new value for the `solr-user`. Write a new username encoded in base64.
- To change `password` - set a new value for `solr-pwd`. Write a new password encoded in base64.

## 6.3.6 Testing

To ensure the correctness of *Helm charts*, a set of tests has been included. To run the tests, execute the following command:

```
helm test meshiq
```

After running this command, you will see a series of messages regarding specific test outcomes. Information about the starting and completion times, as well as phases of each specific test, will be provided. If the phase is **successful**, you can be confident that everything is working as expected. In other cases, further investigation needs to be done.

**meshiq** is the release name in this case, so if you have written a different name, you should change it accordingly.

# Chapter 7: Installing the meshIQ Platform for Basic Management & Monitoring

---

The meshIQ Platform for Basic Management and Monitoring offers the observability features of the Grafana Integration and adds data collection for historical purposes. This edition assumes the use of Grafana or Splunk as a user interface for monitoring.

In a simple configuration, views can be used to collect facts.

- One instance of Solr
- Kafka
- ActiveMQ (to run the user interface)
- 6 services
- Job scheduler
- Web REST / UI
- ZooKeeper
- Grafana for visualizations and alerting

## 7.1 Installing the meshIQ Platform for Basic Management and Monitoring in a Docker Container

This section describes required, recommended, and optional configuration steps for the meshIQ Platform for Basic Management and Monitoring. It also includes wizard-based and manual lifecycle management instructions, and procedures for creating the meshIQ Platform Data Source.

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

### 7.1.1 Download the compressed tar file

1. Download the `meshiq_simple_v11.X.docker.tgz` file.
2. Extract it to a location of your choosing.

### 7.1.2 Required Configuration Steps

In the `bin/set-env.sh` shell script, set the `DOCKER_HOST_IP` variable to the external IP of the machine that is running docker.



NOTE

The IP address can also be set when you create meshIQ components. If the `'DOCKER_HOST_IP'` variable in `'set-env.sh'` is left as `'localhost'`, then after you choose to create meshIQ components using the menu, you will have an opportunity to select an IP address from a list or to set a specific one.

### 7.1.3 Recommended Configuration Steps (Optional)

1. In the `bin/set-env.sh` shell script, pick the `PROJECT_VERSION` variable that matches the meshIQ version you need. It can be defined using a major.minor pattern (e.g., 10.3 or 10.4), a major pattern (e.g., 10 or 11) or 'latest' (to have the latest Docker images used for meshIQ Components).
2. In the `config/meshiq/DB_USER` file, set the database user to be used by meshIQ components.
3. In the `config/meshiq/DB_USER_PWD` file, set the database user password to be used by meshIQ components.
4. In the `config/meshiq/SITE_KEY_PWD` file, set the SSL keystore password.

### 7.1.4 Optional Configuration (Changes are not Recommended Unless Needed)

1. The `config/license/meshiq.lic` file contains the license used by meshIQ WGS, Domain, and Track.
2. `config/cep-wgs/registry.xml` is a meshIQ WGS object registry configuration file.
3. `config/domain/registry.xml` is a meshIQ Domain service object registry configuration file.
4. `config/meshiq-db/meshiq.cnf` is a WGS bound database (Postgres) configuration file.
5. The `meshiq` folder contains docker compose service definition `*.yaml` files.

Configuration parts to consider changing are as follows:

- docker host machine port bindings (if your environment already used some of them, and you get port bind errors running the meshIQ Platform).
  - `manage-ui` environment variables `SSL_KEY_PATH`, `SSL_KEY_PWD`, and `SSL_KEY_ALIAS` (if you need Tomcat to enable HTTPS protocol support).
  - `grafana` environment variables `GF_SERVER_PROTOCOL`, `GF_SERVER_CERT_FILE` and `GF_SERVER_CERT_KEY` (if you need to enable HTTPS protocol support).
6. In `bin/set-env.sh` you can choose the DB initialization actions policy. (The default is demand.)
    - `skip` - skips all DB initialization actions.
    - `create` - performs DB (re)create action.

**WARNING:** this will destroy your existing database!

- `update` - performs only DB update action.



NOTE

This will fail if the database does not exist.

- `demand` - performs DB create/update depending on current environment demand.
7. In the `bin/set-env.sh` shell script, you can set flowing configurations to access an external database:

- a. USE\_EXTERNAL\_DB - indicates whether an external database will be used (default is **false**).
- b. PROJECT\_DB - database URL (default is **jdbc:postgresql://meshiq-db:5432**).



Set the external database user and password in the config/meshiq/DB\_USER and config/meshiq/DB\_USER\_PWD files.

- c. PROJECT\_DB\_DRIVER - database driver (default is **org.postgresql.Driver**).
8. In bin/set-env.sh you can set meshIQ Domain information to access an external Domain:
- a. AP\_DOMAIN\_HOST\_IP - meshIQ Domain server host/ip (default is **domain**).
  - b. DOMAIN\_SERVER\_PORT - meshIQ Domain server port (default is **2323**).
  - c. DOMAIN\_SERVER\_NAME - meshIQ Domain server name (default is **DOMAIN\_SERVER**).
  - d. DOMAIN\_NAME - meshIQ Domain name (default is **DOMAIN**).
9. meshIQ bound Solr database configuration:
- a. In the ADMINISTRATOR\_PWD file, set the Administrator user password for the external Domain.

## 7.1.5 meshIQ Platform Lifecycle: Wizard-driven Menu Script (Recommended)



We strongly recommend that you **do not use** Git Bash, or Mintty in general, to run shell scripts, as this may produce unexpected results.

After setting up *set-env.sh*, you can use the wizard-like menu script *meshiq.sh* to manage the meshIQ Platform lifecycle:

```
./bin/meshiq.sh
```

Or to use a more advanced menu, use *meshiq\_c.sh*:

```
./bin/meshiq_c.sh
```

## 7.1.6 meshIQ Platform Lifecycle: Manual Lifecycle Management (Optional)

### 7.1.6.1 Initial startup: Creates and runs application containers

1. To run the base meshIQ Platform component containers, run the *up\_meshiq.sh* shell script.
2. To run any set of connection manager containers as needed, run these shell scripts:
  - *up\_cm\_mq.sh* - to run IBM MQ connection manager.



- `up_cm_kafka.sh` - to run Kafka connection manager.
- `up_cm_ems.sh` - to run EMS (Enterprise Message Service/JMS) connection manager.
- `up_cm_ace-iib.sh` - to run IBM ACE/IIB connection manager.
- `up_cm_solace.sh` - to run Solace connection manager.
- `up_cm_rabbitmq.sh` - to run RabbitMQ connection manager.

### 7.1.6.2 Migrating existing volumes to new app version

If your meshIQ Platform version tag is defined using the major or major.minor pattern, the volume versions are migrated during the upgrade process. That is because Docker volume naming is bound to the `PROJECT_NAME` variable defined in the `set-env.sh` shell script, which is a combination of the project ID and version.

The dedicated `migrate-volumes.sh` shell script is available to migrate volumes from the previous app version to the new app version. This script runs automatically when you run the `up_meshiq.sh`, `start_all.sh` or `start_wgs.sh` shell scripts.

If your meshIQ Platform version tag is defined as `latest`, there is no need to migrate volume versions because the version tag value does not change.

### 7.1.6.3 Stopping running containers



These procedures stop running containers without destroying them. For instructions on how to destroy containers, see [Permanent shutdown](#).

1. To stop running meshIQ Platform containers, run the `stop_all.sh` shell script.
2. To stop all running meshIQ Platform Connection manager containers, run the `stop_cms.sh` shell script.

### 7.1.6.4 Starting any available stopped application container


1. To start all meshIQ Platform containers that have been created, run the `start_all.sh` shell script.
2. To start the set of meshIQ Platform Connection manager containers that have been created, run the `start_cms.sh` shell script.

### 7.1.6.5 Restarting: Stops running application containers and starts them again

1. To restart running meshIQ Platform containers, run the `restart_all.sh` shell script.
2. To restart the set of running meshIQ Platform Connection manager containers, run the `restart_cms.sh` shell script.

3. To restart a running meshIQ Platform WGS service container, run the `restart_wgs.sh` shell script. For your convenience, it restarts the meshIQ WGS service container along with all running Connection manager containers.

### 7.1.6.6 Permanent shutdown



**IMPORTANT!** These instructions permanently destroy application containers. To merely stop the containers without destroying them, see [Stopping running containers](#).

1. To shut down any running meshIQ Platform Connection manager container, run the `down_cms.sh` shell script.
2. To shut down any running meshIQ Platform base component and Connection manager containers, run the `down_all.sh` shell script.
3. (Optional.) You can remove (destroy) persistent volumes used by the meshIQ Platform by running the `remove-volumes.sh` shell script.

### 7.1.7 Creating a meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
  - Username: admin
  - Password: admin
3. Click on the triple bar icon (☰) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
  - **Name:** Choose a name for your data source.
  - **Service URL:** Set the URL to `http://ds-api:8080/ds-api/jkq1?` (default meshIQ Service URL).  
`http://ds-api:8080/ds-api/jkq1?`
  - **Access Token:** Use `grafana-default` as the default meshIQ Access Token.  
`grafana-default`
  - **Enable Query Autocomplete:** Enable this option by setting it to **true**.
  - **Autocomplete Service URL:** Set the URL to `http://cep-service:7580` (default Autocomplete Service URL).

<http://cep-service:7580>

8. Click **Save & Test** to validate the data source.

## 7.2 Installing the meshIQ Platform for Basic Management & Monitoring using Docker Swarm



Swarm mode is required and must be active to run the meshIQ Platform in Docker Swarm.

### 7.2.1 Download the compressed tar file

1. Download the `meshiq_swarm_simple_v11.1.X.docker.tgz` file.
2. Extract it to a location of your choosing.

### 7.2.2 Docker Swarm initialization

First, Docker Swarm needs to be initialized. It has already been installed with Docker. However, to start Docker Swarm, run this command:

```
docker swarm init
```

To make sure that Docker Swarm is active, you can run this command:

```
docker info | grep -i swarm
```

### 7.2.3 Leave the Docker Swarm mode

To leave the Docker Swarm run this command:

```
docker swarm leave -f
```

### 7.2.4 Check information about services

To check for more information about the service, run this command:

```
docker service inspect --pretty SERVICE_NAME
```

Change `SERVICE_NAME` to your actual service name.

To see which nodes are running the service:

```
docker service ps SERVICE_NAME
```

To see all services:

```
docker service ls
```

## 7.2.5 Setting the number of replicas

Docker Swarm has a tremendous advantage over Docker Compose. Swarm is capable of managing the number of running instances and the container's resilience against faults. You can set the number of service instances in the `set-env.sh` script by changing the number of replicas.

For example:

```
export CM_KAFKA_SWARM_REPLICAS=1
```

The number of Kafka Connection Manager replicas is currently set to 1, although this setting can be adjusted as needed. Additionally, each configuration file specifies how many times the container will attempt to restart after encountering problems.

Continue with section [7.1.2: Required Configuration Steps of Installing the meshIQ Platform for Basic Management and Monitoring in a Docker Container](#).

## 7.3 Installing the meshIQ Platform for Basic Management & Monitoring: Helm Chart for Kubernetes

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the `.tar` file.

### 7.3.1 Prerequisites

- [Docker](#)
- Kubernetes: [Docker Desktop bundled](#), [minikube](#), [Azure](#) or [AWS](#) Kubernetes service, [RH OpenShift](#)
- [Helm](#)

### 7.3.2 Pull the Helm Chart from the Repo

- Add meshIQ helm charts repo (**this is only required the first time**):

- production repo

```
helm repo add meshiq-prod
https://containercomponents.s3.amazonaws.com/helm/charts
```

- development repo

```
helm repo add meshiq-dev
https://containercomponents.s3.amazonaws.com/helm/charts/dev
```

- Update meshIQ helm repo index (we recommend that this be done before every pull to fetch new chart versions):

```
helm repo update
```

- Pull helm chart from repo:

- Prod grade chart

```
helm pull meshiq-prod/meshiq-simple --untar
```

- Dev grade chart

```
helm pull meshiq-dev/meshiq-simple --untar
```

### 7.3.3 Set up and Run the meshIQ Platform on Your Kubernetes Cluster

- Make these changes to the `values.yaml` file:
  - Replace `<YOUR-CLUSTER-PUBLIC-IP>` with your Kubernetes cluster public IP or your PC's IP address if running Kubernetes cluster locally.



NOTE

**DO NOT USE localhost or 127.0.0.1.** On containers, it does not resolve to your PC's address!

- After changing the IP address, run this command:

```
helm install meshiq meshiq-simple/
```

- You can also set the IP address when installing the helm chart by running this command:

```
helm install meshiq meshiq-simple/ --set global.service.external.ip=<YOUR-CLUSTER-PUBLIC-IP>
```

### 7.3.4 Stop Services

To stop services, run this command:

```
helm uninstall meshiq
```

### 7.3.5 Updating the Helm Chart

When a new helm chart version is released, you can upgrade it. The chart upgrade can be done like this:

1. Update helm repository index:

```
helm repo update
```

2. List all available chart versions:

```
helm search repo meshIQ -l
```

3. Make a backup of your current meshiq-simple chart directory:

```
cp -r ./meshiq-simple meshiq-simple-back
```

4. Pull the new version of the helm chart:

```
helm pull <chart_name> --untar
```



You can add the `--version <version>` argument to pull a specific version of the chart:  
`helm pull <chart_name> --untar --version <version>`

5. Restore executable mode for chart bundled shell scripts:

```
chmod +x ./meshiq-simple/scripts/*.sh
```

6. Merge values.yaml of the new helm chart `./meshiq-simple/values.yaml` with one on your backup `./meshiq-simple-back/values.yaml`.

7. Update chart:

```
helm upgrade --install meshiq meshiq-simple/
```

8. See changed pods get terminated and started again.

```
kubectl get pod --watch
```

### 7.3.5.1 Creating the meshIQ Platform Data Source

To create a meshIQ Platform data source in Grafana, follow these steps:

1. Open <http://localhost:3002/> in your web browser (or your custom Grafana URL, if applicable).
2. Log in to Grafana using the following credentials:
  - Username: admin
  - Password: admin



If password was changed, use the one set in the Values.yaml file as `grafana.adminPassword`.

3. Click on the triple bar icon (≡) next to "Home" to expand the menu.
4. Select **Connections** from the expanded menu.
5. Look for the **meshIQ** Data Source and click on it.
6. On the **meshIQ** Data Source page, click the blue **Create a data source** button in the top right-hand corner.
7. Provide the following information:
  - **Name:** Choose a name for your data source.

- **Service URL:** Set the URL to `http://ds-api:8080/ds-api/jkql?` (default meshIQ Service URL).

`http://ds-api:8080/ds-api/jkql?`

- **Access Token:** Use `grafana-default` as the default meshIQ Access Token.

`grafana-default`

- **Enable Query Autocomplete:** Enable this option by setting it to **true**.
- **Autocomplete Service URL:** Set the URL to `http://cep-service:7580` (default Auto-complete Service URL).

`http://cep-service:7580`

8. Click **Save & Test** to validate the data source.

### 7.3.5.2 Changing the Number of Shards

When the collection is too large for one node, it could be broken up and stored in sections by creating multiple shards. So, if you want to change the number of shards, you can do it in the `values.yaml` file.

- `solr.replicas` - number of Solr nodes (by default it is 1)
- `cep-service.solrShards` - number of Solr shards (by default it is 1)
- `cep-service.solrDbRep1` - number of Solr replications (by default it is 1)
- `cep-service.solrDbShardsPerNode` - number of Solr shards per node (by default it is 1)

### 7.3.5.3 Changing the Solr Username and Password

If you would like to change a Solr user configuration, this change should be made in the [solr-secret.yaml](#) file.

- To change `username` - set a new value for `solr-user`. Write a new username encoded in base64.
- To change `password` - set a new value for `solr-pwd`. Write a new password encoded in base64.

## 7.3.6 Log in to Track

To access HTTP enabled Track in browser: `http://<YOUR-CLUSTER-PUBLIC-IP>:8081/track`.

To access HTTPS enabled Track in browser: `https://<YOUR-CLUSTER-PUBLIC-IP>:8441/track`.

## 7.3.7 Testing

To ensure the correctness of *Helm charts*, a set of tests has been included. To run the tests, execute the following command:

```
helm test meshiq
```

After running this command, you will see a series of messages regarding specific test outcomes. Information about the starting and completion times, as well as phases of each specific test, will be provided. If the phase is *successful*, you can be confident that everything is working as expected. In other cases, further investigation needs to be done.

*meshiq* is the release name in this case, so if you have written a different name, you should change it accordingly.

## 7.4 Installing the meshIQ Platform for Basic Management and Monitoring Using the Deployment Pack

Follow the steps below to get started with the meshIQ Platform for Basic Management and Monitoring.



Please verify the prerequisites before beginning installation.

NOTE

### 7.4.1 Prerequisites for Installation

Linux System

- Operating System: Linux 64-bit OS
- CPU: Minimum 8 virtual CPUs
- Memory: Minimum 32GB RAM
- File System: Minimum 100 GB free space, SSD preferred
- Recommended Install folder: /opt/meshiq
- Required: Dedicated Server Instance



Some steps below require root access.

NOTE

1. Create a User ID.

Create the Linux user ID (UID) and group which will be used to start all meshIQ services.

The default user and group are meshiq:meshiq. This can be changed to any preferred user product owner. The UID and group must own all files and directories under \$APIN\_HOME.

2. Disable Swap.

We highly recommend that you turn Swap off within the operating system on all virtual machine servers running meshIQ Track.



To disable swap, perform the following:

- a) Identify configured swap devices and files with the following command:  
`cat /proc/swaps`
- b) Turn off all swap devices and files with the following command:  
`/swapoff -a`
- c) Comment out any swap matching references found in `/etc/fstab`.

### 3. Modify Limits.

Modify `/etc/security/limits.conf` and add the lines below for your user. Replace `meshiq` with the user that will start Track processes.

```
meshiq soft nofile 65536
meshiq hard nofile 65536

meshiq soft nproc 65536
meshiq hard nproc 65536
```

### 4. Software requirements


- Ensure 'curl' is installed. Curl is required to run `sbin/config.sh` script
- Ensure 'python' version `>= 2.7` is installed, if you are planning to run Apache Storm or Machine Learning.
- Ensure that native 'PostgreSQL' is not running. The installation uses its own PostgreSQL database implementation.
- As of version 1.3, the platform requires a Structured Query database server to run the job scheduler. This meshIQ Deployment pack is preconfigured to use PostgreSQL database by default; it is required to run Track.

### 5. Network ports:

The following list contains all processes network ports that are involved in meshIQ Track deployment.

|                             |                          |
|-----------------------------|--------------------------|
| DOMAIN SERVER (ATPNAMES):   | 2323, 3000 (TCP)         |
| CEP SERVER (ATPNODE):       | 3005 (TCP)               |
| STREAMING:                  | 6585 OR HTTPS(443) (TCP) |
| WEB SERVER (apache-tomcat): | 443, 8080 (TCP)          |
| ZOOKEEPER:                  | 2181 (TCP)               |
| ACTIVEMQ SERVER:            | 61616 (TCP)              |
| ACTIVEMQ WEB CONSOLE:       | 8161 (TCP)               |
| KAFKA SERVER:               | 9092 (TCP)               |
| SOLR:                       | 8983 (TCP)               |
| POSTGRESQL (postgres):      | 5432 (TCP)               |
| WORKGROUP SERVER:           | 4010 (TCP & UDP)         |
| REST API:                   | 8019 (TCP)               |
| STREAMING:                  | 6585 OR HTTPS(443) (TCP) |
| MQ AGENT (nsqmq):           | 5010 (TCP & UDP)         |

|                                                |                  |
|------------------------------------------------|------------------|
| MQ AGENT JAVA (nsqcmmq.jar):                   | 5015 (TCP & UDP) |
| MQ CONNECTION MANAGER (nsqcmmq.jar):           | 5025 (TCP)       |
| IBM Listeners: ----- 1414 (TCP)                |                  |
| IIB/ACE CONNECTION MANAGER (nsqcmace.jar):     | 5577 (TCP)       |
| IIB/Ace node: ----- 4414 (TCP)                 |                  |
| EMS CONNECTION MANAGER (nsqcmems.jar)          | 5556 (TCP)       |
| Tibco EMS URL: ----- 7222 (TCP)                |                  |
| KAFKA CONNECTION MANAGER (nsqcmkafka.jar)      | 5566 (TCP)       |
| Kafka Server: ----- 9092 (TCP)                 |                  |
| SOLACE CONNECTION MANAGER (nsqcmsolace.jar)    | 5588 (TCP)       |
| Solace URL PORT: ----- 8980 (TCP)              |                  |
| RABITMQ CONNECTION MANAGER (nsqcmrabbitmq.jar) | 5599 (TCP)       |
| RabbitMQ URL PORT: ----- 15672 (TCP)           |                  |

 **IMPORTANT!** The machine on which the installation will be done must meet all prerequisites described above.


## 7.4.2 Setup steps

These steps assume local or remote installation of all related components.

Log in to the [meshIQ software repository](#) to download the latest version of the meshIQ Platform package for Basic Management and Monitoring. A username and password are required.

1. Untar/unzip the meshIQ\_x.x.x\_Full\_DPackVx.x.tar.Z package to the installation path of your choice.

Shell command: `tar -zxvf meshIQ_x.x.x_Full_DPackVx.x.tar.Z`

 **NOTE** This package contains the directory "meshiq" as a root directory.

2. Review, verify and enable/disable local running services:  
Change to the \$APIN\_HOME/sbin/ folder and modify the file installed\_services.conf

Enter "yes" for enabled or "no" for disabled.

syntax: Service\_Name=yes (with no spaces)

3. To start the manage services, and for initial setup:

Access the `APIN_HOME/sbin/` directory using the first command below, then run the second command and follow the onscreen instructions.

Shell command: `cd APIN_HOME/sbin/`

Shell command: `./start.sh all`



NOTE

The first time you run these commands, you may be asked to specify the untarred/unzipped directory location, after which you will need to re-run the command `./start.sh all..`

You have completed the installation. Please check `$APIN_HOME/logs` for any errors.

If you are new to the meshIQ Platform, we recommend that you check out the following web pages to get started:

**Management**      <https://customers.meshiq.com/hc/en-us/categories/360002068774>

**Tracking**      <https://customers.meshiq.com/hc/en-us/categories/360002083773>

## 7.4.3 Setting the Installation Path (Optional)

### 7.4.3.1 Setting the Installation Path as an Environment Variable

To run startup scripts from anywhere, we recommend that you set environment variables `APIN_HOME` and `PATH` to the `sbin` directory in the user profile.

Copy and paste these two lines toward the end of the user profile file.

```
export APIN_HOME=/opt/meshiq
export PATH=$PATH:$APIN_HOME/sbin
```

Change `/opt/meshiq` to the path to the root directory of your meshiq installation.

### 7.4.3.2 Setting the Installation Path Manually

To set the installation path manually, edit `InstallationPath/sbin/apin_env.sh`

Change the line:

```
export APIN_HOME=ExistingPath
```

Replace `ExistingPath` with `NewPath`.

## **7.4.4 Access meshIQ Manage**

To access the meshIQ manage user interface from a web browser, use the following address:

`http://<serverip>:8080/manage/`

Use the following default log in credentials:

User Name: Admin (case sensitive)

Password: admin (case sensitive)

## **7.4.5 Access meshIQ Track**

To access the meshIQ Track user interface from a web browser, use the following address:

`http://<serverip>:8080/track/login.jsp`

Use the following default log in credentials:

User Name: Admin (case sensitive)

Password: password that was set during the deployment

# Chapter 8: Installing the Manage component

---

This section focuses on installation of the Manage component, which allows you to view, manage, and configure objects throughout the MESH.

## 8.1 Installing the Manage Component in a Docker Container

This section describes required, recommended, and optional configuration steps for the meshIQ Platform Manage edition. It also includes wizard-based and manual lifecycle management instructions.

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

### 8.1.1 Download the compressed tar file

1. Download the `meshiq_manage_v11.X.docker.tgz` file.
2. Extract it to a location of your choosing.

### 8.1.2 Required Configuration Steps

In the `bin/set-env.sh` shell script, set the `DOCKER_HOST_IP` variable to the external IP of the machine that is running docker.



The IP address can also be set when you create meshIQ components. If the `'DOCKER_HOST_IP'` variable in `'set-env.sh'` is left as `'localhost'`, then after you choose to create meshIQ components using the menu, you will have an opportunity to select an IP address from a list or to set a specific one.

### 8.1.3 Recommended Configuration Steps (Optional)

1. In the `bin/set-env.sh` shell script, pick the `PROJECT_VERSION` variable that matches the meshIQ version you need. It can be defined using a major.minor pattern (e.g., 10.3 or 10.4), a major pattern (e.g., 10 or 11) or 'latest' (to have the latest Docker images used for meshIQ Components).
2. In `config/meshiq/DB_USER` set the database user to be used by meshIQ Manage components.
3. In `config/meshiq/DB_USER_PWD` set the database user password to be used by meshIQ Manage components.
4. In `config/meshiq/SITE_KEY_PWD` set the SSL keystore password.

### 8.1.4 Optional Configuration (Changes are not Recommended Unless Needed)

1. The `config/license/meshiq.lic` file contains the license used by the meshIQ WGS and Domain.

2. config/cep-wgs/registry.xml is a meshIQ WGS object registry configuration file.
3. config/domain/registry.xml is a meshIQ Domain service object registry configuration file.
4. config/meshiq-db/manage.cnf is a WGS bound database (Postgres) configuration file.
5. The meshiq folder contains docker compose service definition \*.yaml files. Configuration parts to consider changing are as follows:
  - docker host machine port bindings (if your environment already used some of them, and you get port bind errors running meshIQ Manage).
  - *manage-ui* SSL\_KEY\_PATH environment variable and ./config/meshiq/SITE\_KEY\_PWD file content (if you need Tomcat to enable HTTPS protocol support).
6. In bin/set-env.sh you can choose the DB initialization actions policy (default is demand):
  - skip - skips all DB initialization actions.
  - create - performs DB (re)create action.

**WARNING:** this will destroy your existing database!

- update - performs only DB update action.



This will fail if the database does not exist.

NOTE

- demand - performs DB create/update depending on current environment demand.

## 8.1.5 Wizard-driven Menu Script (Recommended)



We strongly recommend that you **do not use** Git Bash, or Mintty in general, to run shell scripts, as this may produce unexpected results.

NOTE

After setting up *set-env.sh*, you can use the wizard-like menu script *manage.sh* to manage the meshIQ Manage lifecycle:

```
./bin/manage.sh
```

## 8.1.6 Manual Lifecycle Management (Optional)

### 8.1.6.1 Initial startup - creates and runs application containers

1. To run base meshIQ Manage component containers, run the *up\_manage.sh* shell script.
2. To run any set of Connection manager containers as needed, run these shell scripts:
  - *up\_cm\_mq.sh* - to run IBM MQ connection manager.
  - *up\_cm\_kafka.sh* - to run Kafka connection manager.
  - *up\_cm\_ems.sh* - to run EMS (Enterprise Message Service/JMS) connection manager.

- `up_cm_ace-iib.sh` - to run IBM ACE/IIB connection manager.
- `up_cm_solace.sh` - to run Solace connection manager.
- `up_cm_rabbitmq.sh` - to run RabbitMQ connection manager.

### 8.1.6.2 Migrating existing volumes to new app version

If your meshIQ Platform version tag is defined using the major or major.minor pattern, the volume versions are migrated during the upgrade process. That is because Docker volume naming is bound to the `PROJECT_NAME` variable defined in the `set-env.sh` shell script, which is a combination of the project ID and version.

The dedicated `migrate-volumes.sh` shell script is available to migrate volumes from the previous app version to the new app version. This script is run automatically when you run the `up_meshiq.sh`, `start_all.sh` or `start_wgs.sh` shell scripts.

If your meshIQ Platform version tag is defined as latest, there is no need to migrate volume versions because the version tag value does not change.

### 8.1.6.3 Stopping running containers



These procedures stop running containers without destroying them. For instructions on how to destroy containers, see [Permanent shutdown](#).

1. To stop all running meshIQ Manage containers, run the `stop_all.sh` shell script.
2. To stop all running meshIQ Manage Connection manager containers, run the `stop_cms.sh` shell script.

### 8.1.6.4 Starting any available stopped application container

1. To start meshIQ Manage containers that have been created, run the `start_all.sh` shell script.
2. To start the set of meshIQ Manage Connection manager containers that have been created, run the `start_cms.sh` shell script.

### 8.1.6.5 Restarting: Stops running application containers and starts them again

1. To restart all running meshIQ Manage containers, run the `restart_all.sh` shell script.
2. To restart the set of running meshIQ Manage Connection manager containers, run the `restart_cms.sh` shell script.
3. To restart a running meshIQ Manage WGS service container, run the `restart_wgs.sh` shell script. For convenience, it restarts the meshIQ Manage WGS service container along with all running Connection manager containers.

### 8.1.6.6 Permanent shutdown

**IMPORTANT!**

These instructions permanently destroy application containers. To merely stop the containers without destroying them, see section [8.1.6.3, Stopping running containers](#).

1. To shut down any running meshIQ Manage Connection manager container, run the `down_cms.sh` shell script.
2. To shut down any running meshIQ Manage base component and Connection manager container, run the `down_all.sh` shell script.
3. (Optional.) You can remove (destroy) persistent volumes used by meshIQ Manage by running the `remove-volumes.sh` shell script.

## 8.2 Installing the Manage Component using Docker Swarm

**NOTE**

Swarm mode is required and must be active to run the meshIQ Platform in Docker Swarm.

### 8.2.1 Download the compressed tar file

1. Download the `meshiq_swarm_manage_v11.1.X.docker.tgz` file.
2. Extract it to a location of your choosing.

### 8.2.2 Docker Swarm initialization

First, Docker Swarm needs to be initialized. It has already been installed with Docker. However, to start Docker Swarm, run this command:

```
docker swarm init
```

To make sure that Docker Swarm is active, you can run this command:

```
docker info | grep -i swarm
```

### 8.2.3 Leave the Docker Swarm mode

To leave the Docker Swarm run this command:

```
docker swarm leave -f
```

### 8.2.4 Check information about services

To check for more information about the service, run this command:



```
docker service inspect --pretty SERVICE_NAME
```

Change SERVICE\_NAME to your actual service name.

To see which nodes are running the service:

```
docker service ps SERVICE_NAME
```

To see all services:

```
docker service ls
```

## 8.2.5 Setting the number of replicas

Docker Swarm has a tremendous advantage over Docker Compose. Swarm is capable of managing the number of running instances and the container's resilience against faults. You can set the number of service instances in the set-env.sh script by changing the number of replicas.

For example:

```
export CM_KAFKA_SWARM_REPLICAS=1
```

The number of Kafka Connection Manager replicas is currently set to 1, although this setting can be adjusted as needed. Additionally, each configuration file specifies how many times the container will attempt to restart after encountering problems.

Continue with section [8.1.2: Required Configuration Steps of Installing the Manage Component in a Docker Container](#).

If you are not yet familiar with the meshIQ Platform, consider reviewing Appendix B: Terms and Project Layouts to learn more about commonly used meshIQ-related terms and the directory structure extracted from the .tar file.

# Appendix A: References

---

## A.1 meshIQ Documentation

The following documents relevant to meshIQ management applications can be found in the [Resource Center](#).

| Table A-1. meshIQ Documentation |                            |
|---------------------------------|----------------------------|
| Document Number (or higher)     | Title                      |
| MM11.000                        | meshIQ Manage User's Guide |
| MS11.000                        | meshIQ Secure User's Guide |
| MT11.000                        | meshIQ Track User's Guide  |

# Appendix B: Terms and Project Layouts

---

This appendix contains the following three sections:

- [Terms to consider](#)
- [Docker Directory Layouts](#)
- [Helm Chart Layouts](#)

## B.1 Terms to consider

Connection managers

“Connection managers” refers to any set of available meshIQ Connection manager services. All distributions come with the following set of connection managers:

- *nsqcmmq* - IBM MQ connection manager.
- *nsqcmems* - EMS (Enterprise Message Service/JMS) connection manager.
- *nsqcmkafka* - Kafka connection manager.
- *nsqcmace* - IBM ACE/IIB connection manager.
- *nsqcmsolace* - Solace connection manager.
- *nsqcmrabbitmq* - RabbitMQ connection manager.

### Databases

- *meshiq-db.yaml* - WGS bound database: PostgreSQL. All distributions.

### Servers

- *cep-wgs.yaml* - WGS (Workgroup server) having CEP (Complex Event Processing) capabilities. All distributions.
- *Domain.yaml* - meshIQ Domain server. All distributions.
- *solr* - Solr server.

### Services

- *activemq.yaml* - Apache ActiveMQ service.
- *cep-dsx.yaml* - Data Source service having CEP (Complex event processing) capabilities.
- *cep-gw.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Stitch, DbWriter, Compute and Gateway experts.
- *cep-job-scheduler.yaml* - service having CEP (Complex Event Processing) capabilities. Includes meshIQ Platform job scheduler expert.
- *cep-metrics.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Metrics expert.

- *cep-ml.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Prediction expert.
- *cep-rt.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Subscription and Trigger experts.
- *cep-sched.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Scheduler expert.
- *cep-scripts.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Script expert.
- *cep-service.yaml* - service having CEP (Complex Event Processing) capabilities. Includes Query, DbWriter, Service and AutoComplete experts.
- *grafana.yaml* - Grafana UI Web application service with meshIQ plugin.
  
- *kafka.yaml* - Kafka and ZooKeeper services.
- *manage.yaml* - Manage UI Web application and JAX-RS Web services. All distributions.
- *ml.yaml* - Machine Learning services for training and running queries.
- *solr-zookeeper.yaml* - Solr and ZooKeeper services (Docker Compose only)
- *solr.yaml* - Solr service (Docker Swarm only)
- *track.yaml* - meshIQ Track UI Web application service
- *zookeeper.yaml* - ZooKeeper services (Docker Swarm only)

REST endpoints

- *ds-api.yaml* - JKQL REST Endpoint with support for GET, POST, WebSockets

## B.2 Project Directories Layouts: Docker Distributions

The following directories and files are common to all meshIQ Platform Docker distributions:

- ***bin*** - contains Docker Compose application shell scripts.
  - ***scripts*** - contains Docker Compose based meshIQ Platform (or for the Manage distribution, Manage) lifecycle management and maintenance scripts.
- ***config*** - contains meshIQ Manage components configurations:
  - ***cep-wgs*** - contains *cep-wgs* service configuration files.
  - ***domain*** - contains *domain* service configuration files.
  - ***license*** - contains license *meshiq.lic* file.
  - ***manage*** - contains *manage-ui* service configuration files.

- **meshiq-db** - contains `meshiq-db` service configuration files.
- **meshIQ** - contains `meshiq` service configuration files.
  - **keystore** - For the Manage distribution, this contains SSL keys used by `manage-ui` service (Tomcat) and `cep-wgs` REST API service (Jetty). For all other distributions, these keys are also used by track services.
- **nsqcm** - contains `connection-managers` services configuration files.
- **meshiq** - contains Docker Compose services definition scripts.

The meshIQ platform Docker distributions for basic management and monitoring, typical servers for streaming, and larger meshIQ platform deployments also include the following in the config directory:

- **activemq-artemis** - contains `activemq` service configuration files.
- **cep-job-scheduler** - contains `cep-job-scheduler` service configuration files.
- **cep-track** - contains `cep-track` service configuration files.
- **grafana** - contains `grafana` service configuration files.
  - **certs** - contains SSL certificate and key used by `grafana` service.
- **track** - contains `track` services configuration files.
  - **ext/resources** - contains `external track` configuration files.

Only the Grafana integration contains a **cep-dsx** directory in the config directory. The `cep-dsx` directory contains `meshiq` data source service configuration files.

All distributions except Manage include a `solr` directory in the config directory:

- **solr** - contains `solr` service configuration files.
  - **init** - contains `tokens.jkq1` file with meshIQ tokens used to connect to `ds-api` service, Or, for the Grafana integration, to the `cep-dsx` service.

## B.3 Project Directories Layouts: Helm Chart Distributions



The /ext/ directories shown in this section are best suited for keystores/certificates.

As reflected in the installation instructions throughout this document, you must configure the values.yaml file at the root of each Helm Chart. The template files in the templates directories mentioned below typically pull variables from the “global” configuration in values.yaml. The changes that you make in values.yaml will be reflected in the templates directories when you deploy the Helm Chart.

Helm Chart Distributions:

- B.3.1 Helm Chart: Typical Servers for Streaming and Larger Deployments
- B.3.2 Helm Chart: New Grafana Integration
- B.3.3 Helm Chart: Basic Management and Monitoring

### B.3.1 Helm Chart: Typical Servers for Streaming and Larger Deployments

Differences between the Typical Servers installation and Larger Deployments are noted below.

- **templates/** - In the Typical distribution, this directory contains templates/configuration files for MeshIQ Typical Platform (CEP Job Scheduler, Scripts, Gateway, etc.).  
In Larger deployments, files are for MeshIQ Big Platform (Track, Cep-Scripts, Cep-Service, ML, etc.)
  - tests/ - Files to test connectivity/setup for the aforementioned services.
- **charts/** - Helm Chart dependencies for MeshIQ Typical/Larger Platform
  - **grafana/** - Grafana Helm Chart
    - **templates/** - Templates/configuration files for Grafana
      - **tests/** - Files to test Grafana configuration
    - **dashboards/**
      - custom-dashboard.json
    - **ci/**
  - **meshiq-activemq/** - MeshIQ ActiveMQ Helm Chart
    - **templates/** - Templates/configuration for MeshIQ ActiveMQ (Internal)
    - **cfg/** - Configuration file for internal MeshIQ ActiveMQ Broker

- activemq.xml
- **solr/** - Solr Helm Chart
  - **templates/** - Templates/configuration for Solr
- **manage/** - MeshIQ Manage Helm Chart
  - **templates/** - Templates/configuration files for MeshIQ Manage
    - **tests/** - Files to test various Connection Manager connectivities
  - **charts/** - Helm Chart dependencies for MeshIQ Manage
    - **meshiq-db/** - MeshIQ Database Helm Chart
      - **templates/** - Templates/configuration for MeshIQ Database
        - **tests/** - File to test MeshIQ Database connectivity
          - meshiq-db-connection-test.yaml
      - **config/**
        - meshiq-db/
          - manage.cnf
  - **config/** - Configuration files for various Manage services
    - **domain/** - Configuration files for Domain service
      - profile.properties
      - node.properties
      - registry.xml
      - security.dat
      - ext/
        - readme.txt
    - **nsqcmq/** - Configuration files for IBM MQ Connection Manager
      - logback.xml
      - nsqcmq.properties
      - **ext/**
        - readme.txt
    - **cep-wgs/** - Configuration files for MeshIQ Workgroup Server
      - node.properties
      - registry.xml
      - wgse\_man\_reg.xml

- wgs11.properties
- ext/
  - readme.txt
- **nsqcmrabbitmq/** - Configuration files specific to RabbitMQ Connection Manager
  - nsqcmrabbitmq.properties
  - log4j2.xml
  - ext/
    - readme.txt
- **nsqcmkafka/** - Configuration files specific to Apache Kafka connection Manager
  - log4j2.xml
  - nsqcmkafka.properties
  - **ext/**
    - readme.txt
- **manage/** - Configuration files specific to MeshIQ Manage (Frontend)
  - server.xml
  - server.key
  - server.crt
  - site.jks
  - meshiq.lic
  - ext/
    - apwmq\_samlssso.xml - Used for SSO
    - readme.txt
- **nsqcmace/** - Configuration files specific to ACE connection manager
  - nsqcmace.properties
  - log4j2.xml
  - ext/
    - readme.txt
- **nsqcmsolace/** - Configuration files specific to Solace connection manager
  - nsqcmsolace.properties



- log4j2.xml
      - ext/
        - readme.txt
    - **nsqcmems/** - Configuration files specific to TIBCO EMS Connection Manager
      - nsqcmems.properties
      - log4j2.xml
      - ext/
        - readme.txt
    - **scripts/** - Auxiliary scripts used for pre-deployment/setup
  - **solr-operator/** - Helm Chart for Solr Operator
    - **templates/** Templates/configuration files for Solr Operator
    - **crds/**
      - crds.yaml
    - **charts/** - Helm Chart Dependencies for Solr Operator
      - **zookeeper-operator/** - Helm Chart for Zookeeper Operator
        - **templates/** - Templates/configuration files for Zookeeper Operator
  - **strimzi-kafka-operator/** - Helm Chart for Strimzi Kafka Operator
    - **templates/** - Templates/configuration for Strimzi Kafka Operator
    - **crds/**
    - **files/** - Directory containing Strimzi dashboards for Grafana
      - grafana-dashboards/
- **config/** - Configuration/authentication files for MeshIQ Typical Helm Chart
  - **cep-gw/** - CEP-Gateway configuration
    - registry.xml
  - **cep-ml/** - CEP-MachineLearning configuration (Larger deployments only)
    - registry.xml
  - **track/** - Track specific configuration (including SAMLSSO)
    - server.xml
    - **ext/**
      - **resources/**
        - track-samlssso.xml

- **cep-scripts/** - CEP-Scripts configuration
  - registry.xml
- **cep-sched/** - CEP-Schedule configuration
  - registry.xml
- **meshiq/** - MeshIQ "global" configuration
  - DB\_USER\_PWD
  - SITE\_KEY\_PWD
  - server.xml
  - DB\_USER
  - server.key
  - server.crt
  - site.jks
- **license/** - License file
  - meshiq.lic
- **cep-service/** - CEP-Service configuration/authentication
  - registry.xml
  - tokens.jkql
- **cep-metrics/** - CEP-Metrics configuration (Larger deployments only)
  - registry.xml
- **cep-job-scheduler/** - CEP-Job-Scheduler configuration
  - registry.xml
- **scripts/** - Auxiliary scripts used for pre-deployment/setup

## B.3.2 Helm Chart: New Grafana Integration

The Project Directory structure for the New Grafana Integration Helm Chart is listed below.

- **templates/** - Templates/configuration files for MeshIQ Thin Data Services
  - tests/ - Files to test Data Services configuration
- **charts/** - Helm Chart dependencies for MeshIQ Thin Platform
  - **grafana/** - Grafana Helm Chart
    - **templates/** - Templates/configuration files for Grafana
      - tests/ - File to test Grafana configuration
    - **dashboards/**
      - custom-dashboard.json
    - **ci/**
  - **solr/** - Solr Helm Chart
    - templates/ - Templates/configuration files for Solr
  - **manage/** - MeshIQ Manage Helm Chart
    - **templates/** - Templates/configuration files for MeshIQ Manage
      - **tests/** - Files to test MeshIQ Manage configuration/connection manager connections
    - **charts/** - Helm Chart dependencies for MeshIQ Manage
      - **meshiq-db/** - MeshIQ Database Helm Chart
        - **templates/** - Templates/configuration files for MeshIQ Database
          - **tests/** - File to test database connectivity
            - meshiq-db-connection-test.yaml
        - **config/** - MeshIQ Database configuration file
          - meshiq-db/
            - manage.cnf
    - **config/** - Configuration files for Manage services
      - **domain/** - Configuration files specific to Domain
        - profile.properties
        - node.properties
        - registry.xml
        - security.dat
        - ext/

- readme.txt
- **nsqcmmq/** - Configuration files specific to IBM MQ Connection Manager
  - logback.xml
  - nsqcmmq.properties
  - ext/
    - readme.txt
- **cep-wgs/** - Configuration files specific to MeshIQ Workgroup Server
  - node.properties
  - registry.xml
  - wgse\_man\_reg.xml
  - wgs11.properties
  - ext/
    - readme.txt
- **nsqcmrabbitmq/** - Configuration files specific to RabbitMQ Connection Manager
  - nsqcmrabbitmq.properties
  - log4j2.xml
  - ext/
    - readme.txt
- **nsqcmkafka/** - Configuration files specific to Apache Kafka connection Manager
  - log4j2.xml
  - nsqcmkafka.properties
  - ext/
    - readme.txt
- **manage/** - Configuration files specific to MeshIQ Manage (Frontend)
  - server.xml
  - server.key
  - server.crt
  - site.jks
  - meshiq.lic

- ext/
  - apwmq\_samlssso.xml - Used for SSO
  - readme.txt
- **nsqcmace/** - Configuration files specific to ACE connection manager
  - nsqcmace.properties
  - log4j2.xml
  - ext/
    - readme.txt
- **nsqcmsolace/** - Configuration files specific to Solace connection manager
  - nsqcmsolace.properties
  - log4j2.xml
  - ext/
    - readme.txt
- **nsqcmems/** - Configuration files specific to TIBCO EMS Connection Manager
  - nsqcmems.properties
  - log4j2.xml
  - ext/
    - readme.txt
- **scripts/** - Auxiliary scripts used for pre-deployment/setup
- **solr-operator/** - Helm Chart for Solr Operator
  - **templates/** - Templates/configuration files for Solr Operator
  - **crds/**
  - **charts/** - Helm Chart dependencies for Solr Operator
    - **zookeeper-operator/** - Helm Chart for Zookeeper Operator
      - Chart.yaml
      - README.md
      - values.yaml
      - templates/ - Templates/configuration files for Zookeeper Operator
- **config/** - Configuration/authentication files for MeshIQ Thin Helm Chart

- **grafana/**
  - **certs/**
    - grafana.key
    - grafana.crt
- **cep-dsx/**
  - registry.xml
  - tokens.jkql
- **meshiq/**
  - DB\_USER\_PWD
  - SITE\_KEY\_PWD
  - server.xml
  - DB\_USER
  - server.key
  - server.crt
  - site.jks
- **license/**
  - meshiq.lic
- **scripts/** - Auxiliary scripts used for pre-deployment/setup

### B.3.3 Helm Chart: Basic Management and Monitoring

The project directory structure for the Basic Management and Monitoring Helm Chart is listed below.

- **config**
  - **cep-dsx** - Contains cep-dsx (Data Services) configuration files
    - registry.xml
    - tokens.jkql
  - **cep-gw** - Contains cep-gw (Gateway) configuration file
    - registry.xml
  - **cep-job-scheduler** - Contains cep-job-scheduler service configuration file
    - registry.xml
  - **cep-metrics** - Contains cep-metrics configuration file
    - registry.xml
  - **cep-ml** - Contains cep-ml (Machine Learning) configuration file
    - registry.xml
  - **cep-rt** - Contains cep-rt configuration file
    - registry.xml
  - **cep-sched** - Contains cep-sched (Schedule Expert) configuration file
    - registry.xml
  - **cep-scripts** - Contains cep-scripts (Scripts for CEP) configuration file
    - registry.xml
  - **cep-service** - Contains cep-service configuration files
    - registry.xml
    - tokens.jkql
  - **configure** - Contains configure configuration file (for future use)
    - server.xml
  - **grafana** - Contains grafana certs/authentication
    - certs
      - grafana.crt
      - grafana.key
  - **license** - Contains MeshIQ Platform license

- meshiq.lic
- **meshiq** - Contains MeshIQ Platform configuration files
  - DB\_USER - username for database
  - DB\_USER\_PWD - password for database user (DB\_USER)
  - SITE\_KEY\_PWD - keystore password
  - server.xml - Tomcat server.xml configuration
- **track** - Contains track configuration files
  - ext/resources
    - track-samlssso.xml
  - server.xml - Tomcat server.xml configuration
- **scripts** - Contains auxiliary scripts for pre-deployment/setup
- **templates** - Contains low-level configuration/setup for each image/container.
- **charts** - Dependencies for the MeshIQ Platform Helm Chart
  - **meshiq-activemq** - ActiveMQ Helm Chart to be used internally
    - cfg
      - broker.xml
  - **grafana** - Grafana Helm Chart
    - dashboards
      - custom-dashboard.json
  - **manage** - MeshIQ Manage Helm Chart
    - **charts** - Dependencies for Manage Helm Chart
      - meshiq-db
        - config
    - **config**
      - **cep-wgs** - Workgroup Server specific configuration
        - node.properties
        - registry.xml
        - wgs11.properties
        - wgse\_man\_reg.xml
        - ext
      - **domain** - Domain specific configuration
        - node.properties



- profile.properties
- registry.xml
- security.dat
- ext
- **manage** - Manage (Frontend) specific configuration
  - meshiq.lic - License file
  - server.crt/.key - Certs for Tomcat server
  - server.xml - Manage Tomcat configuration file
  - ext
    - apwmq\_samlss.xml - Used for SSO
- **nsqcmace** - ACE Connection Manager configuration
  - log4j2.xml
  - nsqcmace.properties
  - ext
- **nsqcmems** - TIBCO EMS Connection Manager configuration
  - log4j2.xml
  - nsqcmems.properties
  - ext
- **nsqcmkafka** - Apache Kafka Connection Manager configuration
  - log4j2.xml
  - nsqcmkafka.properties
  - ext
- **nsqcmmq** - IBM MQ Connection Manager configuration
  - logback.xml
  - nsqcmmq.properties
  - ext
- **nsqcmrabbitmq** - RabbitMQ Connection Manager configuration
  - log4j2.xml
  - nsqcmrabbitmq.properties
  - ext
- **nsqcmsolace** - Solace Connection Manager configuration

- log4j2.xml
- nsqcmsolace.properties
- ext
- **solr** - Solr Helm Chart
- **solr-operator** - Solr Operator Helm Chart
- **strimzi-kafka-operator** - Strimzi Kafka Helm Chart to be used internally