Hazelcast Management Center Reference Manual

Table of Contents

Welcome to the Hazelcast Management Center Reference Manual. This manual includes concepts and instructions to guide you on how to use Management Center to monitor your Hazelcast IMDG and Jet Clusters.

Preface

Hazelcast Management Center enables you to monitor and manage your Hazelcast IMDG and Jet clusters.

Naming

- **Hazelcast Management Center** or **Management Center** refers to the Hazelcast IMDG and Jet

cluster monitoring tool provided by Hazelcast, Inc.

- **Hazelcast IMDG** or just **Hazelcast** refers to the Hazelcast in-memory data grid middleware. **Hazelcast** is also the name of the company (Hazelcast, Inc.) providing Hazelcast IMDG and Hazelcast Jet.
- **Hazelcast Jet** or **Jet** refers to the distributed data processing engine provided by Hazelcast, Inc.

Hazelcast IMDG

Hazelcast Management Center is delivered with Hazelcast IMDG. It can also be downloaded as a separate package from the hazelcast.org website.

See the Hazelcast IMDG Reference Manual for all Hazelcast IMDG topics including the clusters and their operations, clients, data structures, computing and WAN replication.

Licensing

Hazelcast Management Center requires either of the following:

- Management Center license
- Hazelcast IMDG Enterprise license
- Hazelcast IMDG Enterprise HD license
- Hazelcast Jet Enterprise license

It also has a free version which lets you to monitor your clusters having up to three IMDG and Jet members each.

Trademarks

Getting Help

Support is provided via the following channels:

- Troubleshooting section of this guide
- Stack Overflow (ask a question on how to use Management Center properly and troubleshoot your setup)
- Hazelcast mailing list (propose features and discuss your ideas with the team)
- Hazelcast Community Slack (discuss anything related to Hazelcast IMDG, Jet and Management Center with other Hazelcast users and Hazelcast developers)

Release Notes

See the Release Notes document for the new features, enhancements and fixes performed for each Hazelcast Management Center release.

1. Supported Environments

You need Java Runtime Environment 1.8 or later to run Hazelcast Management Center.

Management Center uses RocksDB library to store metrics data on disk. Because of that dependency, macOS users have to be on at least 10.15 version (Catalina) and Windows users need to have Microsoft Visual C++ 2015 Redistributable installed. Otherwise, you can use the official Management Center docker image.

1.1. IMDG/Jet Compatibility

This version of Hazelcast Management Center is tested with Hazelcast IMDG 4.0.1 and Jet 4.2. We officially support any IMDG 4.0.z and Jet 4.2.z version.

1.2. Browser Compatibility

Hazelcast Management Center is tested and works on the following browsers:

- Last two versions of Chrome at the time of release
- Last two versions of Firefox at the time of release
- Latest Safari
- Internet Explorer 11
- Latest Microsoft Edge

2. Getting Started

To start using the Management Center:

1. download the Hazelcast Management Center package
2. start the Management Center service
3. launch the Management Center user interface.

2.1. Downloading Management Center

Hazelcast Management Center is included in the Hazelcast IMDG download package. You can download it from the download page of Hazelcast's website.

When a new Hazelcast IMDG version is released, it comes with a Management Center that is compatible with IMDG. There may be times when a new Management Center version is released

before a new version of Hazelcast IMDG. In that case, you may download the new Management Center from its download page as a separate package.

2.2. Starting the Management Center Service

You have the following options to start the Management Center service:

- Start Hazelcast Management Center from the command line.
- Use the scripts that come with the download package.
- Deploy the file `hazelcast-management-center-4.2020.08.war` on your Java application server/servlet container.

2.2.1. Using the Command Line

After you downloaded, extract the Hazelcast IMDG or Management Center package. The extracted directory, i.e., `hazelcast-management-center-4.2020.08`, contains the `hazelcast-management-center-4.2020.08.jar` file.

You can start this file directly from the command line using the following command:

```
java -jar hazelcast-management-center-4.2020.08.jar
```

The above command starts the Hazelcast Management Center service on the port `8080` with the root context path (`http://localhost:8080`).

In Management Center 4.0, the default context path changed from `hazelcast-mancenter` to the root context path.

You can use `-Dhazelcast.mc.http.port` and `-Dhazelcast.mc.contextPath` command line options to start Management Center on a different port and context path:

```
java -Dhazelcast.mc.http.port=8083 -Dhazelcast.mc.contextPath='hazelcast-mc' -jar
hazelcast-management-center-4.2020.08.jar
```

The above command starts Management Center on port `8083` and context path `hazelcast-mc`. You can access it by opening `http://localhost:8083/hazelcast-mc` in your browser.

Then, you need to let Management Center know the Hazelcast IMDG member addresses as explained in the Connecting IMDG Members to Management Center chapter.

For the options you can provide when starting with the command line, see the Configuring Management Center chapter. In that chapter, you can learn about topics including how to start with a license or extra classpath, how to configure the security, disk usage, update interval or logging.

2.2.2. Deploying to Application Server

Instead of starting at the command line, you can deploy the Management Center to your application server (Tomcat, Jetty, etc.).

2.2.3. Using Scripts in the Package

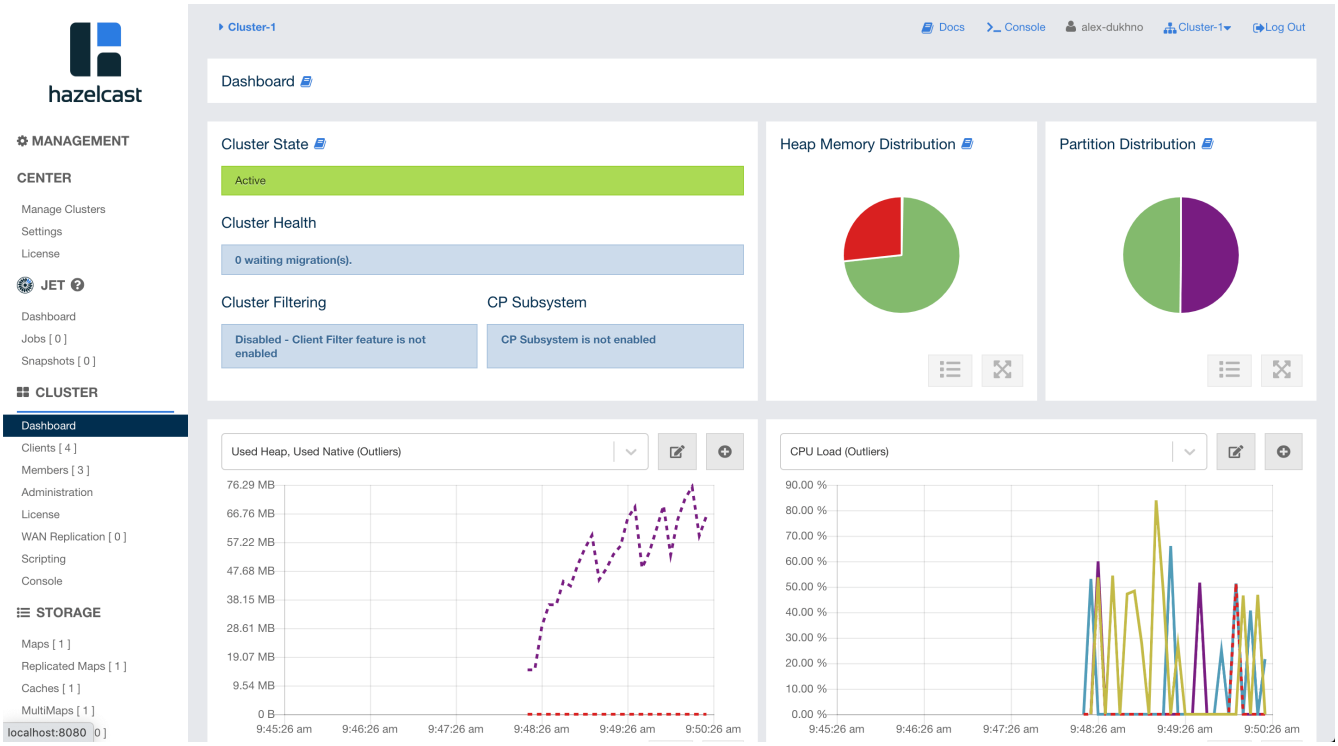
As another option, you can use the `start.bat` or `start.sh` scripts, which come with the download package, to start the Management Center. You can find these scripts under the extracted directory.

2.3. Quick Launch

After the above steps are completed, let's quickly open the Management Center user interface with the default settings.

Assuming that you have started your Hazelcast IMDG and Jet clusters each including a single member on your local machine (`localhost`) and started the Management Center service on `http://localhost:8080`, follow the below instructions:

1. Go to `http://localhost:8080` using your preferred browser.
2. Select `Dev Mode` as the security provider, and click `Save` (note: this configuration is not recommended for production environments).
3. Login with your newly created credentials, and "Manage Clusters" page appears.
4. Click on the "Add Cluster Config" button to connect Management Center to your IMDG cluster:
    a. Click on the `Save` button after providing your IMDG cluster's name and member address (`localhost`).
5. Again, click on the "Add Cluster Config" button to connect Management Center to your Jet cluster:
    a. Click on the `Save` button after providing your Jet cluster's name (use the default `jet`) and member address (`localhost`).
6. On the "Manage Cluster" page, click on the `Select` button for a desired cluster to monitor it; the dashboard of your cluster appears as shown below:



See Launching the Management Center User Interface chapter for more details.

Also see here and here for information on starting the IMDG and Jet clusters, respectively.

You also have the **Dev Mode** option for development or evaluation purposes that provides quick access to Management Center without requiring any security credentials. Please see here for details.

3. Configuring Management Center

This chapter explains how you can configure Hazelcast Management Center according to your needs.

The communication between the IMDG/Jet cluster members and the Management Center instance is based on Hazelcast open binary client protocol. The clusters that Management Center should connect to are configured within Management Center as described in the Connecting Management Center to IMDG Members section.

In previous versions, the URL of the Management Center instance was configured within the `<hazelcast></hazelcast>` section of `hazelcast.xml`. This configuration is now deprecated and ignored.

3.1. Providing a License

When starting the Management Center from the command line, a license can be provided using the system property `hazelcast.mc.license`. For example by using the command line parameter:

```
java -Dhazelcast.mc.license=<key> -jar hazelcast-management-center-4.2020.08.jar
```

When this option is used, the license provided takes precedence over any license set and stored previously using the user interface. Previously stored licenses are not affected and will be used again when the Management Center is started without the `hazelcast.mc.license` property. This also means no new license can be stored when the property is used.

3.2. Providing an Extra Classpath

You can also start the Management Center with an extra classpath entry (for example, when using JAAS authentication) by using the following command:

```
java -cp "hazelcast-management-center-4.2020.08.jar:/path/to/an/extra.jar" com
.hazelcast.webmonitor.Launcher
```

On Windows, the command becomes as follows (semicolon instead of colon):

```
java -cp "hazelcast-management-center-4.2020.08.jar;/path/to/an/extra.jar"
com.hazelcast.webmonitor.Launcher
```

3.3. Configuring the Client Used by Management Center

You can configure the client instance that is used for connecting to the cluster by using the following command line parameters:

- `-Dhazelcast.mc.client.initial.backoff.millis`: Duration, in milliseconds, to wait after the first failure before retrying. It is in milliseconds. The default value is `1000` ms. Set values have to be in range of `1000` to `60000` ms.
- `-Dhazelcast.mc.client.backoff.multiplier`: Factor with which to multiply backoff after a failed retry. Default value is `2`. Set values have to be in range of `1` to `10`.
- `-Dhazelcast.mc.client.max.backoff.millis`: When backoff reaches this upper bound, it does not increase any more. It is in milliseconds. Default value is `32000` ms. Set values have to be in range of `32000` to `600000` ms.

Note that these parameters apply to all clients that are started by Management Center.

You can also pass in a custom client configuration file to Management Center. These parameters override any configuration you pass as part of adding a new cluster connection. Following is a list of all other client configuration parameters that are overriden by Management Center when creating clients to connect to clusters:

- `InstanceName`: Generated based on the cluster name.
- `ConnectionStrategyConfig.asyncStart`: Set to `true`. It means that the client starts without waiting to get a cluster connection.
- `ConnectionStrategyConfig.clusterConnectTimeoutMillis`: Set to infinity. The client never gives up

trying to connect to the cluster.

- `NetworkConfig.smartRouting`: Set to `true`. It means that client will open connections to all members.

3.4. Configuring Metadata Polling Period of Management Center

Management Center polls the cluster members periodically for their metadata, which includes the list of data structures it has and their configurations. You can change the polling frequency by using the `hazelcast.mc.state.reschedule.delay.millis` command line parameter. Its default value is `1000`.

3.5. Configuring Disk Usage

You can control the disk space used by the Management Center to avoid exceeding available disk space. That can be done by configuring the Metrics Persistence.

You can use the `hazelcast.mc.metrics.disk.ttl.days` system property to configure the Management Center's metrics persistence. This property specifies Time-to-Live (TTL) setting for each record in the metrics persistence. It is in days and its default value is `1`.

It is important to understand that the TTL setting acts as a *soft* limit and gives you an indirect control over Management Center disk usage. The final disk usage depends on the volume of metrics persistence data generated by your clusters, i.e., the number of cluster members and data structures with enabled statistics that you have in the connected clusters.

You may want to decrease the TTL setting in situations when the Management Center uses too much disk space.

3.6. Enabling Health Check Endpoint

When running the Management Center from the command line, you can enable the Health Check endpoint. This endpoint responds with `200 OK` HTTP status code once the Management Center web application has started. The endpoint is available on port `<Management Center HTTP port> + 1` with context path `<Management Center context path>/health` (by default, its URL is `http://localhost:8081/health`). Note that the HTTP protocol is always used for the Health Check endpoint, independently of TLS/SSL settings, and no additional authentication is enforced for it.

If you want to enable the Health Check endpoint, use the following command line argument:

```
-Dhazelcast.mc.healthCheck.enable=true
```

3.7. Configuring Sessions

This section provides information on how to configure the Management Center sessions for various aspects including timeouts and login/logout operations.

3.7.1. Configuring Session Timeout

If you have started the Management Center from the command line by using the JAR file, by default, the sessions that are inactive for 30 minutes are invalidated. To change this, you can use the `-Dhazelcast.mc.session.timeout.seconds` command line parameter.

For example, the following command starts the Management Center with a session timeout period of 1 minute:

```
java -Dhazelcast.mc.session.timeout.seconds=60 -jar hazelcast-management-center-
4.2020.08.jar
```

If you have deployed the Management Center on an application server/container, you can configure the default session timeout period of the application server/container to change the session timeout period for the Management Center. If your server/container allows application specific configuration, you can use it to configure the session timeout period for the Management Center.

3.7.2. Disabling Multiple Simultaneous Login Attempts

Normally, a user account on the Management Center can be used from multiple locations at the same time. If you want to forbid others from logging in, when there's already someone logged in with the same username, you can start the Management Center with the `-Dhazelcast.mc.allowMultipleLogin=false` command line parameter.

3.7.3. Disable Login Configuration

In order to prevent password guessing attacks, logging in is disabled temporarily after a number of failed login attempts. When not configured explicitly, the default values are used, i.e., logging in is disabled for 5 seconds when a username is failed to log in consecutively 3 times. During this 5 seconds of period, logging in is not allowed even when the correct credentials are used. After 5 seconds, the user will be able to log in using the correct credentials.

Assuming the configuration with the default values, if the failed attempts continue (consecutively 3 times) after the period of disabled login passes, this time the disable period is multiplied by 10: logging in is disabled for 50 seconds. The whole process repeats itself until the user logs in successfully. By default, there's no upper limit to the disable period, but can be configured by using the `-Dhazelcast.mc.maxDisableLoginPeriod` parameter.

Here is a scenario, in the given order, with the default values:

1. You try to login with your credentials consecutively 3 times but failed.
2. Logging in is disabled and you have to wait for 5 seconds.
3. After 5 seconds have passed, logging in is enabled.
4. You try to login with your credentials consecutively 3 times but again failed.
5. Logging in is disabled again and this time you have to wait for 50 seconds until your next login attempt.

6. And so on; each 3 consecutive login failures causes the disable period to be multiplied by 10.

You can configure the number of failed login attempts, initial and maximum duration of the disabled login and the multiplier using the following command line parameters:

- `-Dhazelcast.mc.failedAttemptsBeforeDisableLogin`: Number of failed login attempts that cause the logging in to be disabled temporarily. Default value is `3`.
- `-Dhazelcast.mc.initialDisableLoginPeriod`: Initial duration for the disabled login in seconds. Default value is `5`.
- `-Dhazelcast.mc.disableLoginPeriodMultiplier`: Multiplier used for extending the disable period in case the failed login attempts continue after disable period passes. Default value is `10`.
- `-Dhazelcast.mc.maxDisableLoginPeriod`: Maximum amount of time for the disable login period. This parameter does not have a default value. By default, disabled login period is not limited.

3.7.4. Forcing Logout on Multiple Simultaneous Login Attempts

If you haven't allowed multiple simultaneous login attempts explicitly, the first user to login with a username stays logged in until that username explicitly logs out or its session expires. In the meantime, no one else can login with the same username. If you want to force logout for the first user and let the newcomer login, you need to start Management Center with the `-Dhazelcast.mc.forceLogoutOnMultipleLogin=true` command line parameter.

3.8. Configuring and Enabling Security

This section provides information on how to use and manage the Management Center with TLS/SSL and mutual authentication. You will also learn how to force the users to specify passwords that are hard to guess.

3.8.1. Using Management Center with TLS/SSL Only

To encrypt data transmitted over all channels of the Management Center using TLS/SSL, make sure you do all of the following:

- Deploy the Management Center on a TLS/SSL enabled container or start it from the command line with TLS/SSL enabled. See Installing the Management Center.
  - Another option is to place the Management Center behind a TLS-enabled reverse proxy. In that case, make sure your reverse proxy sets the necessary HTTP header (`X-Forwarded-Proto`) for resolving the correct protocol.
- Enable TLS/SSL communication to the Management Center for your Hazelcast cluster. See Connecting Hazelcast members to the Management Center.
- If you're using Clustered JMX on the Management center, enable TLS/SSL for it. See Enabling TLS/SSL for Clustered JMX.
- If you're using LDAP authentication, make sure you use LDAPS or enable the "Start TLS" field. See LDAP Authentication.

You can configure how Management Center treats `X-Forwarded-*` headers using the system property `hazelcast.mc.forwarded.requests.enabled`. If its value is set to `true`, Management Center accepts and treats them as set by a reverse proxy in front of it, otherwise, they are ignored. Its default value is `true`.

3.8.2. Enabling TLS/SSL When Starting with JAR File

When you start the Management Center from the command line, it serves the pages unencrypted by using "http", by default. To enable TLS/SSL, use the following command line parameters when starting the Management Center:

- `-Dhazelcast.mc.tls.enabled`: Specifies whether TLS/SSL is enabled. Its default value is false (disabled).
- `-Dhazelcast.mc.tls.keyStore`: Path to the keystore.
- `-Dhazelcast.mc.tls.keyStorePassword`: Password of the keystore.
- `-Dhazelcast.mc.tls.trustStore`: Path to the truststore.
- `-Dhazelcast.mc.tls.trustStorePassword`: Password of the truststore.

You can leave the truststore and truststore password values empty to use the system JVM's own truststore.

The following is an example on how to start the Management Center with TLS/SSL enabled from the command line:

```
java -Dhazelcast.mc.tls.enabled=true
-Dhazelcast.mc.tls.keyStore=/some/dir/selfsigned.jks
-Dhazelcast.mc.tls.keyStorePassword=yourpassword -jar hazelcast-management-center-
4.2020.08.jar
```

You can access the Management Center from the following HTTPS URL on port 8443: `https://localhost:8443`.

On the member side, you need to configure the Management Center URL as `https://localhost:8443` and also set the following JVM arguments when starting the member:

```
-Djavax.net.ssl.trustStore=path to your truststore
-Djavax.net.ssl.trustStorePassword=yourpassword
```

If you plan to use a self-signed certificate, make sure to create a certificate with the hostname of the machine you will deploy the Management Center on. Otherwise, you will see a line similar to the following in the member logs:

```
javax.net.ssl.SSLHandshakeException: java.security.cert.CertificateException:
No subject alternative names matching IP address 127.0.0.1 found
```

To override the HTTPS port, you can use the `-Dhazelcast.mc.https.port` command line option when starting the Management Center. For example:

```
java -Dhazelcast.mc.tls.enabled=true \
     -Dhazelcast.mc.tls.keyStore=/dir/to/certificate.jks \
     -Dhazelcast.mc.tls.keyStorePassword=yourpassword \
     -Dhazelcast.mc.https.port=443 \
     -jar hazelcast-management-center-4.2020.08.jar
```

This starts the Management Center on HTTPS port 443.

You can encrypt the keystore/truststore passwords and pass them as command line arguments in encrypted form for improved security. See the Variable Replacers section for more information.

Enabling HTTP Port

By default, HTTP port is disabled when you enable TLS. If you want to have an open HTTP port that redirects to the HTTPS port, use the following command line argument:

```
-Dhazelcast.mc.tls.enableHttpPort=true
```

Managing TLS Enabled Clusters

If a Hazelcast cluster is configured to use TLS for communication between its members using a self-signed certificate, the Management Center will not be able to perform some of the operations that use the cluster's HTTP endpoints (such as shutting down a member or getting the thread dump of a member). This is so because self-signed certificates are not trusted by default by the JVM. For these operations to work, you need to configure a truststore containing the public key of the self-signed certificate when starting the JVM of the Management Center using the following command line parameters:

- `-Dhazelcast.mc.httpClient.tls.trustStore`: Path to the truststore.
- `-Dhazelcast.mc.httpClient.tls.trustStorePassword`: Password of the truststore.
- `-Dhazelcast.mc.httpClient.tls.trustStoreType`: Type of the truststore. Its default value is JKS.
- `-Dhazelcast.mc.httpClient.tls.trustManagerAlgorithm`: Name of the algorithm based on which the authentication keys are provided. System default is used if none is provided. You can find out the default by calling the `javax.net.ssl.TrustManagerFactory#getDefaultAlgorithm` method.

You can encrypt the truststore password and pass it as a command line argument in encrypted form for improved security. See the Variable Replacers section for more information.

By default, JVM also checks for the validity of the hostname of the certificate. If this test fails, you will see a line similar to the following in the Management Center logs:

```
javax.net.ssl.SSLHandshakeException: java.security.cert.CertificateException:
No subject alternative names matching IP address 127.0.0.1 found
```

If you want to disable this check, start the Management Center with the following command line parameter:

```
-Dhazelcast.mc.disableHostnameVerification=true
```

3.8.3. Mutual Authentication

You can configure Management Center to require mutual authentication. With this setup, any client (be it a Web browser or an HTTP client such as curl) needs to present their TLS certificate and the Management Center needs to have its truststore configured so that the Management Center can know which clients it can trust. To enable mutual authentication, you need to use the following command line parameters when starting the Management Center:

```
-Dhazelcast.mc.tls.mutualAuthentication=REQUIRED
```

See the below snippet to see the full command to start the Management Center:

```
java -Dhazelcast.mc.tls.enabled=true \
     -Dhazelcast.mc.tls.trustStore=path to your truststore \
     -Dhazelcast.mc.tls.trustStorePassword=password for your truststore \
     -Dhazelcast.mc.tls.mutualAuthentication=REQUIRED \
     -jar hazelcast-management-center-4.2020.08.jar
```

The parameter `-Dhazelcast.mc.tls.mutualAuthentication` has two options:

- `REQUIRED`: If the client does not provide a keystore or the provided keys are not included in the Management Center's truststore, the client will not be authenticated.
- `OPTIONAL`: If the client does not provide a keystore, it will be authenticated. But if the client provides keys that are not included in the Management Center's truststore, the client will not be authenticated.

Managing Mutual Authentication Enabled Clusters

If mutual authentication is enabled for the cluster (as described here), the Management Center needs to have a keystore to identify itself. For this, you need to start the Management Center with the following command line parameters:

- `-Dhazelcast.mc.httpClient.tls.keyStore`: Path to the keystore.
- `-Dhazelcast.mc.httpClient.tls.keyStorePassword`: Password of the keystore.
- `-Dhazelcast.mc.httpClient.tls.keyStoreType`: Type of the keystore. Its default value is JKS.

- `-Dhazelcast.mc.httpClient.tls.keyManagerAlgorithm`: Name of the algorithm based on which the authentication keys are provided. System default is used if none is provided. You can find out the default by calling the `javax.net.ssl.KeyManagerFactory#getDefaultAlgorithm` method.

Excluding Specific TLS/SSL Protocols

When you enable TLS on the Management Center, it will support the clients connecting with any of the TLS/SSL protocols that the JVM supports by default. In order to disable specific protocols, you need to set the `-Dhazelcast.mc.tls.excludeProtocols` command line argument to a comma separated list of protocols to be excluded from the list of supported protocols. For example, to allow only TLSv1.2, you need to add the following command line argument when starting the Management Center:

```
-Dhazelcast.mc.tls.excludeProtocols=SSLv3,SSLv2Hello,TLSv1,TLSv1.1
```

When you specify the above argument, you should see a line similar to the following in the Management Center log:

```
2017-06-21 12:35:54.856:INFO:oejus.SslContextFactory:Enabled Protocols
[TLSv1.2] of [SSLv2Hello, SSLv3, TLSv1, TLSv1.1, TLSv1.2]
```

3.8.4. Using a Dictionary to Prevent Weak Passwords

In order to prevent certain words from being included in the user passwords, you can start the Management Center with `-Dhazelcast.mc.security.dictionary.path` command line parameter which points to a text file that contains a word on each line. As a result, the user passwords will not contain any dictionary words, making them harder to guess.

The words in the dictionary need to be at least three characters long in order to be used for checking the passwords. The shorter words are ignored to prevent them from blocking the usage of many password combinations. You can configure the minimum length of words by starting the Management Center with `-Dhazelcast.mc.security.dictionary.minWordLength` command line parameter and setting it to a number.

An example to start the Management Center using the aforementioned parameters is shown below:

```
java -Dhazelcast.mc.security.dictionary.path=/usr/MCtext/pwd.txt \
     -Dhazelcast.mc.security.dictionary.minWordLength=3 \
     -jar hazelcast-management-center-4.2020.08.jar
```

3.8.5. Including and/or Excluding Specific Cipher Suites

When you configure TLS you also can provide which cipher suites Management Center can use for establishing TLS connection. You can include cipher suites with `-Dhazelcast.mc.include.cipher.suites` and exclude with `-Dhazelcast.mc.exclude.cipher.suites` system properties during Management Center startup. You can use the exact cipher suite name or a

regular expression. For example:

```
"-Dhazelcast.mc.include.cipher.suites=^SSL_.*$"
"-Dhazelcast.mc.exclude.cipher.suites=^.*_(MD5|SHA|SHA1)$,^TLS_RSA_.*$,^.*_NULL_.*$"
```

3.9. Configuring Logging

The Management Center uses Logback for its logging. By default, it uses the following configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <statusListener class="ch.qos.logback.core.status.NopStatusListener" />

    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>
                %d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level %logger{36} - %msg%n
            </pattern>
        </encoder>
    </appender>

    <logger level="ERROR" name="com.hazelcast" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="INFO" name="com.hazelcast.webmonitor" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="WARN" name="org.eclipse.jetty" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="WARN" name="org.springframework" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="ERROR" name=
"org.springframework.web.servlet.mvc.method.annotation.ExceptionHandlerExceptionResolv
er"
            additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="WARN" name="org.hibernate.validator" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="WARN" name="org.flywaydb" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>
    <logger level="WARN" name="com.zaxxer.hikari" additivity="false">
        <appender-ref ref="STDOUT"/>
    </logger>

    <root level="INFO">
        <appender-ref ref="STDOUT"/>
    </root>
</configuration>
```

To change the logging configuration, you can create a custom Logback configuration file and start the Management Center with the `-Dlogback.configurationFile` option pointing to your configuration file.

For example, you can create a file named `logback-custom.xml` with the following content and set

logging level to `DEBUG`. To use this file as the logging configuration, you need to start the Management Center with the `-Dlogback.configurationFile=/path/to/your/logback-custom.xml` command line parameter:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>
                %d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level %logger{36} - %msg%n
            </pattern>
        </encoder>
    </appender>

    <root level="DEBUG">
        <appender-ref ref="STDOUT"/>
    </root>
</configuration>
```

To write log messages into rolling log files (in parallel with printing them into the console), you can use a similar Logback configuration file:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <property name="pattern" value="%d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level
%logger{36} - %msg%n" />

    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>${pattern}</pattern>
        </encoder>
    </appender>

    <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
        <file>${user.home}/mc-logs/mc.log</file>
        <!-- daily rollover with last 7 days history -->
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
            <fileNamePattern>${user.home}/mc-logs/mc.%d{yyyy-MM-
dd}.log</fileNamePattern>
            <maxHistory>7</maxHistory>
        </rollingPolicy>
        <encoder>
            <pattern>${pattern}</pattern>
        </encoder>
    </appender>

    <root level="INFO">
        <appender-ref ref="STDOUT"/>
        <appender-ref ref="FILE"/>
    </root>
</configuration>
```

3.9.1. Enabling Audit Logging

You may enable additional security audit logging by setting the `hazelcast.mc.auditlog.enabled` system property to `true`. Log entries from the audit logging will be marked with the `hazelcast.auditlog` logging category.

An example log entry looks like the following:

```
2019-11-05 12:16:48 [qtp1551870003-37] INFO  hazelcast.auditlog - MC-2001 [Auth]:User
logged in:{username=JohnHallaign}
```

`MC-2001 [Auth]` you see in this example represents the log's type. The following table lists the current log categories along with their types:

Event Category	Log Type/Description
Management Center Configuration Logs	• `MC-0001 [Config]`: Metrics Persistence is enabled. • `MC-0002 [Config]`: Metrics Persistence is disabled. • `MC-0003 [Config]`: User is created. • `MC-0004 [Config]`: User is edited. • `MC-0005 [Config]`: User's password is changed. • `MC-0006 [Config]`: User is deleted. • `MC-0009 [Config]`: License is set.
Cluster Configuration Logs	• `MC-1001 [Cluster Config]`: Map's configuration is changed. • `MC-1003 [Cluster Config]`: Cluster's state is changed. • `MC-1004 [Cluster Config]`: Cluster is shut down. • `MC-1005 [Cluster Config]`: Member is shut down. • `MC-1006 [Cluster Config]`: Lite member is promoted. • `MC-1007 [Cluster Config]`: Cluster version is changed.
Authentication Logs	• `MC-2001 [Auth]`: User logs in. • `MC-2002 [Auth]`: User logs out. • `MC-2003 [Auth]`: Login failures.
Scripting Logs	• `MC-3001 [Script]`: Script is executed on a member.
Console Logs	• `MC-4001 [Console]`: Console command is executed on the cluster.
Map/Cache Logs	• `MC-5001 [Browser]`: User browses through a map screen in Management Center. • `MC-5002 [Browser]`: User browses through a cache screen in Management Center.
Hot Restart Logs	• `MC-6001 [Hot Restart]`: Force start is run. • `MC-6002 [Hot Restart]`: Partial start is run. • `MC-6003 [Hot Restart]`: Hot Restart backup operation is triggered. • `MC-6004 [Hot Restart]`: Hot Restart backup operation is interrupted.

Event Category	Log Type/Description
WAN Replication Logs	• `MC-7001 [WAN]`: WAN configuration is added. • `MC-7002 [WAN]`: WAN consistency check operation is run. • `MC-7003 [WAN]`: WAN synchronization on a map is run. • `MC-7004 [WAN]`: State of the WAN publisher is changed. • `MC-7005 [WAN]`: Clear operation for the WAN events queue is run.
CP Subsystem Logs	• `MC-8001 [CP Subsystem]`: Member is promoted to be a CP subsystem member. • `MC-8002 [CP Subsystem]`: Member is removed from CP subsystem. • `MC-8003 [CP Subsystem]`: CP subsystem is reset.
Jet Logs	• `MC-9001 [Jet]`: Jet job is restarted. • `MC-9002 [Jet]`: Jet job is suspended. • `MC-9003 [Jet]`: Jet job is resumed. • `MC-9004 [Jet]`: Jet job is cancelled. • `MC-9005 [Jet]`: Jet snapshot is deleted. • `MC-9006 [Jet]`: Jet snapshot is exported. • `MC-9007 [Jet]`: Jet job is cancelled and snapshot is exported.

To write security audit logging into separate rolling log files, you can use a similar Logback configuration file:

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <property name="pattern" value="%d{yyyy-MM-dd HH:mm:ss} [%thread] %-5level
%logger{36} - %msg%n" />

    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>${pattern}</pattern>
        </encoder>
    </appender>

    <appender name="AUDIT_FILE" class="
ch.qos.logback.core.rolling.RollingFileAppender">
        <file>${user.home}/mc-logs/audit.log</file>
        <!-- daily rollover with last 30 days history -->
        <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
            <fileNamePattern>${user.home}/mc-logs/audit.%d{yyyy-MM-
dd}.log</fileNamePattern>
            <maxHistory>30</maxHistory>
        </rollingPolicy>
        <encoder>
            <pattern>${pattern}</pattern>
        </encoder>
    </appender>

    <logger level="INFO" name="hazelcast.auditlog">
        <appender-ref ref="AUDIT_FILE"/>
    </logger>

    <root level="INFO">
        <appender-ref ref="STDOUT"/>
    </root>
</configuration>
```

3.10. Using Variable Replacers

Variable replacers are used to replace custom strings during loading the configuration, either passed as command line arguments, used during UI based Management Center configuration, or configured with Management Center Configuration Tool. They can be used to mask sensitive information such as usernames and passwords. Of course their usage is not limited to security related information.

Variable replacers implement the interface `com.hazelcast.webmonitor.configreplacer.spi.ConfigReplacer` and they are configured via the following command line arguments:

- `-Dhazelcast.mc.configReplacer.class`: Full class name of the replacer.
- `-Dhazelcast.mc.configReplacer.failIfValueMissing`: Specifies whether the loading configuration

process stops when a replacement value is missing. It is an optional attribute and its default value is `true`.

- Additional command line arguments specific to each replacer implementation. All of the properties for the built-in replacers are explained in the upcoming sections.

The following replacer classes are provided by Hazelcast as example implementations of the `ConfigReplacer` interface. Note that you can also implement your own replacers.

- `EncryptionReplacer`
- `PropertyReplacer`

Each example replacer is explained in the following sections.

3.10.1. EncryptionReplacer

This example `EncryptionReplacer` replaces the encrypted variables with its plain form. The secret key for encryption/decryption is generated from a password which can be a value in a file and/or environment specific values, such as MAC address and actual user data.

Its full class name is `com.hazelcast.webmonitor.configreplacer.EncryptionReplacer` and the replacer prefix is `ENC`. Here are the properties used to configure this example replacer:

- `hazelcast.mc.configReplacer.prop.cipherAlgorithm`: Cipher algorithm used for the encryption/decryption. Its default value is AES.
- `hazelcast.mc.configReplacer.prop.keyLengthBits`: Length (in bits) of the secret key to be generated. Its default value is 128.
- `hazelcast.mc.configReplacer.prop.passwordFile`: Path to a file whose content should be used as a part of the encryption password. When the property is not provided, no file is used as a part of the password. Its default value is null.
- `hazelcast.mc.configReplacer.prop.passwordNetworkInterface`: Name of the network interface whose MAC address should be used as a part of the encryption password. When the property is not provided no network interface property is used as a part of the password. Its default value is null.
- `hazelcast.mc.configReplacer.prop.passwordUserProperties`: Specifies whether the current user properties (`user.name` and `user.home`) should be used as a part of the encryption password. Its default value is true.
- `hazelcast.mc.configReplacer.prop.saltLengthBytes`: Length (in bytes) of a random password salt. Its default value is 8.
- `hazelcast.mc.configReplacer.prop.secretKeyAlgorithm`: Name of the secret key algorithm to be associated with the generated secret key. Its default value is AES.
- `hazelcast.mc.configReplacer.prop.secretKeyFactoryAlgorithm`: Algorithm used to generate a secret key from a password. Its default value is PBKDF2WithHmacSHA256.
- `hazelcast.mc.configReplacer.prop.securityProvider`: Name of a Java Security Provider to be used for retrieving the configured secret key factory and the cipher. Its default value is null.

Older Java versions may not support all the algorithms used as defaults. Use the property values supported by your Java version.

As a usage example, let's create a password file and generate the encrypted strings out of this file as shown below:

1. Create the password file: `echo '/Za-uG3dDfpd,5.-' > /opt/master-password`
2. Define the encrypted variables:

```
java -cp hazelcast-management-center-4.2020.08.jar \
     -Dhazelcast.mc.configReplacer.prop.passwordFile=/opt/master-password \
     -Dhazelcast.mc.configReplacer.prop.passwordUserProperties=false \
     com.hazelcast.webmonitor.configreplacer.EncryptionReplacer \
     "aPasswordToEncrypt"

Output:

$ENC{wJxe1vfHTgg=:531:WkAEdSi//YWEbwvVNoU9mUyZ0DE49acJeaJmGalHHfA=}
```

3. Configure the replacer and provide the encrypted variables as command line arguments while starting the Management Center:

```
java \
 -Dhazelcast.mc.configReplacer.class=com.hazelcast.webmonitor.configreplacer
.EncryptionReplacer \
 -Dhazelcast.mc.configReplacer.prop.passwordFile=/opt/master-password \
 -Dhazelcast.mc.configReplacer.prop.passwordUserProperties=false \
 -Dhazelcast.mc.tls.enabled=true \
 -Dhazelcast.mc.tls.keyStore=/opt/mc.keystore \
 -Dhazelcast.mc.tls.keyStorePassword=
'$ENC{wJxe1vfHTgg=:531:WkAEdSi//YWEbwvVNoU9mUyZ0DE49acJeaJmGalHHfA=}' \
 -jar hazelcast-management-center-4.2020.08.jar
```

3.10.2. PropertyReplacer

`PropertyReplacer` replaces variables by properties with the given name. Usually the system properties are used, e.g., `${user.name}`.

Its full class name is `com.hazelcast.webmonitor.configreplacer.PropertyReplacer` and the replacer prefix is empty string ("").

3.10.3. Implementing Custom Replacers

You can also provide your own replacer implementations. All replacers have to implement the three methods that have the same signatures as the methods of the following interface:

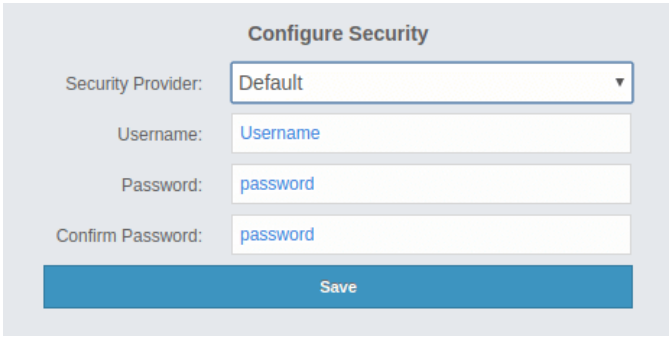
```
import java.util.Properties;

public interface ConfigReplacer {
    void init(Properties properties);
    String getPrefix();
    String getReplacement(String maskedValue);
}
```

4. Launching the Management Center User Interface

If you have the open source edition of Hazelcast, the Management Center can be used for at most three IMDG and Jet members each. To use it for more members, you need to have either a Management Center license, Hazelcast IMDG Enterprise license, Hazelcast IMDG Enterprise HD or Hazelcast Jet Enterprise license. The license should be entered within the Management Center as described in the following paragraphs.

Even if you have IMDG Enterprise, IMDG Enterprise HD or Jet Enterprise license key, and you set it as explained in the Setting the License Key section, you still need to enter this same license within the Management Center. See the following paragraphs to learn how you can do this.

After you downloaded Management Center, configured it as needed and started its service as described in the previous chapters, browse to `http://localhost:8080` on your preferred browser. Since you are going to use the Management Center for the first time, the following page appears:



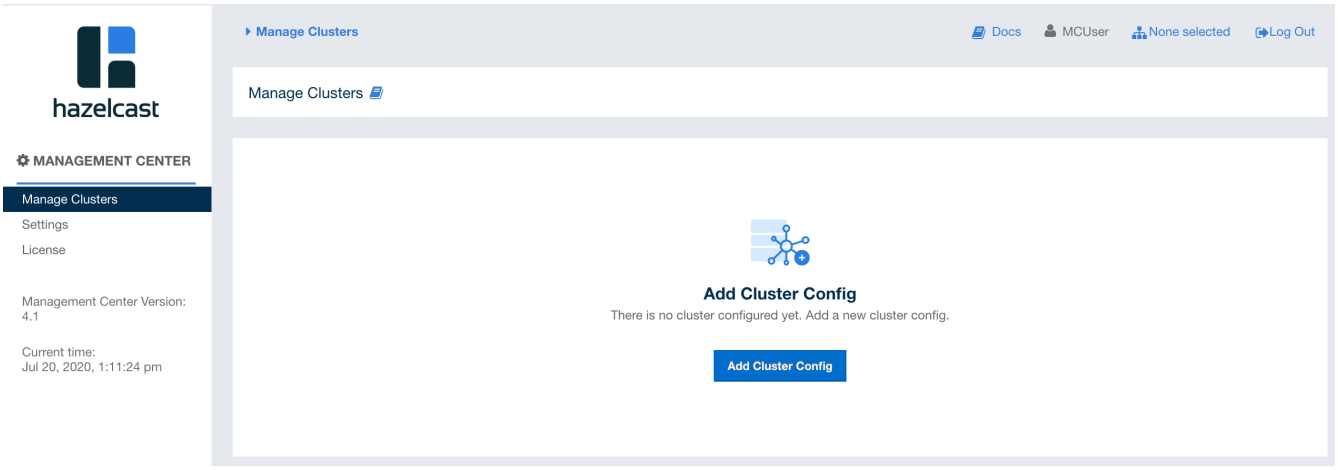
If you already configured security before, a login page appears instead.

It asks you to choose your security provider and create a username and password, and also gives you the option to use Management Center in **Dev Mode** (developer mode).

Available security providers are Default, Active Directory, LDAP and JAAS, which are described in the following subsections. If you choose to use it in Dev Mode, please see this section.

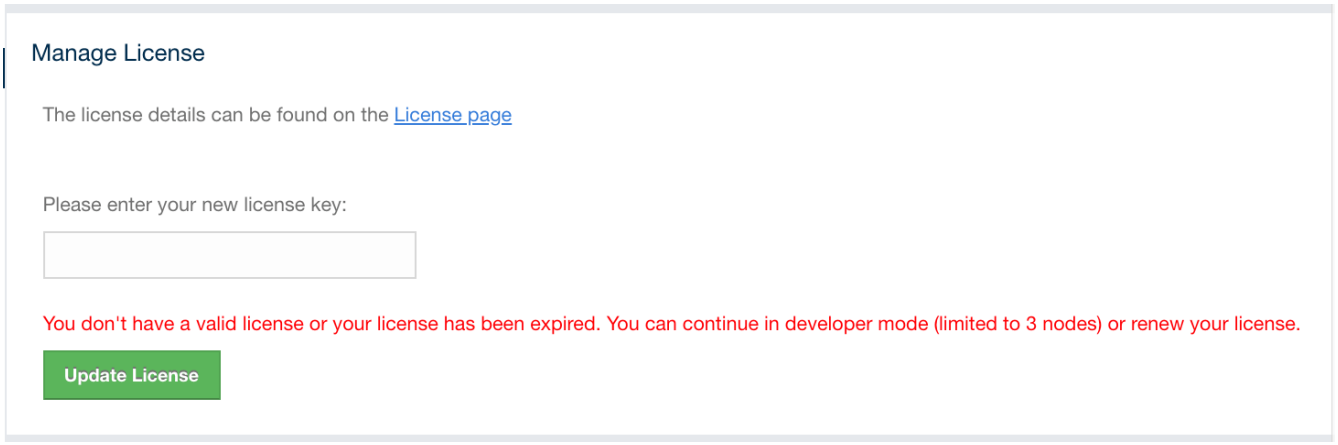
Once you press the **Save** button, your security provider configuration is saved and you can log in with your credentials.

After you successfully login, the following page appears, regardless of whether you created your credentials or selected the development mode (Dev Mode):



Now, you can create the IMDG or Jet cluster configuration. See the Managing Clusters chapter.

If you have either of the aforementioned licenses, you can enter it by clicking on the **Administration** button on the left menu and opening the **Manage License** tab. Here you can enter your license key and press the **Update License** button, as shown below:



Note that a license can likewise be provided using the system property `hazelcast.mc.license` (see the Starting with a License for details).

When you try to connect to a cluster that has more than three members without entering a license key or if your license key is expired, the following warning message is shown at the top:

If you choose to continue without a license, please remember that the Management Center works if your cluster has **at most** three members.

The Management Center creates a folder with the name `hazelcast-mc` under your user's home folder to save data files and above settings/license information. You can change the data folder by setting the `hazelcast.mc.home` system property.

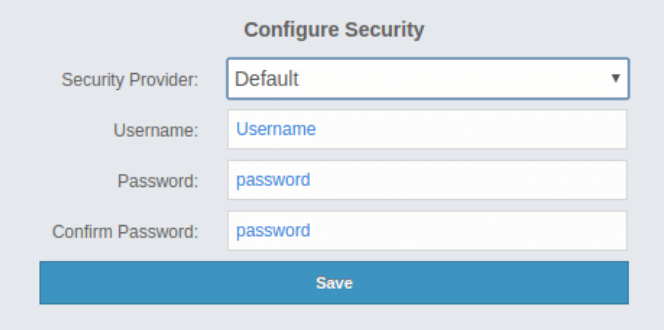
Now, let's see the details of security provider configurations and Dev Mode in the following subsections.

4.1. Authentication Options

As mentioned above, available security providers are Default, Active Directory, LDAP and JAAS. They are described in the following sections.

4.1.1. Default Authentication

You can use the default security provider for authentication/authorization on the Management Center. In this case, the user accounts are stored in the Management Center's database.



Provide the details in this form for the default security provider:

- **Username:** Username for the initial administrator user account.
- **Password, Confirm Password:** Password for the initial administrator user account.

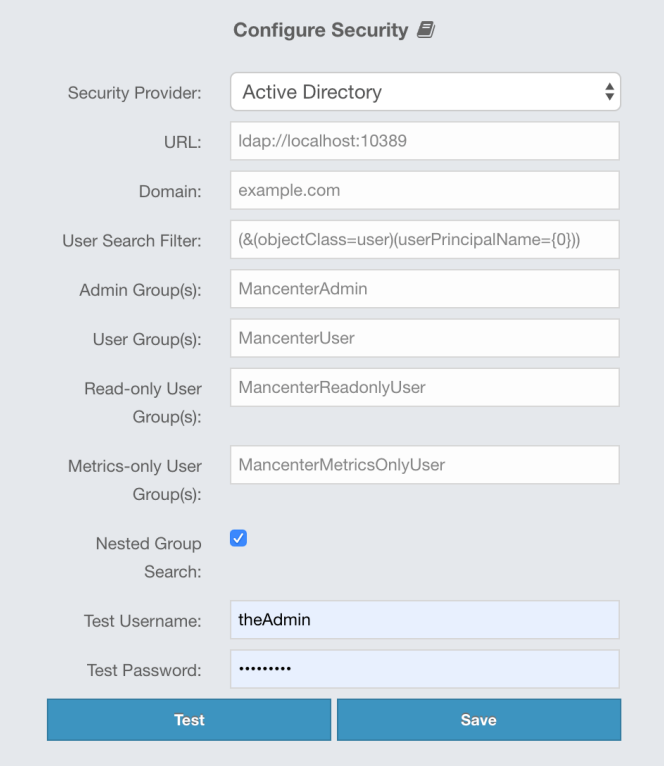
The password you enter should match the following criteria:

- It should be at least 8 characters long.
- It should not include the username.
- It should contain at least one special character, which is not a letter, digit or space.
- It should contain at least one letter.
- It should contain at least one digit.
- It should not contain two or more of the same letter following each other.
- It should not contain a 3-character or longer sequence of letters following or preceeding each other, e.g., "abc" or "fed".
- It should not contain a 3-character or longer sequence of digits following or preceeding each other, e.g., "123" or "987".
- It should not match any of the words listed in the dictionary (you can find more details in the Using a Dictionary to Prevent Weak Passwords section).

You can also use the `create-user` command in the MC Conf tool to configure the default security provider without any UI interactions. See this command's description for details.

4.1.2. Active Directory Authentication

You can use your existing Active Directory server for authentication/authorization on the Management Center. In the "Configure Security" page, select **Active Directory** from the "Security Provider" combo box, and the following form page appears:



Provide the details in this form for your Active Directory server:

- **URL:** URL of your Active Directory server, including the schema (`ldap://` or `ldaps://`) and port.
- **Domain:** Domain of your organization on Active Directory.
- **User Search Filter:** LDAP search filter expression to search for the users. `{0}` will be replaced with `username@domain` and `{1}` will be replaced with only the `username`. You can use both placeholders, only one of them or none in your search filter. For example, `(&(objectClass=user)(userPrincipalName={0}))` searches for a username that matches with the `userPrincipalName` attribute and member of the object class `user`.
- **Admin Group(s):** Members of this group and its nested groups have admin privileges on the Management Center. To use more than one group, separate them with a semicolon (;).
- **User Group(s):** Members of this group and its nested groups have read and write privileges on the Management Center. To use more than one group, separate them with a semicolon (;).
- **Read-only User Group(s):** Members of this group and its nested groups have only read privilege on the Management Center. To use more than one group, separate them with a semicolon (;).
- **Metrics-only Group(s):** Members of this group and its nested groups have the privilege to see only the metrics on Management Center. To use more than one group, separate them with a semicolon (;).
- **Nested Group Search:** Disable if you have a large LDAP group structure and it takes a long time to query all nested groups during login.

- **Test Username:** Username to test the Active Directory configuration with. Note that this value will not be saved and only be used for testing the Active Directory configuration.
- **Test Password:** Password to test the Active Directory configuration with. Note that this value will not be saved and only be used for testing the Active Directory configuration.

Before saving the configuration, you can test it by clicking the **Test** button. Note that the user you test with needs to be a member of one of the groups you have configured for the Management Center.

You can use the `hazelcast.mc.ldap.timeout` system property to specify both connect and read timeout values for Active Directory search queries. It is in milliseconds and its default value is `3000` milliseconds.

Configuration Hot Reload

Once configured, Active Directory settings are saved in a local database managed by Management Center. If you need to update your settings afterwards, you need to provide the import properties file under `<hazelcast-mc>/import/securityHotReload.properties`, and then click on the **Reload Security Config** button on the login page. The `securityHotReload.properties` should contain the following properties:

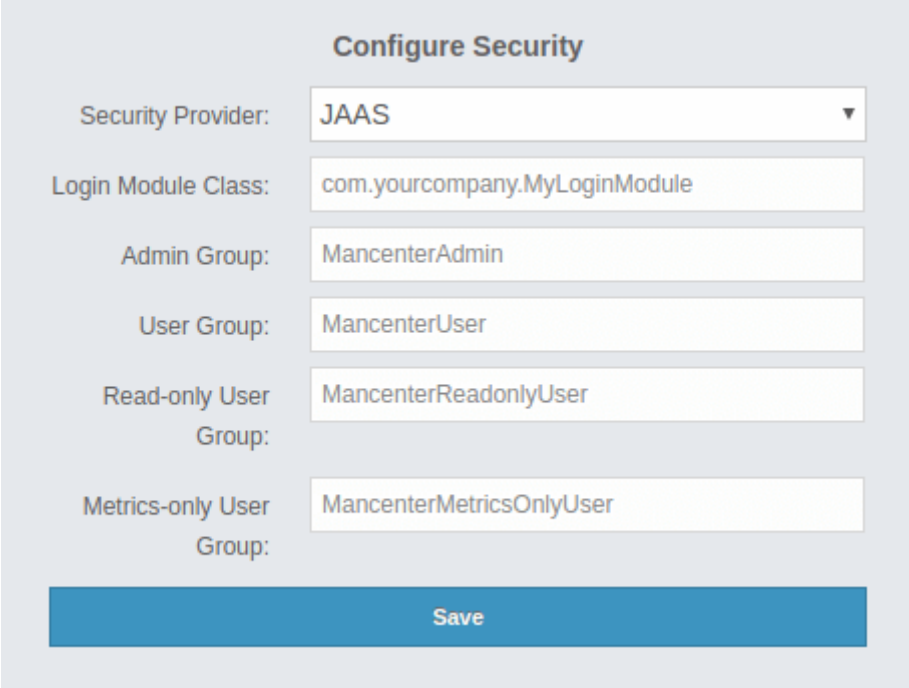
```
url=<active directory instance url>
domain=<domain>
adminGroup=<Admin group(s). Use ';' to separate multiple groups>
userGroup=<Read-write group(s). Use ';' to separate multiple groups>
readonlyUserGroup<Read-only group(s). Use ';' to separate multiple groups>
metricsOnlyGroup<Metrics-only group(s). Use ';' to separate multiple groups>
```

The **Reload Security Config** button will only appear when the `<hazelcast-mc>/import/securityHotReload.properties` file is present. After a successful import, the file will be renamed as `importedSecurityHotReload-<import_timestamp>.properties.bak`.

Alternatively, you can use MC-Conf tool's `security reset` and `active-directory configure` tasks to configure the Active Directory security provider from scratch, but you need to stop the Management Center service for this configuration option. See the Management Center Configuration Tool section for more information.

4.1.3. JAAS Authentication

You can use your own `javax.security.auth.spi.LoginModule` implementation for authentication/authorization on the Management Center. In the "Configure Security" page, select **JAAS** from the "Security Provider" combo box, and the following page appears:



Provide the details in this form for your JAAS `LoginModule` implementation:

- **Login Module Class**: Fully qualified class name of your `javax.security.auth.spi.LoginModule` implementation.
- **Admin Group:** Members of this group have admin privileges on the Management Center.
- **User Group:** Members of this group have read and write privileges on the Management Center.
- **Read-only User Group:** Members of this group have only read privilege on the Management Center.
- **Metrics-only Group:** Members of this group have the privilege to see only the metrics on the Management Center.

The following is an example implementation. Note that we return two `java.security.Principal` instances; one of them is the username and the other one is a group name, which you will use when configuring JAAS security as described above.

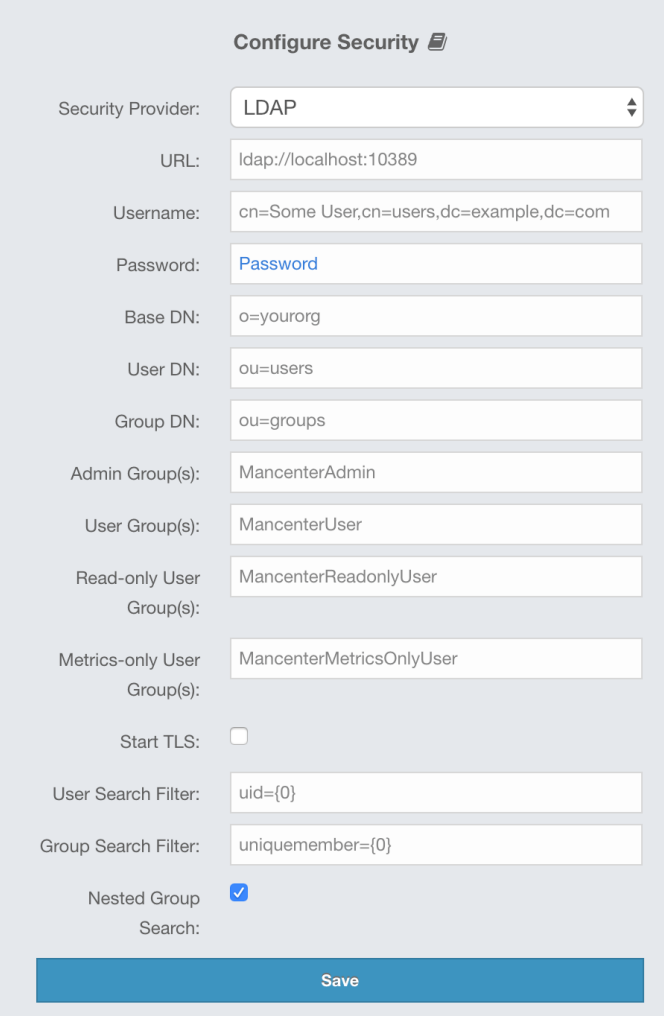
```
import javax.security.auth.Subject;
import javax.security.auth.callback.Callback;
import javax.security.auth.callback.CallbackHandler;
import javax.security.auth.callback.NameCallback;
import javax.security.auth.callback.PasswordCallback;
import javax.security.auth.login.LoginException;
import javax.security.auth.spi.LoginModule;
import java.security.Principal;
import java.util.Map;

public class SampleLoginModule implements LoginModule {
    private Subject subject;
    private String password;
    private String username;
```

```
    @Override
    public void initialize(Subject subject, CallbackHandler callbackHandler, Map
<String, ?> sharedState, Map<String, ?> options) {
        this.subject = subject;

        try {
            NameCallback nameCallback = new NameCallback("prompt");
            PasswordCallback passwordCallback = new PasswordCallback("prompt", false);

            callbackHandler.handle(new Callback[] {nameCallback, passwordCallback });

            password = new String(passwordCallback.getPassword());
            username = nameCallback.getName();
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    @Override
    public boolean login() throws LoginException {
        if (!username.equals("emre")) {
            throw new LoginException("Bad User");
        }

        if (!password.equals("pass1234")) {
            throw new LoginException("Bad Password");
        }

        subject.getPrincipals().add(new Principal() {
            public String getName() {
                return "emre";
            }
        });

        subject.getPrincipals().add(new Principal() {
            public String getName() {
                return "HazelcastMCAdmin";
            }
        });

        return true;
    }

    @Override
    public boolean commit() throws LoginException {
        return true;
    }

    @Override
    public boolean abort() throws LoginException {
        return true;
```

```
    }

    @Override
    public boolean logout() throws LoginException {
        return true;
    }
}
```

4.1.4. LDAP Authentication

You can use your existing LDAP server for authentication/authorization on the Management Center. In the "Configure Security" page, select **LDAP** from the "Security Provider" combo box, and the following form page appears:



Provide the details in this form for your LDAP server:

- **URL:** URL of your LDAP server, including schema (`ldap://` or `ldaps://`) and port.
- **Distinguished name (DN) of user:** DN of a user that has admin privileges on the LDAP server. It is used to connect to the server when authenticating users.
- **User DN:** DN to be used for searching users.
- **Group DN:** DN to be used for searching groups.
- **Admin Group(s):** Members of this group and its nested groups have admin privileges on the

Management Center. To use more than one group, separate them with a semicolon (;).

- **User Group(s):** Members of this group and its nested groups have read and write privileges on the Management Center. To use more than one group, separate them with a semicolon (;).
- **Read-only User Group(s):** Members of this group and its nested groups have only read privilege on the Management Center. To use more than one group, separate them with a semicolon (;).
- **Metrics-only Group(s):** Members of this group and its nested groups have the privilege to see only the metrics on the Management Center. To use more than one group, separate them with a semicolon (;).
- **Start TLS:** Enable if your LDAP server uses **Start TLS** operation.
- **User Search Filter:** LDAP search filter expression to search for the users. For example, `uid={0}` searches for a username that matches with the `uid` attribute.
- **Group Search Filter:** LDAP search filter expression to search for the groups. For example, `uniquemember={0}` searches for a group that matches with the `uniquemember` attribute.
- **Nested Group Search:** Disable if you have a large LDAP group structure and it takes a long time to query all nested groups during login.

Values for **Admin**, **User**, **Read-only** and **Metrics-Only** group names must be given as plain names. They should not contain any LDAP attributes such as `CN`, `OU` and `DC`.

You can use the `hazelcast.mc.ldap.timeout` system property to specify connect and read timeout values for LDAP search queries. It is in milliseconds and its default value is `3000` milliseconds.

Configuration Hot Reload

Once configured, LDAP settings are saved in a local database managed by Management Center. If you need to update your settings afterwards, you need to provide the import properties file under `<hazelcast-mc>/import/securityHotReload.properties`, and then click on the **Reload Security Config** button on the login page. The `securityHotReload.properties` should contain the following properties:

```
url=<Ldap URL>
username=<Distinguished name (DN) of user>
password=<password>
userDn=<User DN>
groupDn<Group DN>
startTls=<true|false>
adminGroup=<Admin group(s). Use ';' to separate multiple groups>
userGroup=<Read-write group(s). Use ';' to separate multiple groups>
readonlyUserGroup<Read-only group(s). Use ';' to separate multiple groups>
metricsOnlyGroup<Metrics-only group(s). Use ';' to separate multiple groups>
userSearchFilter=<User Search Filter>
groupSearchFilter=<Group Search Filter>
nestedGroupSearch=<true|false>
```

The **Reload Security Config** button will only appear when the `<hazelcast-mc>/import/securityHotReload.properties` file is present. After a successful import, the file will be renamed as `importedSecurityHotReload-<import_timestamp>.properties.bak`.

Alternatively, you can use MC-Conf tool's `security reset` and `active-directory configure` tasks to configure the Active Directory security provider from scratch, but you need to stop the Management Center service for this configuration option. See the Management Center Configuration Tool section for more information.

Enabling TLS/SSL for LDAP

If your LDAP server is using `ldaps` (LDAP over SSL) protocol or the **Start TLS** operation, use the following command line parameters for your Management Center deployment:

- `-Dhazelcast.mc.ldap.ssl.trustStore`: Path to the truststore. This truststore needs to contain the public key of your LDAP server.
- `-Dhazelcast.mc.ldap.ssl.trustStorePassword`: Password of the truststore.
- `-Dhazelcast.mc.ldap.ssl.trustStoreType`: Type of the truststore. Its default value is JKS.
- `-Dhazelcast.mc.ldap.ssl.trustManagerAlgorithm`: Name of the algorithm based on which the authentication keys are provided. System default is used if none is provided. You can find out the default by calling the `javax.net.ssl.TrustManagerFactory#getDefaultAlgorithm` method.

Password Encryption

By default, the password that you use in the LDAP configuration is stored in a plain text in a local database. This might pose a security risk. To store the LDAP password in an encrypted form, we offer the following options:

- **Provide a keystore password:** This creates and manages a Java keystore under the Management Center home directory. The LDAP password is stored in this keystore in an encrypted form.
- **Configure an external Java keystore:** This uses an existing Java keystore. This option might also be used to store the password in an HSM that provides a Java keystore API.

In the case of using either one of the options, the LDAP password you enter on the initial configuration UI dialog will be stored in an encrypted form in a Java keystore instead of the local database.

You can also encrypt the password, and provide it in an encrypted form, when configuring LDAP security provider. See the Variable Replacers section for more information.

Providing a Master Key for Encryption

There are two ways to provide a master key for encryption:

- If you deploy the Management Center on an application server, you need to set the

`MC_KEYSTORE_PASS` environment variable before starting Management Center. This option is less secure. You should clear the environment variable once you make sure you can log in with your LDAP credentials to minimize the security risk.

- If you're starting the Management Center from the command line, you can start it with `-Dhazelcast.mc.askKeyStorePassword`. The Management Center asks for the keystore password upon start and use it as a password for the keystore it creates. This option is more secure as it only stores the keystore password in the memory.

By default, the Management Center creates a Java keystore file under the Management Center home directory with the name `mc.jceks`. You can change the location of this file by using the `-Dhazelcast.mc.keyStore.path=/path/to/keyStore.jceks` JVM argument.

Configuring an External Java KeyStore

If you don't want the Management Center to create a keystore for you and use an existing one that you've created before (or an HSM), set the following JVM arguments when starting the Management Center:

- `-Dhazelcast.mc.useExistingKeyStore=true`: Enables use of an existing keystore.
- `-Dhazelcast.mc.existingKeyStore.path=/path/to/existing/keyStore.jceks`: Path to the keystore. You do not have to set it if you use an HSM.
- `-Dhazelcast.mc.existingKeyStore.pass=somepass`: Password for the keystore. You do not have to set it if HSM provides another means to unlock HSM.
- `-Dhazelcast.mc.existingKeyStore.type=JCEKS`: Type of the keystore.
- `-Dhazelcast.mc.existingKeyStore.provider=com.yourprovider.MyProvider`: Provider of the keystore. Leave empty to use the system provider. Specify the class name of your HSM's `java.security.Provider` implementation if you use an HSM.

Make sure your keystore supports storing `SecretKey`s.

Updating Encrypted Passwords

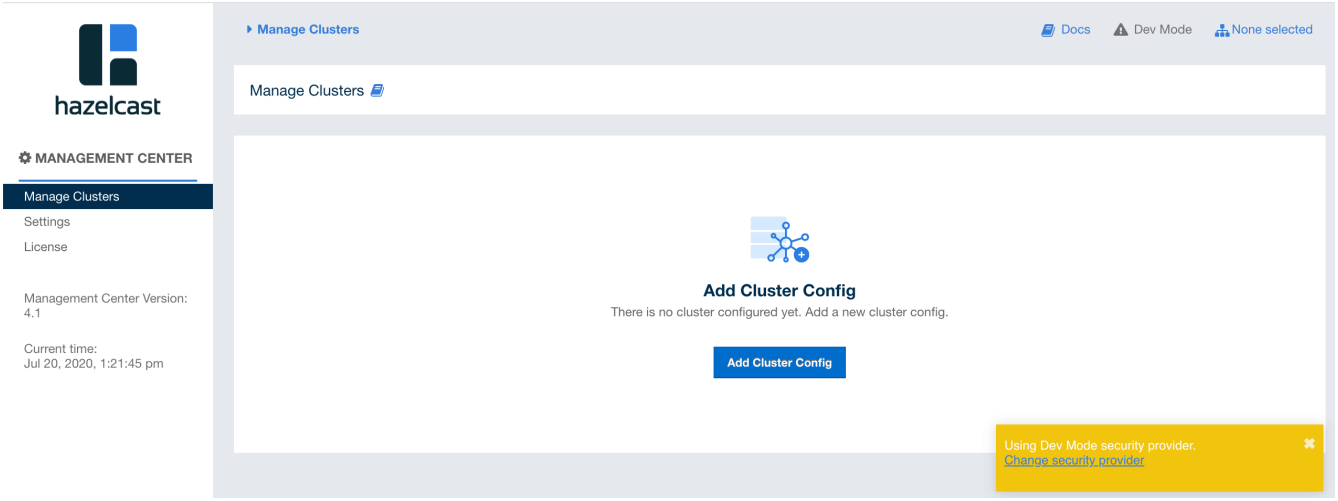
You can use the `update-ldap-password` command in the MC Conf tool to update the encrypted LDAP password stored in the keystore. See this command's description for details.

4.2. Dev Mode

As mentioned previously, Management Center gives you to option to use it in its development mode (Dev Mode). You can select this mode when you launch Management Center, see the introduction of this chapter.

Dev Mode is for the Hazelcast clusters running on your local for development or evaluation purposes and it provides quick access to the Management Center without requiring any security credentials. Keep in mind that this mode should not be used with production clusters.

Once you select "Dev Mode" after you launch Management Center, the following page appears:



The page is similar to the one when you configure one of the authentication options described in the previous sections. Only difference is that it gives you the option to change the security provider whenever you need using the yellow box as shown above. You can also use the "Dev Mode" item located on top right menu of the page for the same purpose.

5. Connecting to the Clusters

After you start and/or configure the Management Center service as explained in the Starting the Management Center Service and Configuring the Management Center chapters, make sure that `http://localhost:8080` is up.

To connect the Management Center to your clusters, you have two options to be performed in Management Center:

1. You can provide the IP addresses or hostnames of one or more IMDG and Jet members.
2. Or, you can upload a configuration file in XML or YAML format. Management Center starts a client for each of IMDG and Jet clusters. This configuration file is for that Management Center client. See here for details on this configuration file. You can think of Management Center as a client connecting to the Hazelcast IMDG and Jet clusters.

If you enabled TLS/SSL on a Hazelcast cluster, then you need to perform the 2. step listed above, i.e., upload a configuration file. This configuration file needs to contain the parameters same as the ones you provide in the case when a client connects to a TLS/SSL enabled Hazelcast cluster. See here for details on TLS/SSL configurations.

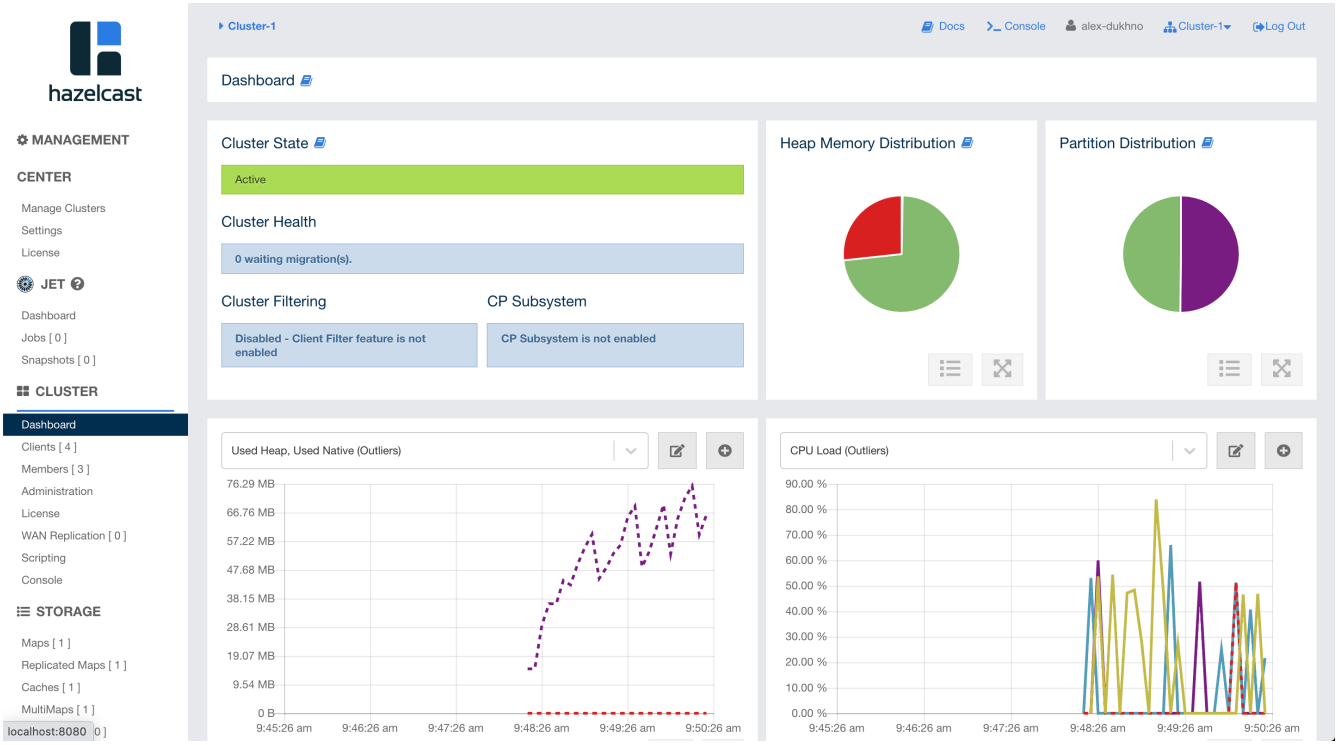
When connecting to multiple clusters from the same Management Center, please ensure that the cluster names are unique. Management Center does not support simultaneous connections to more than one cluster with the same cluster name.

If the cluster uses Advanced Network, then the provided member address should be the client address of the member.

6. User Interface Overview

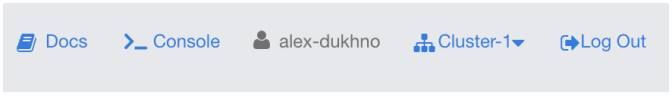
In this chapter, we briefly go over the toolbar and left menu items of the user interface itself, regardless of which cluster we monitor (IMDG or Jet). For now, you can ignore the body content which shows the cluster data. Those are explained in the following chapters throughout this document.

The following is the Management Center user interface:



The user interface has a toolbar on the top and a menu on the left.

6.1. Toolbar



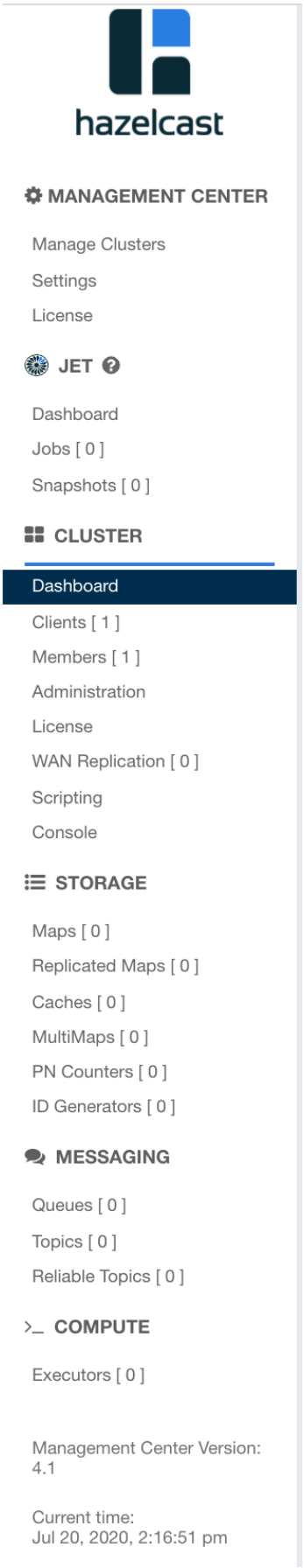
The toolbar has the following elements:

- **Docs**: Opens the Management Center documentation in a new browser tab.
- **Console**: Opens the in-page console so that you can easily execute commands in your cluster. See the Executing Console Commands chapter. Note that this top menu item appears only when you select a cluster to monitor.
- **User name**: Shows the current user's name. When you hover your mouse cursor onto it, the user's last login time is shown as a tooltip, for security purposes.
- **Cluster Selector**: Switches between the clusters. When clicked on, a dropdown list of available clusters appears. You can select any cluster; once selected, the page immediately loads with the selected cluster's information. This selector has the same functionality as selecting a cluster in the "Manage Clusters" menu of the user interface.

- **Logout**: Closes the current user's session.

6.2. Menu

The Management Center user interface includes a menu on the left which lists the data structures and jobs in your clusters, cluster members and clients connected to the cluster (numbers in square brackets show the instance count for each entity), as shown below. You can also see an overview state of your cluster, execute commands and scripts, and perform user/license operations using this menu:



WAN Replication button is only visible with the Hazelcast IMDG Enterprise license.

The menu consists of the following parent menu headings:

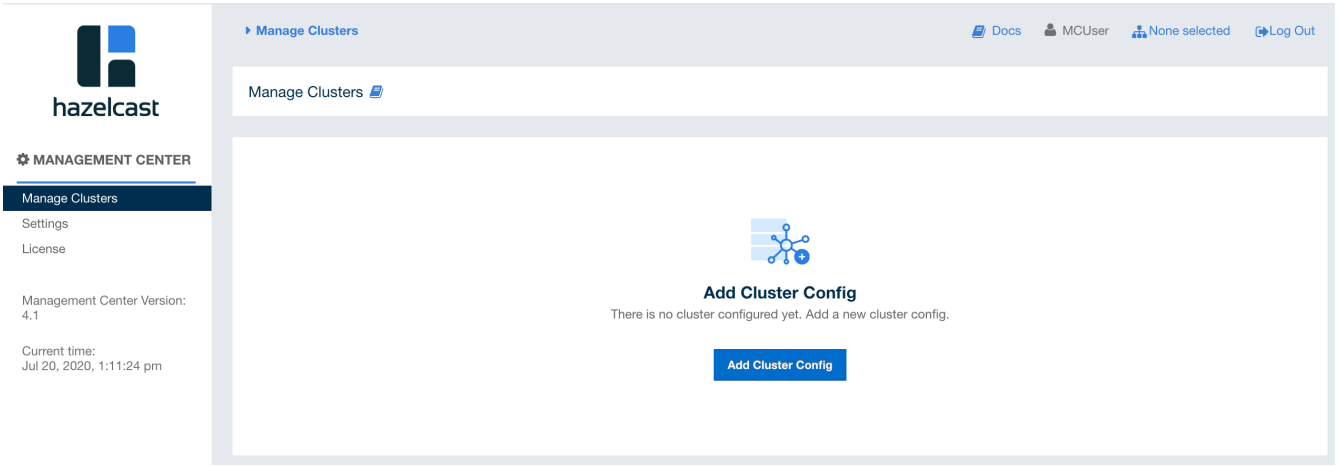
- **MANAGEMENT CENTER**: Includes menu items to manage clusters, user and license operations, and examine the user interface logs.
- **JET**: Includes menu items to export snapshots and manage Jet jobs.
- **CLUSTER**: Includes menu items for general cluster operations, monitoring clients/cluster members and executing scripts/commands.
- **STORAGE**: Includes menu items to monitor data structures such as the maps, caches and PN counters in your cluster.
- **MESSAGING**: Includes menu items to monitor queues and topics in your cluster.
- **COMPUTE**: Includes the "Executors" menu item to monitor the executor services in your cluster.

If you have not selected a cluster to monitor, only the "Management Center" menu items are shown in the user interface.

The following sections describe each menu item from top to bottom.

7. Managing Clusters

"Manage Clusters" is the first page that appears after you login to Management Center for the first time. It is used to add, remove or edit cluster configurations, and looks like the following:



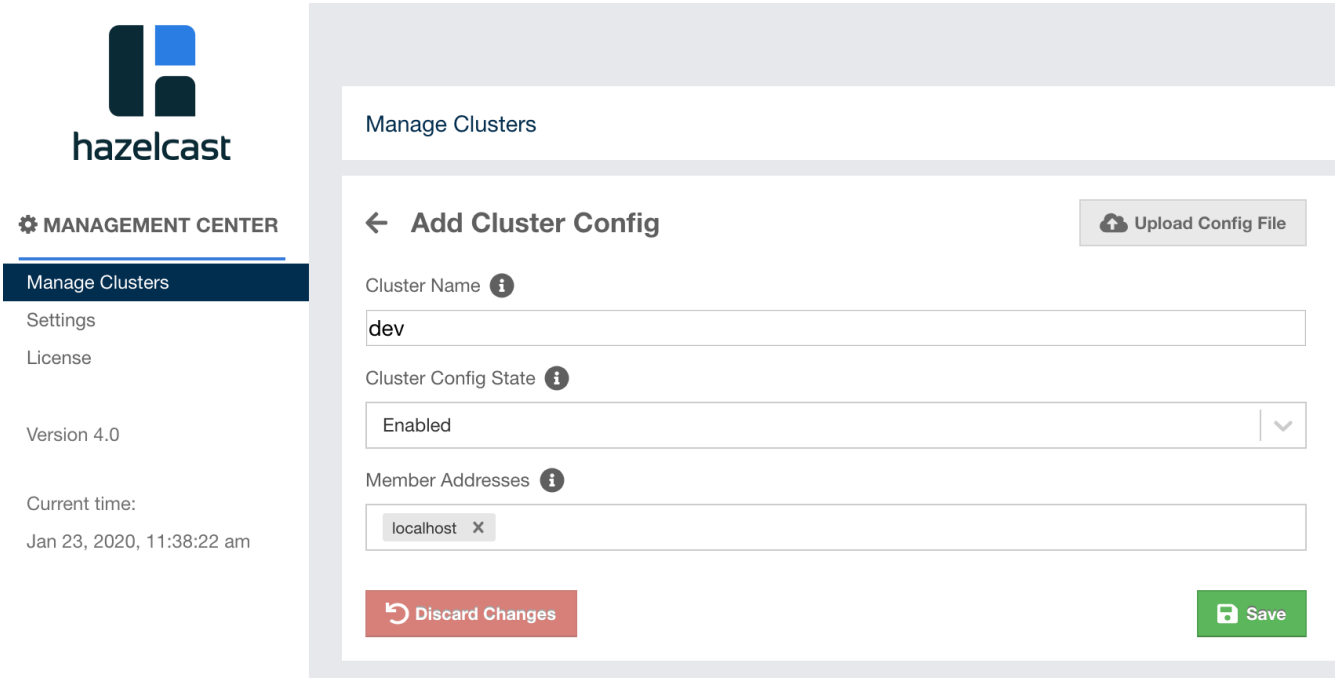
If you are using Management Center for the first time, you need to add your cluster's configuration using this page You can always go to this page during your Management Center usage by clicking on the "Manage Clusters" menu item under the "Management Center" parent menu heading.

To start monitoring your cluster you have already created (on your local machine, another development environment, etc.) you need to first define a cluster configuration to connect Management Center to your cluster. After you click on the "Add Cluster Config" button shown in the above screenshot, you can do this by either of the following options:

- Using a form by providing the cluster name and member addresses
- Uploading a configuration file

7.1. Creating a Cluster Configuration Using Form

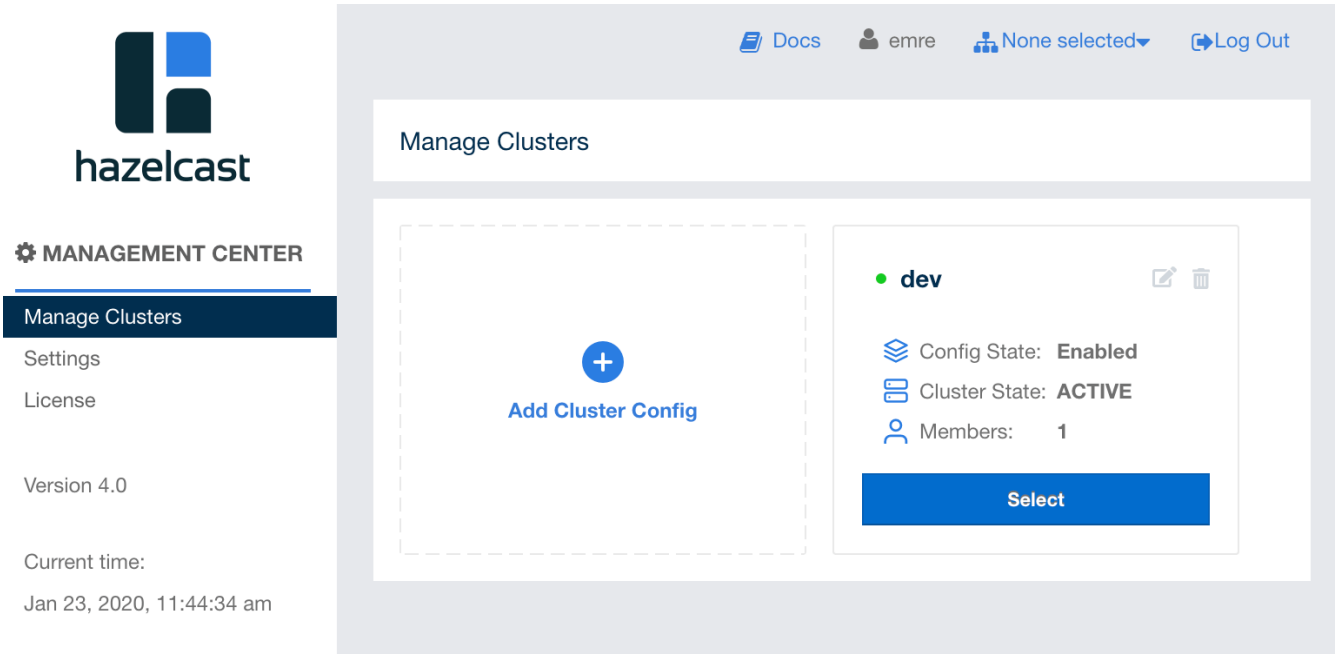
Click on the "Add Cluster Config" button as seen in the above screenshot. The following page appears:



- `Cluster Name`: Enter the name of your cluster, which was defined when the cluster was created. (The default cluster name is `dev` for IMDG clusters and `jet` for Jet Clusters)
- `Cluster Config State`: Select "Enabled" to activate the cluster configuration. You can also select "Disabled" which will create the cluster configuration in a deactivated state for use at a later time. Disabled cluster configurations are ignored until enabled.
- `Member Addresses`: Give the cluster members' addresses (one or preferably more) in the form of `hostname`, `hostname:port`, `IP address` or `IP address:port`. If no port is provided, the system tries the ports through 5701 and 5703 in a random order. Type the member address and press enter (or return) to make it visible in the "Member Addresses" field.

As soon as Management Center connects to one of the members, it will auto-discover the other members of the cluster. It is recommended to enter more than one member address in case that member in the configuration is no longer available in the future.

Click on the `Save` button and the "Manage Clusters" page loads with the new cluster configuration added, as shown below:

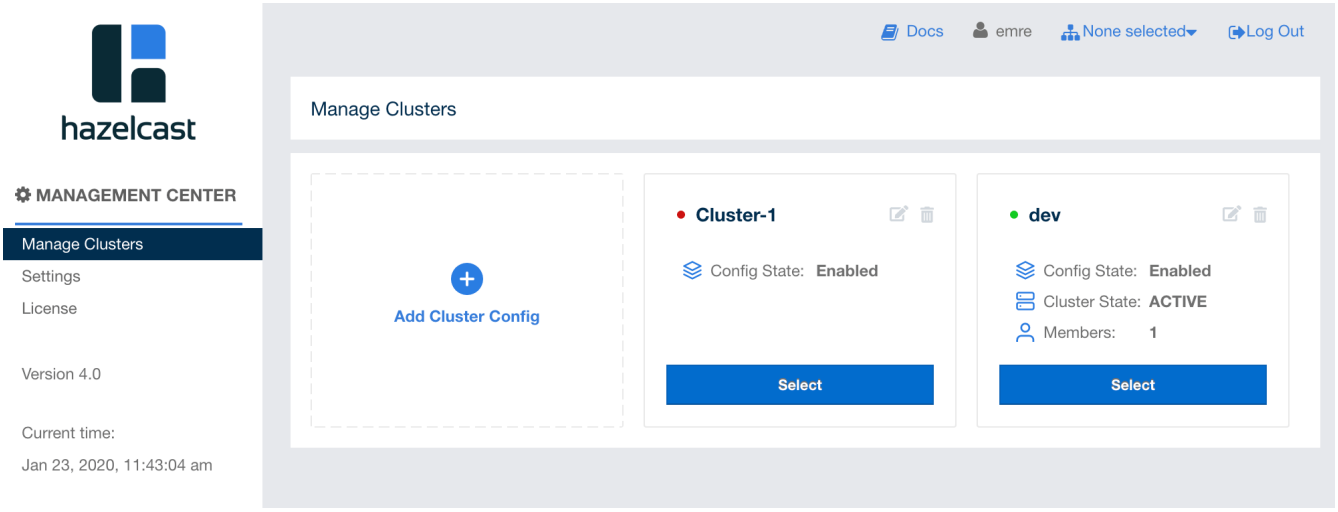


Here, you can see the state of the cluster and member count. Also, at the top left of the cluster box, you see the green colored dot, meaning that Management Center is connected to your cluster.

You can add multiple cluster configurations using the same mechanism as explained above.

When connecting to multiple clusters from the same Management Center, please ensure that the cluster names are unique. Management Center does not support simultaneous connections to more than one cluster with the same cluster name.

The following page shows an example where Management Center is and is not able to connect one of clusters:



You can edit or delete a cluster configuration using the icons on any cluster's box area in this page.

To see the status of a cluster, click on its `Select` button. Management Center internally decides whether the selected one is an IMDG cluster or a Jet one, and shows the related "Dashboard" page of that cluster.

The "Dashboard" page of the related cluster will load; see the Dashboard Page chapter for the details. Clicking on the `Select` button and selecting a cluster in the cluster selector (located at the

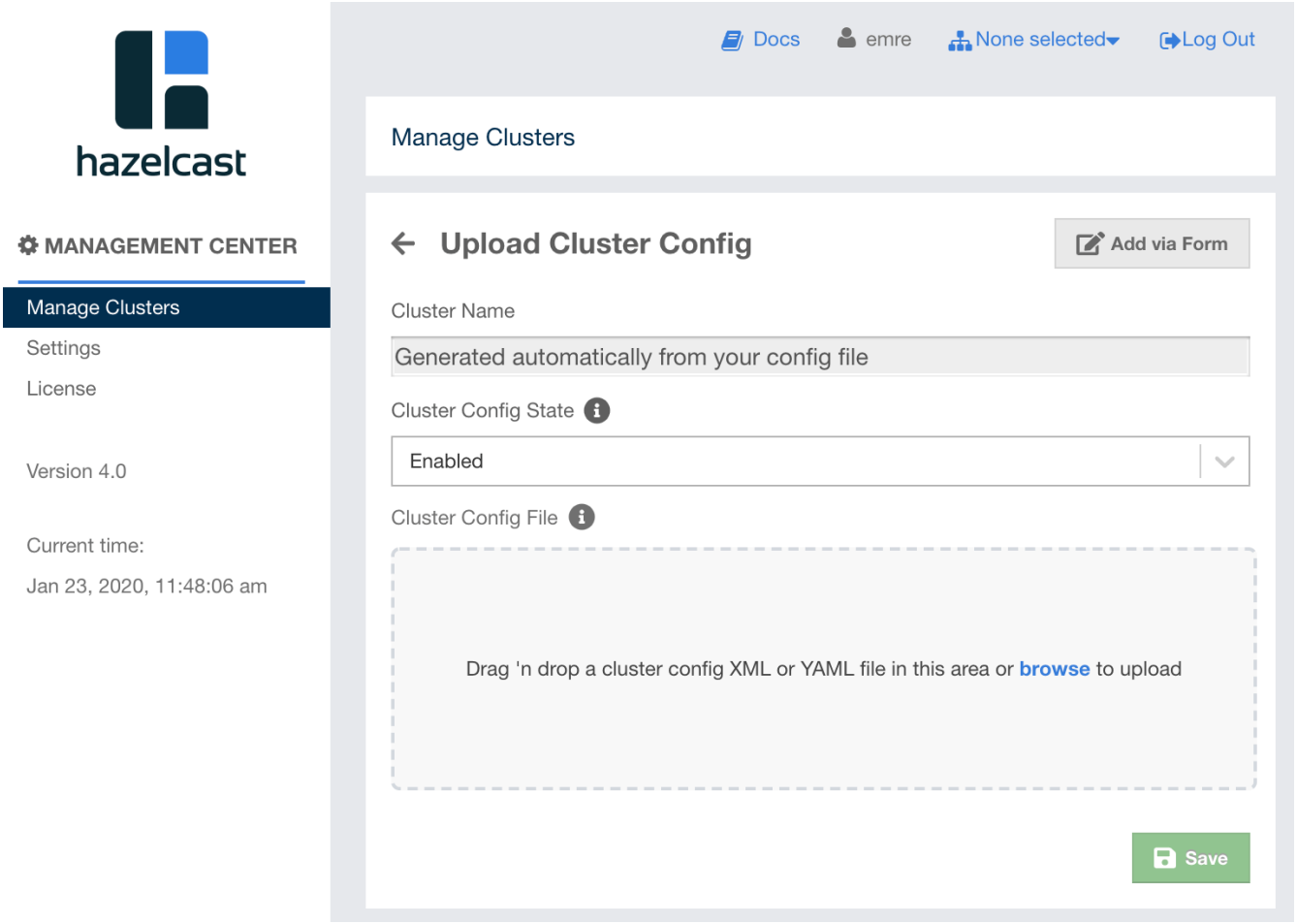
toolbar) have the same functionality.

7.2. Creating a Cluster Configuration by Uploading File

Alternatively, you can upload a configuration file for the Management Center client created for your IMDG cluster. For this, click on the `Upload Config File` button which is seen after you click on the "Add Cluster Config" button.

This is the recommended method for more complex configurations.

The following page appears:



The name of your cluster will be automatically generated as specified in the configuration file you upload. As mentioned before in the connecting members chapter, you can think of Management Center as a client that connects to your IMDG cluster you create. Therefore, this configuration file is a well-known `hazelcast.client` in XML or YAML language.

Here is an example configuration file:

Example hazelcast-client.xml file

```
<hazelcast-client>
    <cluster-name>dev</cluster-name>
    <network>
        <cluster-members>
            <address>127.0.0.1</address>
            <address>127.0.0.2</address>
        </cluster-members>
    </network>
</hazelcast-client>
```

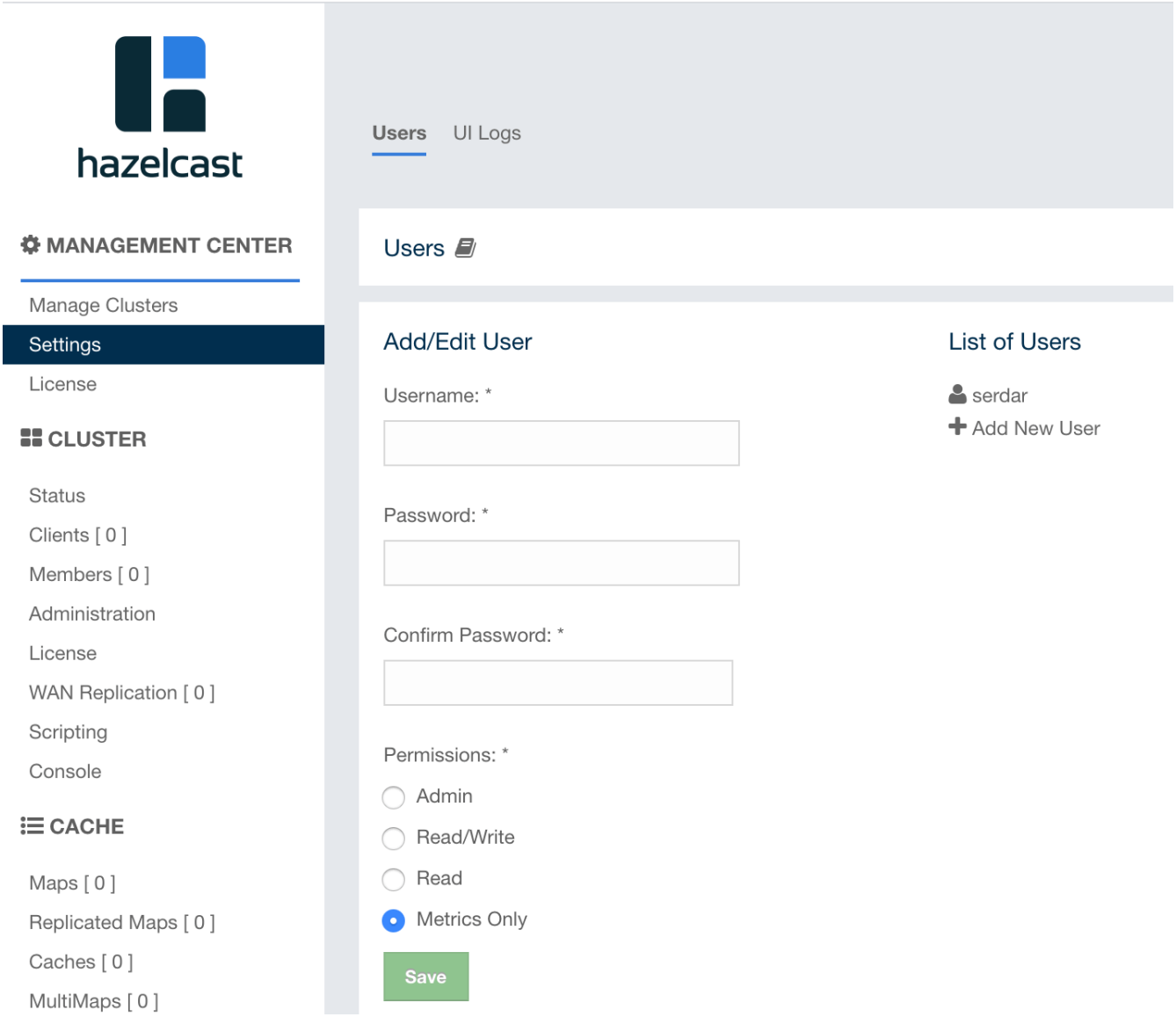
You can simply drag this file onto the `Cluster Config File` box in this page, or find it using the `browse` link in the same box.

Select "Enabled" in the `Cluster Config State` combobox to activate the configuration. You can also select "Disabled" just to create the configuration and use it at a later time. In this case, the configuration is ignored until enabled.

You can always go back to the form page (explained in the previous section) by clicking on the `Add via Form` on this page.

8. Managing Users

User management is only available for the default security provider. See the Default Authentication section for more information.



To add a user to the system, specify the username, e-mail and password in the **Add/Edit User** part of the page. Then provide the user's privilege status using the checkboxes under `Permissions`:

- **Admin**: Select if the user to be added will have the administrator privileges.
- **Read/Write**: If this permission is given to the user, **Home**, **Scripting**, **Console** and **Documentation** items will be visible. The users with this permission can update a map configuration and perform operations on the members.
- **Read**: If this permission is given to the user, only **Home** and **Documentation** items will be visible at the toolbar on that user's session. Also, the users with this permission cannot update a map configuration, run a garbage collection and take a thread dump on a cluster member, or shutdown a member (see the Monitoring Members section).
- **Metrics Only**: If this permission is given to the user, only **Home** and **Documentation** items will be visible at the toolbar on that user's session. Also, the users with this permission cannot browse a map or a cache to see their contents, cannot update a map configuration, run a garbage collection and take a thread dump on a cluster member, or shutdown a member (see the Monitoring Members section).

After you enter/select all the fields, click on the **Save** button to create the user. You will see the newly created user's username on the right side, in the **List of Users** part of the page.

To edit or delete a user, click on its username listed in the **List of Users**. Selected user information appears on the left side of the page. To update the user information, change the fields as desired and click on the **Save** button; note that you can only change the username and/or its privilege.

You can also change a user's password or delete the user account. To change the user's password, click on the **Change Password** button while editing the user. To delete the user from the system, click on the **Delete** button. Note that changing the password of a user and deleting the user account both require you to enter your own password.

Certain user management operations are also available via the MC Conf tool. See the Management Center Configuration Tool section for more information.

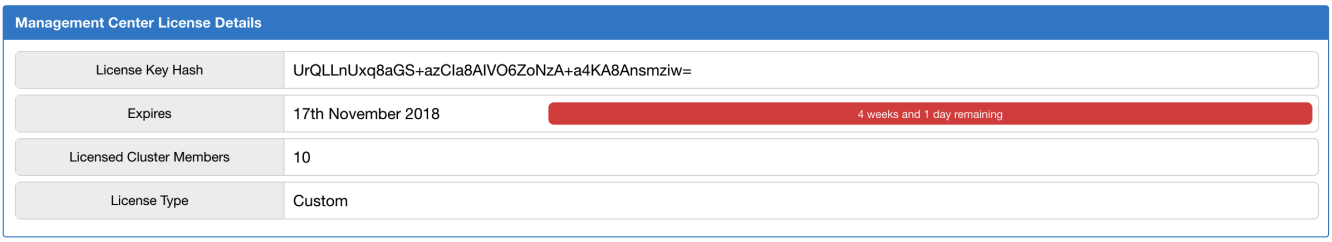
9. Managing Licenses

You can manage the following types of licenses in Management Center:

- License for Management Center
- License for your Hazelcast product

9.1. Management Center License

Using the "License" menu item under the "Management Center" parent menu heading, you can view the details of your Management Center license. An example screenshot is shown below.



It shows the expiration date, total licensed member count and type of your Management Center license.

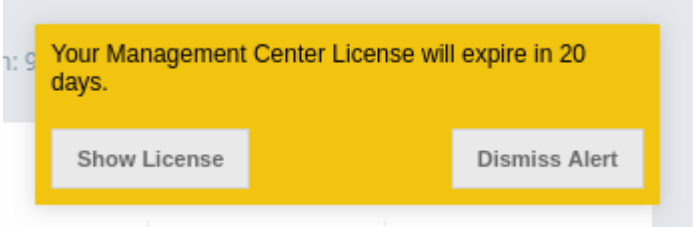
For security reasons, the license key itself is not shown. Instead, SHA-256 hash of the key as a Base64 encoded string is shown.

If there are any problems related to your Management Center license, "License" menu item is highlighted with red exclamation points. In this case, please check this screen to see what the problem is. The following are the possible problems:

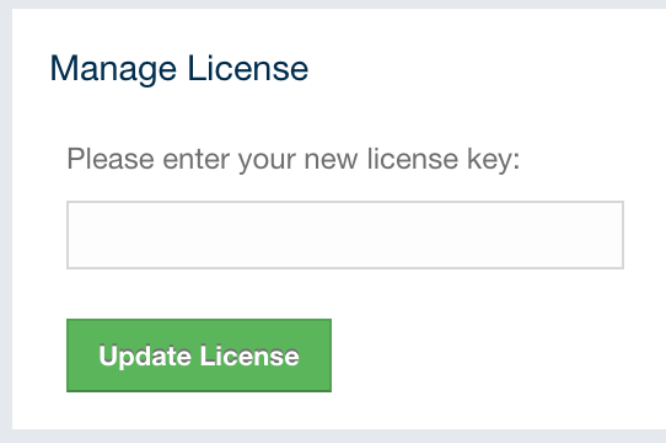
- Your Management Center license is expired.
- The count of your cluster members is more than the allowed count by the license.

If the Management Center license expires in 30 days or less, or has already expired, a warning will appear in the upper right corner once a day. Warning will contain time remaining until license expires or how long the license is expired. There will also be two buttons - the first one with label "Show License" will redirect you to the license page, the second one with label "Dismiss Alert" will

dismiss the alert. An example screenshot is shown below.



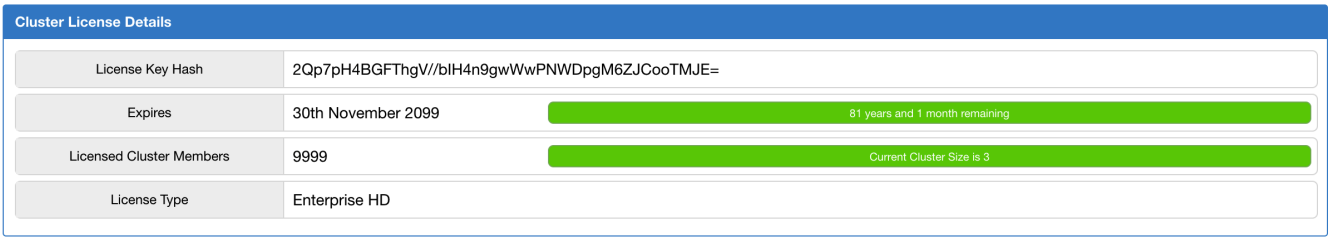
To update the Management Center license, you can open the **Manage License** tab and click on the **Update License** button and enter the new license code.



Alternatively, a license can be provided using the `hazelcast.mc.license` system property (see the Starting with a License section for details).

9.2. Cluster License

Using the "License" menu item under the "Cluster" parent menu heading, you can view the details of your cluster's license (Hazelcast IMDG license). An example screenshot is shown below.



It shows the expiration date, total licensed member count and type of your Hazelcast IMDG license.

For security reasons, the license key itself is not shown. Instead, SHA-256 hash of the key as a Base64 encoded string is shown.

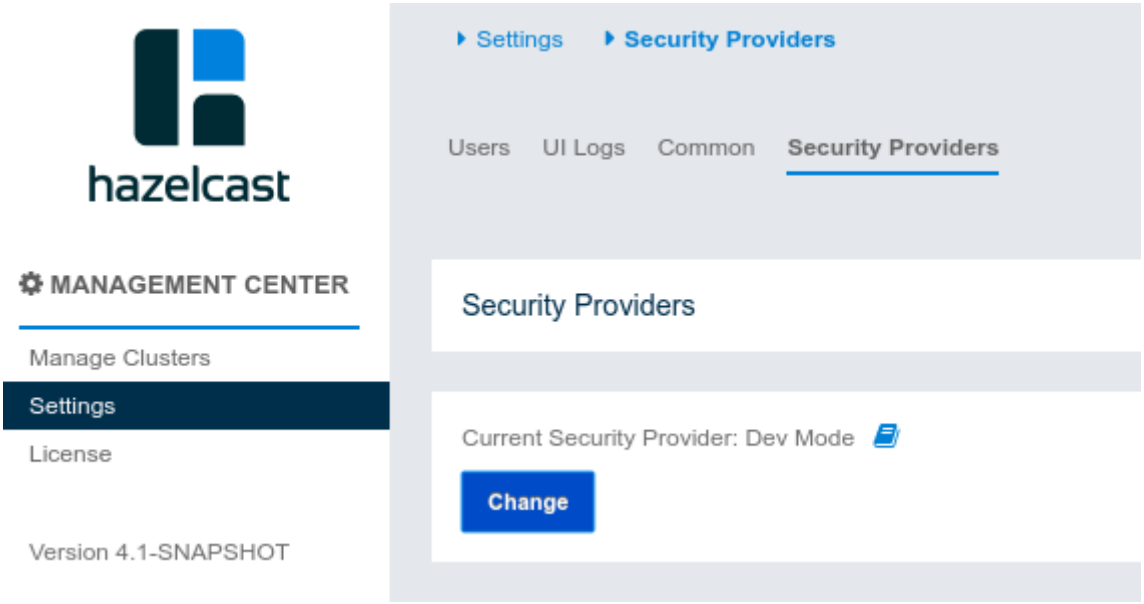
If there are any problems related to your cluster license, "License" menu item is highlighted with red exclamation points. In this case, please check this screen to see what the problem is. The following are the possible problems:

- Your Hazelcast IMDG license is expired.
- The count of your cluster members is more than the allowed count by the license.

If the cluster license expires in 30 days or less, or has already expired, a warning will appear in the upper right corner once a day, similar to the one for the Management Center license. Warning will contain time remaining until license expires or how long the license is expired. There will also be two buttons - the first one with label "Show License" will redirect you to the license page, the second one with label "Dismiss Alert" will dismiss the alert.

10. Managing Security Providers

In case you selected the `Dev Mode` as the security provider when you first launched Management Center, then it is possible to switch to an other security provider using the `Security Providers` tab in the `Settings` menu:



Important: this is a one-time option, you can change the security provider via the Management Center UI only if you currently use `Dev Mode`. Due to security considerations, any subsequent changes to the security provider configuration is only available via the Management Center Configuration Tool.

11. Monitoring an IMDG Cluster

This chapter details the monitoring and administering of an IMDG cluster using Management Center.

11.1. Dashboard Page

This is the first page appearing after logging in. It gives an overview of the connected cluster. The following subsections describe each portion of the page.

11.1.1. Memory Utilization

This part of the page provides information related to the memory usages for each member, as shown below:

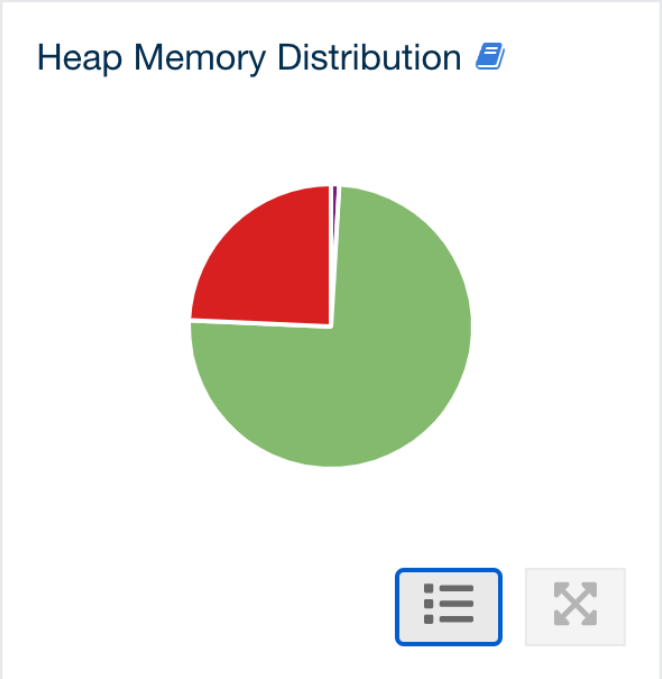
Memory Utilization

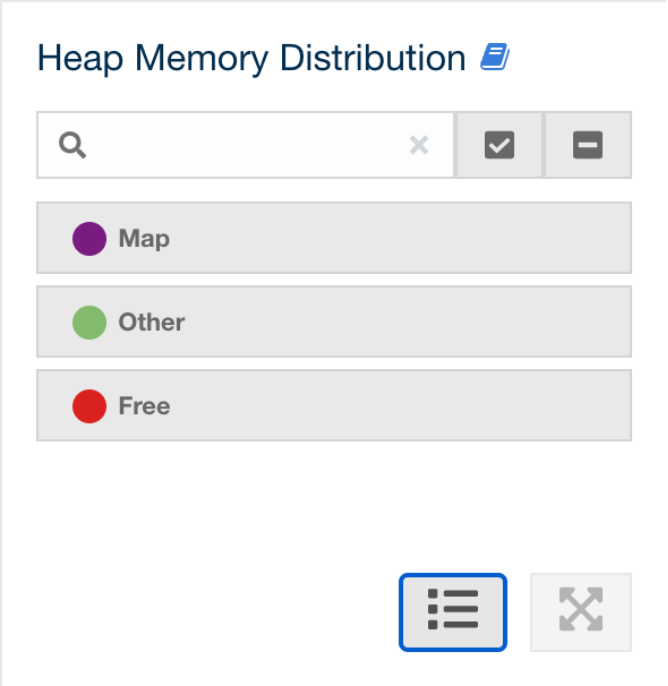
Member	Used Heap	Committed Heap	Heap Usage	Used Native Me...	Committed Nati...	GC Major Count	GC Major Time	GC Minor Count	GC Minor Time
127.0.0.1:5701	337.84 MB	640.00 MB	52.79 %	0 B	0 B	0	0ms	727	7s 622ms
127.0.0.1:5702	337.85 MB	640.00 MB	52.79 %	0 B	0 B	0	0ms	727	7s 622ms
127.0.0.1:5703 (lite)	338.05 MB	640.00 MB	52.82 %	0 B	0 B	0	0ms	727	7s 622ms

The first column lists the members with their IPs and ports. The next columns show the used and free memories out of the total memory reserved for Hazelcast IMDG usage, in real-time. The **Committed Heap** column lists the Java Heap memory capacity of each member, and the **Heap Usage** column lists the percentage value of used memory out of the committed heap memory. The **Used Heap** column shows how much the Java Heap memory is used of members graphically. The **Committed Native Memory** column lists the Java Off Heap memory reserved by Hazelcast IMDG member, and the **Used Native Memory** column shows how much Java Off Heap memory is used by members. When you move the mouse cursor on a desired graph, you can see the memory usage at the time point where the cursor is placed. Graphs under this column show the memory usages approximately for the last 2 minutes.

11.1.2. Heap Memory Distribution

This part of the page graphically provides the cluster wise breakdown of heap memory, as shown below. The purple area is the heap memory used by the maps (including all owned/backup entries, any near cache usage and cost of the Merkle tree). The green area is the heap memory used by both non-Hazelcast entities and all Hazelcast entities except the map, i.e., the heap memory used by all entities subtracted by the heap memory used by map. The red area is the free heap memory out of the whole cluster's total committed heap memory.





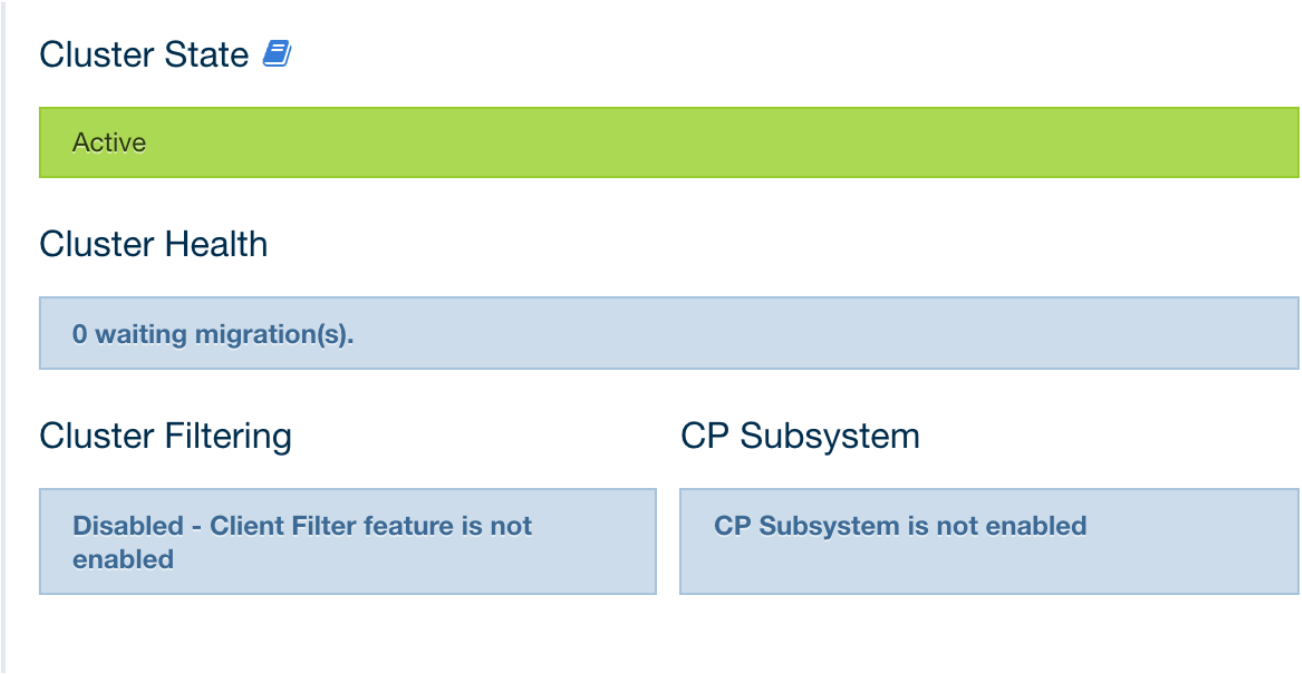
In the above example, you can see a small amount of the total heap memory is used by Hazelcast IMDG maps, about 25% is used by both non-Hazelcast entities and all Hazelcast entities except the map and the rest of the total heap memory is free. You can see the exact percentages by placing the mouse cursor on the chart.

11.1.3. Cluster State/Health/Client Filtering/CP Subsystem

This part has the following status indicator elements:

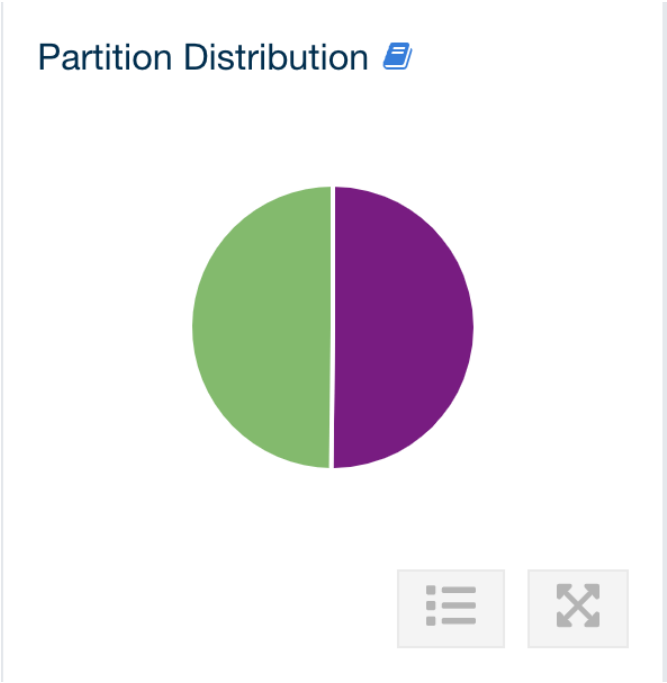
- **Cluster State:** Shows the current cluster state. For more information on cluster states, see the Cluster State section.
- **Cluster Health:** Shows how many migrations are taking place currently.
- **Cluster Filtering:** Shows values for the current cluster client filtering status and type. For more information on the cluster client filtering, see the Changing Cluster Client Filtering section.
- **CP Subsystem:** Shows the CP subsystem status. For more information on the CP subsystem support in the Management Center, see the CP Subsystem section.

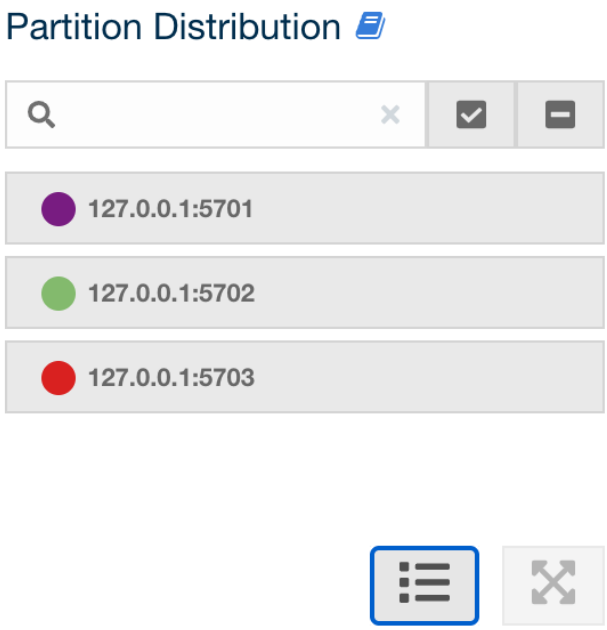
Cluster client filtering is only available with Hazelcast IMDG license that includes the Cluster Client Filtering feature.



11.1.4. Partition Distribution

This chart shows what percentage of partitions each cluster member has, as shown below:



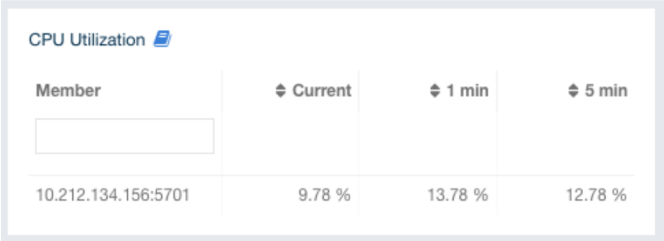


You can see each member's partition percentages by placing the mouse cursor on the chart. In the above example, you can see that two members out of three share the total partition count (which is 271 by default and configurable; see the `hazelcast.partition.count` property explained in the System Properties section of the Hazelcast IMDG Reference Manual).

The partition distribution chart does not show any information until you create your distributed objects. When you add new members to your cluster, there will be no partition migration since partitions do not exist yet. Once you connect to your cluster and, for example, create a map (using `hazelcastInstance.getMap()`), only then this chart starts to show partition distribution information.

11.1.5. CPU Utilization

This part of the page provides load and utilization information for the CPUs for each cluster member, as shown below:



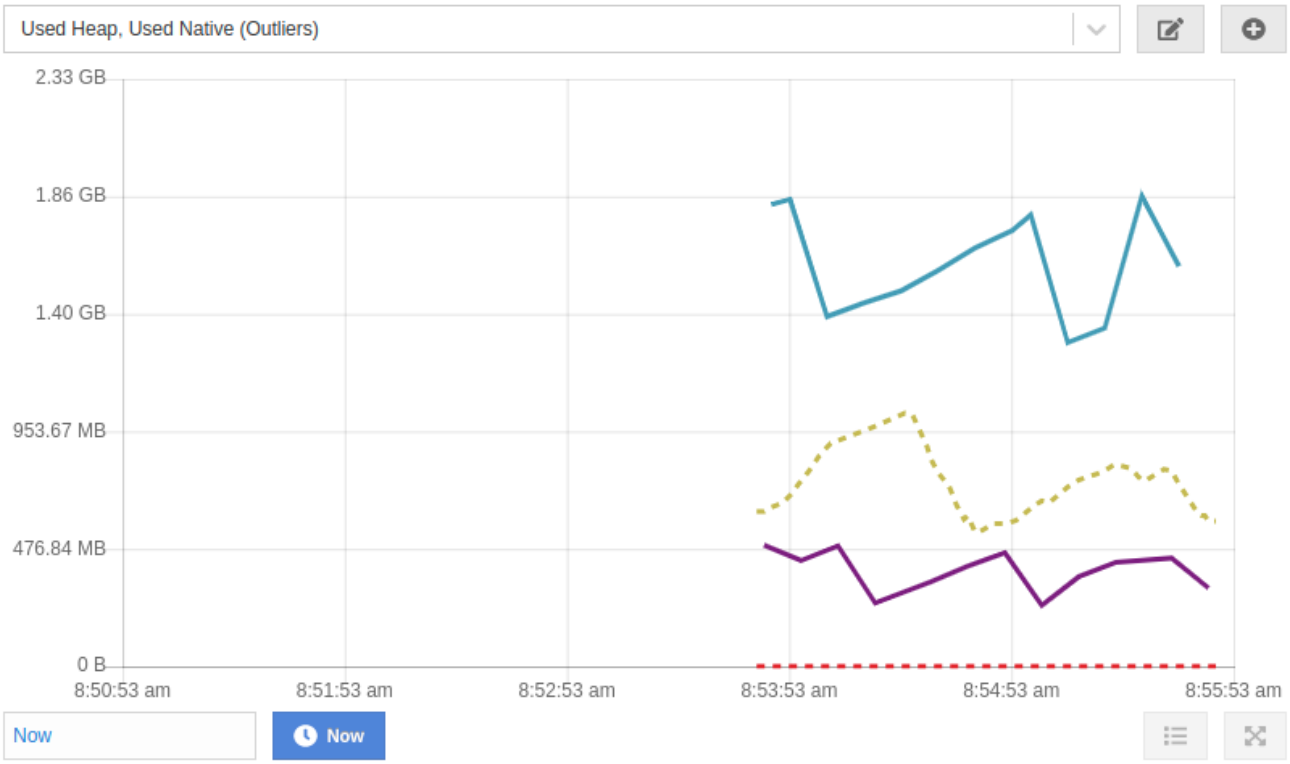
The first column lists the members with their IPs and ports. The next columns list the system load averages on each member for the current moment, last 1 and 5 minutes. These average values are calculated as the sum of the count of runnable entities running on and queued to the available CPUs averaged over the moment being, last 1 and 5 minutes. This calculation is operating system specific, typically a damped time-dependent average. If system load average is not available, these columns show negative values.

11.1.6. Widgets

The dashboard shows two widgets that show various metrics of the cluster. See the Widget section for more information.

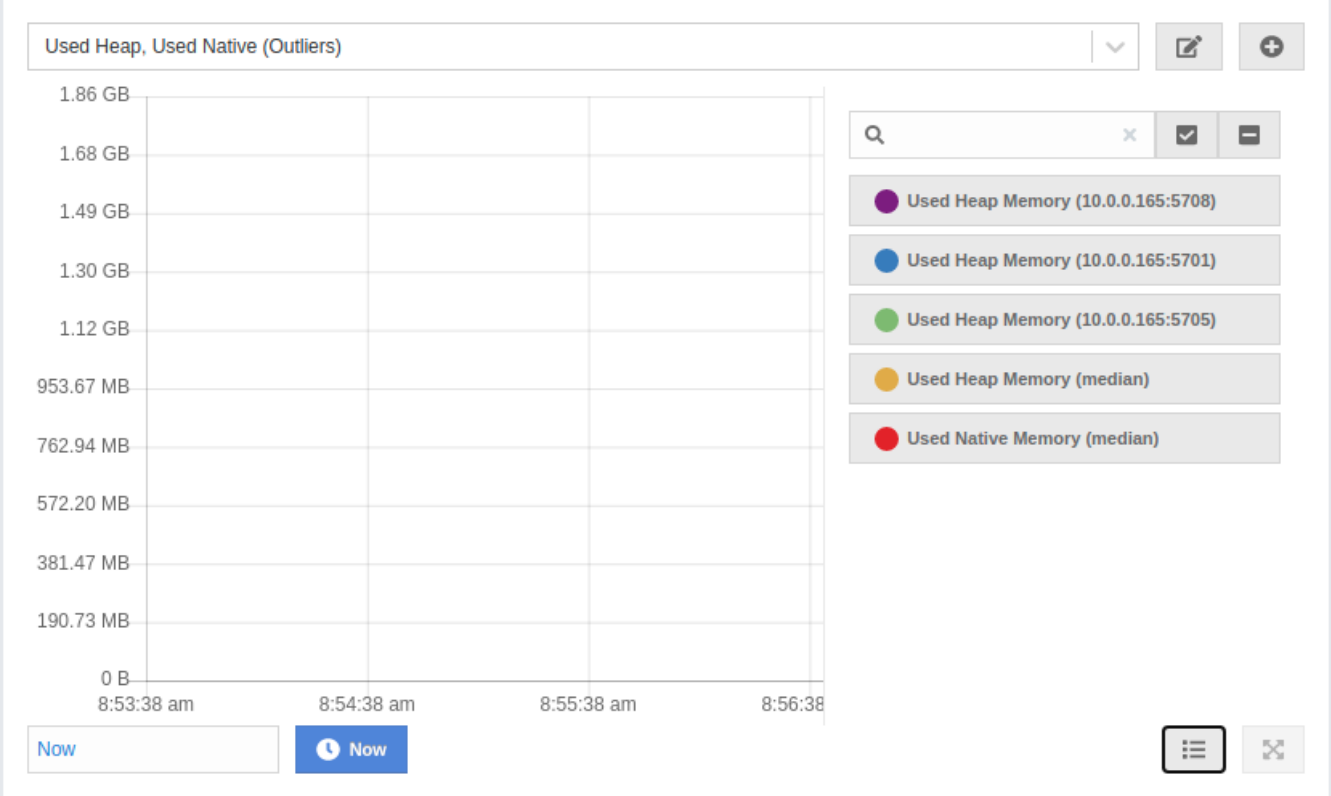
11.2. Widget

Widget is a special component designed to inspect cluster metrics as graphs. It is used on the dashboard and on the details page of any data structure (with a few exceptions).



11.2.1. Legend

Widget can display up to 12 graphs at the same time. Legend allows you to adjust which graphs are visible and which ones are not. By default, all graphs are visible. A title of a hidden graph is displayed as struck through.



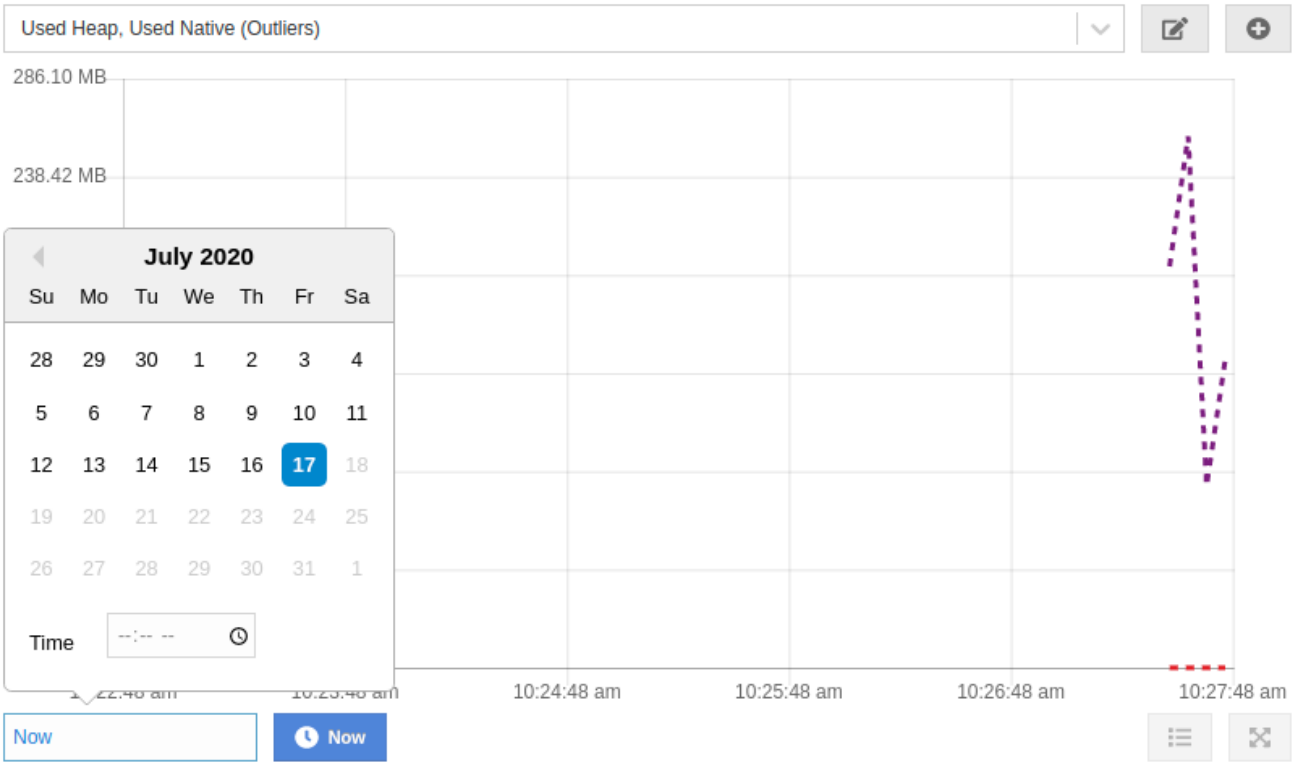
11.2.2. Current and Historical Data

Widget works in two modes - current and historical. It fetches current data in the current mode and historical data in the historical mode.

The current mode is always available. The historical mode requires Metrics Persistence to be enabled.

The input on the bottom controls what data is displayed. The input is disabled if only current data is available. You can select any arbitrary point in the past to see the historical data from that period. You can drag the chart to adjust the time point.

Widget shows up to 5 minutes of data. In the current mode it requests data in chunks every 5 seconds, and concatenates it. In the historical mode it requests 5 minutes of data up to the selected time point once.



When the historical mode is activated, the bottom input displays the selected time point. You can press the "Now" button to switch back to the current mode.

11.2.3. Presets

Widget provides access to an extensive list of metrics and uses a selection of filters to show only relevant data matching a certain condition on the graph. A preset is a combination of metrics and a filter. You can add/edit new presets and remove the existing ones. Each widget is shipped with at

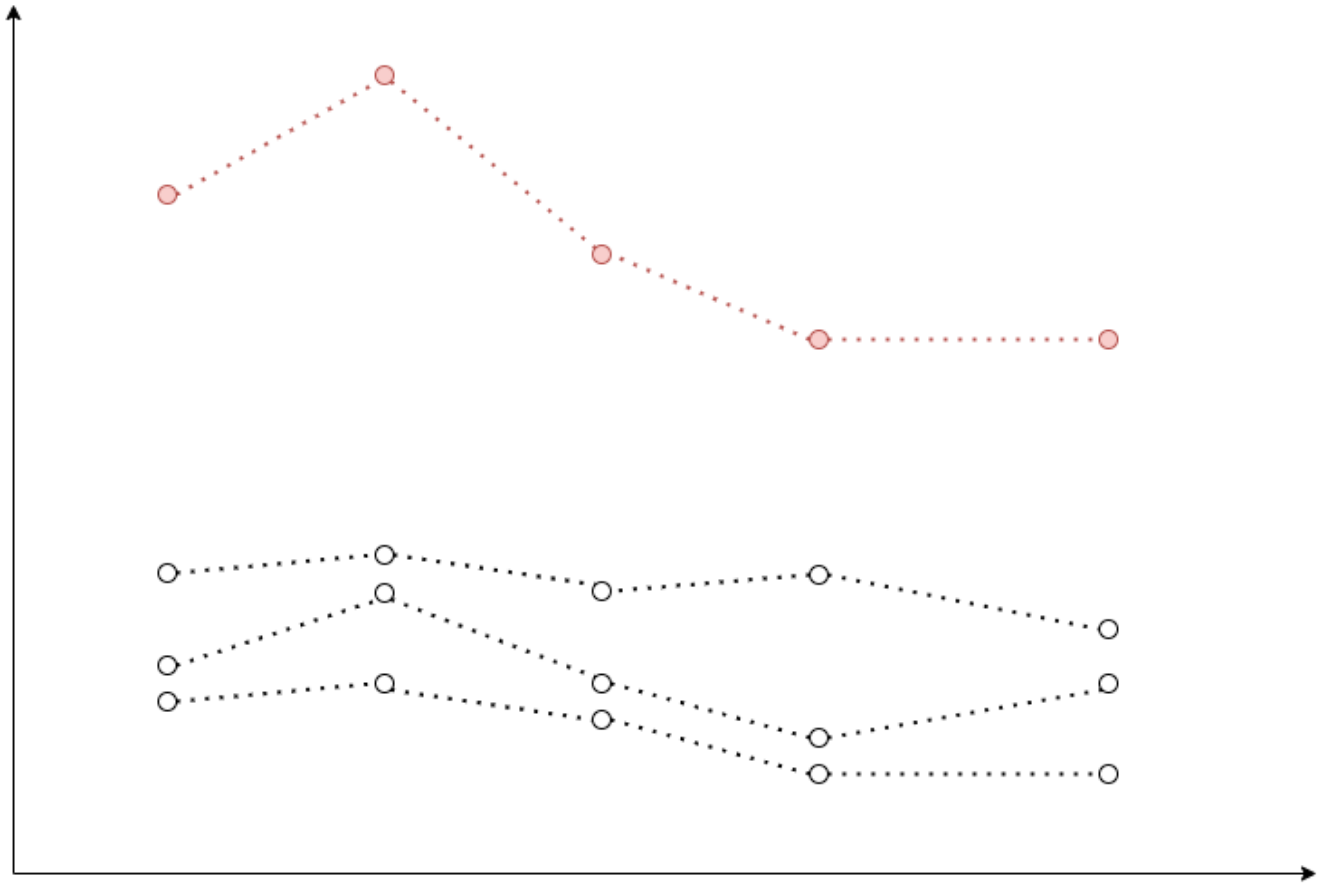
least one default preset. You cannot edit or remove a default preset.

The top dropdown shows the name of the preset and allows you to switch between them. The pencil button next to it opens an edit preset view. The plus button opens a new preset view.

11.2.4. Metrics

Every member collects its own metrics. Management Center periodically polls the members for their metrics and stores them in a dedicated metric storage. Later on, it exposes individual member metrics and their aggregates. Aggregated metric is a single value calculated for a metric across all cluster members for a single point in time. It can be a sum of values for all cluster members for a single time point, it can be their average, mean, etc.

Here is an example of the sum aggregate (colored in red):



Every metric is measured using some unit of measurement. You can select metrics with up to two different units.

11.2.5. Filters

Widget supports the following types of filters:

1. Outliers (auto)
2. Peak Top 3
3. Peak Bottom 3
4. Average Top 3

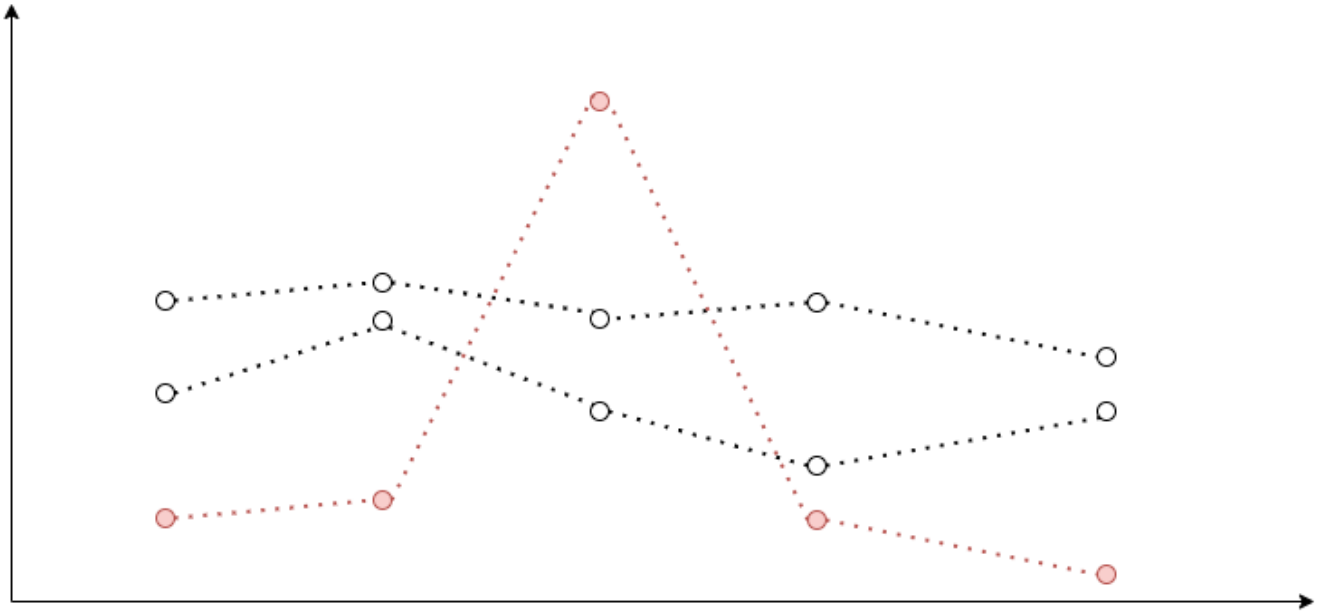
5. Average Bottom 3
6. Manual Selection

Peak Top 3 and Peak Bottom 3

It works as follows:

1. Widget fetches individual member metrics for all members.
2. It finds a maximum/minimum (for Top 3 and Bottom 3 correspondingly) value in the list of data points for every member.
3. It shows top 3/bottom 3 graphs based on the maximum/minimum value.

On the image below a graph with the highest peak value is colored in red:

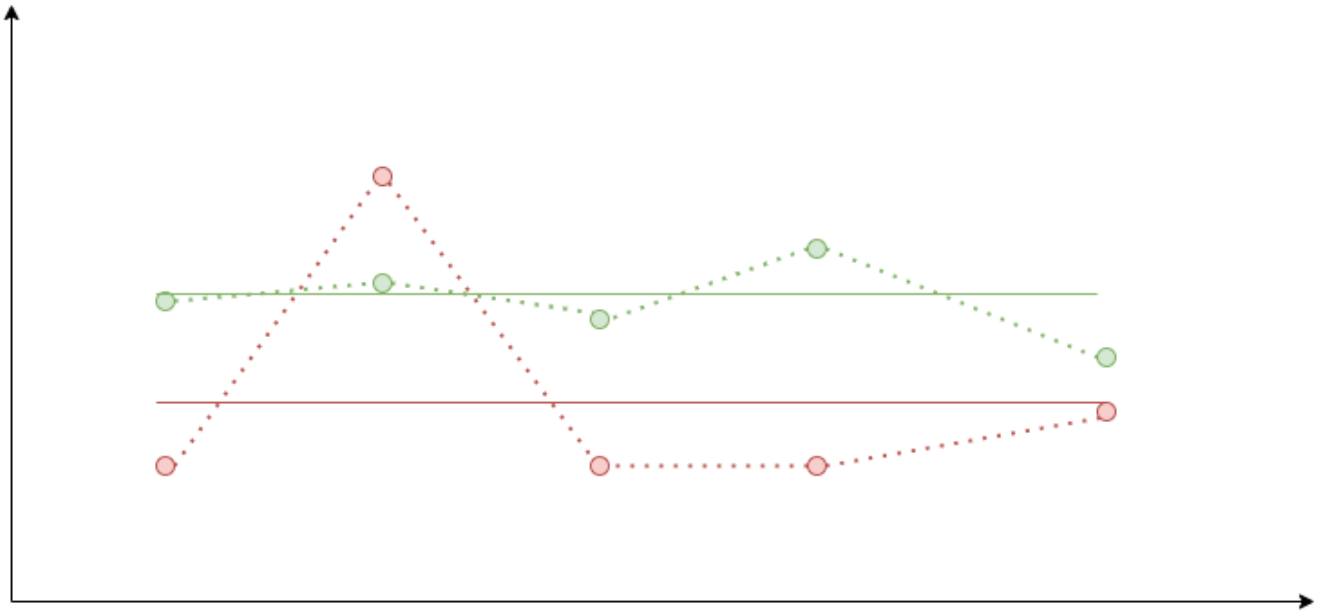


Average Top 3 and Average Bottom 3

It works as follows:

1. Widget fetches individual member metrics for all members.
2. It calculates an average value for the list of data points for every member.
3. It shows top 3/bottom 3 graphs based on the average value.

On the image below green and red graphs are displayed with their corresponding average values. As you can see, the red graph has a higher peak value, but a lower average value.

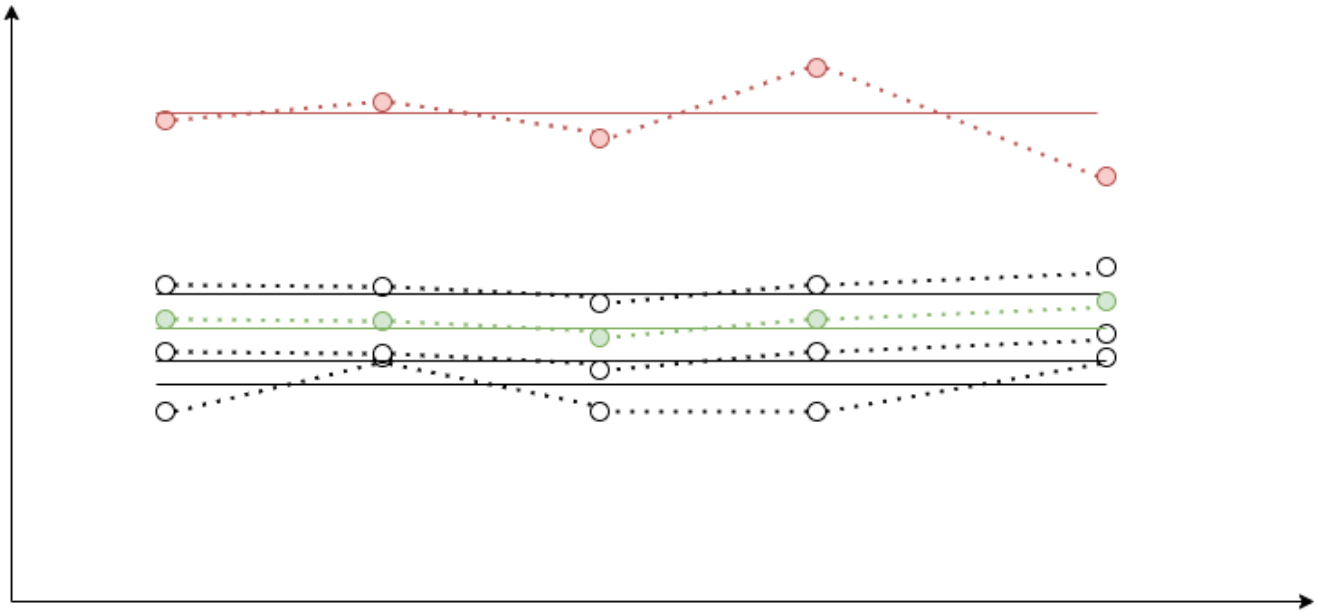


Outliers (auto)

It works as follows:

1. Widget fetches individual members metrics for all members.
2. It calculates an average value for the list of data points for every member.
3. Widget fetches a median aggregate for the selected metric.
4. It calculates an average value for the median aggregate graph.
5. It shows the median aggregate graph as a dashed line.
6. If the average value for any member is greater than the average value for the median aggregate, it shows the graph for that member. This graph is called an outlier. The graph shows up to 3 outliers with the greatest deviation from the median.

The below image shows 4 member graphs with their corresponding averages. A median aggregate with its average is colored in green. The member graph that is considered an outlier is colored in red.



Manual Selection

It works as follows:

1. You select the members.
2. Widget fetches individual member metrics for the selected members.
3. It calculates an average value for the list of data points for every member.
4. It shows the graphs for the selected metrics of the selected members.

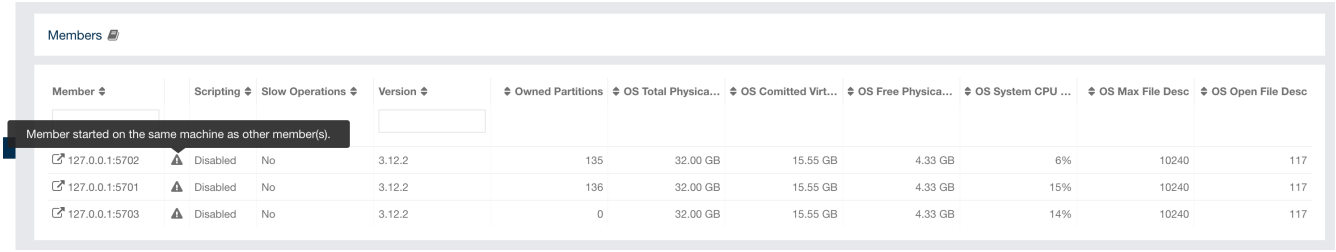
11.3. Monitoring Members

Use this menu item to monitor each cluster member and perform operations like running garbage collection (GC) and taking a thread dump.

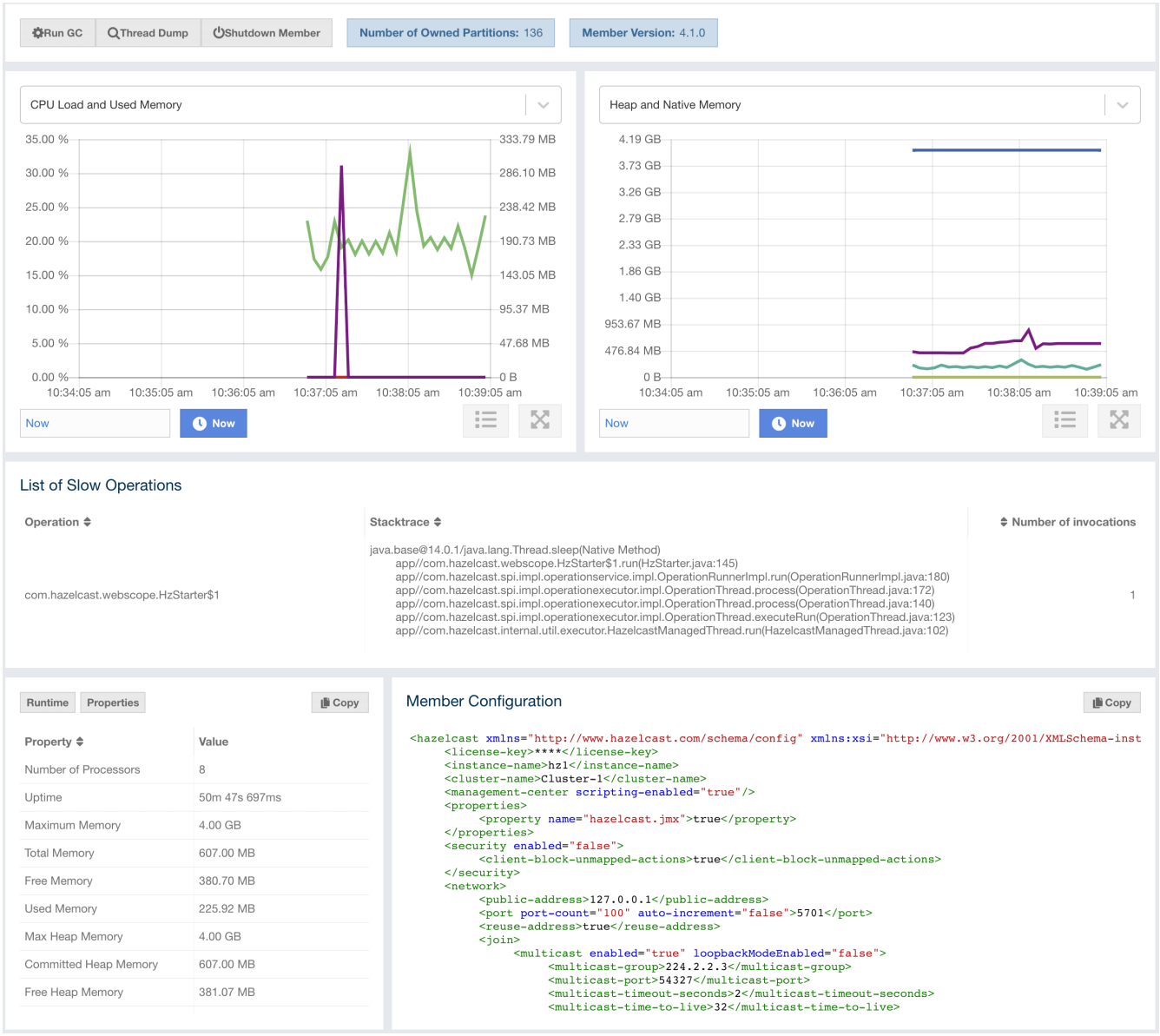
You can see a list of all the members in your cluster by clicking on the **Members** menu item on the left panel. A new page is opened on the right, as shown below.

Member	Scripting	Slow Operations	Owned Partitions	Version	OS Total Physica...	OS Comitted Virt...	OS Free Physical...	OS System CPU ...	OS Max File Des...	OS Open File De...
127.0.0.1:5702	Disabled	No	136	3.12.0	15.47 GB	9.53 GB	1.44 GB	67%	1048576	115
127.0.0.1:5701	Disabled	Yes	135	3.12.0	15.47 GB	9.53 GB	1.41 GB	21%	1048576	115
127.0.0.1:5703	Disabled	No	0	3.12.0	15.47 GB	9.53 GB	1.44 GB	15%	1048576	115

You may see a warning icon with exclamation mark in the list when your runtime or hardware configuration does not follow the performance recommendations. See IMDG Deployment and Operations Guide for more information.



You can filter the members shown and you can also sort the table by clicking on the column headers. Members that participate in the CP subsystem are marked with the CP icon. Clicking on a member name opens a new page for monitoring that member on the right, as shown below.



You can perform the following operations on the selected member using the buttons located at the top left of the page:

- **Run GC**: Executes garbage collection on the selected member. A notification stating that the GC execution was successful is shown.
- **Thread Dump**: Takes a thread dump of the selected member and shows it in a separate dialog.
- **Shutdown Member**: Shuts down the selected member.

- **Promote Member**: It is only shown for the lite members. When pressed, the lite member becomes a data member.

Shutdown member operation requires enabled REST API in the IMDG cluster. See the IMDG documentation for more information.

Next to the above operation buttons, you can see the informative buttons as described below:

- **Number of Owned Partitions**: Shows how many partitions are assigned to the selected member.
- **Member Version**: Shows the Hazelcast IMDG cluster version which the selected member belongs to.
- **CP Member UUID**: Shows CP member UUID if the member participates in the CP subsystem of the cluster.

The **CPU Utilization** chart shows the percentage of CPU usage on the selected member. The **Heap/Memory Utilization** charts show the memory usage on the selected member with three different metrics (maximum, used and total memory). You can open each chart as a separate dialog using the  button placed at top right of them; this gives you a clearer view of the chart.

Runtime is a dynamically updated window tab showing the processor number, the start and up times, and the maximum, total and free memory sizes of the selected member. These values are collected from the default MXBeans provided by the Java Virtual Machine (JVM). Descriptions from the Javadocs and some explanations are below:

- **Number of Processors**: Number of processors available to the member (JVM).
- **Start Time**: Start time of the member (JVM) in milliseconds.
- **Up Time**: Uptime of the member (JVM) in milliseconds
- **Maximum Memory**: Maximum amount of memory that the member (JVM) will attempt to use.
- **Free Memory**: Amount of free memory in the member (JVM).
- **Used Heap Memory**: Amount of used memory in bytes.
- **Max Heap Memory**: Maximum amount of memory in bytes that can be used for memory management.
- **Used Non-Heap Memory**: Amount of used memory in bytes.
- **Max Non-Heap Memory**: Maximum amount of memory in bytes that can be used for memory management.
- **Total Loaded Classes**: Total number of classes that have been loaded since the member (JVM) has started execution.
- **Current Loaded Classes**: Number of classes that are currently loaded in the member (JVM).
- **Total Unloaded Classes**: Total number of classes unloaded since the member (JVM) has started execution.
- **Total Thread Count**: Total number of threads created and also started since the member (JVM) started.

- **Active Thread Count**: Current number of live threads including both daemon and non-daemon threads.
- **Peak Thread Count**: Peak live thread count since the member (JVM) started or peak was reset.
- **Daemon Thread Count**: Current number of live daemon threads.
- **OS: Free Physical Memory**: Amount of free physical memory in bytes.
- **OS: Committed Virtual Memory**: Amount of virtual memory that is guaranteed to be available to the running process in bytes.
- **OS: Total Physical Memory**: Total amount of physical memory in bytes.
- **OS: Free Swap Space**: Amount of free swap space in bytes. Swap space is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space.
- **OS: Total Swap Space**: Total amount of swap space in bytes.
- **OS: Maximum File Descriptor Count**: Maximum number of file descriptors. File descriptor is an integer number that uniquely represents an opened file in the operating system.
- **OS: Open File Descriptor Count**: Number of open file descriptors.
- **OS: Process CPU Time**: CPU time used by the process on which the member (JVM) is running in nanoseconds.
- **OS: Process CPU Load**: Recent CPU usage for the member (JVM) process. This is a double with a value from 0.0 to 1.0. A value of 0.0 means that none of the CPUs were running threads from the member (JVM) process during the recent period of time observed, while a value of 1.0 means that all CPUs were actively running threads from the member (JVM) 100% of the time during the recent period being observed. Threads from the member (JVM) include the application threads as well as the member (JVM) internal threads.
- **OS: System Load Average**: System load average for the last minute. The system load average is the average over a period of time of this sum: (the number of runnable entities queued to the available processors) + (the number of runnable entities running on the available processors). The way in which the load average is calculated is operating system specific but it is typically a damped time-dependent average.
- **OS: System CPU Load**: Recent CPU usage for the whole system represented as a percentage value. `0%` means that all CPUs were idle during the recent period of time observed, while `100%` means that all CPUs were actively running during the recent period being observed.

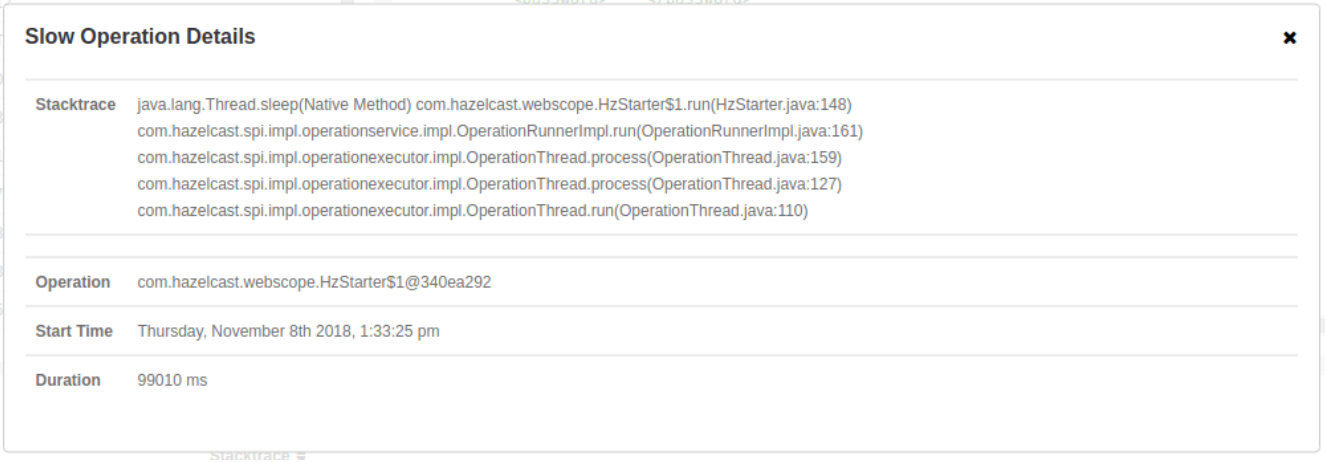
These descriptions may vary according to the JVM version or vendor.

Next to the **Runtime** tab, the **Properties** tab shows the system properties.

The **Member Configuration** window shows the XML configuration of the connected Hazelcast cluster.

The **List of Slow Operations** gives an overview of detected slow operations which occurred on that member. The data is collected by the SlowOperationDetector.

Click on an entry to open a dialog which shows the stacktrace and detailed information about each slow invocation of this operation.

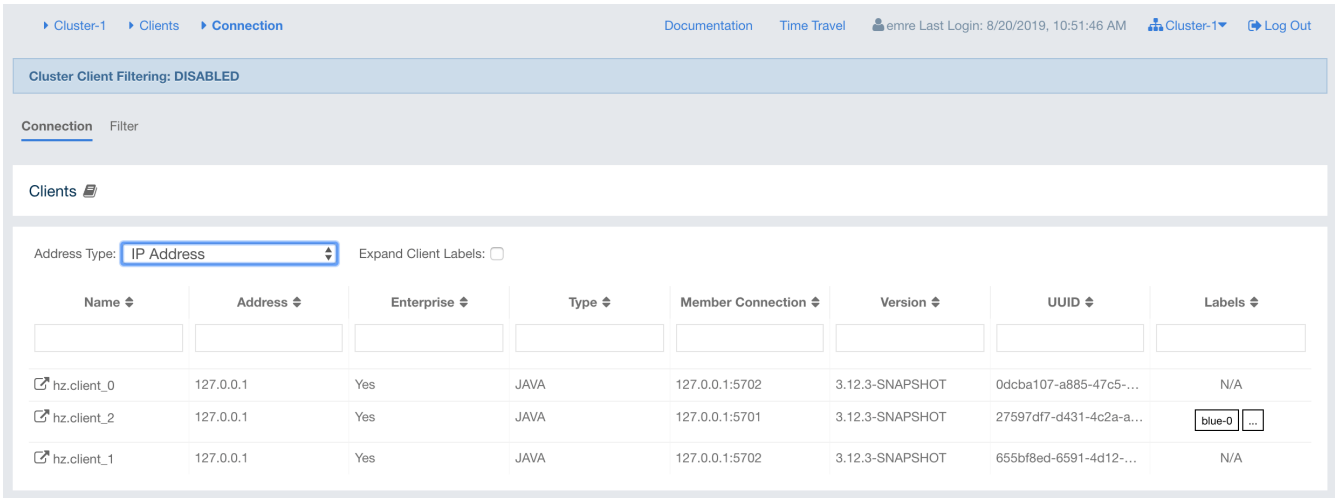


11.4. Monitoring Clients

You can use the **Clients** menu item to monitor all the clients that are connected to your Hazelcast cluster.

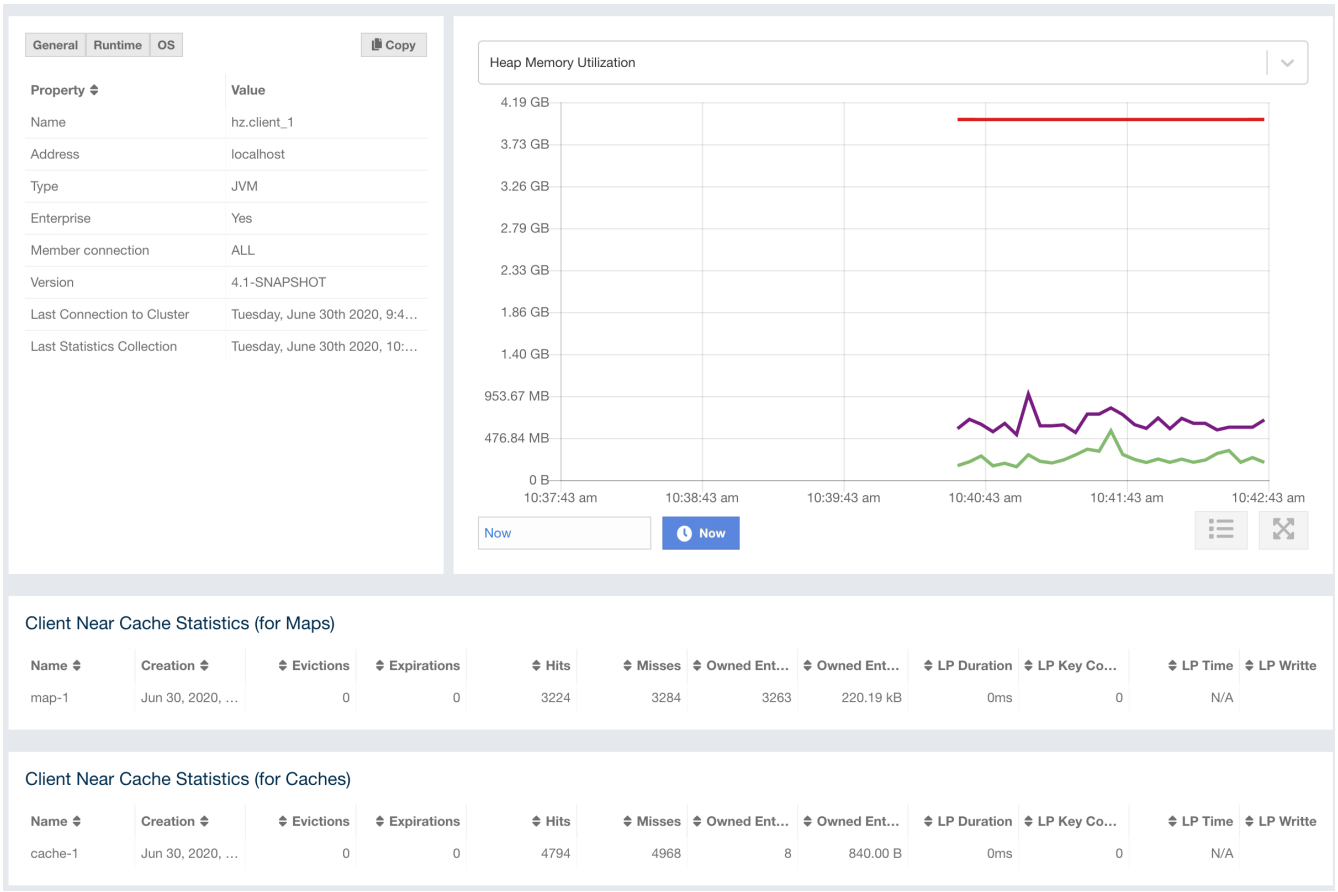
Only basic information for clients, like client instance name, address, type and labels, is shown by default. The values for other fields are shown as **N/A**. As a prerequisite for seeing the full information, you need to enable the client statistics before starting your clients. This can be done by setting the `hazelcast.client.statistics.enabled` system property to `true` on the **client**. Please see the Client System Properties section in the Hazelcast IMDG Reference Manual for more information. After you enable the client statistics, you can monitor your clients using the Management Center.

You can see a list of all the clients in your cluster by clicking on the **Clients** menu item on the left panel. A new page is opened on the right, as shown below. The page has two tabs: **Connection** and **Filter**. The Connection tab is opened by default. This tab shows the list of all the clients. See the Changing Cluster Client Filtering section for the Filter tab's description.



By default, hostname of the client is shown in the address column. You can change it to show its IP address or its canonical hostname instead by using the **Address Type** combo box. Note that this will also cause the client details page to show the IP address or the canonical hostname.

You can filter the clients shown and you can also sort the table by clicking on the column headers. Clicking on a client name will open a new page for monitoring that client on the right, as shown below.



The **Heap Memory Utilization** chart shows the memory usage on the selected client with three different metrics (maximum, used and total memory) represented by different colors. You can open this chart as a separate window using the button placed at top right of it; this gives you a clearer view of the chart.

General is a dynamically updated window tab showing general information about the client. Below are brief explanations for each piece of information:

- **Name**: Name of the client instance.
- **Address**: Local IP address of the client that is used for connecting to members.
- **Type**: Type of the client.
- **Enterprise**: Yes, if the client is an Hazelcast IMDG Enterprise client.
- **Member Connection**: Shows to which member a client is currently connected to. Please note that **ALL** means a client is configured so that it **might** connect to all members of a cluster, i.e., it might not have a connection to all members all the time.
- **Version**: Version of the client.

- **Last Connection to Cluster**: Time that the client connected to the cluster. It is reset on each reconnection.
- **Last Statistics Collection**: Time when the latest update for the statistics is collected from the client.
- **User Executor Queue Size**: Number of waiting tasks in the client user executor.
- **Labels**: List of client labels (as defined by the client).

Next to the **General** tab, the **Runtime** tab shows the processor number, uptime, and maximum, total and free memory sizes of the selected client. These values are collected from the default MXBeans provided by the Java Virtual Machine (JVM). Descriptions from the Javadocs and some explanations are below:

- **Number of Processors**: Number of processors available to the client (JVM).
- **Up Time**: Uptime of the client (JVM).
- **Maximum Memory**: Maximum amount of memory that the client (JVM) will attempt to use.
- **Total Memory**: Amount of total heap memory currently available for current and future objects in the client (JVM).
- **Free Memory**: Amount of free heap memory in the client (JVM).
- **Used Memory**: Amount of used heap memory in the client (JVM).

Next to the **Runtime** tab, the **OS** tab shows statistics about the operating system of the client. These values are collected from the default MXBeans provided by the Java Virtual Machine (JVM). Descriptions from the Javadocs and some explanations are below:

- **Free Physical Memory**: Amount of free physical memory.
- **Committed Virtual Memory**: Amount of virtual memory that is guaranteed to be available to the running process.
- **Total Physical Memory**: Total amount of physical memory.
- **Free Swap Space**: Amount of free swap space. Swap space is used when the amount of physical memory (RAM) is full. If the system needs more memory resources and the RAM is full, inactive pages in memory are moved to the swap space.
- **Total Swap Space**: Total amount of swap space.
- **Maximum File Descriptor Count**: Maximum number of file descriptors. File descriptor is an integer number that uniquely represents an opened file in the operating system.
- **Open File Descriptor Count**: Number of open file descriptors.
- **Process CPU Time**: CPU time used by the process on which the member (JVM) is running.
- **System Load Average**: System load average for the last minute. The system load average is the average over a period of time of this sum: (the number of runnable entities queued to the available processors) + (the number of runnable entities running on the available processors). The way in which the load average is calculated is operating system specific but it is typically a damped time-dependent average.

Some of the Runtime/OS statistics may not be available for your client's JVM implementation/operating system. **N/A** is shown for these types of statistics. Please refer to your JVM/operating system documentation for further details.

The **Client Near Cache Statistics** table shows statistics related to the Near Cache of a client. There are two separate tables; one for maps and one for caches.

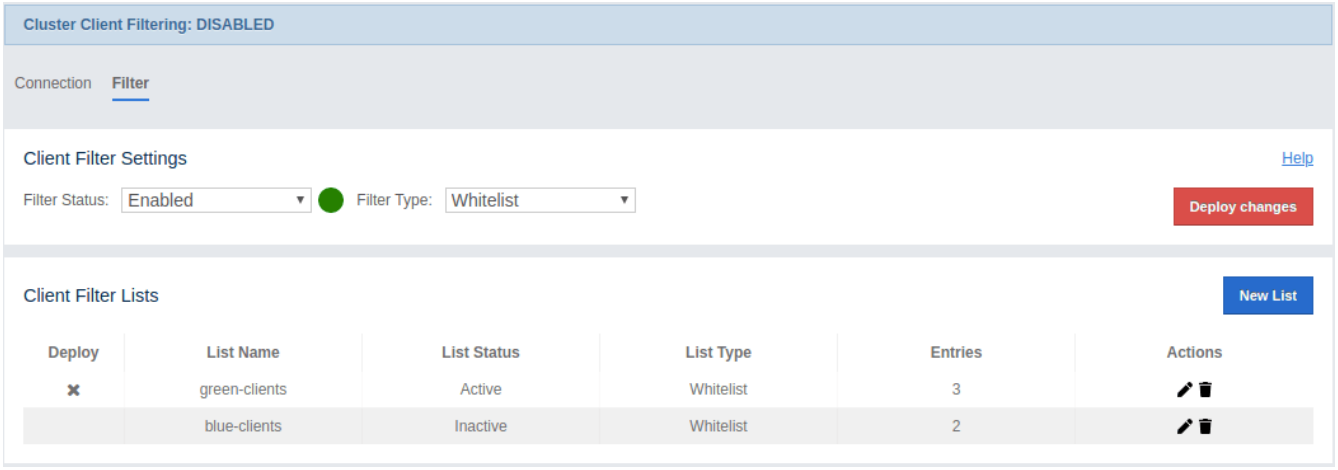
- **Map/Cache Name**: Name of the map or cache.
- **Creation Time**: Creation time of this Near Cache on the client.
- **Evictions**: Number of evictions of Near Cache entries owned by the client.
- **Expirations**: Number of TTL and max-idle expirations of Near Cache entries owned by the client.
- **Hits**: Number of hits (reads) of Near Cache entries owned by the client.
- **Misses**: Number of misses of Near Cache entries owned by the client.
- **Owned Entry Count**: Number of Near Cache entries owned by the client.
- **Owned Entry Memory Cost**: Memory cost of Near Cache entries owned by the client.
- **LP Duration**: Duration of the last Near Cache key persistence (when the pre-load feature is enabled).
- **LP Key Count**: Number of Near Cache key persistences (when the pre-load feature is enabled).
- **LP Time**: Time of the last Near Cache key persistence (when the pre-load feature is enabled).
- **LP Written Bytes**: Written number of bytes of the last Near Cache key persistence (when the pre-load feature is enabled).
- **LP Failure**: Failure reason of the last Near Cache persistence (when the pre-load feature is enabled).

Please note that you can configure the time interval for which the client statistics are collected and sent to the cluster, using the system property `hazelcast.client.statistics.period.seconds`. See the System Properties section in the Hazelcast IMDG Reference Manual for more information.

11.5. Changing Cluster Client Filtering

The Filter tab is only available with Hazelcast IMDG license that includes the Cluster Client Filtering feature.

The **Filter** tab includes **Cluster Client Filtering** status, **Cluster Client Filter Settings** and **Client Filtering Lists** sections, as shown below.



The **Cluster Client Filtering** status section describes if there is a deployed client filtering list available to all cluster members (**Enabled** status), or if the feature is disabled for the cluster and the members allow any clients (**Disabled** status).

The **Cluster Client Filter Settings** section allows to specify the status of the feature and the filtering type and to deploy any modifications made in client filtering lists to the deployed list available to all cluster members. On the deploy action the following happens:

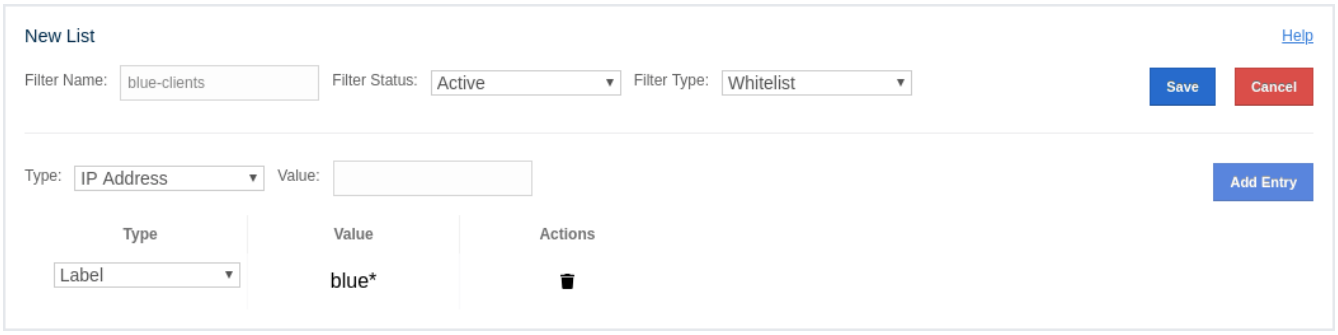
- If the status to be deployed is **Disabled**, the deployed client filtering list available to all cluster members is cleaned up and the members start allowing any client to connect.
- If the status to be deployed is **Enabled**, all entries of the matching lists from the Client Filtering Lists section are copied into the deployed client filtering list and applied to all cluster members. Matching lists are selected by their status (**List Status** must be **Active**) and type (**List Type** must match the value of the **Client Filter Type** selection).

Once a cluster member receives the deployed client filtering list from the Management Center, it immediately applies the list to all currently connected clients and then uses it for newly connecting clients. Blacklisted clients may connect to another cluster if they are configured to support blue-green deployment. Please see the Blue-Green Deployment and Disaster Recovery section in the Hazelcast IMDG Reference Manual for more information.

If some of the cluster members are not reachable from the Management Center, those members keep using the last client list applied to them.

The deploy action in the Cluster Client Filter Settings section is available by clicking on the **Deployed/Deploy Changes** button. This button also describes if there were any changes in client filtering lists that would lead to changes in the deployed client filtering list as the result of the deploy (**Deploy Changes** label), or there were no such changes (**Deployed** label).

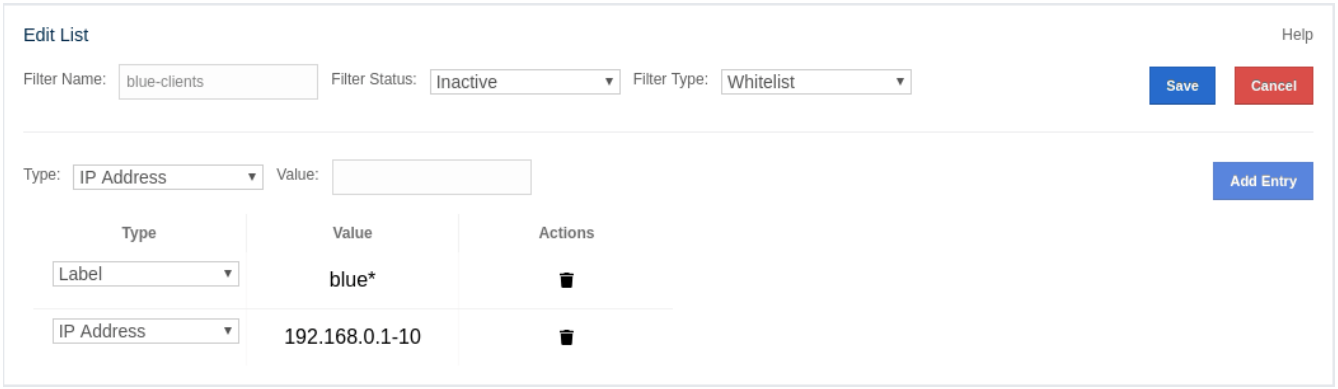
The **Client Filtering Lists** section allows creation, editing and deletion of the client filtering lists. To create a new client filtering list, you need to click the **Add New List** button, which will open the Create List form, as shown below. Once you enter all fields and entries for the new list, click the **Save** button to save your modifications.



The following formats of list entry values are supported:

- For the IP Address entry type you can specify IP address (IPv4 or IPv6) with optional range characters (`*` and `-`) instead of any byte group. For instance, `10.3.10.*` refers to IPs between `10.3.10.0` and `10.3.10.255`. The `10.3.10.4-18` refers to IPs between `10.3.10.4` and `10.3.10.18` (4 and 18 included).
- For the Label entry type you can specify any string with optional wildcard characters (`*`). For instance, `green*` refers to any label values that start with the `green` string.
- For the Instance Name entry type you can specify any string with optional wildcard characters (`*`). For instance, `*-client` refers to any label values that end with the `-client` string.

To modify an existing client filtering list, you need to click the **Edit** button, which will open the Edit List form, as shown below.



To delete an existing client filtering list, you need to click the **Delete** button and confirm your action in the opened dialog.

Any modifications made in the Client Filtering Lists section will become available to members only after the deploy action.

11.6. Monitoring Data Structures

This chapter provides information on how you can monitor the Hazelcast data structures in your cluster.

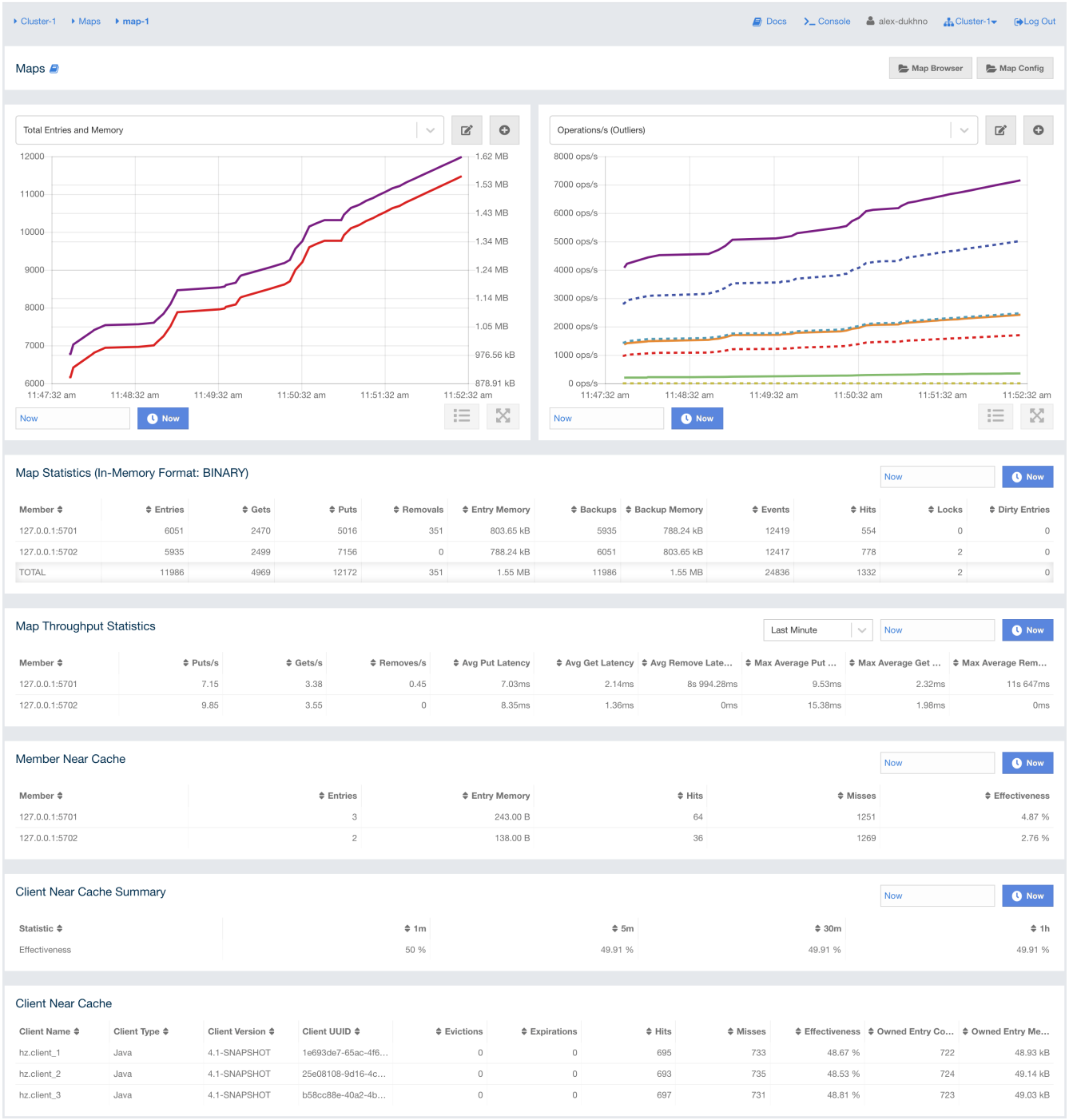
11.6.1. Maps

You can see a list of all the maps in your cluster by clicking on the **Maps** menu item on the left panel. A new page is opened on the right, as shown below.

Name	Entries	EntryMemory	BackupMemory	Events	Hits	Locks	DirtyEntries
map-45	50	7.62 kB	7.62 kB	0	0	0	0
map-46	50	7.62 kB	7.62 kB	0	0	0	0
map-47	50	7.62 kB	7.62 kB	0	0	0	0
map-48	50	7.62 kB	7.62 kB	0	0	0	0
map-49	50	7.62 kB	7.62 kB	0	0	0	0
map-5	50	7.62 kB	7.62 kB	0	0	0	0
map-50	50	7.62 kB	7.62 kB	0	0	0	0
map-51	50	7.62 kB	7.62 kB	0	0	0	0
map-52	50	7.62 kB	7.62 kB	0	0	0	0
map-53	50	7.62 kB	7.62 kB	0	0	0	0
map-54	50	7.62 kB	7.62 kB	0	0	0	0
map-55	50	7.62 kB	7.62 kB	0	0	0	0
map-56	50	7.62 kB	7.62 kB	0	0	0	0
map-57	50	7.62 kB	7.62 kB	0	0	0	0
map-58	50	7.62 kB	7.62 kB	0	0	0	0
map-59	50	7.62 kB	7.62 kB	0	0	0	0
map-6	50	7.62 kB	7.62 kB	0	0	0	0
map-60	50	7.62 kB	7.62 kB	0	0	0	0
map-61	50	7.62 kB	7.62 kB	0	0	0	0
map-62	50	7.62 kB	7.62 kB	0	0	0	0

« ‹ 1 2 3 4 5 6 › »

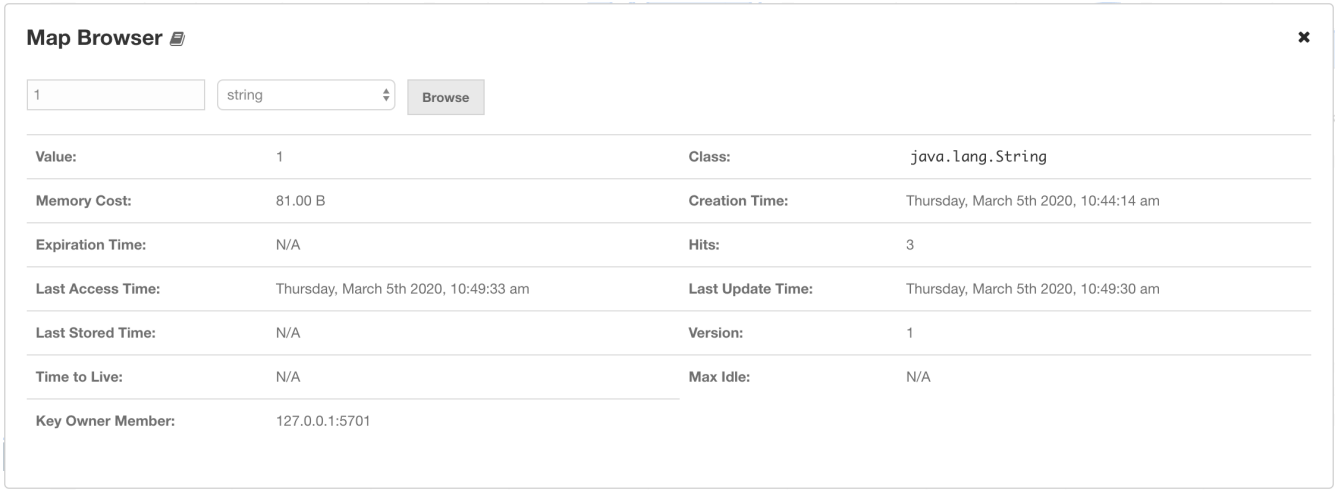
You can filter the maps shown and you can also sort the table by clicking on the column headers. Clicking on a map name opens a new page for monitoring that map instance on the right, as shown below.



The below subsections explain the portions of this window.

Map Browser

Use the Map Browser tool to retrieve properties of the entries stored in the selected map. To open the Map Browser tool, click on the **Map Browser** button, located at the top right of the window. Once opened, the tool appears as a dialog, as shown below.

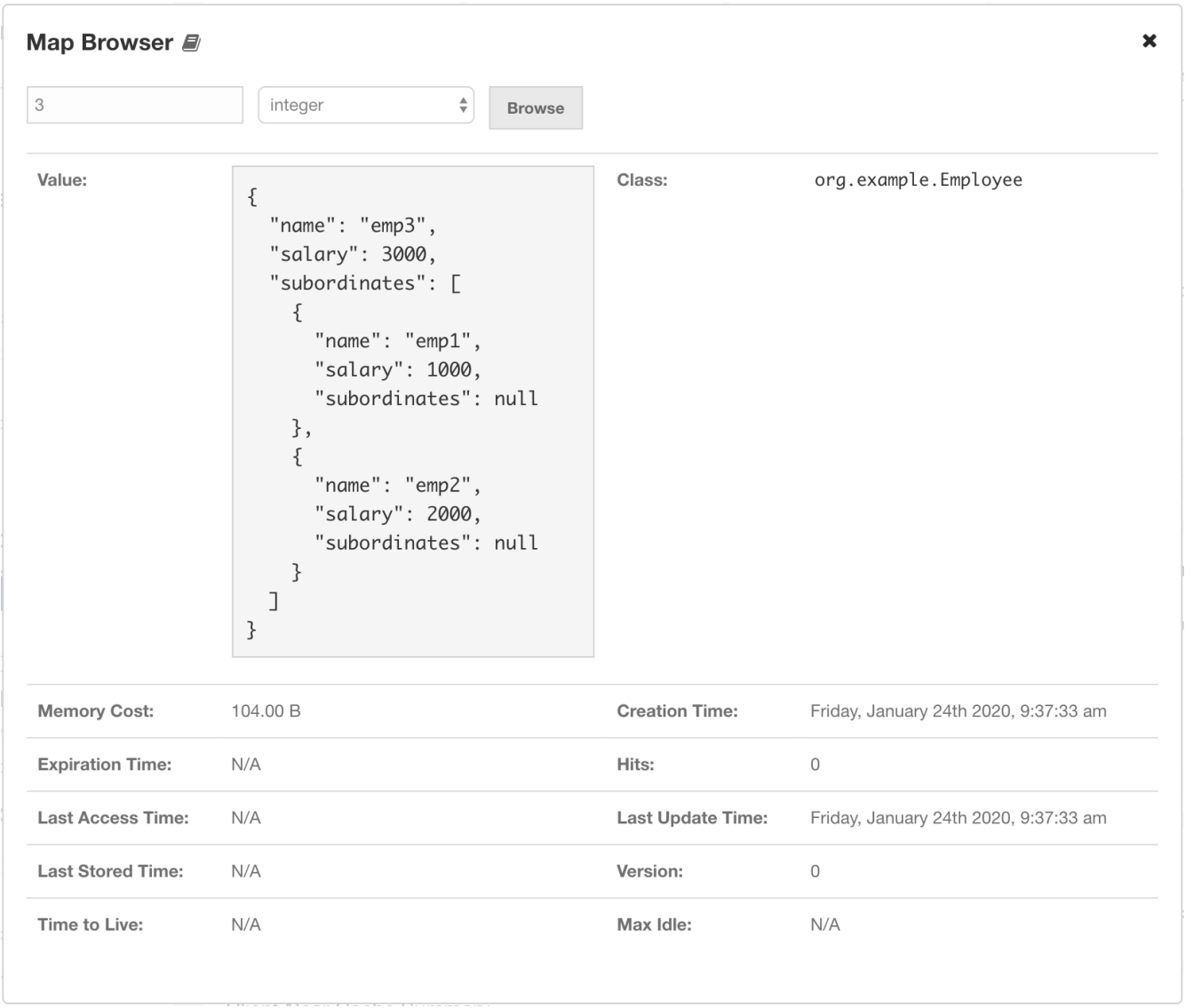


Once the key and the key's type are specified and the **Browse** button is clicked, the key's properties along with its value are listed.

Fields Description

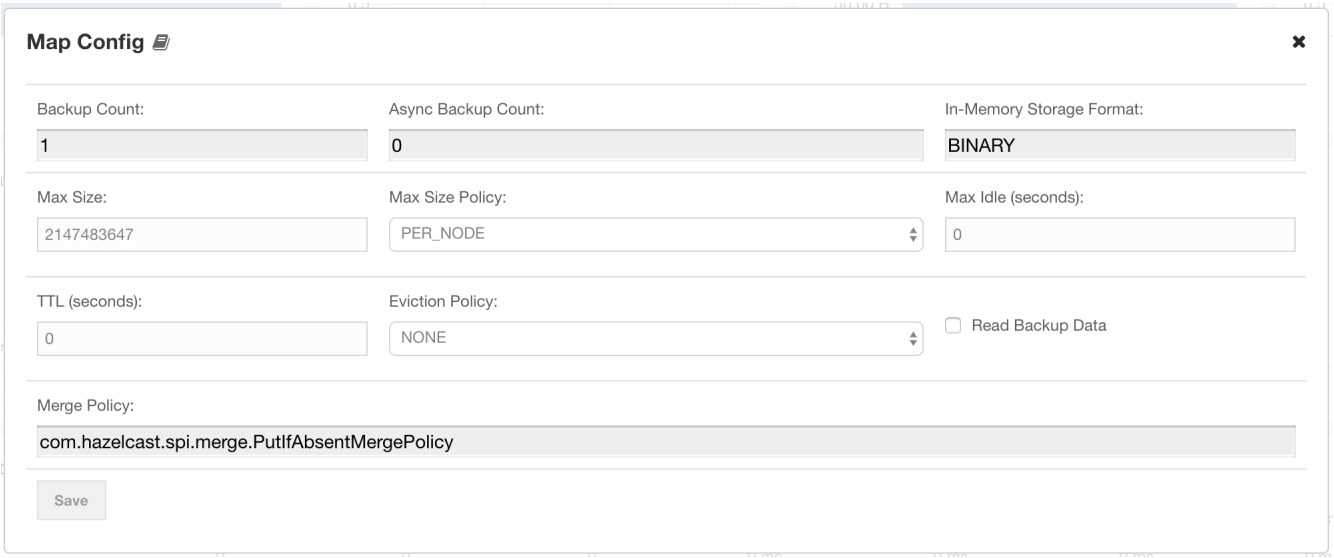
- **Value**: Value that is associated with the key in the map.
- **Class**: Key's Java class.
- **Memory Cost**: Space, in bytes, the key and value cost to be hold in memory.
- **Creation Time**: First time the key was put.
- **Expiration Time**: Time the key was expired and deleted from the map (depends on expiration policy configuration).
- **Hits**: Count of reads by the clients for the key's value in a map.
- **Last Access Time**: Last time the clients read the key's value.
- **Last Stored Time**: Last time the key was stored into the storage.
- **Version**: Count of times where the key's value has been overwritten.
- **Time to Live**: Last set time to live (in milliseconds).
- **Max Idle**: Last set max idle time (in milliseconds).

If you are using a serialization mechanism other than standard Java serialization for storing values in your map, you need to configure the client that Management Center uses for connecting to the cluster (as described here). Note that if you have any custom classes, you need to add the JAR containing them to Management Center's classpath before starting it as described in Providing an Extra Classpath. In case the value stored in the map is not a `String` or of another Java primitive type, it is rendered as a JSON value as shown below:



Map Config

Use the Map Config tool to set the selected map's attributes, such as the backup count, TTL, and eviction policy. To open the Map Config tool, click on the **Map Config** button, located at the top right of the window. Once opened, the tool appears as a dialog, as shown below.



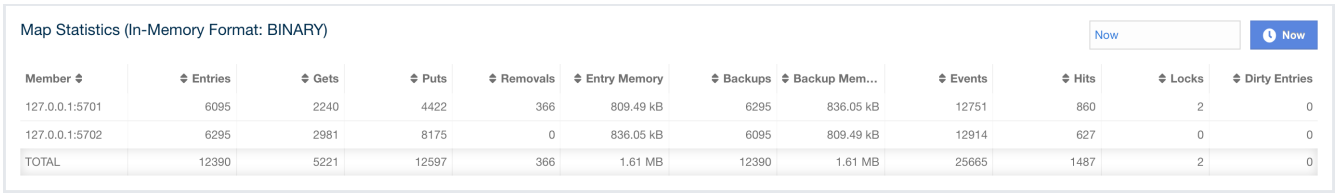
You can change any attribute and click the **Update** button to save your changes.

Map Monitoring

Besides the Map Browser and Map Config tools, the map monitoring page shows two widgets that show various metrics of the map. See the Widget section for more information.

Under these charts are **Map Statistics**, **Map Throughput**, **Member Near Cache**, and **Client Near Cache** data tables.

Map Statistic Table provides statistics distributed over the members, as shown below.



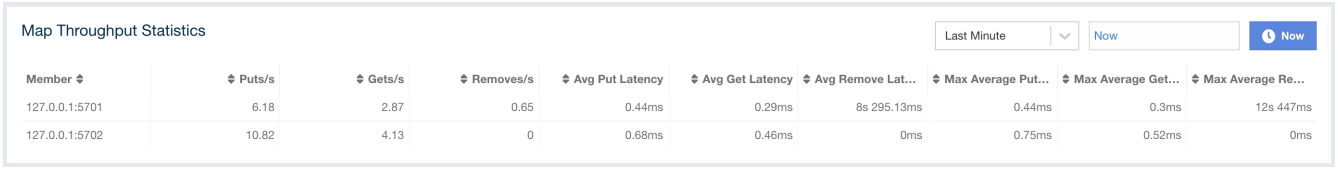
Member	Entries	Gets	Puts	Removals	Entry Memory	Backups	Backup Mem...	Events	Hits	Locks	Dirty Entries
127.0.0.1:5701	6095	2240	4422	366	809.49 kB	6295	836.05 kB	12751	860	2	0
127.0.0.1:5702	6295	2981	8175	0	836.05 kB	6095	809.49 kB	12914	627	0	0
TOTAL	12390	5221	12597	366	1.61 MB	12390	1.61 MB	25665	1487	2	0

From left to right, this table lists the following:

- **Members**: IP address and port of the member.
- **Entries**: Number of entries owned by the member.
- **Gets**: Number of get operations received by the member.
- **Puts**: Number of put operations received by the member.
- **Removes**: Number of remove operations received by the member.
- **Entry Memory**: Memory cost of owned entries in the member.
- **Backups**: Number of backup entries held by the member.
- **Backup Memory**: Memory cost of backup entries held by the member.
- **Events**: Number of events received by the member.
- **Hits**: Number of hits (reads) of the entries that are owned by the member, including those which are no longer in the map (for example, may have been evicted). The number of hits may be inaccurate after a partition is migrated to a new owner member.
- **Locks**: Number of currently locked entries owned by the member.
- **Dirty Entries**: Number of entries that the member owns and are dirty (updated but not persisted yet). In the cases where **MapStore** is enabled, these are the entries that are put to/removed from the map but not written to/removed from a database yet.

You can ascend or descend the order of the listings by clicking on the column headings.

Map Throughput Table provides information about the operations (get, put, remove) performed on each member in the map, as shown below.



Member	Puts/s	Gets/s	Removes/s	Avg Put Latency	Avg Get Latency	Avg Remove Lat...	Max Average Put...	Max Average Get...	Max Average Re...
127.0.0.1:5701	6.18	2.87	0.65	0.44ms	0.29ms	8s 295.13ms	0.44ms	0.3ms	12s 447ms
127.0.0.1:5702	10.82	4.13	0	0.68ms	0.46ms	0ms	0.75ms	0.52ms	0ms

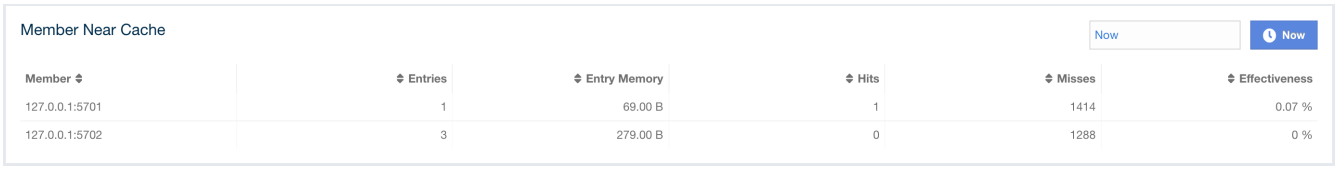
From left to right, this table lists the following:

- **Members**: IP address and port of the member.
- **Puts/s**: Number of put operations per second on the member.
- **Gets/s**: Number of get operations per second on the member.
- **Removes/s**: Number of remove operations per second on the member.
- **Avg Put Latency**: Average latency of put operations on the member.
- **Avg Get Latency**: Average latency of get operations on the member.
- **Avg Remove Latency**: Average latency of remove operations on the member.
- **Max Avg Put Latency**: Maximum average latency of put operations on the member.
- **Max Avg Get Latency**: Maximum average latency of get operations on the member.
- **Max Avg Remove Latency**: Maximum average latency of remove operations on the member.

You can select the time period in the combo box placed on the top right corner of the window, for which the table data will be shown. Available values are **Since Beginning**, **Last Minute**, **Last 10 Minutes** and **Last 1 Hour**.

To ascend or descend the order of the listings, click on the column headings.

Member Near Cache Table provides information about the Member Near Caches, if available, on each member, as shown below.



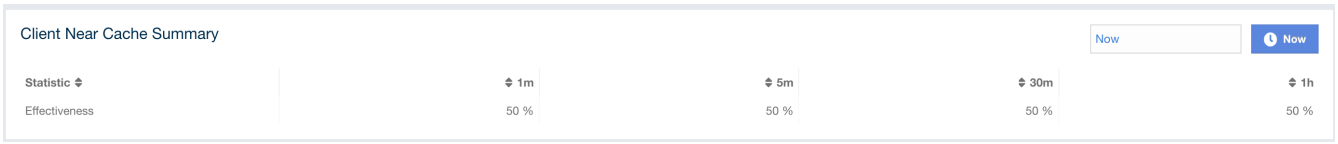
From left to right, this table lists the following:

- **Members**: IP address and port of the member which has Near Caches defined for the maps.
- **Entries**: Count of the entries in each Near Cache.
- **Entry Memory**: Memory cost of the entries in each Near Cache.
- **Hits**: Count of the entries read from the Near Cache.
- **Misses**: Count of the entries which cannot be found in the Near Cache when requested to read.
- **Ratio**: Hits/Misses ratio.

To ascend or descend the order of the listings, click on the column headings.

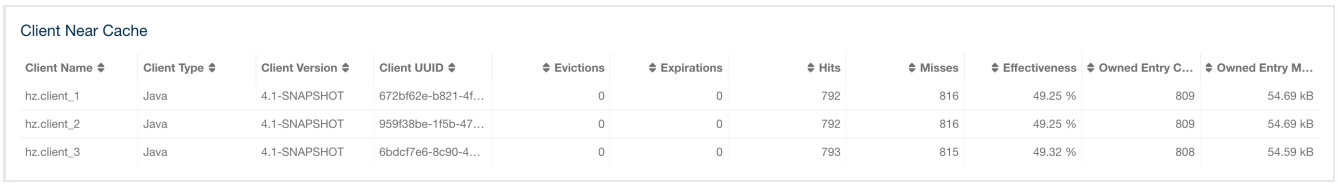
Client Near Cache Summary provides summary information related to the Near Cache statistics aggregated for all the clients that have Near Cache enabled for this map. Aggregated statistics are shown for the following periods: *1 minute*, *5 minutes*, *30 minutes* and *60 minutes*. Currently, the table shows overall Near Cache effectiveness, calculated as hits/total reads ratio.

You need to enable the statistics for clients to see them here. Please refer to Monitoring Clients for details.



Client Near Cache Table provides information about the Near Caches statistics, if available, on each client that has Near Cache enabled for this map, as shown below.

You need to enable the statistics for clients to see them here. Please refer to Monitoring Clients for details.



From left to right, this table lists the following:

- **Client Name**: Name of the client instance which has Near Cache defined for the map.
- **Client Type**: Type of the client.
- **Client Version**: Version of the client.
- **Client UUID**: Client unique identifier.
- **Evictions**: Number of evictions of Near Cache entries owned by the client.
- **Expirations**: Number of TTL and max-idle expirations of Near Cache entries owned by the client.
- **Hits**: Number of hits (reads) of Near Cache entries owned by the client.
- **Misses**: Number of misses of Near Cache entries owned by the client.
- **Effectiveness**: Hits/total reads ratio.
- **Owned Entry Count**: Number of Near Cache entries owned by the client.
- **Owned Entry Memory Cost**: Memory cost of Near Cache entries owned by the client.

To ascend or descend the order of the listings, click on the column headings.

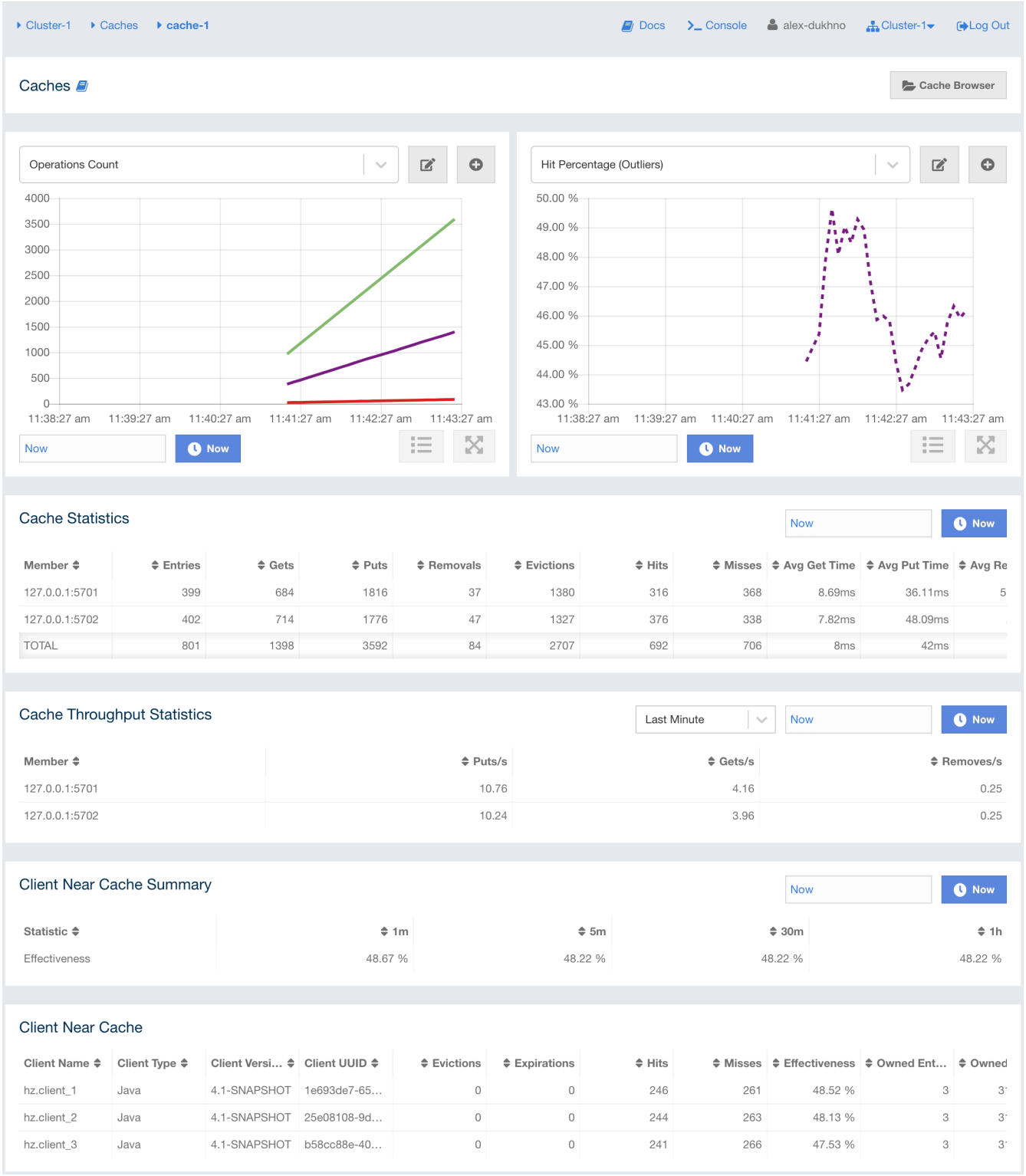
11.6.2. Caches

You can see a list of all the caches in your cluster by clicking on the **Caches** menu item on the left panel. A new page is opened on the right, as shown below.

Name	Entries	Hits	Misses
cache-42	19	0	0
cache-43	17	0	0
cache-44	7	0	0
cache-45	18	0	0
cache-46	14	0	0
cache-47	12	0	0
cache-48	8	0	0
cache-49	31	0	0
cache-5	6	0	0
cache-50	15	0	0
cache-51	13	0	0
cache-52	24	0	0
cache-53	12	0	0
cache-54	8	0	0
cache-55	15	0	0
cache-56	29	0	0
cache-57	12	0	0
cache-58	6	0	0
cache-59	13	0	0
cache-6	7	0	0

« ‹ 1 2 3 4 5 6 › »

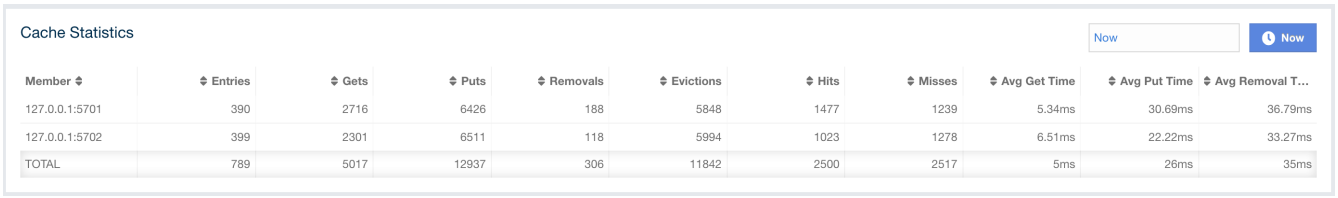
You can filter the caches shown and you can also sort the table by clicking on the column headers. Clicking on the cache name opens a new page for monitoring that cache instance on the right, as shown below.



On top of the page, there are two widgets that show various metrics of the cache. See the Widget section for more information.

Under these charts are **Cache Statistics**, **Cache Throughput**, and **Client Near Cache** data tables.

Cache Statistics Table provides the selected cache's statistics distributed over the members, as shown below.



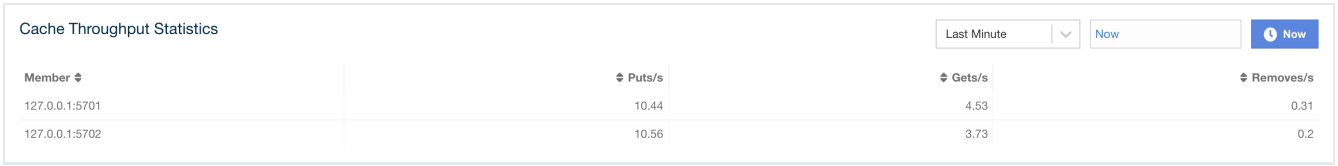
Member	Entries	Gets	Puts	Removals	Evictions	Hits	Misses	Avg Get Time	Avg Put Time	Avg Removal T...
127.0.0.1:5701	390	2716	6426	188	5848	1477	1239	5.34ms	30.69ms	36.79ms
127.0.0.1:5702	399	2301	6511	118	5994	1023	1278	6.51ms	22.22ms	33.27ms
TOTAL	789	5017	12937	306	11842	2500	2517	5ms	26ms	35ms

From left to right, this table lists the following in real time:

- **Members**: IP address and port of the member.
- **Entries**: Number of entries in this cache owned by the member.
- **Gets/Puts/Removals**: Number of the get/put/remove operations for this cache received by the member.
- **Evictions**: Number of evictions of Cache entries owned by the member.
- **Hits**: Number of the reads performed for this cache's entries.
- **Misses**: Number of the entries which cannot be found in the cache when requested to read.
- **Avg Get/Put/Removal Time**: Average elapsed time for the get/put/removal operations for the cache on each member.

To ascend or descend the order of the listings, click on the column headings.

Cache Throughput Statistic Table provides information about the operations (get, put, remove) performed on each member for the selected cache.



Member	Puts/s	Gets/s	Removes/s
127.0.0.1:5701	10.44	4.53	0.31
127.0.0.1:5702	10.56	3.73	0.2

From left to right, this table lists the following:

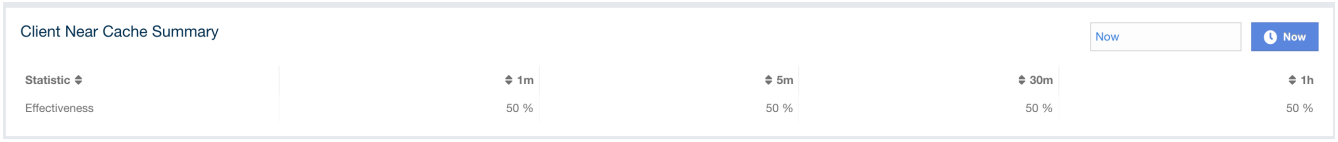
- IP address and port of each member.
- Put, get and remove operation rates on each member for this cache.

You can select the period in the combo box placed at the top right corner of the window, for which the table data will be shown. Available values are **Since Beginning**, **Last Minute**, **Last 10 Minutes** and **Last 1 Hour**.

You can ascend or descend the order of the listings in each column by clicking on column headings.

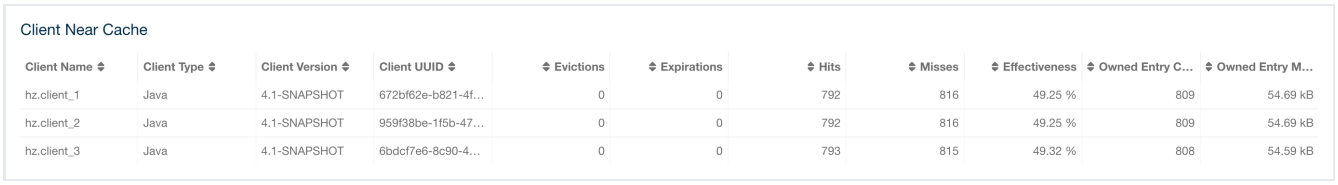
Client Near Cache Summary provides summary information related to the Near Cache statistics aggregated for all the clients that have Near Cache enabled for this cache. Aggregated statistics are shown for the following periods: *1 minute*, *5 minutes*, *30 minutes* and *60 minutes*. Currently, the table shows overall Near Cache effectiveness, calculated as hits/total reads ratio.

You need to enable the statistics for clients to see them here. Please refer to Monitoring Clients for details.



Client Near Cache Table provides information about the Near Caches statistics, if available, on each client that has Near Cache enabled for this cache, as shown below.

You need to enable the statistics for clients to see them here. Please refer to Monitoring Clients for details.



From left to right, this table lists the following:

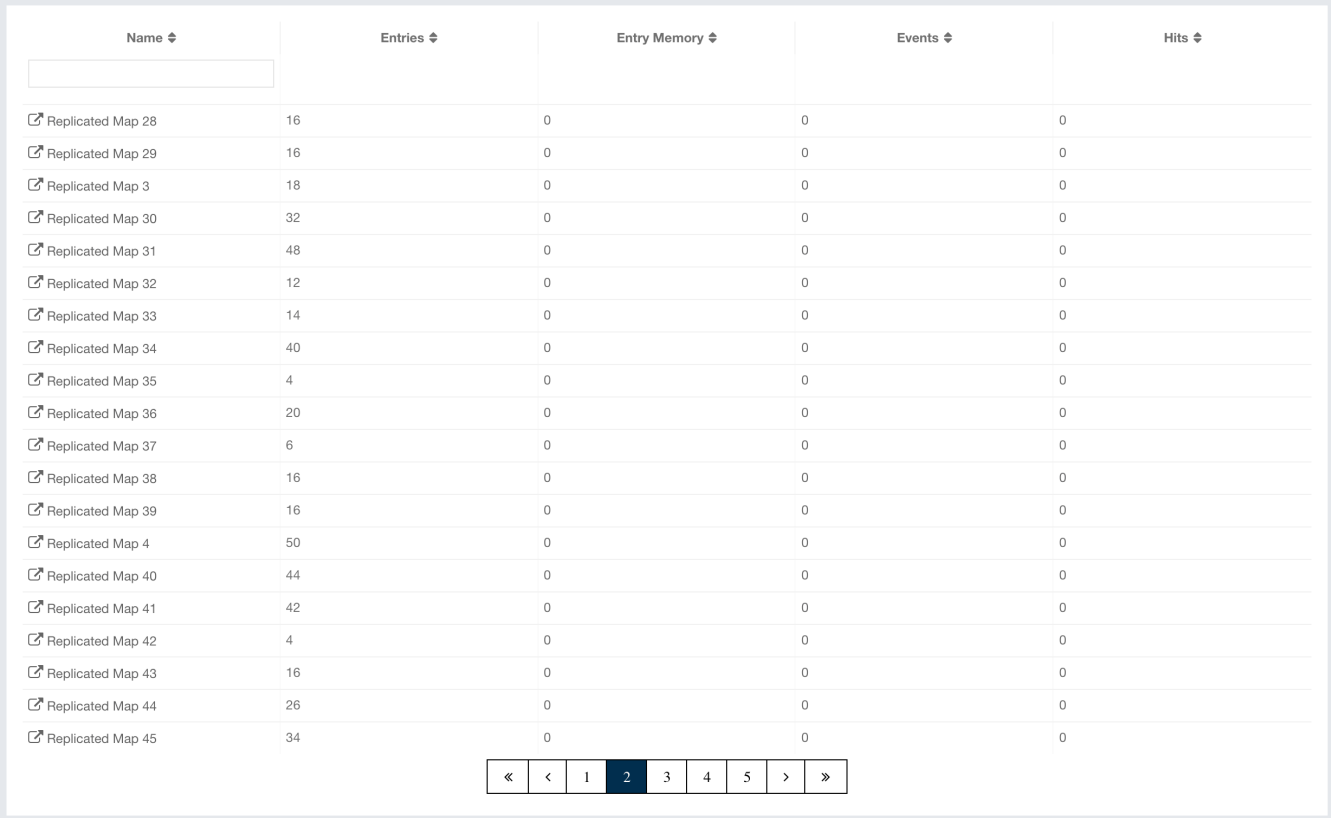
- **Client Name**: Name of the client instance which has Near Cache enabled for the map.
- **Client Type**: Type of the client.
- **Client Version**: Version of the client.
- **Client UUID**: Client unique identifier.
- **Evictions**: Number of evictions of Near Cache entries owned by the client.
- **Expirations**: Number of TTL and max-idle expirations of Near Cache entries owned by the client.
- **Hits**: Number of hits (reads) of Near Cache entries owned by the client.
- **Misses**: Number of misses of Near Cache entries owned by the client.
- **Effectiveness**: Hits/total reads ratio.
- **Owned Entry Count**: Number of Near Cache entries owned by the client.
- **Owned Entry Memory Cost**: Memory cost of Near Cache entries owned by the client.

To ascend or descend the order of the listings, click on the column headings.

You need to enable the statistics for caches to monitor them in the Management Center. Use the `<statistics-enabled>` element or `setStatisticsEnabled()` method in declarative or programmatic configuration, respectively, to enable the statistics. Please refer to the JCache Declarative Configuration section for more information.

11.6.3. Replicated Maps

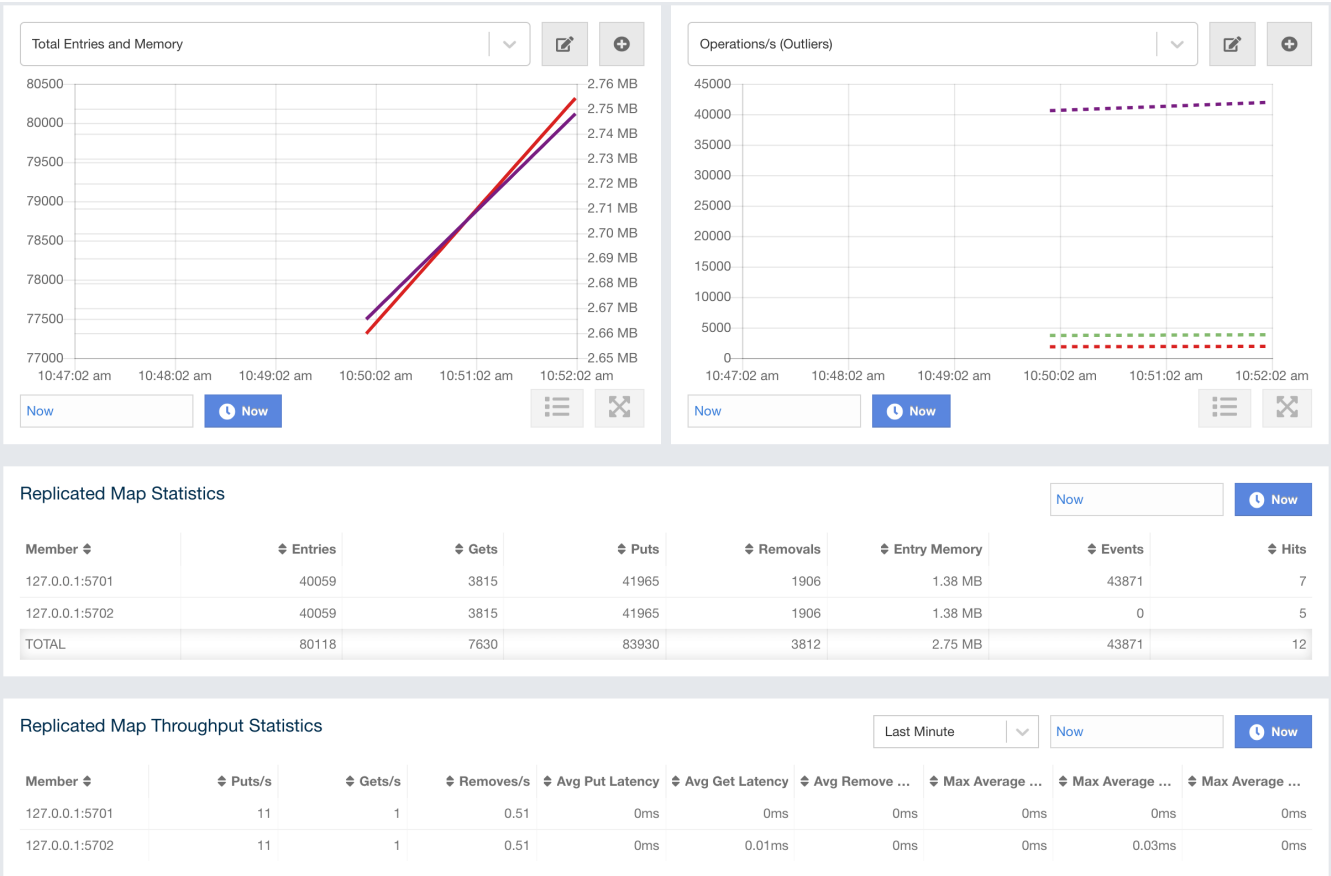
You can see a list of all the Replicated Maps in your cluster by clicking on the **Replicated Maps** menu item on the left panel. A new page is opened on the right, as shown below.



Name	Entries	Entry Memory	Events	Hits
Replicated Map 28	16	0	0	0
Replicated Map 29	16	0	0	0
Replicated Map 3	18	0	0	0
Replicated Map 30	32	0	0	0
Replicated Map 31	48	0	0	0
Replicated Map 32	12	0	0	0
Replicated Map 33	14	0	0	0
Replicated Map 34	40	0	0	0
Replicated Map 35	4	0	0	0
Replicated Map 36	20	0	0	0
Replicated Map 37	6	0	0	0
Replicated Map 38	16	0	0	0
Replicated Map 39	16	0	0	0
Replicated Map 4	50	0	0	0
Replicated Map 40	44	0	0	0
Replicated Map 41	42	0	0	0
Replicated Map 42	4	0	0	0
Replicated Map 43	16	0	0	0
Replicated Map 44	26	0	0	0
Replicated Map 45	34	0	0	0

« ‹ 1 2 3 4 5 › »

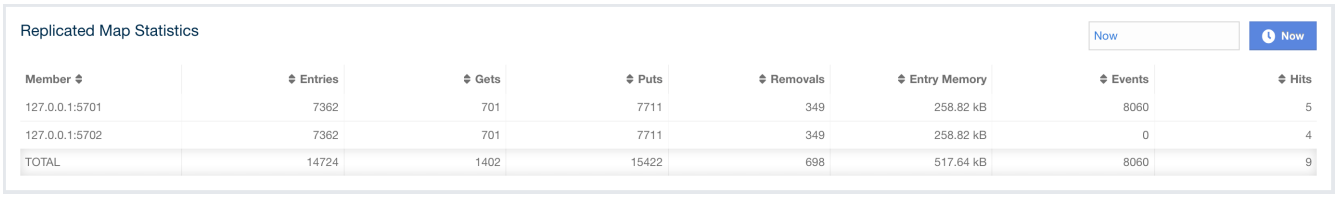
You can filter the Replicated Maps shown and you can also sort the table by clicking on the column headers. Clicking on a Replicated Map name opens a new page for monitoring that Replicated Map instance on the right, as shown below.



In this page, you can monitor metrics of the selected Replicated Map. The page shows two widgets that show various metrics of the Replicated Map. See the Widget section for more information.

Under these charts are **Replicated Map Statistics** and **Replicated Map Throughput** data tables.

Replicated Map Statistics Table provides statistics distributed over the members, as shown below.

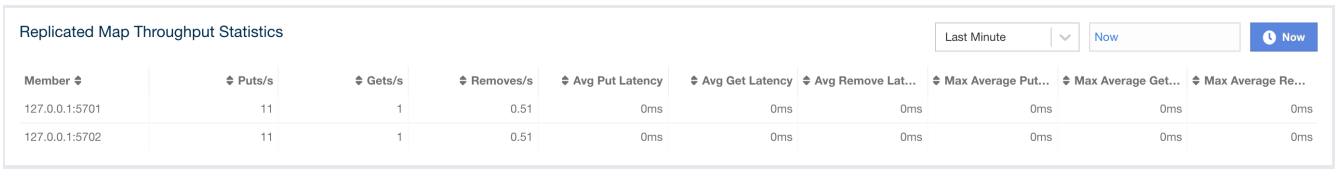


Member	Entries	Gets	Puts	Removals	Entry Memory	Events	Hits
127.0.0.1:5701	7362	701	7711	349	258.82 kB	8060	5
127.0.0.1:5702	7362	701	7711	349	258.82 kB	0	4
TOTAL	14724	1402	15422	698	517.64 kB	8060	9

From left to right, this table lists the following:

- **Members**: IP address and port of the member.
- **Entries**: Number of entries in this Replicated Map owned by the member.
- **Gets/Puts/Removals**: Number of the get/put/remove operations for this Replicated Map received by the member.
- **Entry Memory**: Memory cost of the owned entries in the member.
- **Events**: Number of the events received by the member.
- **Hits**: Number of the reads performed for this Replicated Map's entries.

Replicated Map Throughput Table provides information about operations (get, put, remove) performed on each member in the selected Replicated Map.



Member	Puts/s	Gets/s	Removes/s	Avg Put Latency	Avg Get Latency	Avg Remove Lat...	Max Average Put...	Max Average Get...	Max Average Re...
127.0.0.1:5701	11	1	0.51	0ms	0ms	0ms	0ms	0ms	0ms
127.0.0.1:5702	11	1	0.51	0ms	0ms	0ms	0ms	0ms	0ms

From left to right, this table lists the following:

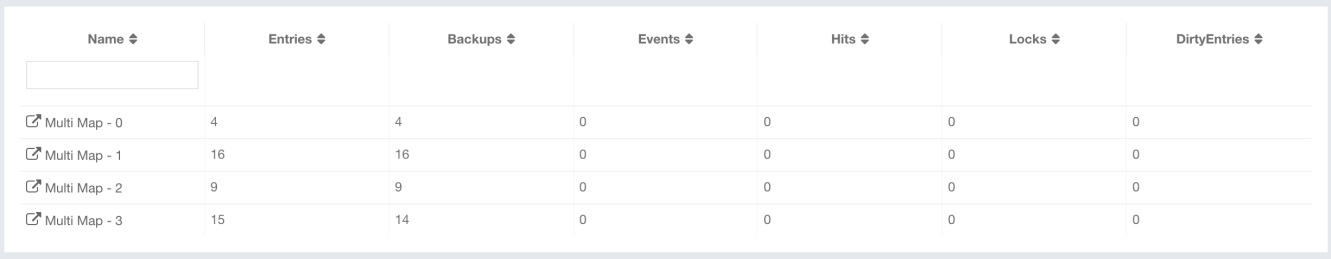
- IP address and port of each member
- put, get, and remove operations on each member
- average put, get, and remove latencies
- maximum average put, get, and remove latencies on each member.

You can select the period from the combo box placed at the top right corner of the window, in which the table data is shown. Available values are **Since Beginning**, **Last Minute**, **Last 10 Minutes** and **Last 1 Hour**.

To ascend or descend the order of the listings, click on the column headings.

11.6.4. MultiMaps

You can see a list of all the MultiMaps in your cluster by clicking on the **MultiMaps** menu item on the left panel. A new page is opened on the right, as shown below.



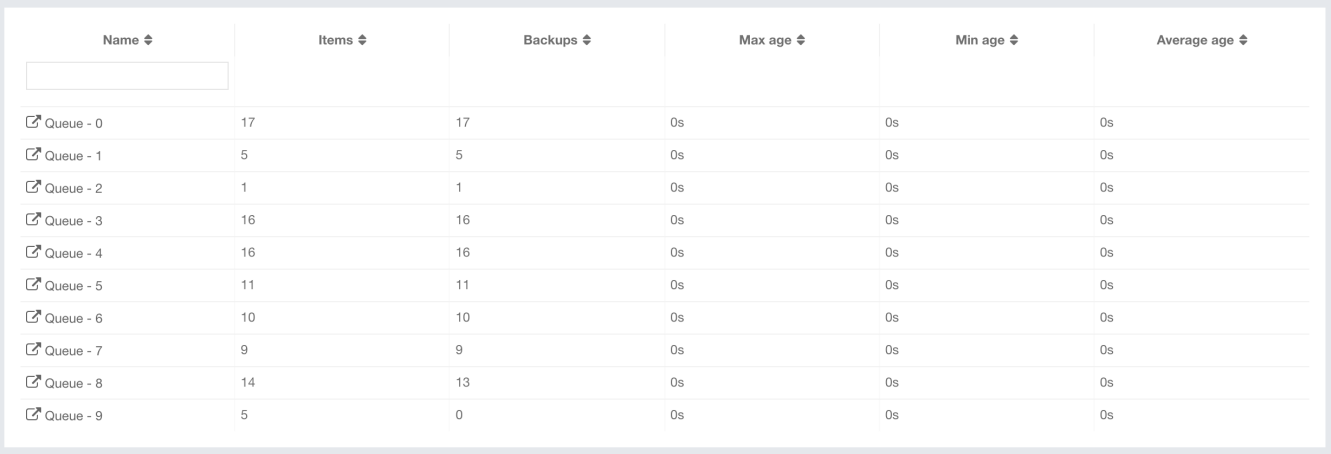
Name	Entries	Backups	Events	Hits	Locks	DirtyEntries
Multi Map - 0	4	4	0	0	0	0
Multi Map - 1	16	16	0	0	0	0
Multi Map - 2	9	9	0	0	0	0
Multi Map - 3	15	14	0	0	0	0

You can filter the MultiMaps shown and you can also sort the table by clicking on the column headers. Clicking on a MultiMap name opens a new page for monitoring that MultiMap instance on the right.

MultiMap is a specialized map where you can associate a key with multiple values. This monitoring option is similar to the **Maps** option: the same monitoring charts and data tables monitor MultiMaps. The differences are that you cannot browse the MultiMaps and re-configure it. Please see the Managing Maps.

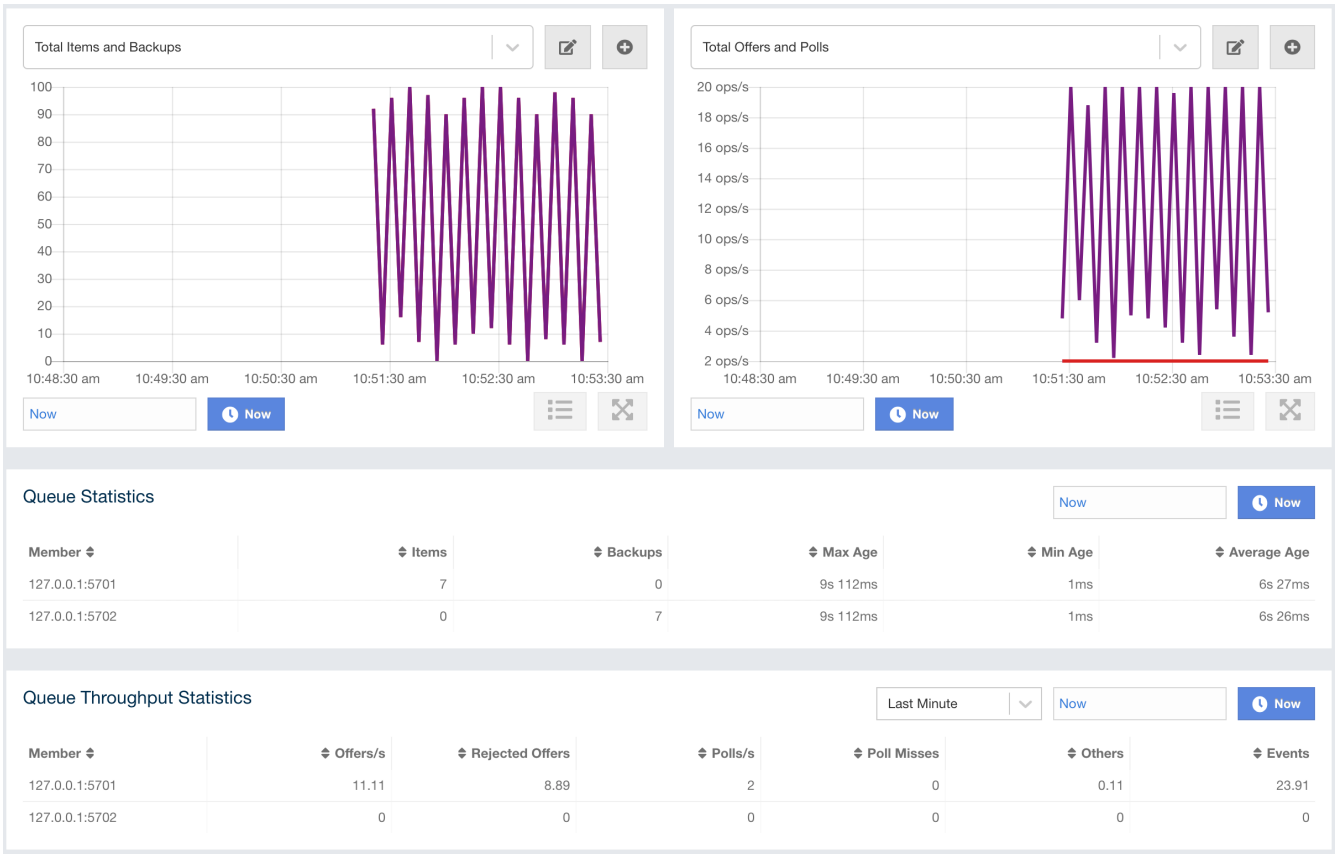
11.6.5. Queues

You can see a list of all the queues in your cluster by clicking on the **Queues** menu item on the left panel. A new page is opened on the right, as shown below.



Name	Items	Backups	Max age	Min age	Average age
Queue - 0	17	17	0s	0s	0s
Queue - 1	5	5	0s	0s	0s
Queue - 2	1	1	0s	0s	0s
Queue - 3	16	16	0s	0s	0s
Queue - 4	16	16	0s	0s	0s
Queue - 5	11	11	0s	0s	0s
Queue - 6	10	10	0s	0s	0s
Queue - 7	9	9	0s	0s	0s
Queue - 8	14	13	0s	0s	0s
Queue - 9	5	0	0s	0s	0s

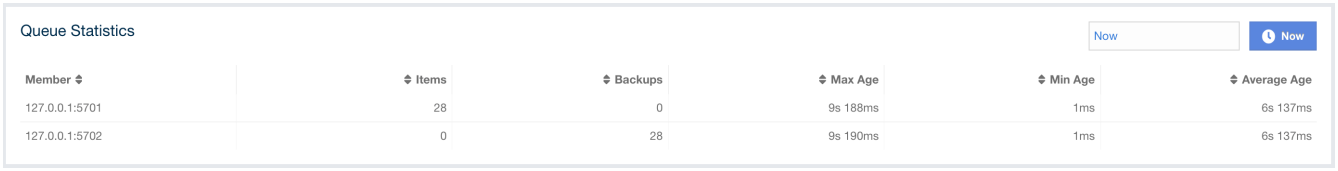
You can filter the queues shown and you can also sort the table by clicking on the column headers. Clicking on a queue name opens a new page for monitoring that queue instance on the right, as shown below.



On top of the page, there are two widgets that show various metrics of the queue. See the Widget section for more information.

Under these charts are **Queue Statistics** and **Queue Throughput Statistics** tables.

Queue Statistics table provides item and backup item counts in the queue and age statistics of items and backup items at each member, as shown below.



From left to right, this table lists the IP address and port, items and backup items on the queue of each member, and maximum, minimum and average age of items in the queue. The order of the listings in each column can be ascended or descended by clicking on the column headings.

Queue Operation Statistics table provides information about the operations (offers, polls, events) performed on the queues, as shown below.

From left to right, this table lists the IP address and port of each member, and counts of offers, rejected offers, polls, poll misses and events.

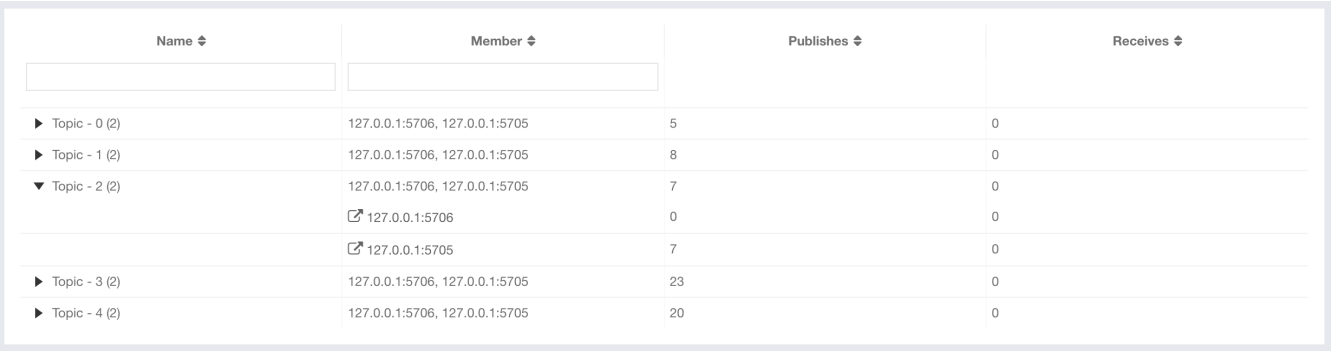
You can select the period in the combo box placed at the top right corner of the window to show the

table data. Available values are **Since Beginning**, **Last Minute**, **Last 10 Minutes** and **Last 1 Hour**.

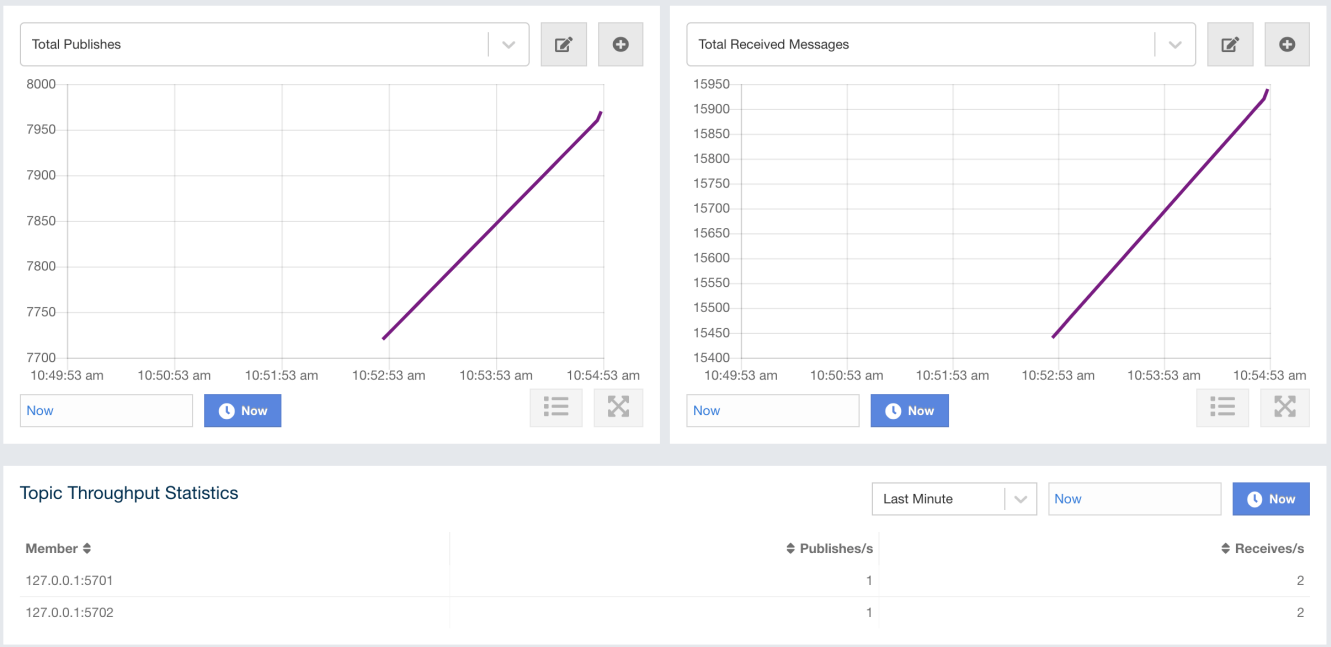
Click on the column headings to ascend or descend the order of the listings.

11.6.6. Topics

You can see a list of all the topics in your cluster by clicking on the **Topics** menu item on the left panel. A new page is opened on the right, as shown below.



You can filter the topics shown and you can also sort the table by clicking on the column headers. Clicking on a topic name opens a new page for monitoring that topic instance on the right, as shown below.



On top of the page, there are two widgets that show various metrics of the topic. See the Widget section for more information.

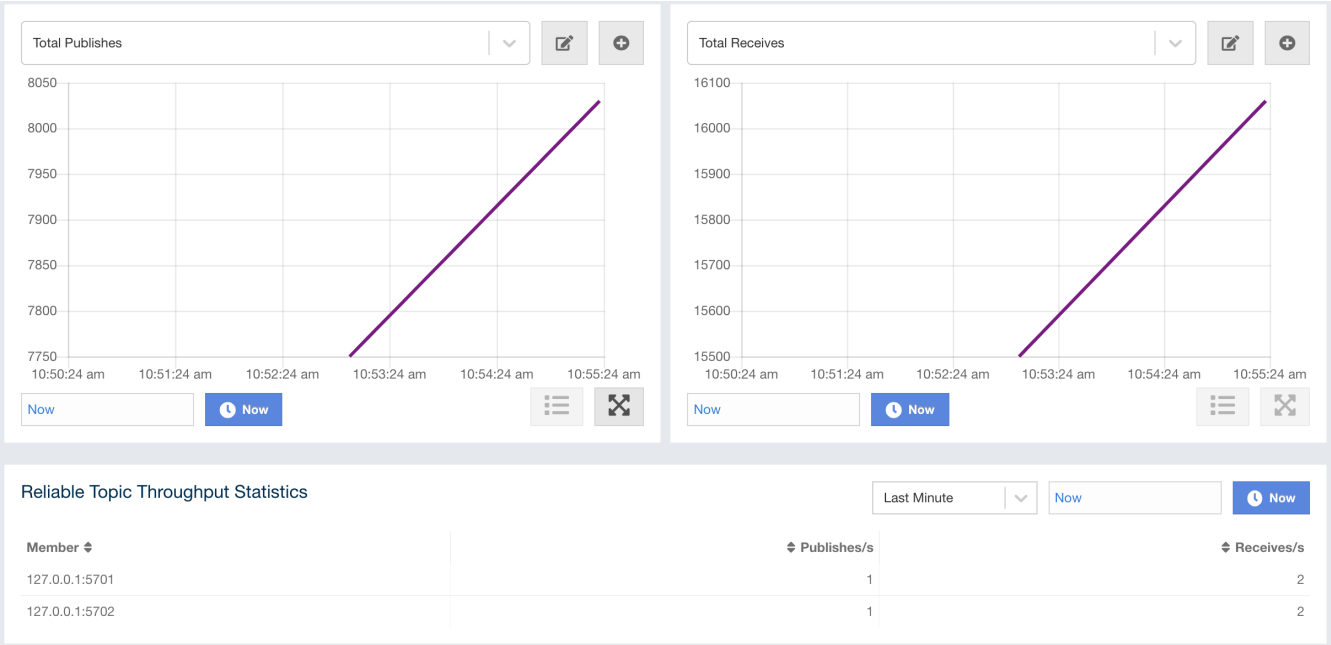
Under these charts is the Topic Operation Statistics table. From left to right, this table lists the IP addresses and ports of each member, and counts of the messages published and received per second in real-time. You can select the period in the combo box placed at top right corner of the table to show the table data. The available values are **Since Beginning**, **Last Minute**, **Last 10 Minutes** and **Last 1 Hour**.

Click on the column heading to ascend or descend the order of the listings.

11.6.7. Reliable Topics

You can see a list of all the Reliable Topics in your cluster by clicking on the **Reliable Topics** menu item on the left panel. A new page is opened on the right, as shown below.

You can filter the Reliable Topics shown and you can also sort the table by clicking on the column headers. Clicking on a Reliable Topic name opens a new page for monitoring that Reliable Topic instance on the right, as shown below.



On top of the page, there are two widgets that show various metrics of the reliable topic. See the Widget section for more information.

Under these charts is the Reliable Topic Operation Statistics table. From left to right, this table lists the IP addresses and ports of each member, and counts of the messages published and received per second in real-time. You can select the period in the combo box placed at top right corner of the table to show the table data. The available values are **Since Beginning**, **Last Minute**, **Last 10 Minutes** and **Last 1 Hour**.

Click on the column heading to ascend or descend the order of the listings.

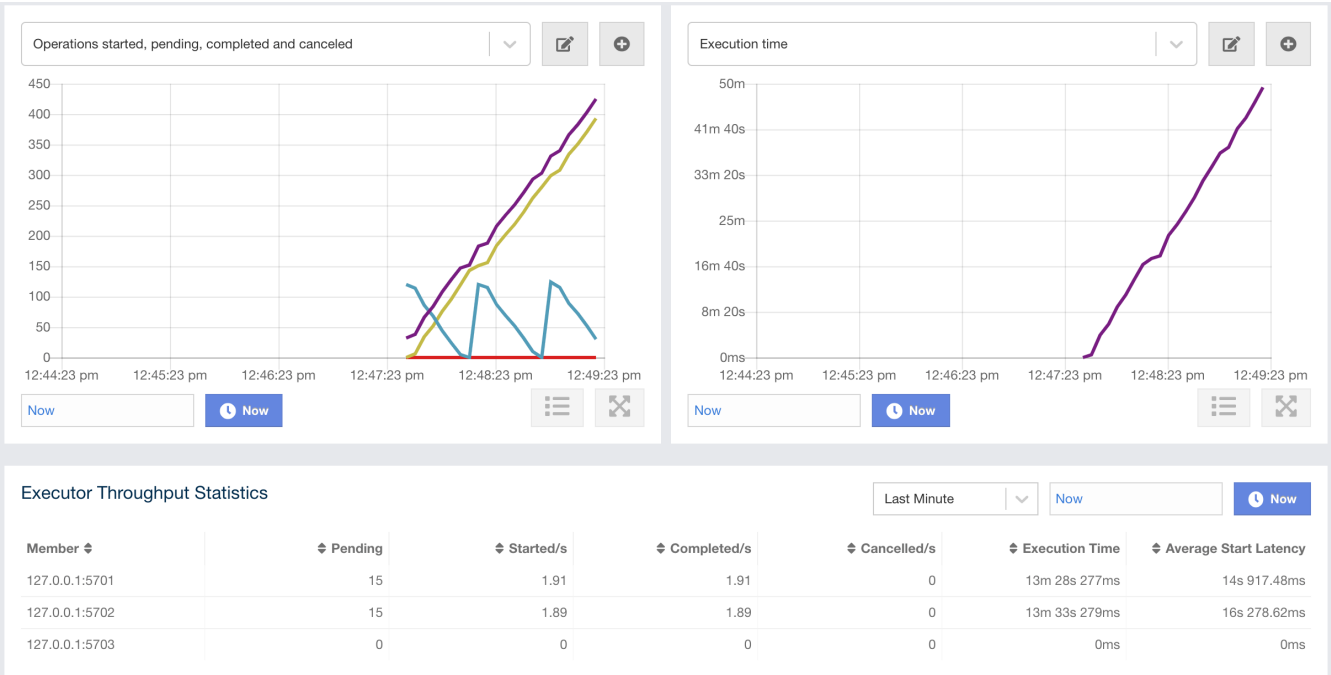
11.6.8. Executors

You can see a list of all the Executors in your cluster by clicking on the **Executors** menu item on the left panel. A new page is opened on the right, as shown below.

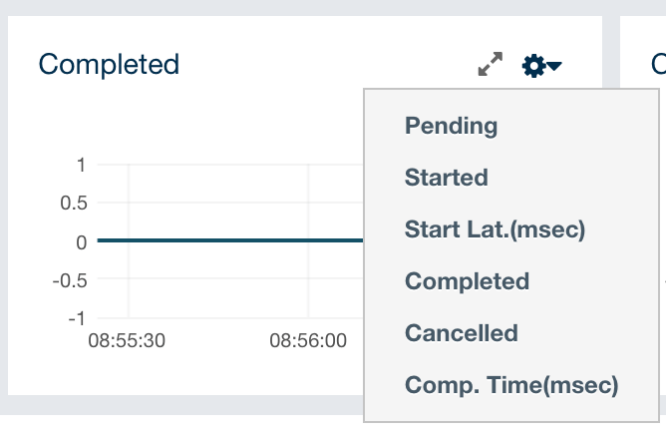
Name	Member	Pending	Started	Completed	Cancelled	Execution Time	Avg Start Latency
Executor - 0 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		0, 0
Executor - 1 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		1, 0
Executor - 10 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		1, 2
Executor - 11 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		1, 1
Executor - 12 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		1, 1
Executor - 13 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		0, 0
Executor - 14 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		5, 5
Executor - 15 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		5, 5
Executor - 16 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		4, 0
	127.0.0.1:5706	0	1	1	0	1m 30s	4
	127.0.0.1:5705	0	1	1	0	28m 51s	0
Executor - 17 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		4, 4
Executor - 18 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		3, 4
Executor - 19 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		3, 3
Executor - 2 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		0, 1
Executor - 20 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		3, 3
Executor - 21 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		2, 2
Executor - 22 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		1, 1
Executor - 23 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		0, 1
Executor - 24 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		1, 0
Executor - 25 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		5, 6
Executor - 26 (2)	127.0.0.1:5706, 127.0....	0, 0	1, 1	1, 1	0, 0		5, 5

« ‹ 1 2 3 4 5 › »

You can filter the Executors shown and you can also sort the table by clicking on the column headers. Clicking on an Executor name opens a new page for monitoring that Executor instance on the right, as shown below.



On top of the page, there are two widgets that show various metrics of the Executor. See the Widget section for more information.



When you click on a desired monitoring, the chart loads with the selected option. To open a chart as a separate dialog, click on the button placed at top right of each chart. The below monitoring charts are available:

- **Pending**: Monitors the pending executors. Y-axis is the executor count.
- **Started**: Monitors the started executors. Y-axis is the executor count.
- **Start Lat. (msec.)**: Shows the latency when executors are started. Y-axis is the duration in milliseconds.
- **Completed**: Monitors the completed executors. Y-axis is the executor count.
- **Comp. Time (msec.)**: Shows the completion period of executors. Y-axis is the duration in milliseconds.

Under these charts is the **Executor Operation Statistics** table, as shown below.

From left to right, this table lists the IP address and port of members, the counts of pending, started and completed executors per second, and the execution time and average start latency of executors on each member. Click on the column heading to ascend or descend the order of the listings.

11.6.9. Locks

You can use the scripting feature of the Management Center to monitor the locks in your cluster. See the Scripting section to learn how to use this feature.

You can use the below scripts to retrieve various information about the locks in your cluster.

To find the number of active locks in your cluster, use the following script:

```
var findLocks = function() {
    var lockstr = '';
    var node = hazelcast.getCluster().getLocalMember();

    var locks =
hazelcast.node.nodeEngine.getService('hz:impl:lockService').getAllLocks();
    return "Active Lock Count : " + locks.size();

}

findLocks();
```

To print the locks in your cluster, use the following script:

```
var findLocks = function() {
        var lockStr = '';
        var distributedObjects = hazelcast.getDistributedObjects();
        for each(distributedObject in distributedObjects) {
            if(distributedObject.getServiceName().equals("hz:impl:lockService")){
                lockStr += distributedObject.getName() + '\n';
            }

        }
        return lockStr;
}

findLocks();
```

To force unlock a lock in your cluster, use the following script:

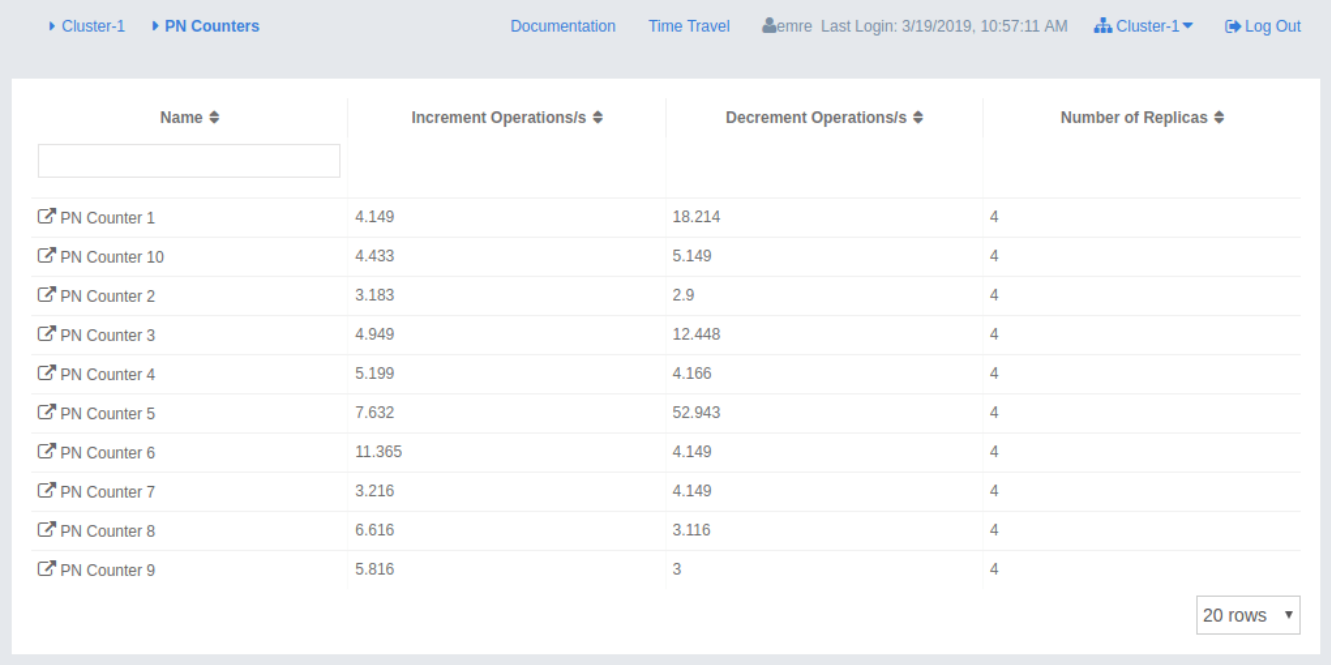
```
var forceUnlock = function(lockName) {

    hazelcast.getLock(lockName).forceUnlock();
    return 'OK';

}

forceUnlock('your_Lock_Name');
```

To check if a lock is being hold by a member, use the following script:

```
var isLocked = function(lockName) {

    var locked = hazelcast.getLock(lockName).isLocked();
    return lockName + ' -> ' + locked;

}

isLocked('your_Lock_Name');
```

11.6.10. PN Counters

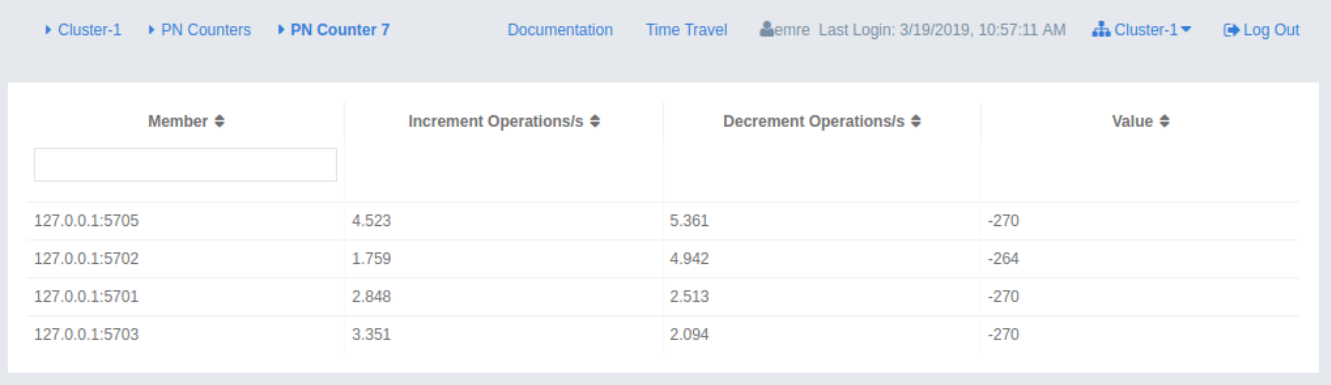
You can see a list of all the PN counters in your cluster by clicking on the **Counters** menu item on the left panel. A new page is opened on the right, as shown below.



You can filter the counters shown and you can also sort the table by clicking on the column headers. The monitoring data available are:

- **Increment Operations/s**: Average number of times the counter was incremented per second during the last timeslice.
- **Decrement Operations/s**: Average number of times the counter was decremented per second during the last timeslice.
- **Number of Replicas**: Number of member instances that have a state for the counter.

Clicking on a counter name opens a new page for monitoring that specific counter instance, as shown below.

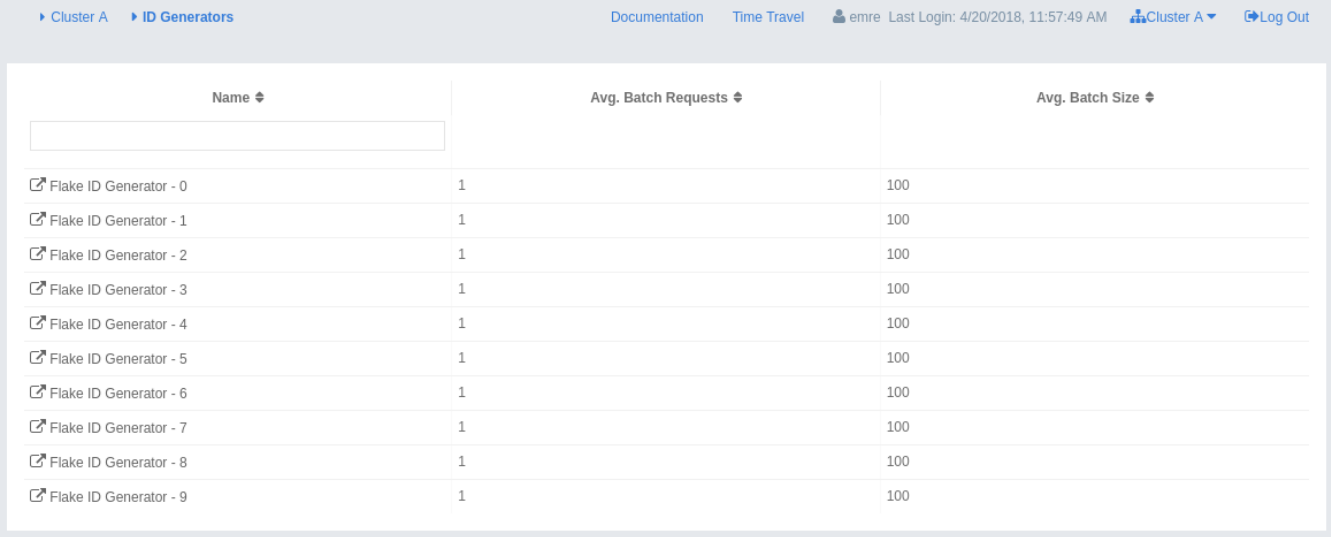


The table can likewise be sorted by clicking the column headers. It shows IP and port of the members that have a state for the specific counter named in the page's title. The monitoring data available are:

- **Increment Operations/s**: Average number of times the counter was incremented on that member per second during the last timeslice
- **Decrement Operations/s**: Average number of times the counter was decremented on that member per second during the last timeslice
- **Value**: Current value of the counter on that member.

11.6.11. Flake ID Generators

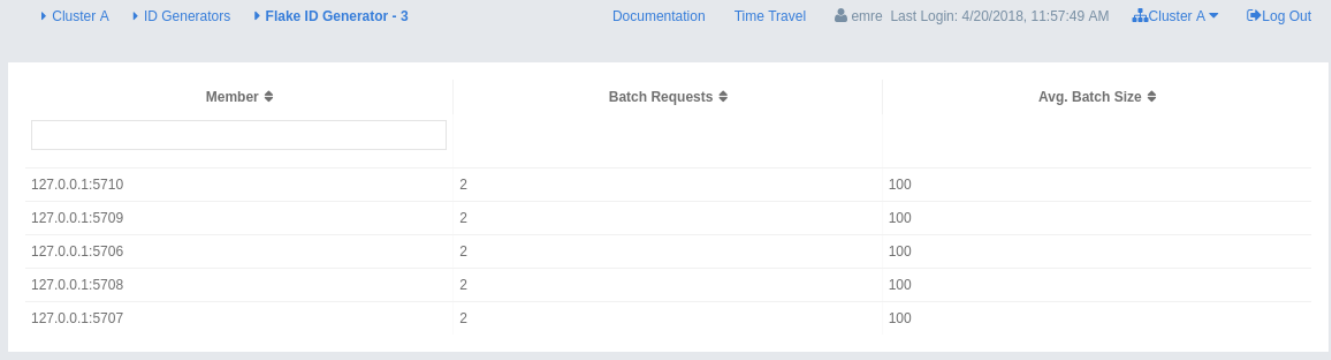
You can see a list of all Flake ID Generators in your cluster by clicking on the **ID Generators** menu item on the left panel. A new page is opened on the right, as shown below.



You can filter the generators shown and you can also sort the table by clicking on the column headers. The monitoring data available are:

- **Avg. Batch Requests:** Average count of batch requests coming from all the members to a generator, i.e., total batch requests from all members to a generator divided by the member count for that generator.
- **Avg. Batch Size:** Average size of the ID batches created by a generator, i.e., total number of IDs generated (the sum of IDs for all batches) for all members divided by the total count of batch requests coming from all members.

Clicking on a generator name opens a new page for monitoring that specific generator instance, as shown below.



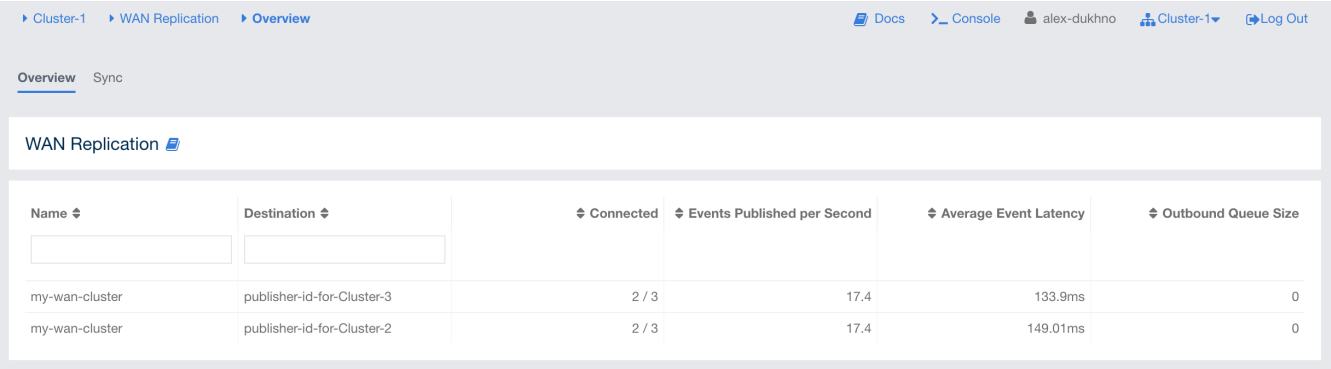
The table can likewise be sorted by clicking the column headers. It shows IP and port of the members that have a state for the specific generator named in the page's title. The monitoring data available are:

- **Batch Requests:** Total count of batch requests to a generator by this member.
- **Avg. Batch Size:** Average size of the ID batches created for this member, i.e., total number of IDs generated (the sum of IDs for all batches) for this member divided by the total count of batch requests coming from this member.

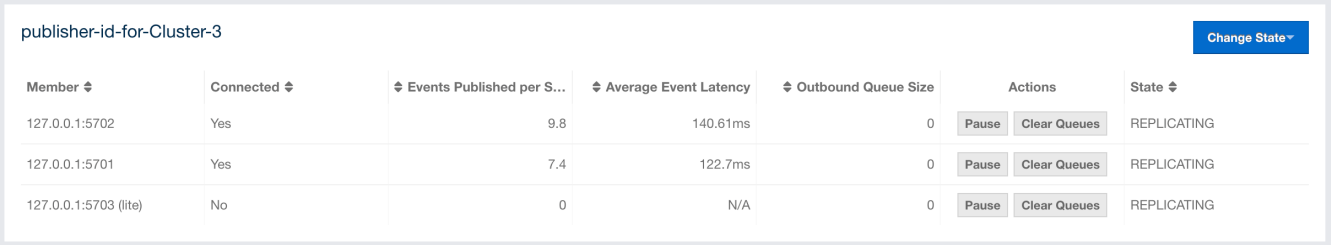
The operations per second is not the number of new IDs generated or used but the number of ID batches. The batch size is configurable, usually it contains hundreds or thousands of IDs. A client uses all IDs from a batch before a new batch is requested.

11.7. Monitoring WAN Replication

WAN replication schemes are listed under the **WAN Replication** menu item on the left. When you click on a scheme, a new page for monitoring the targets which that scheme has appears on the right, as shown below:



In this page, you see the **WAN Replication Operations Table** for each target which belongs to this scheme. One of the example tables is shown below:



- **Connected**: Status of the member connection to the target. "Yes" means that the member is sending WAN events to the target and connected to it. "No" means the member is not sending WAN events to the target due to, e.g., having not enough events to be sent and other members are already sending the events in hand.
- **Events Published per Second**: Number of published events per second. Please see the paragraph below.
- **Average Event Latency**: Average latency of sending a record to the target from this member. Please see the paragraph below.
- **Outbound Queue Size**: Number of records waiting in the queue to be sent to the target.
- **Action**: Pause, stop or resume replication of a member's records. You can also clear the event queues in a member using the "Clear Queues" action. For instance, if you know that the target cluster is being shut down, decommissioned, put out of use and it will never come back, you may additionally clear the WAN queues to release the consumed heap after the publisher has been switched. Or, when a failure happens and queues are not replicated anymore, you could manually clear the queues using, again the "Clear Queues" action.
- **State**: Shows current state of the WAN publisher on a member. See Changing WAN Publisher State for the list of possible WAN publisher states.

Events Published per Second and **Average Event Latency** are based on the following internal statistics:

- Total published event count (TBEC): Total number of events that are successfully sent to the target cluster since the start-up of the member.
- Total latency (TL): Grand total of each event's waiting time in the queue, including network transmit and receiving ACK from the target.

Each member sends these two statistics to the Management Center at intervals of 3 seconds (update interval). Management Center derives **Events Published per Second** and **Average Event Latency** from these statistics as formulated below:

Events Published per Second = (Current TBEC - Previous TBEC) / Update Interval

Average Event Latency = (Current TL - Previous TL) / (Current TBEC - Previous TBEC)

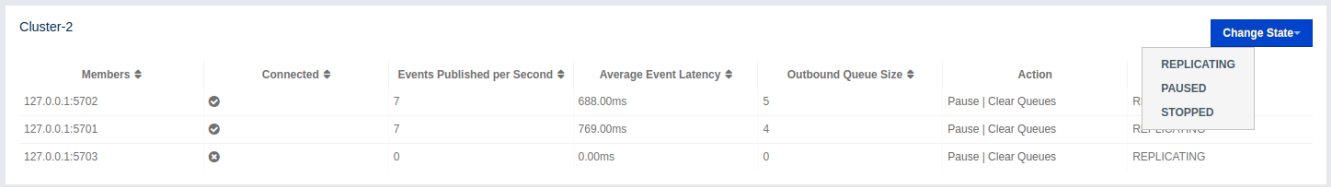
11.7.1. Changing WAN Publisher State

A WAN publisher can be in one of the following states:

- `REPLICATING` (default): State where both enqueuing new events is allowed, enqueued events are replicated to the target cluster.

- `PAUSED`: State where new events are enqueued but they are not dequeued. Some events which have been dequeued before the state was switched may still be replicated to the target cluster but further events will not be replicated.
- `STOPPED`: State where neither new events are enqueued nor dequeued. As with the `PAUSED` state, some events might still be replicated after the publisher has switched to this state.

You can change a WAN publisher's state by clicking the `Change State` dropdown button on top right hand corner of the WAN Replication Operations Table.



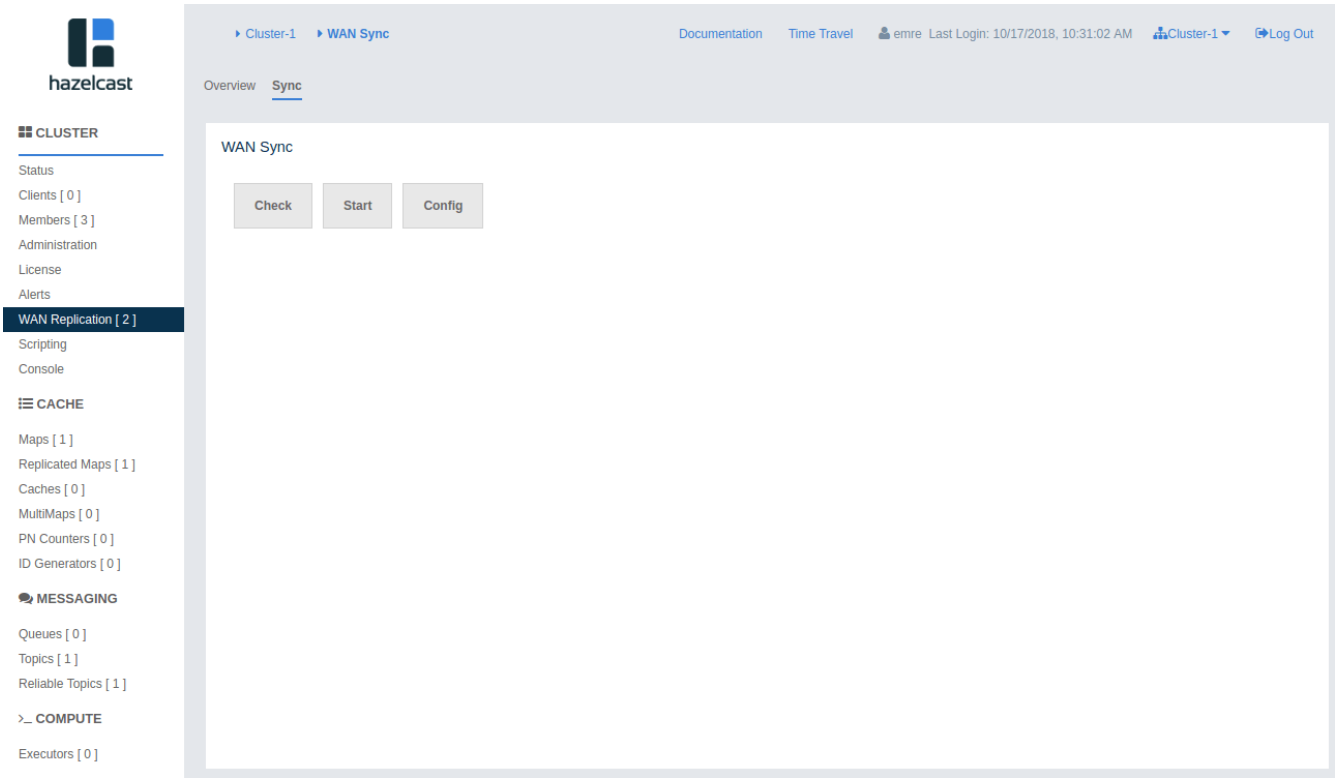
11.7.2. WAN Sync

You can initiate a synchronization operation on an IMap for a specific target cluster. This operation is useful if two remote clusters lost their synchronization due to WAN queue overflow or in restart scenarios.

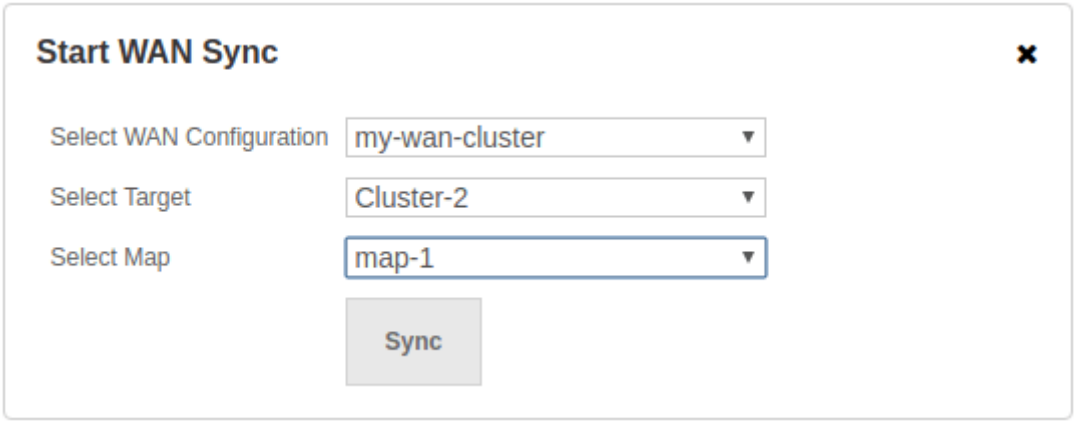
Hazelcast provides the following synchronization options:

1. Default WAN synchronization operation: It sends all the data of an IMap to a target cluster to align the state of target IMap with the source IMap. See here for more information.
2. WAN synchronization using Merkle trees: To initiate this type of synchronization, you need to configure the cluster members. See the Delta WAN Synchronization section in Hazelcast IMDG Reference Manual for details about configuring them. Make sure you meet all the requirements to use Delta WAN Synchronization and do the configuration on both the source and target clusters.

To initiate WAN Sync, open the **WAN Replication** menu item on the left and navigate to the **Sync** tab.

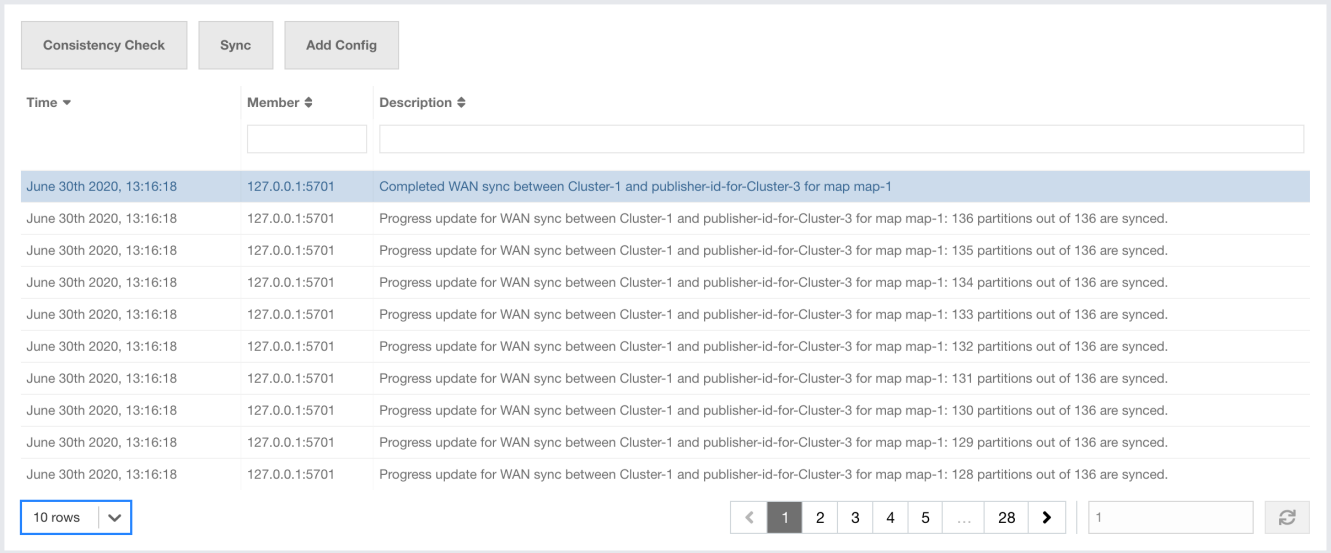


Click **Start** button to open the dialog, enter the target details for the sync operation and click **Sync** to start the operation.



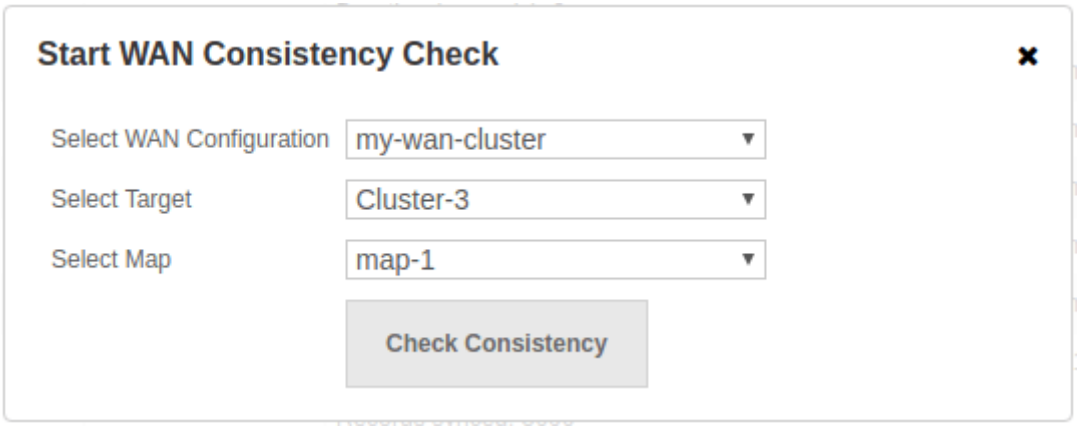
You can also use the "All Maps" option in the above dialog if you want to synchronize all the maps in source and target cluster.

You can see the progress of the operation once you initiate it.

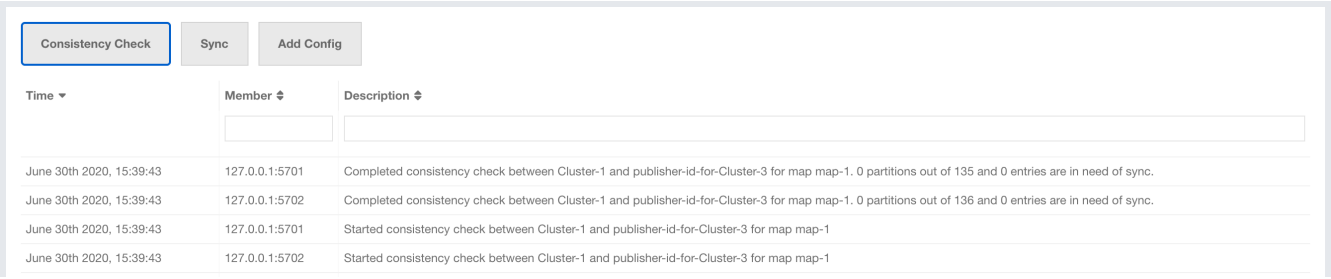


11.7.3. WAN Consistency Check

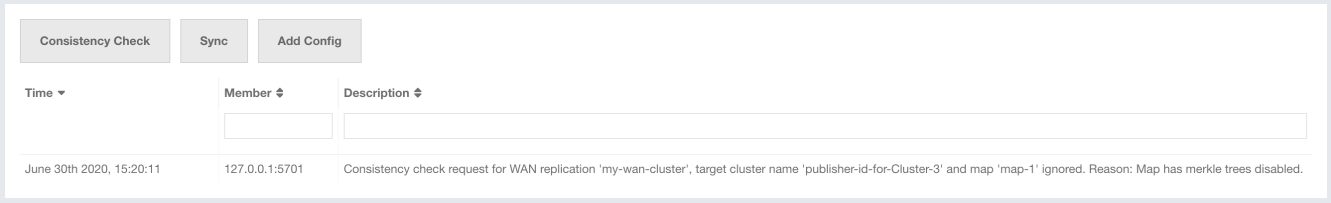
You can check if an IMap is in sync with a specific target cluster. Click **Check** button to open the dialog, enter the target details for the consistency check operation and click **Check Consistency** to start the operation.



You can see the progress of the operation once you initiate it.



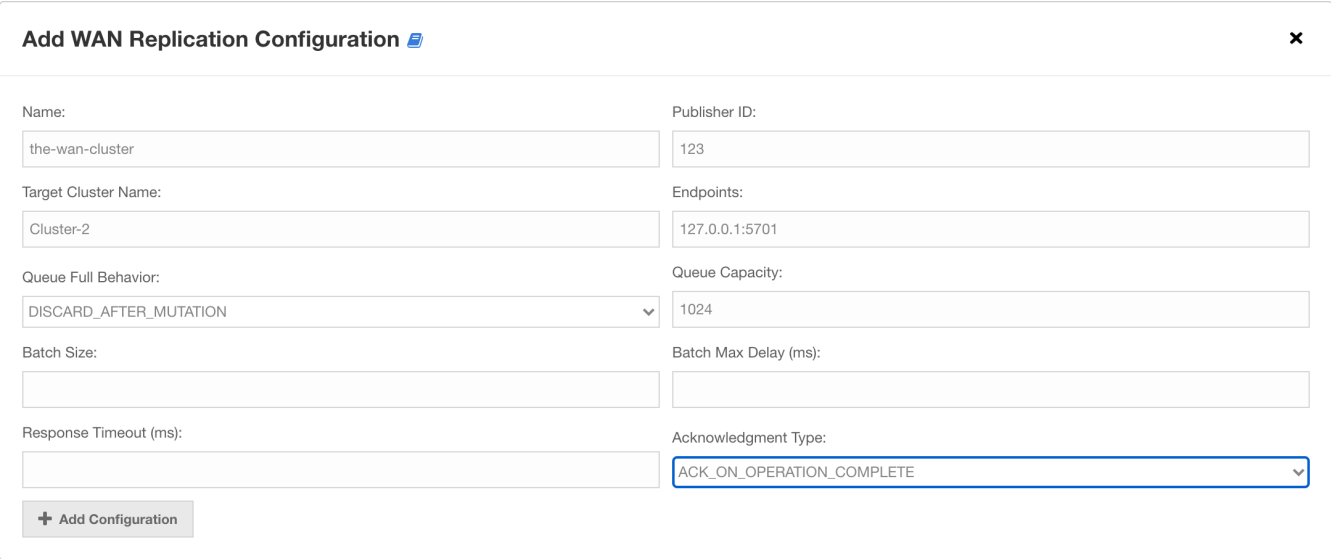
You need to use Merkle trees for WAN synchronization to be able to check for the consistency between two clusters. You need to configure the Merkle trees on both the source and target clusters. If you do not configure it for the source cluster, consistency check is ignored. If it's enabled for the source cluster but not for the target cluster, all entries are reported as if they need a sync because a sync operation will be a full sync in the absence of Merkle trees.



11.7.4. Add Temporary WAN Replication Configuration

You can add a temporary WAN replication configuration dynamically to a cluster. It is useful for having one-off WAN sync operations. The added configuration has two caveats:

- It is not persistent, so it does not survive a member restart.
- It cannot be used as a target for regular WAN replication. It can only be used for WAN sync.

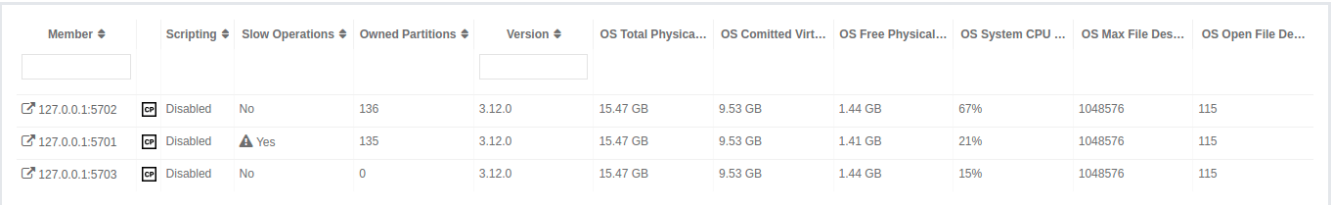


See the WAN Replication section in Hazelcast IMDG Reference Manual for details about the fields and their possible values.

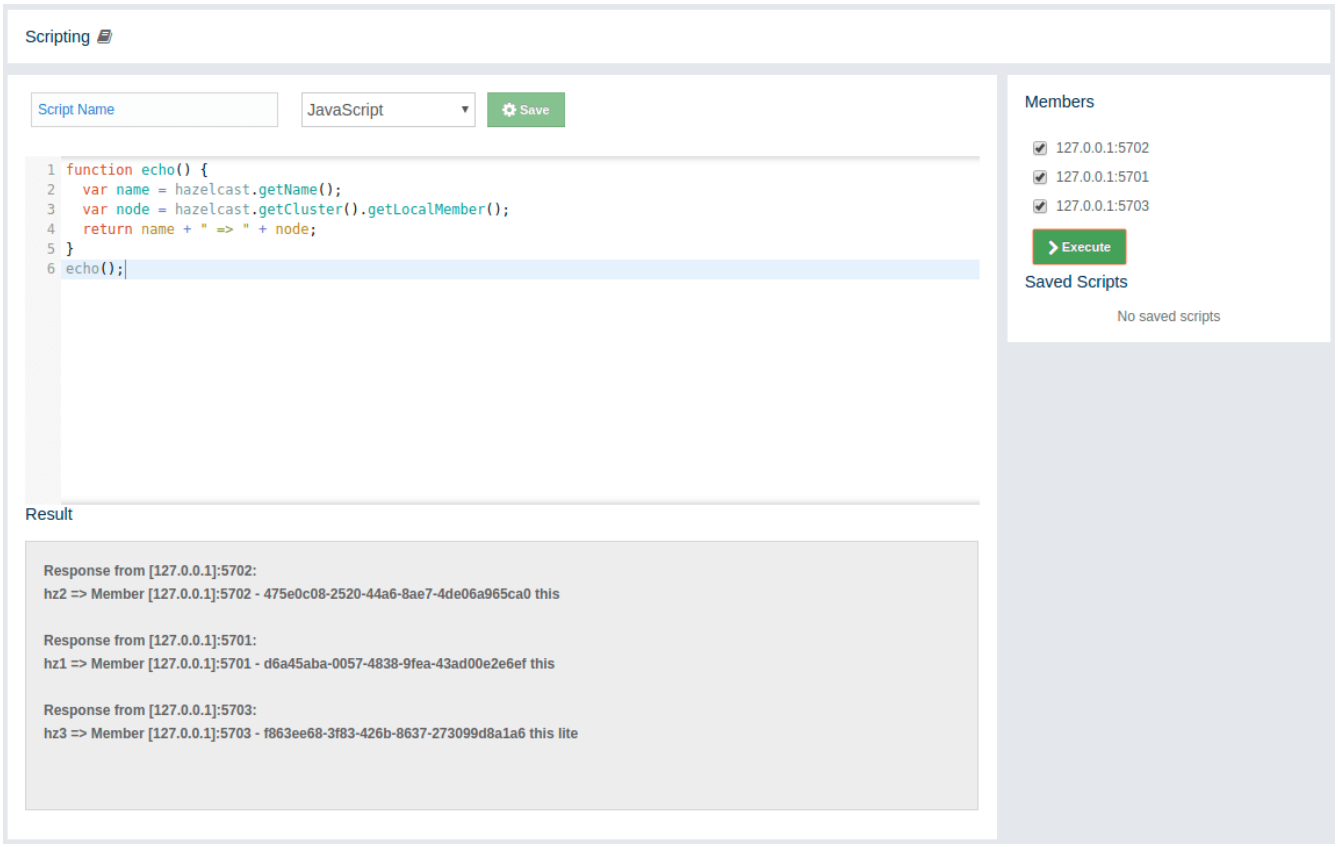
After clicking the **Add Configuration** button, the new WAN replication configuration is added to the cluster. You can see the new configuration when you try to initiate a WAN sync operation as described in the previous section.

11.8. Scripting

You can use the scripting feature of the Management Center to execute scripts on the cluster. Note that it's disabled by default and you need to enable it in your cluster's configuration. Please see the Toggle Scripting Support section in the Hazelcast IMDG Reference Manual for details. Members list shows whether scripting is enabled or disabled for each member.



To use this feature, click on the **Scripting** menu item on the left panel. Once selected, the scripting feature opens as shown below.



In this window, the left panel is the actual script code editor. The combo box located at the top part of the editor enables you to select a scripting language: currently, JavaScript, Ruby, Groovy and Python languages are supported. You can select the members on which the code will execute from the **Members** list shown at the right side of the window. After you write your script and press the **Execute** button, you can see the execution result in the **Result** part of the window.

To use the scripting languages other than JavaScript on a member, the libraries for those languages should be placed in the classpath of that member. See the Enabling Additional Scripting Languages section for more details.

There are **Save** and **Delete** buttons on the top right of the scripting editor. To save your scripts, press the **Save** button after you type a name for your script into the field next to this button. The scripts you saved are listed in the **Saved Scripts** part of the window, located at the bottom right of the page. Click on a saved script from this list to execute or edit it. If you want to remove a script that you wrote and saved before, select it from this list and press the **Delete** button.

In the scripting engine you have a `HazelcastInstance` bonded to a variable named `hazelcast`. You can invoke any method that `HazelcastInstance` has via the `hazelcast` variable. You can see an example usage for JavaScript below:

```
var name = hazelcast.getName();
var node = hazelcast.getCluster().getLocalMember();
var employees = hazelcast.getMap("employees");
employees.put("1","John Doe");
employees.get("1"); // will return "John Doe"
```

11.8.1. Enabling Additional Scripting Languages

IMDG members use the `javax.script.ScriptEngineManager` API for the execution of the script. Only the JavaScript script engine is available in most JVMs by default. To use a scripting language other than JavaScript on a member, you need to add the corresponding scripting engine in the classpath of that member. Here is an example of Maven configuration for several scripting engines:

```
<dependencies>
    <!-- Groovy -->
    <dependency>
        <groupId>org.codehaus.groovy</groupId>
        <artifactId>groovy-jsr223</artifactId>
        <version>2.5.6</version>
    </dependency>
    <!-- Ruby (JRuby) -->
    <dependency>
        <groupId>org.mule.modules</groupId>
        <artifactId>mule-module-scripting-jruby</artifactId>
        <version>3.9.0</version>
    </dependency>
    <dependency>
        <groupId>joda-time</groupId>
        <artifactId>joda-time</artifactId>
        <version>2.10.2</version>
    </dependency>
    <!-- Python (Jython) -->
    <dependency>
        <groupId>org.python</groupId>
        <artifactId>jython-standalone</artifactId>
        <version>2.7.1</version>
    </dependency>
</dependencies>
```

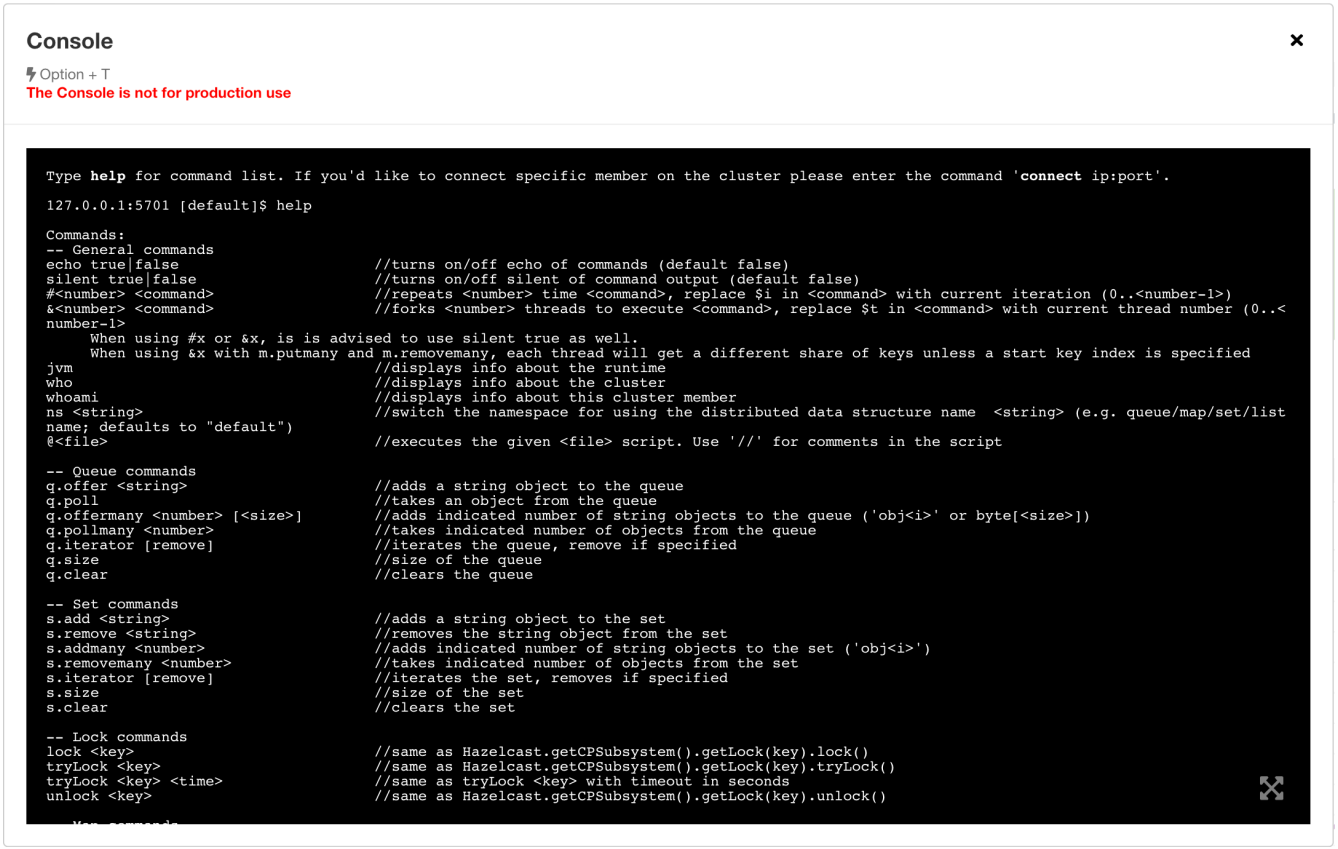
11.9. Executing Console Commands

Disclaimer: It is not recommended for production use.

The Management Center has a console feature that enables you to execute commands on the cluster. For example, you can perform `put`s and `get`s on a map, after you set the namespace with the command `ns <name of your map>`. The same is valid for queues, topics and other data structures that can be monitored on the Management Center. To execute your command, type it into the field

below the console and press **Enter**. Type `help` to see all the commands that you can use.

Open a console window by clicking on the **Console** button located on the left panel or the one located at the top menu. You can also open it by using `ALT + T` keyboard shortcut on Windows/Linux and `Option + T` on macOS. Below is a sample view with the `help` command executed.



The Management Center sends commands to one of the cluster members; for this, it makes an HTTP request to the REST endpoint on that member. As you can see in the above screenshot, the console screen shows the IP address of the member which receives the console commands. Basically, it connects to the port that member listens to, which is configured on the member side as described here. An example configuration on the member side is shown below:

```
<hazelcast>
    ...
    <network>
        <port port-count="20" auto-increment="true">5701</port>
    </network>
    ...
</hazelcast>
```

The direction of this communication is from an ephemeral port number on the Management Center to the port number 5701 (according to the above example) on the member.

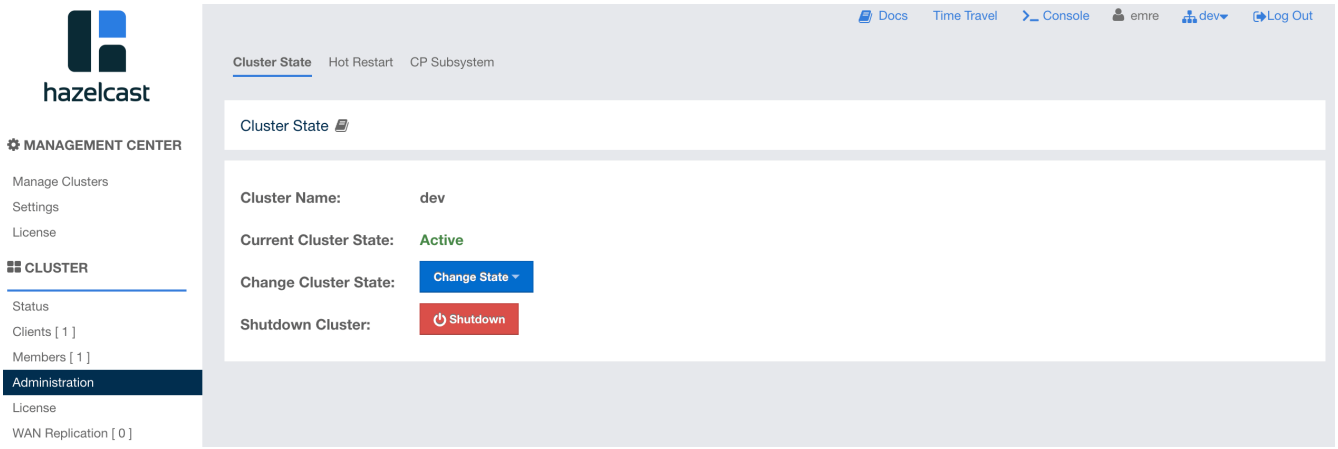
There is no configuration option to specify which outbound ports the Management Center will use; it picks on from the ephemeral port pool. Note that, this can create an issue where a firewall is placed between the cluster and Management Center.

The Management Center Console does not support data structures with spaces or special characters in their names.

11.10. Cluster Administration

Using the "Administration" menu item, you can change the state of your cluster, shut down it, update your Management Center license and perform Rolling Upgrade or Hot Restart on your cluster.

When you click on the "Administration" menu item, the following page shows up:

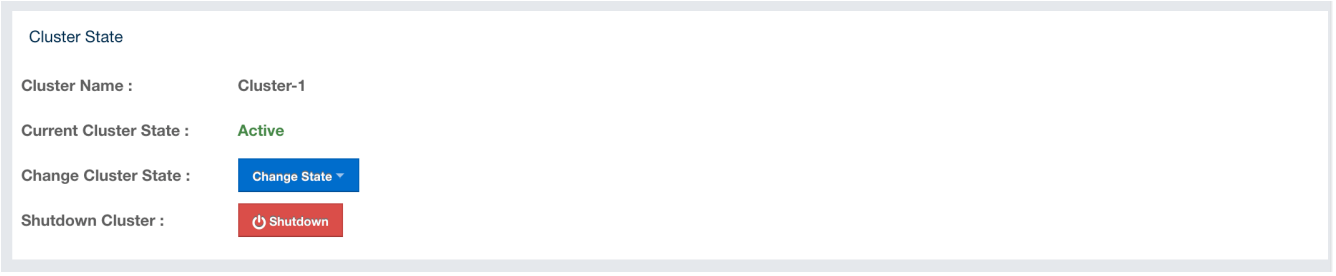


This menu item is available only to admin users.

You can perform the aforementioned administrative tasks using the tabs on this page. Below sections explain each tab.

11.10.1. Cluster State

The admin user can see and change the cluster state and shut down the cluster using the buttons listed in this page as shown below.



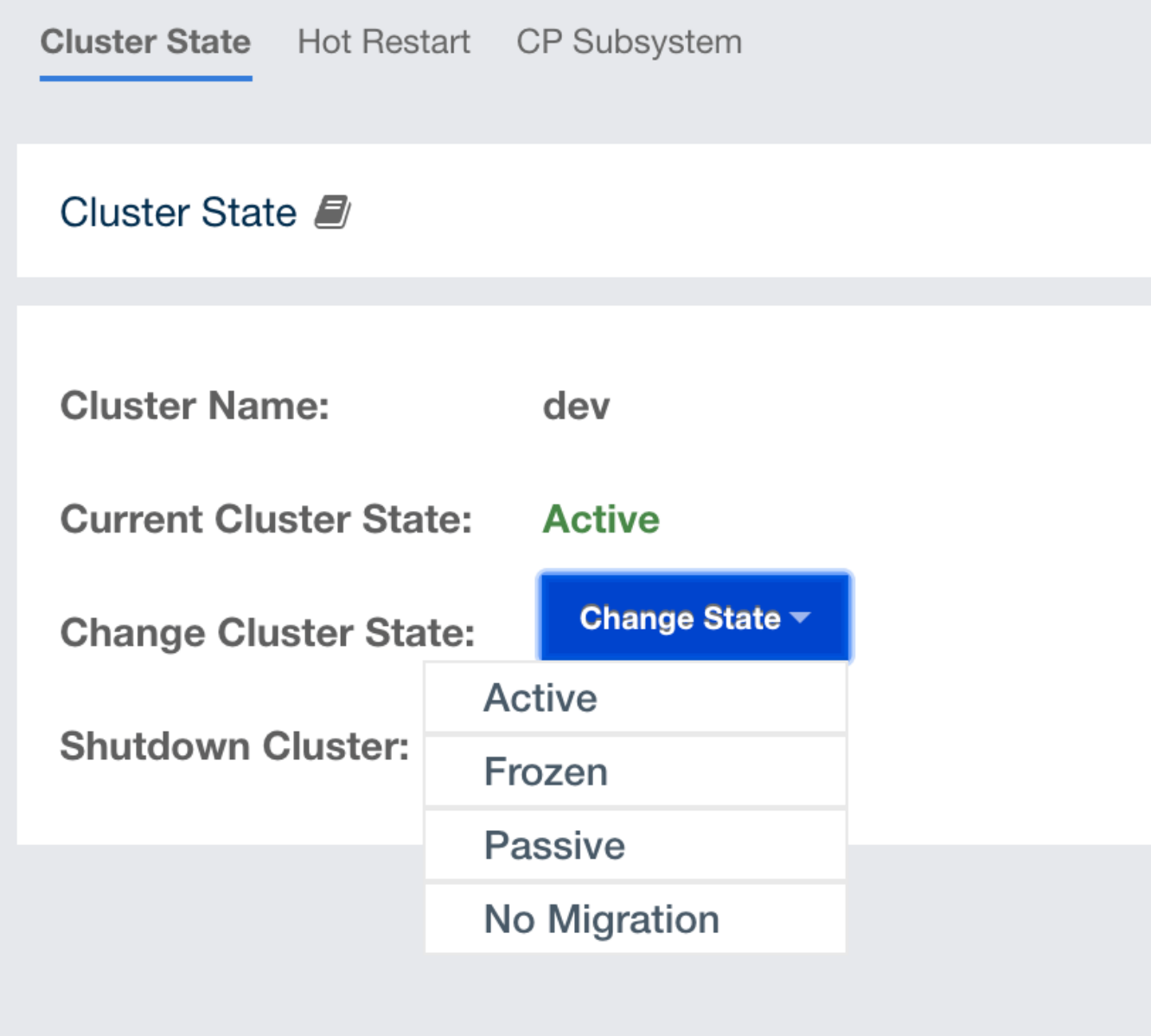
Cluster States:

- **Active**: The cluster continues to operate without any restriction. All operations are allowed. This is the default state of a cluster.
- **No Migration**: Migrations (partition rebalancing) and backup replications are not allowed. The cluster continues to operate without any restriction. All other operations are allowed.
- **Frozen**: New members are not allowed to join, except the members left in **this** or the **Passive**

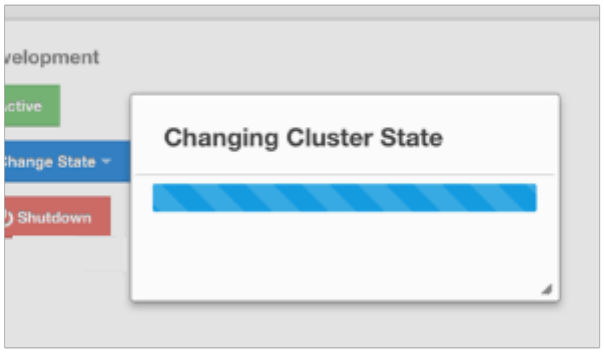
state. All other operations except migrations are allowed and the cluster operates without any restriction.

- **Passive**: New members are not allowed to join, except the members left in **this** or the **Frozen** state. All operations, except the ones marked with `AllowedDuringPassiveState`, are rejected immediately.
- **In Transition**: Shows that the cluster state is in transition. This is a temporary and intermediate state. It is not allowed to set it explicitly.

Changing the Cluster State

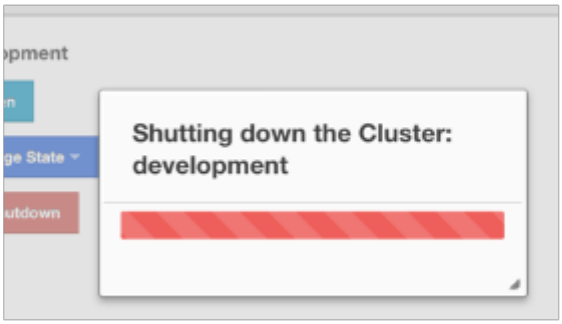


- Click the dropdown menu and choose the state to which you want your cluster to change. A pop-up appears and stays on the screen until the state is successfully changed.



Shutting Down the Cluster

- Click the **Shutdown** button. A pop-up appears and stays on the screen until the cluster is successfully shutdown.

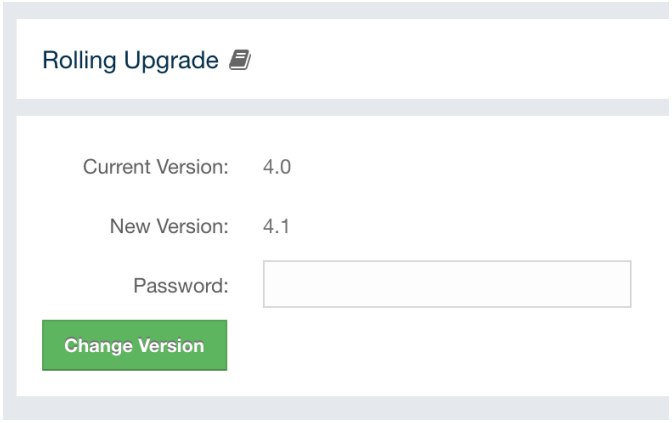


If an exception occurs during the state change or shutdown operation on the cluster, this exception message is shown on the screen as a notification.

11.10.2. Rolling Upgrade

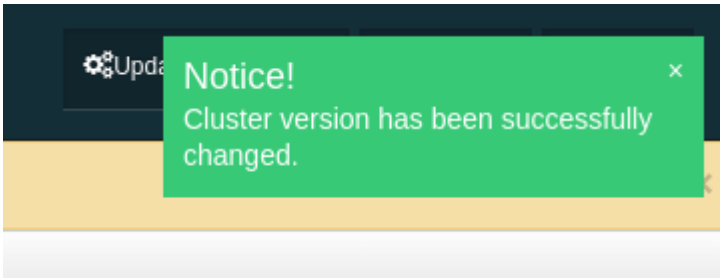
The admin user can upgrade the cluster version once all members of the cluster have been upgraded to the intended codebase version as described in the Rolling Upgrade Procedure section of the Hazelcast IMDG Reference Manual.

Open the **Rolling Upgrade** tab to perform a Rolling Upgrade and change the cluster's version.



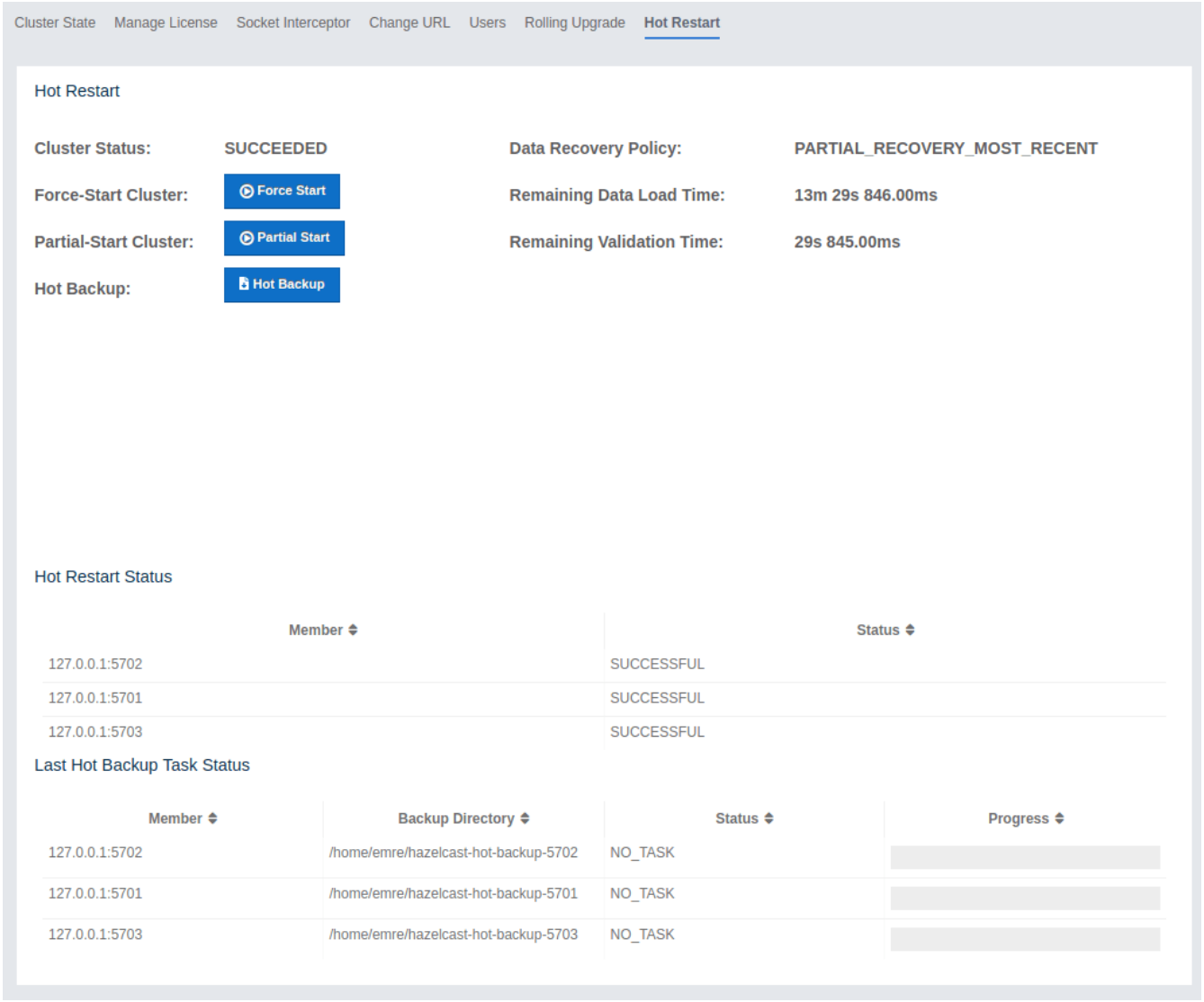
Enter the password of the cluster if security is enabled on the cluster (leave it empty if not), and click on the **Change Version** button.

Once the operation succeeds, you will see the following notification:



11.10.3. Hot Restart

Using the Hot Restart tab, you can perform force and partial start of the cluster and see the Hot Restart status of the cluster members. You can also take snapshots of the Hot Restart Store (Hot Backup). When you click on this tab, the following page is shown:



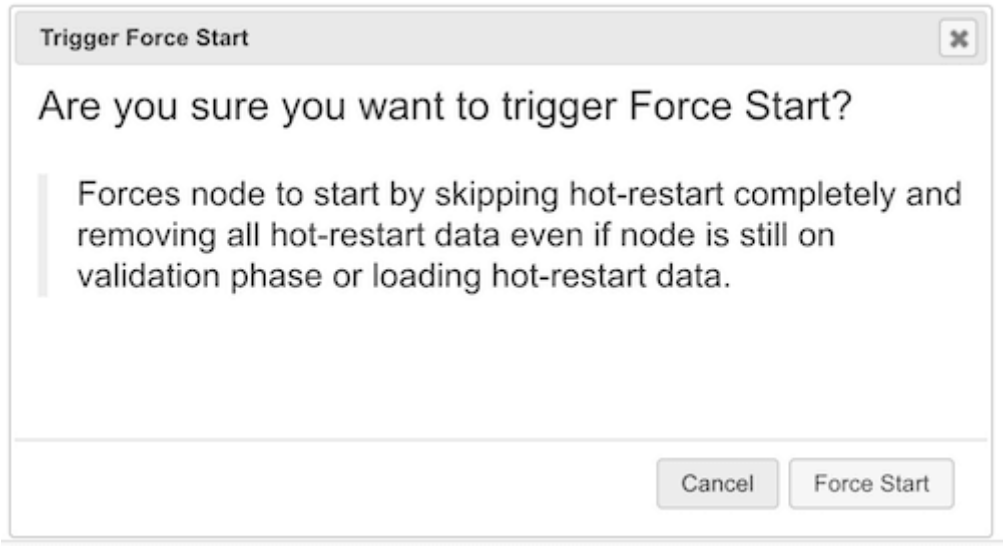
Below sections explain each operation.

Force Start

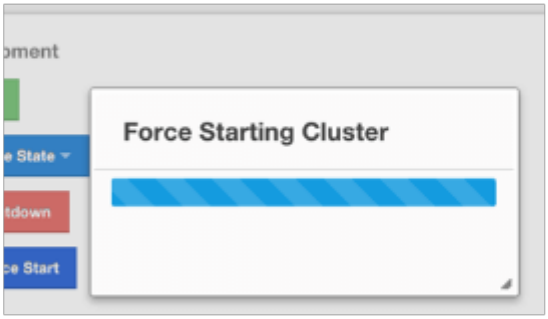
Restart process cannot be completed if a member crashes permanently and cannot recover from the failure since it cannot start or it fails to load its own data. In that case, you can force the cluster to clean its persisted data and make a fresh start. This process is called **force start**.

See the Force Start section in the Hazelcast IMDG Reference Manual for more information on this operation.

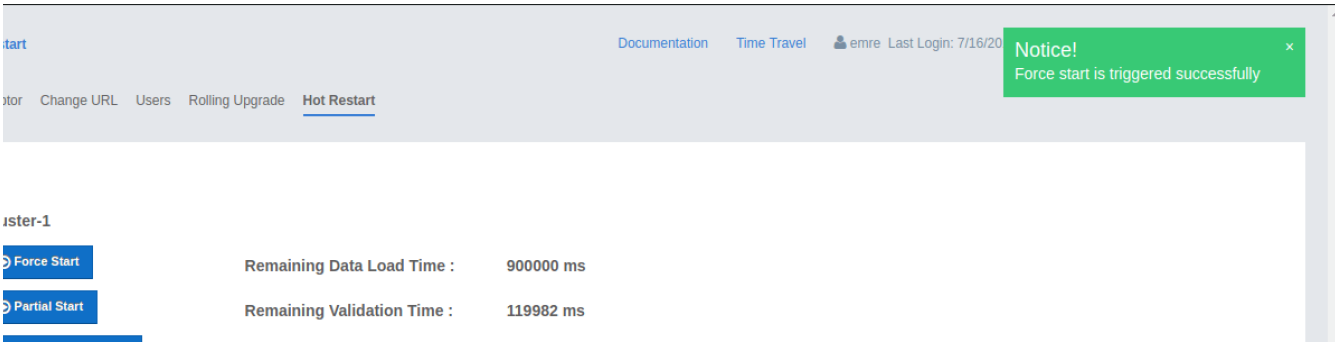
To perform a force start on the cluster, click on the **Force Start** button. A confirmation dialog appears as shown below.



Once you click on the **Force Start** button on this dialog, the cluster starts the force start process and the following progress dialog shows up while doing so.



This dialog stays on the screen until the operation is triggered. Once it is done, the success of force start operation is shown as a notice dialog, as shown below.



If an exception occurs, this exception message is shown on the screen as a notification.

Partial Start

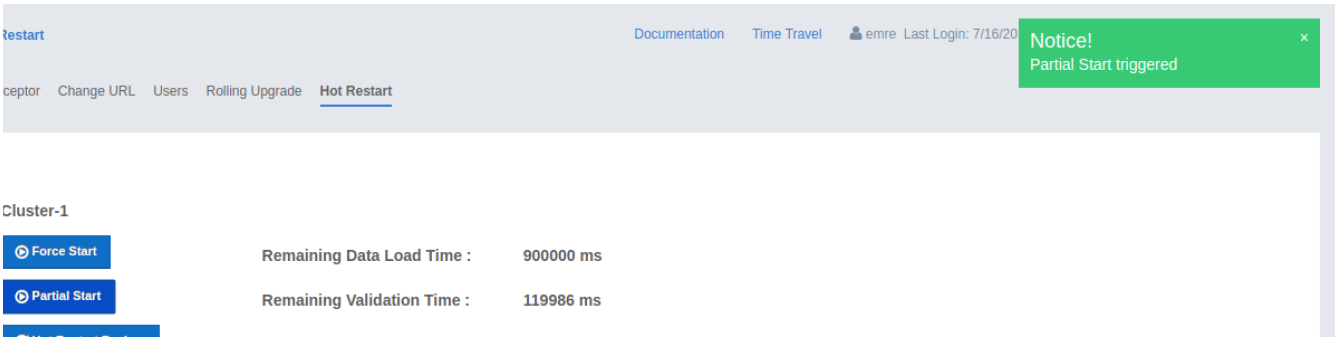
When one or more members fail to start or have incorrect Hot Restart data (stale or corrupted data) or fail to load their Hot Restart data, the cluster becomes incomplete and the restart mechanism cannot proceed. One solution is to use Force Start and make a fresh start with the existing

members, as explained above. Another solution is to do a partial start.

Partial start means that the cluster will start with an incomplete set of members. Data belonging to the missing members is assumed lost and the Management Center tries to recover the missing data using the restored backups. For example, if you have minimum two backups configured for all the maps and caches, then a partial start up to two missing members is safe against data loss. If there are more than two missing members or there are maps/caches with less than two backups, then data loss is expected.

See the Partial Start section in the Hazelcast IMDG Reference Manual for more information on this operation and how to enable it.

To perform a partial start on the cluster, click on the **Partial Start** button. A notice dialog appears as shown below.



You can also see two fields related to Partial Start operation: "Remaining Data Load Time" and "Remaining Validation Time", as shown in the above screenshot.

- **Remaining Validation Time**: When partial start is enabled, Hazelcast can perform a partial start automatically or manually, in case of some members are unable to restart successfully. Partial start proceeds automatically when some members fail to start and join to the cluster in `validation-timeout-seconds`, which you can configure. After this duration is passed, Hot Restart chooses to perform a partial start with the members present in the cluster. This field, i.e., "Remaining Validation Time" shows how much time is left to the automatic partial start, in seconds. You can always request a manual partial start, by clicking on the **Partial Start** button, before this duration passes.
- **Remaining Data Load Time**: The other situation to decide to perform a partial start is the failures during the data loading phase. When Hazelcast learns the data loading result of all members which have passed the validation step, it automatically performs a partial start with the ones which have successfully restored their Hot Restart data. Note that partial start does not expect every member to succeed in the data loading step. It completes the process when it learns the data loading result for every member and there is at least one member which has successfully restored its Hot Restart data. Relatedly, if it cannot learn the data loading result of all members before `data-load-timeout-seconds` duration, it proceeds with the ones which have already completed the data loading process. This field, i.e., "Remaining Data Load Time" shows how much time (in seconds) is left for Hazelcast to know at least one member has successfully restored its Hot Restart data and perform an automatic partial start.

See the Configuring Hot Restart section in the Hazelcast IMDG Reference Manual for more information on the configuration elements `validation-timeout-seconds` and `data-load-timeout-seconds` mentioned above and how to configure them.

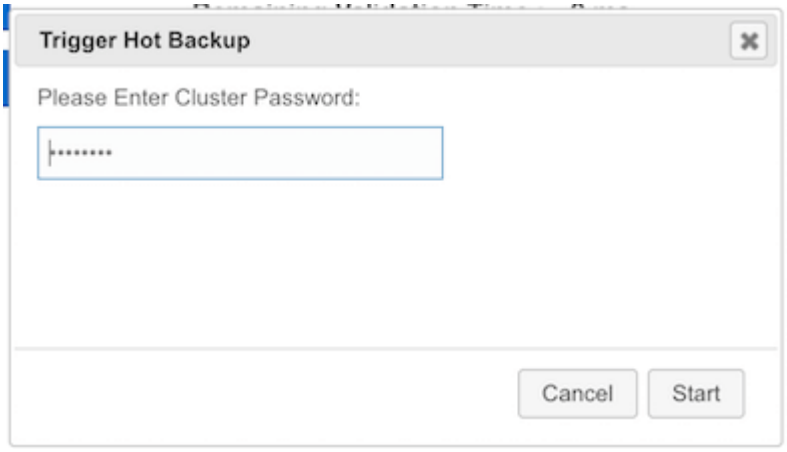
Force and partial start operations can also be performed using the REST API and the script `cluster.sh`. See the Using REST API for Cluster Management section and Using the Script cluster.sh section in the Hazelcast IMDG Reference Manual.

Hot Backup

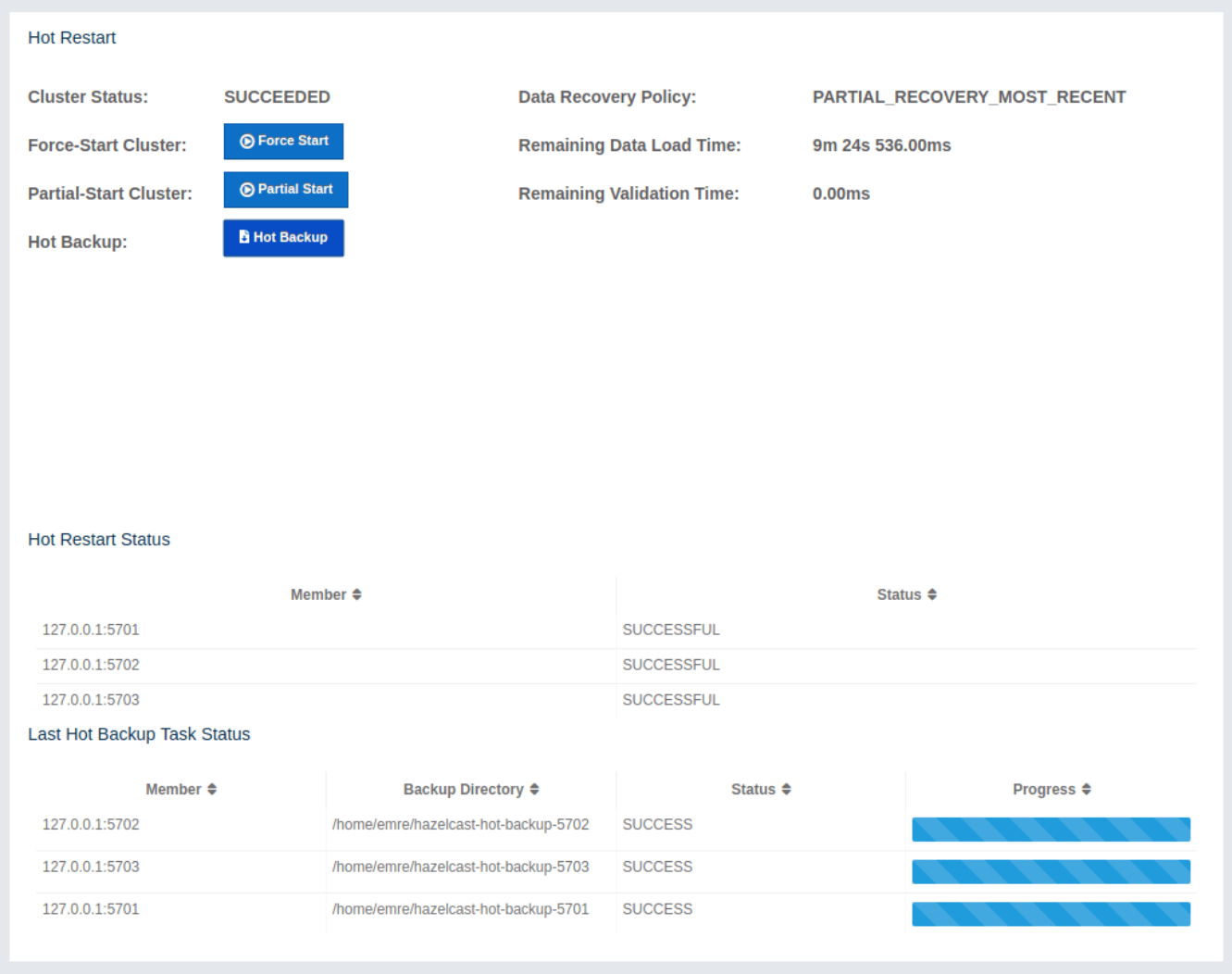
During Hot Restart operations, you can take a snapshot of the Hot Restart data at a certain point in time. This is useful when you wish to bring up a new cluster with the same data or parts of the data. The new cluster can then be used to share load with the original cluster, to perform testing, quality assurance or reproduce an issue on the production data.

Note that you must first configure the Hot Backup directory programmatically (using the method `setBackupDir()`) or declaratively (using the element `backup-dir`) to be able to take a backup of the Hot Restart data. See the Configuring Hot Backup section in the Hazelcast IMDG Reference Manual.

If the backup directory is configured, you can start to perform the backup by clicking on the **Hot Backup** button. The Management Center first asks the cluster password as shown in the following dialog.

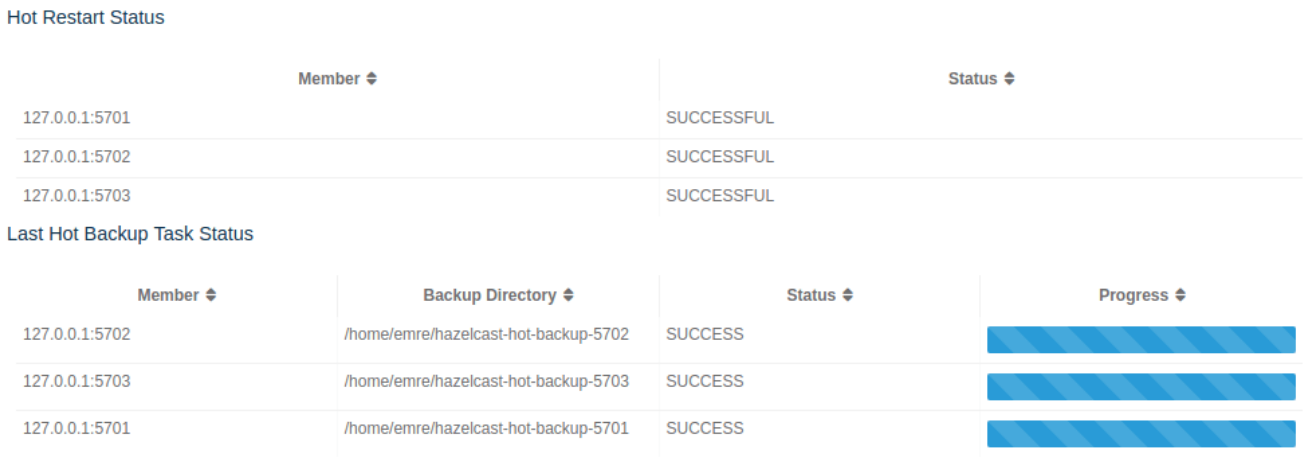


Once you entered the password correctly and click on the "Start" button on this dialog, you will see a notification dialog stating that the backup process starts. You can see the progress of the backup operation under the "Last Hot Backup Task Status" part of the page, as shown below.



Status Information

At the bottom of "Hot Restart" tab, you can see the Hot Restart and Hot Backup statuses of cluster members, as shown below.



You can see the status and progress of your Hot Backup operation under "Last Hot Backup Task Status". It can be IN_PROGRESS and SUCCESS/FAILURE according to the result of the operation.

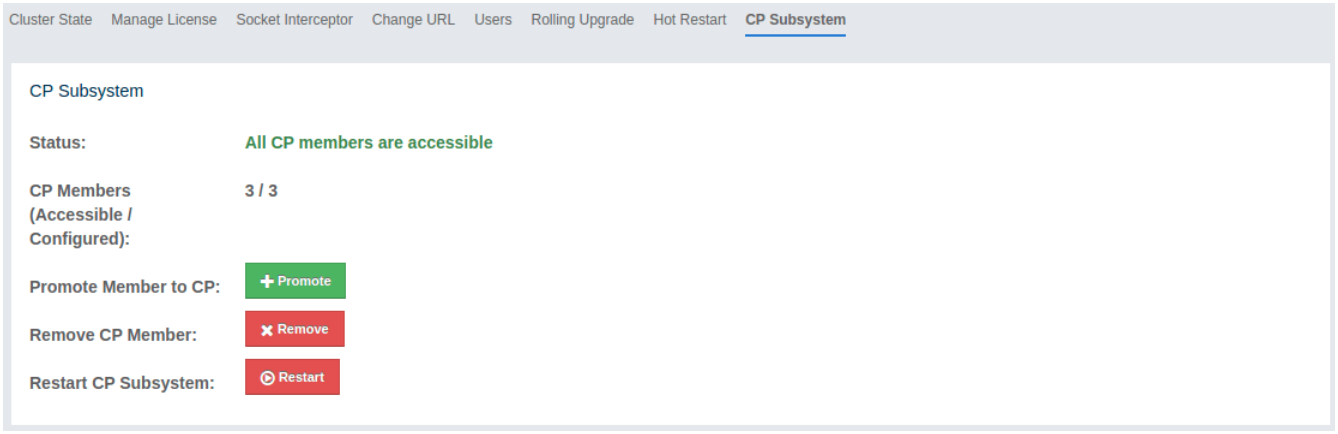
You can also see the status of Hot Restart operation of your cluster members, under "Hot Restart Status". It can be PENDING and SUCCESSFUL/FAILED according to the result of Hot Restart

operation.

11.10.4. CP Subsystem

CP subsystem management operations require enabled REST API in the IMDG cluster. See the IMDG documentation for more information.

The **CP Subsystem** tab can be used to monitor overall status of the CP subsystem in the current cluster and perform certain management operations.



Monitoring CP Subsystem

The **Status** field shows a summary of the current CP subsystem status. It may have one of the following values:

- **CP Subsystem is not supported by this cluster:** Shown for IMDG clusters with version prior to 3.12.
- **CP Subsystem is not enabled:** Shown if CP subsystem is not enabled for the current cluster.
- **All CP members are accessible:** Shown if there are at least the same amount of accessible CP members as the configured CP member count.
- **CP Subsystem warning: one CP member is not accessible:** Shown if there is one missing CP member and the minority count in the CP subsystem is greater than 1. For example, this value is shown when there are 6 accessible CP members and the configured count is 7. In this example, the minority count is 3 members and the majority count is 4 members.
- **CP Subsystem alert: multiple CP members are not accessible:** Shown if there are multiple missing CP members, but their count is less than the minority.
- **CP Subsystem error: minority of the CP members are not accessible:** Shown if the minority of CP members are missing.
- **CP Subsystem error: majority of the CP members are not accessible:** Shown if the majority of CP members are missing.

The **CP Members (Accessible/Configured)** field shows the current count of accessible CP members and the configured CP members count.

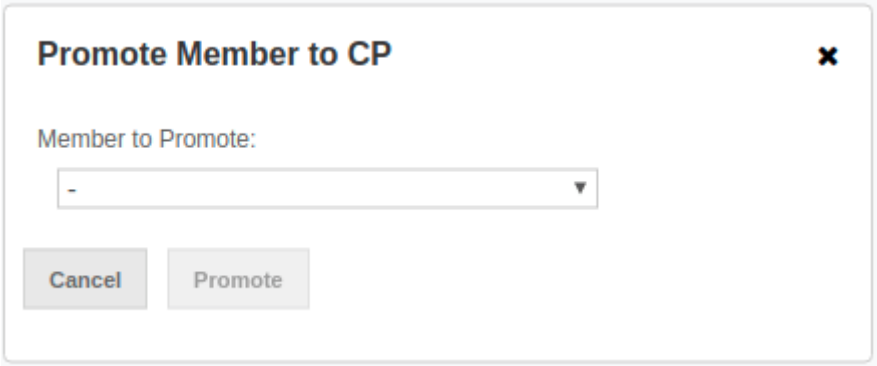
You may promote additional members or remove inaccessible CP members, so the total count of members that participate in the CP subsystem may be greater or less than the configured CP member count. As the Status field considers the configured CP member count as the total CP member count, it should be treated only as a basic health indicator for the CP subsystem.

Managing CP Subsystem

You can also use the CP Subsystem tab to start the following management operations.

Promote Member to CP Subsystem

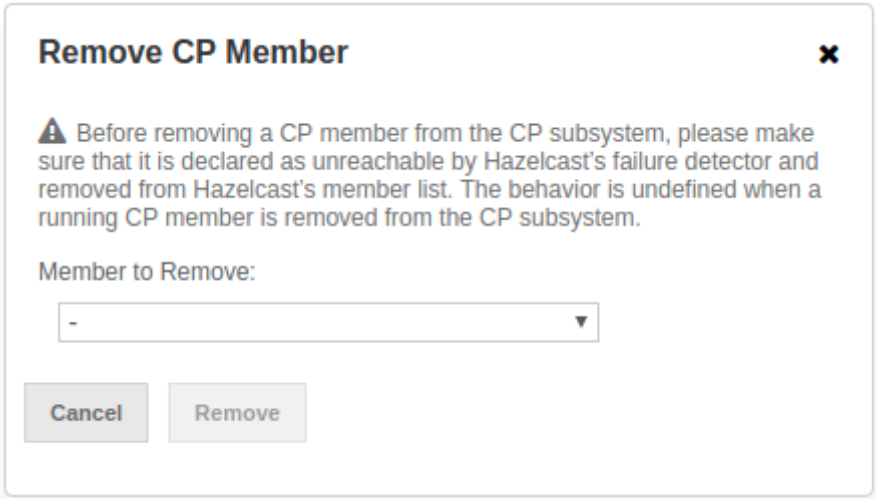
To promote one of the AP members to become a CP member, click on the **Promote** button. A confirmation dialog appears as shown below.



It asks you to choose one of AP members, i.e., one of the members that do not participate in the CP subsystem. Note that lite members are not shown in the dropdown list as lite members do not store data. Once you press the **Promote** button, the CP subsystem starts the promote operation for the given member.

Remove CP Member

To remove one of the inaccessible CP members from the CP subsystem, click on the **Remove** button. A confirmation dialog appears as shown below.

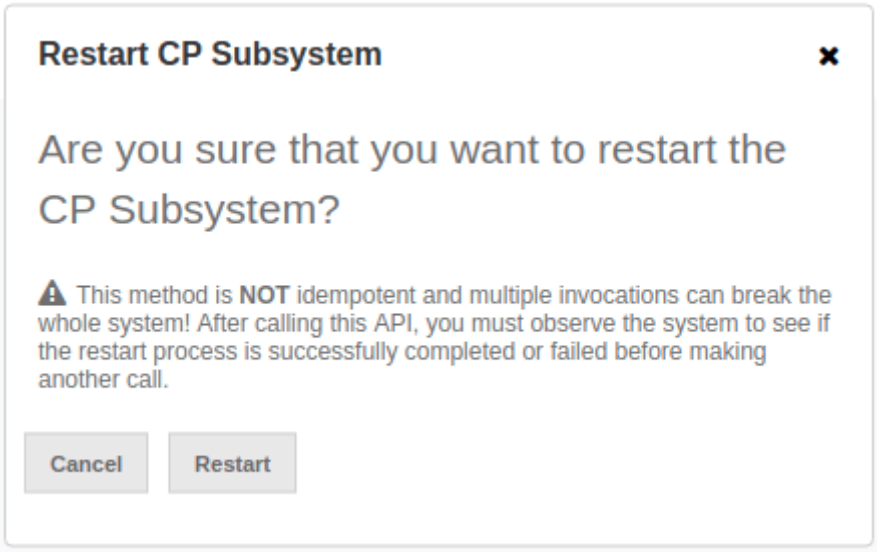


It asks you to choose one of the members that is not connected to the Management Center, but is known by the cluster's CP subsystem. Once you press the **Remove** button, the CP subsystem starts

the remote operation for the given member.

Restart CP Subsystem

To wipe and restart the whole CP subsystem of the cluster, click on the **Restart** button. A confirmation dialog appears as shown below.



Once you press the **Restart** button, CP subsystem proceeds with the restart operation.

The CP subsystem restart operation is **NOT** idempotent and multiple invocations can break the whole system! After using this dialog, you must observe the system to see if the restart process is successfully completed or failed before starting this operation again.

12. Metrics Persistence

Your clusters collect and report metrics data for the connected Management Center. Metrics data includes various number of time series, such as CPU load, memory consumption, and operation counters. See the Metrics section of the Hazelcast IMDG Reference Manual for details about configuring metrics collection.

By default, Management Center persists the collected metrics on disk. You can turn off the switch with the "Metrics Persistence" label to disable on disk persistence for collected metrics data and have them stored only in memory.

Once it is **ON**, the metrics generated by your clusters will be stored on disk. You can configure the Time-to-Live setting to control how long metrics data points are kept on disk. By default, Management Center stores metrics data for one day.

Management Center stores the metrics in a database file, under the Management Center home directory on the disk. The data files can be found in the `<User's Home Directory>/hazelcast-mc/metrics` directory, e.g., `/home/someuser/hazelcast-mc/metrics`. This directory can be changed using the `hazelcast.mc.home` property on the server where Management Center is running.

When on-disk persistence is enabled, you may see the following warning messages in the Management Center log:

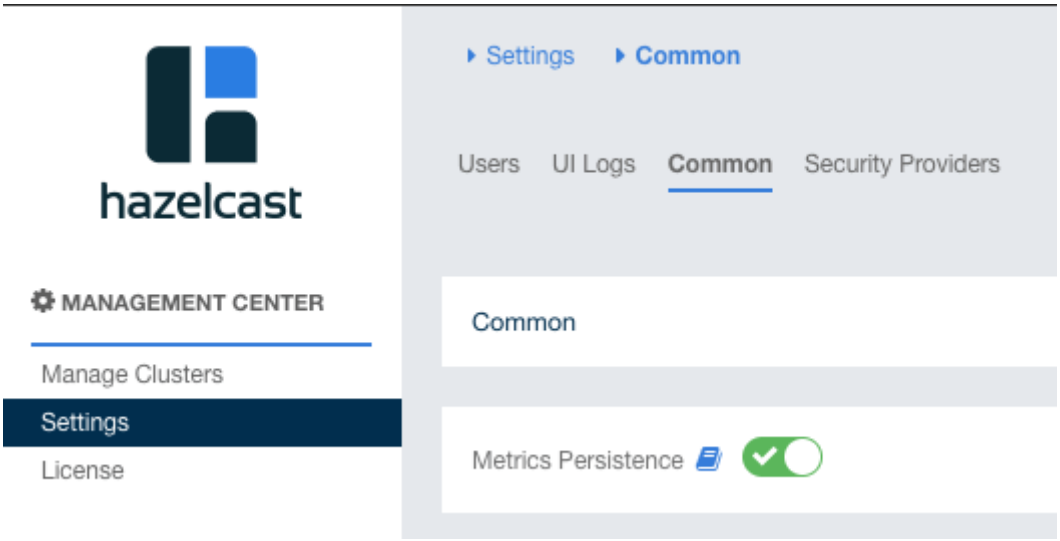
```
Could not find 10230 metrics time series out of 100500 during background persistence
run.
Consider increasing metrics in-memory cache size by setting `hazelcast.mc.metrics
.inMemory.max.size` system property.
```

This message indicates that Management Center could not persist some of the collected data points. That happens when the in-memory cache size is not enough to fit all metrics generated by your clusters and its size should be increased. You can increase it by using the `hazelcast.mc.metrics.inMemory.max.size` system property. This value is `300000` by default, which is enough to store up to 3 minutes of data for 100,000 metrics.

Each in-memory cache entry takes around 0.5 KB of memory, so you may have to increase the maximum heap size in JVM when changing this setting to a greater value.

12.1. Using Metrics Persistence

The **Metrics Persistence** toggle item allows you to check the status of the cluster at a time in the past. You can select it on the Settings:



When you turn the switch on, you can go back in time using the date picker on the left corners of charts and check your cluster's situation at the selected time. All the data structures and members can be monitored as if you are using the Management Center normally (charts and data tables for each data structure and members). It shows the status if metrics persistence has been **ON** at the selected time in the past; otherwise, it shows empty charts and tables.

You can press the Now button next to the date picker to see the latest data. Note that this will only show you the latest data on a chart and not cause the other charts and data tables to refresh.

12.2. Clustered REST

Hazelcast IMDG Enterprise

The Clustered REST API is exposed from the Management Center to allow you to monitor clustered statistics of distributed objects.

12.2.1. Enabling Clustered REST

To enable Clustered REST on your Management Center, pass the following system property at startup. This property is disabled by default.

```
-Dhazelcast.mc.rest.enabled=true
```

12.2.2. Clustered REST API Root

The entry point for the Clustered REST API is `/rest/`. This resource does not have any attributes.

All parameters that are used in the REST API URLs, like cluster names and distributed data structure names, must be URL encoded when composing a valid request for Clustered REST. Such parameters are marked in braces ({ and }) in the URL description for each endpoint. As an example, `name.with/special@chars` parameter value would be encoded as `name.with%2Fspecial%40chars`.

All endpoints return HTTP status code 404 if no data about a cluster, member, client or data structure can be found in the Management Center.

Retrieve Management Center License Expiration Time

This endpoint returns the expiration time in milliseconds (since epoch) of the license key assigned for the Management Center. Returns `-1` if no license is assigned.

- **Request Type:** GET
- **URL:** `/rest/license`
- **Request:**

```
curl http://localhost:8080/rest/license
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "licenseExpirationTime": 4099755599515
}
```

12.2.3. Clusters Resource

This resource returns a list of clusters that are connected to the Management Center.

Retrieve Clusters

- **Request Type:** GET
- **URL:** `/rest/clusters`
- **Request:**

```
curl http://localhost:8080/rest/clusters
```

- **Response:** `200` (application/json)
- **Body:**

```
["dev", "qa"]
```

12.2.4. Cluster Resource

This resource returns information related to the provided cluster name.

Retrieve Cluster Information

This endpoint returns address of the oldest cluster member, the expiration time in milliseconds (since epoch) of the license key assigned for the cluster and the cluster type (`IMDG` or `JET`). Returns `-1` for license expiration time if no license is assigned.

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "masterAddress": "192.168.2.78:5701",
  "licenseExpirationTime": 4099755599515,
  "clusterType": "IMDG"
}
```

12.2.5. Members Resource

This resource returns a list of the members belonging to the provided clusters.

Retrieve Members [GET] [/rest/clusters/{clustername}/members]

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/members
```

- **Response:** `200` (application/json)
- **Body:**

```
[
  "192.168.2.78:5701",
  "192.168.2.78:5702",
  "192.168.2.78:5703",
  "192.168.2.78:5704"
]
```

12.2.6. Member Resource

This resource returns information related to the provided member.

Retrieve Member Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "address": "192.168.2.78:5701",
  "uuid": "11adba52-e19d-4407-a9e9-e0a271cef14a",
  "cpMemberUuid": "f5a8f8a4-f278-4a13-a23e-5accf5b02f42",
  "maxHeapMemory": 129957888,
  "ownedPartitionCount": 68,
  "usedHeapMemory": 60688784,
  "freeHeapMemory": 24311408,
  "committedHeapMemory": 85000192,
  "connectedClientCount": 1,
  "master": true
}
```


Retrieve Connection Manager Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/connectionManager`
- **Request:**

```
curl
http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701/connectionManager
```

- **Response:** `200` (application/json)
- **Body:**


```
{
  "clientConnectionCount": 2,
  "activeConnectionCount": 5,
  "connectionCount": 5
}
```


Retrieve Operation Service Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/operationService`
- **Request:**

```
curl
http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701/operationService
```

- **Response:** `200` (application/json)

- **Body:**

```
{
  "responseQueueSize": 0,
  "operationExecutorQueueSize": 0,
  "runningOperationsCount": 0,
  "remoteOperationCount": 1,
  "executedOperationCount": 461139,
  "operationThreadCount": 8
}
```

Retrieve Event Service Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/eventService`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701/eventService
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "eventThreadCount": 5,
  "eventQueueCapacity": 1000000,
  "eventQueueSize": 0
}
```

Retrieve Partition Service Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/partitionService`
- **Request:**

```
curl
http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701/partitionService
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "partitionCount": 271,
  "activePartitionCount": 68
}
```

Retrieve Proxy Service Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/proxyService`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701/proxyService
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "proxyCount": 8,
  "createdCount": 13,
  "destroyedCount": 5
}
```

Retrieve All Managed Executors

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/managedExecutors`
- **Request:**

```
curl
http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701/managedExecutors
```

- **Response:** `200` (application/json)
- **Body:**

```
["hz:system", "hz:async", "hz:scheduled", "hz:client", "hz:client-query",
"hz:client-blocking-tasks",
"hz:query", "hz:io", "hz:offloadable", "hz:map-load", "hz:map-loadAllKeys", "hz:mc
"]
```

Retrieve a Managed Executor

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/members/{member}/managedExecutors/{managedExecutor}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/members/192.168.2.78:5701
      /managedExecutors/hz:system
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "name": "hz:system",
  "queueSize": 0,
  "poolSize": 0,
  "remainingQueueCapacity": 2147483647,
  "maximumPoolSize": 4,
  "completedTaskCount": 12
}
```

12.2.7. Client Endpoints Resource

This resource returns a list of the client endpoints belonging to the provided cluster. Consider using the newly added Client Statistics Resource as it contains more detailed information about the clients.

Retrieve List of Client Endpoints

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/clients`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/clients
```

- **Response:** `200` (application/json)
- **Body:**

```
["192.168.2.78:61708"]
```

Retrieve Client Endpoint Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/clients/{client}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/clients/192.168.2.78:61708
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "uuid": "6fae7af6-7a7c-4fa5-b165-cde24cf070f5",
  "address": "192.168.2.78:61708",
  "clientType": "JAVA",
  "name": "hz.client_1",
  "labels": [
    "label1"
  ],
  "ipAddress": "192.168.2.78",
  "canonicalHostName": "localhost"
}
```

12.2.8. Maps Resource

This resource returns a list of maps belonging to the provided cluster.

Retrieve List of Maps

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/maps`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/maps
```

- **Response:** `200` (application/json)
- **Body:**

```
["customers", "orders"]
```

Retrieve Map Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/maps/{mapName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/maps/customers
```

- **Response:** `200` (application/json)
- **Body:**


```
{
  "cluster": "dev",
  "name": "customers",
  "ownedEntryCount": 5085,
  "backupEntryCount": 5076,
  "ownedEntryMemoryCost": 833940,
  "backupEntryMemoryCost": 832464,
  "heapCost": 1666668,
  "lockedEntryCount": 2,
  "dirtyEntryCount": 0,
  "hits": 602,
  "lastAccessTime": 1532689094579,
  "lastUpdateTime": 1532689094576,
  "creationTime": 1532688789256,
  "putOperationCount": 5229,
  "getOperationCount": 2162,
  "removeOperationCount": 150,
  "otherOperationCount": 3687,
  "events": 10661,
  "maxPutLatency": 48,
  "maxGetLatency": 35,
  "maxRemoveLatency": 18034,
  "totalPutLatency": 1715433,
  "totalGetLatency": 945421,
  "totalRemoveLatency": 66558323
}
```


12.2.9. MultiMaps Resource

This resource returns a list of multimaps belonging to the provided cluster.

Retrieve List of MultiMaps

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/multimaps`

- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/multimaps
```

- **Response:** `200` (application/json)
- **Body:**

```
["customerAddresses"]
```

Retrieve MultiMap Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/multimaps/{multimapname}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/multimaps/customerAddresses
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "customerAddresses",
  "ownedEntryCount": 4862,
  "backupEntryCount": 4860,
  "ownedEntryMemoryCost": 0,
  "backupEntryMemoryCost": 0,
  "heapCost": 0,
  "lockedEntryCount": 1,
  "dirtyEntryCount": 0,
  "hits": 22,
  "lastAccessTime": 1532689253314,
  "lastUpdateTime": 1532689252591,
  "creationTime": 1532688790593,
  "putOperationCount": 5125,
  "getOperationCount": 931,
  "removeOperationCount": 216,
  "otherOperationCount": 373570,
  "events": 0,
  "maxPutLatency": 8,
  "maxGetLatency": 1,
  "maxRemoveLatency": 18001,
  "totalPutLatency": 487805,
  "totalGetLatency": 14931,
  "totalRemoveLatency": 16388472
}
```


12.2.10. ReplicatedMaps Resource

This resource returns a list of replicated maps belonging to the provided cluster.

Retrieve List of ReplicatedMaps

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/replicatedmaps`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/replicatedmaps
```

- **Response:** `200` (application/json)
- **Body:**


```
["replicated-map-1"]
```

Retrieve ReplicatedMap Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/replicatedmaps/{replicatedmapname}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/replicatedmaps/replicated-map-1
```

- **Response:** `200` (application/json)
- **Body:**


```
{
  "cluster": "dev",
  "name": "replicated-map-1",
  "ownedEntryCount": 10955,
  "ownedEntryMemoryCost": 394380,
  "hits": 15,
  "lastAccessTime": 1532689312581,
  "lastUpdateTime": 1532689312581,
  "creationTime": 1532688789493,
  "putOperationCount": 11561,
  "getOperationCount": 1051,
  "removeOperationCount": 522,
  "otherOperationCount": 355552,
  "events": 6024,
  "maxPutLatency": 1,
  "maxGetLatency": 1,
  "maxRemoveLatency": 1,
  "totalPutLatency": 64,
  "totalGetLatency": 12,
  "totalRemoveLatency": 11
}
```


12.2.11. Caches Resource

This resource returns a list of caches belonging to the provided cluster.

Retrieve List of Caches

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/caches`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/caches
```

- **Response:** `200` (application/json)
- **Body:**

```
["cache-1"]
```

Retrieve Cache Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/caches/{cacheName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/caches/cache-1
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "cache-1",
  "creationTime": 1532688789493,
  "hits": 100,
  "misses": 11,
  "getOperationCount": 231,
  "putOperationCount": 42,
  "removeOperationCount": 1,
  "evictions": 3,
  "averageGetTime": 10.5,
  "averagePutTime": 21.12,
  "averageRemoveTime": 23,
  "lastAccessTime": 1403602693411,
  "lastUpdateTime": 1403602693411,
  "ownedEntryCount": 300
}
```

12.2.12. Queues Resource

This resource returns a list of queues belonging to the provided cluster.

Retrieve List of Queues

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/queues`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/queues
```

- **Response:** `200` (application/json)
- **Body:**

```
["messages"]
```

Retrieve Queue Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/queues/{queueName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/queues/messages
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "messages",
  "ownedItemCount": 55408,
  "backupItemCount": 55408,
  "minAge": 0,
  "maxAge": 0,
  "averageAge": 0,
  "offerOperationCount": 55408,
  "rejectedOffers": 0,
  "pollOperationCount": 0,
  "emptyPolls": 0,
  "otherOperationCount": 0,
  "events": 0,
  "creationTime": 1403602694196
}
```

12.2.13. Topics Resource

This resource returns a list of topics belonging to the provided cluster.

Retrieve List of Topics

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/topics`

- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/topics
```

- **Response:** `200` (application/json)
- **Body:**

```
["news"]
```

Retrieve Topic Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/topics/{topicName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/topics/news
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "news",
  "publishOperationCount": 56370,
  "receiveOperationCount": 56370,
  "creationTime": 1403602693411
}
```

Retrieve List of Reliable Topics

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/reliabletopics`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/reliabletopics
```

- **Response:** `200` (application/json)
- **Body:**

```
["news"]
```

Retrieve Reliable Topic Information

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/reliabletopics/{reliableTopicName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/reliabletopics/news
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "news",
  "publishOperationCount": 56370,
  "receiveOperationCount": 56370,
  "creationTime": 1403602693411,
}
```

12.2.14. Executors Resource

This resource returns a list of executors belonging to the provided cluster.

Retrieve List of Executors

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/executors`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/executors
```

- **Response:** `200` (application/json)
- **Body:**

```
["order-executor"]
```

Retrieve Executor Information [GET] [/rest/clusters/{clustername}/executors/{executorName}]

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/executors/{executorName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/executors/order-executor
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "order-executor",
  "creationTime": 1403602694196,
  "pendingTaskCount": 0,
  "startedTaskCount": 1241,
  "completedTaskCount": 1241,
  "cancelledTaskCount": 0,
  "totalExecutionTime": 1000,
  "totalStartLatency": 400
}
```

12.2.15. PNCounters Resource

This resource returns a list of PN-counters belonging to the provided cluster.

Retrieve List of PNCounters

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/pncounters`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/pncounters
```

- **Response:** `200` (application/json)
- **Body:**

```
["order-pncounter"]
```

Retrieve PNCounter Information [GET] [/rest/clusters/{clustername}/pncounters/{pnCounterName}]

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/pncounters/{pnCounterName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/pncounters/order-pncounter
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "order-pncounter",
  "creationTime": 1403602694196,
  "statsPerMember": {
    "192.168.2.78:5701": {
      "value": 1,
      "incOperationCount": 1,
      "decOperationCount": 0
    },
    "192.168.2.79:5701": {
      "value": 1,
      "incOperationCount": 0,
      "decOperationCount": 0
    }
  }
}
```

12.2.16. FlakeIdGenerators Resource

This resource returns a list of flake ID generators belonging to the provided cluster.

Retrieve List of FlakeIdGenerators

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/flakeidgenerators`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/flakeidgenerators
```

- **Response:** `200` (application/json)
- **Body:**

```
["order-idgenerator"]
```

Retrieve FlakeIdGenerator Information [GET] [/rest/clusters/{clustername}/flakeidgenerators/{flakeIdGeneratorName}]

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/flakeidgenerators/{flakeIdGeneratorName}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/flakeidgenerators/order-idgenerator
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "name": "order-idgenerator",
  "creationTime": 1403602694196,
  "statsPerMember": {
    "192.168.2.78:5701": {
      "batchRequests": 1,
      "idCount": 100
    },
    "192.168.2.79:5701": {
      "batchRequests": 0,
      "idCount": 0
    }
  }
}
```

12.2.17. Client Statistics Resource

This resource returns a list of clients belonging to the provided cluster.

Retrieve List of Client UUIDs

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/clientStats`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/clientStats
```

- **Response:** `200` (application/json)

- **Body:**

```
[
  "f3b1e0e9-ea67-41b2-aba5-ea7480f02a93",
  "cebf4dc9-852c-4605-a181-ffe1cca371a4",
  "2371eed5-26e0-4470-92c1-41ea17110ef6",
  "139990b3-fbc0-43a8-9c12-be53913333f7",
  "d0364a1e-8665-46a8-af1d-be1af5580d07",
  "7f337f8a-3538-4b5c-8ffc-9d4ae459e956",
  "6ef9b6e5-5add-40d9-9319-ce502f55b5fc",
  "fead3a99-19de-431c-9dd0-d6ecc4a4b9c8",
  "e788e04e-2ded-4992-9d76-52c1973216e5",
  "654fc9fb-c5c1-48a0-9b69-0c129fce860f"
]
```

Retrieve Detailed Client Statistics [GET] [/rest/clusters/{clustername}/clientStats/{clientUuid}]

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/clientStats/{clientUuid}`
- **Request:**

```
curl http://localhost:8080/rest/clusters/dev/clientStats/2371eed5-26e0-4470-92c1-
41ea17110ef6
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "type": "JAVA",
  "name": "hz.client_7",
  "address": "127.0.0.1",
  "clusterConnectionTimestamp": 1507874427419,
  "enterprise": true,
  "lastStatisticsCollectionTime": 1507881309434,
  "committedVirtualMemorySize": 12976173056,
  "freePhysicalMemorySize": 3615662080,
  "freeSwapSpaceSize": 8447324160,
  "maxFileDescriptorCount": 1000000,
  "openFileDescriptorCount": 191,
  "processCpuTime": 252980000000,
  "systemLoadAverage": 83.0,
  "totalPhysicalMemorySize": 16756101120,
  "totalSwapSpaceSize": 8447324160
  "availableProcessors": 12,
  "freeHeapMemory": 135665432,
  "maxHeapMemory": 3724541952,
```

```
  "totalHeapMemory": 361234432,
  "usedHeapMemory": 225569000,
  "uptime": 6894992,
  "memberConnection": "ALL",
  "version": "UNKNOWN",
  "nearCacheStats":{
    "CACHE":{
      "cache-1":{
        "clientUuid":"805a6342-eebb-412d-aeba-21c55fadedc3",
        "creationTime":1588752664324,
        "evictions":0,
        "expirations":0,
        "hits":148,
        "misses":164,
        "ownedEntryCount":4,
        "ownedEntryMemoryCost":420,
        "lastPersistenceDuration":0,
        "lastPersistenceKeyCount":0,
        "lastPersistenceTime":0,
        "lastPersistenceWrittenBytes":0
      }
    },
    "MAP":{
      "map-1":{
        "clientUuid":"805a6342-eebb-412d-aeba-21c55fadedc3",
        "creationTime":1588752664321,
        "evictions":0,
        "expirations":0,
        "hits":92,
        "misses":116,
        "ownedEntryCount":109,
        "ownedEntryMemoryCost":7701,
        "lastPersistenceDuration":0,
        "lastPersistenceKeyCount":0,
        "lastPersistenceTime":0,
        "lastPersistenceWrittenBytes":0
      }
    }
  }
}
```


Retrieve WAN Publisher Statistics [GET] [/rest/clusters/{clustername}/wanStats/{wanReplication}/publishers/{publisher}]

- **Request Type:** GET
- **URL:** `/rest/clusters/{clustername}/wanStats/{wanReplication}/publishers/{publisher}`
- **Request:**

```
curl
http://localhost:8080/rest/clusters/dev/wanStats/devWanConfig/publishers/devPublish
er
```

- **Response:** `200` (application/json)
- **Body:**

```
{
  "cluster": "dev",
  "configName": "devWanConfig",
  "publisherId": "devPublisher",
  "totalPublishedEventCount": 1023,
  "totalPublishLatency": 14200,
  "outboundQueueSize": 10
}
```

12.3. Clustered JMX

Hazelcast IMDG Enterprise

Clustered JMX via Management Center allows you to monitor clustered statistics of distributed objects from a JMX interface.

12.3.1. Configuring Clustered JMX

In order to configure Clustered JMX, use the following command line parameters for your Management Center deployment.

- `-Dhazelcast.mc.jmx.enabled=true` (default is false)
- `-Dhazelcast.mc.jmx.port=9000` (optional, default is 9999)
- `-Dcom.sun.management.jmxremote.ssl=false`
- `-Dhazelcast.mc.jmx.rmi.port=9001` (optional, default is 9998)
- `-Dhazelcast.mc.jmx.host=localhost` (optional, default is server's host name)

With embedded Jetty, you do not need to deploy your Management Center application to any container or application server.

You can start the Management Center application with Clustered JMX enabled as shown below.

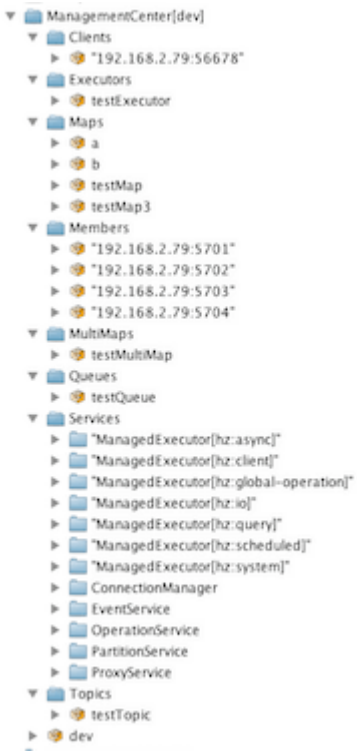
```
java -Dhazelcast.mc.jmx.enabled=true
-Dhazelcast.mc.jmx.port=9999
-Dcom.sun.management.jmxremote.ssl=false -jar hazelcast-management-center-
4.2020.08.jar
```

Once the Management Center starts, you should see a log similar to the one below:

```
INFO: Management Center 3.3
Jun 05, 2014 11:55:32 AM com.hazelcast.webmonitor.service.jmx.impl.JMXService
INFO: Starting Management Center JMX Service on port :9999
```

You should be able to connect to the Clustered JMX interface using the address `localhost:9999`.

You can use `jconsole` or any other JMX client to monitor your Hazelcast IMDG cluster. As an example, below is the `jconsole` screenshot of the Clustered JMX hierarchy.



Enabling TLS/SSL for Clustered JMX

By default, Clustered JMX is served unencrypted. To enable TLS/SSL for Clustered JMX, use the following command line parameters for your Management Center deployment:

- `-Dhazelcast.mc.jmx.ssl=true` (default is false)
- `-Dhazelcast.mc.jmx.ssl.keyStore=path to your keystore`
- `-Dhazelcast.mc.jmx.ssl.keyStorePassword=password for your keystore`

The following is an example on how to start the Management Center with a TLS/SSL enabled Clustered JMX service on port 65432:

```
java -Dhazelcast.mc.jmx.enabled=true
-Dhazelcast.mc.jmx.port=65432
-Dhazelcast.mc.jmx.ssl=true
-Dhazelcast.mc.jmx.ssl.keyStore=/some/dir/selfsigned.jks
-Dhazelcast.mc.jmx.ssl.keyStorePassword=yourpassword -jar hazelcast-management-center-
4.2020.08.jar
```

You can encrypt the keystore password and pass it as a command line argument in encrypted form for improved security. See the Variable Replacers section for more information.

Then, you can use the following command to connect to the Clustered JMX service using JConsole with the address `localhost:65432`:

```
jconsole -J-Djavax.net.ssl.trustStore=/some/dir/selftrusted.ts -J-Djavax.net.ssl
.trustStorePassword=trustpass
```

Additional TLS/SSL Configuration Options

The following are some additional command line arguments that you can use to configure TLS/SSL for clustered JMX:

- `-Dhazelcast.mc.jmx.ssl.keyStoreType`: Type of the keystore. Its default value is JKS.
- `-Dhazelcast.mc.jmx.ssl.keyManagerAlgorithm`: Name of the algorithm based on which the authentication keys are provided. The system default is used if none is provided. You can find out the default by calling the `javax.net.ssl.KeyManagerFactory#getDefaultAlgorithm` method.

12.3.2. Clustered JMX API

The management beans are exposed with the following object name format:

```
ManagementCenter[`*cluster name*`]:type=<`*object type*`>,name=<`*object
name*`>,member="<`*cluster member IP address*`>"
```

The object name starts with the `ManagementCenter` prefix. Then it has the cluster name in brackets followed by a colon. After that, `type`, `name` and `member` attributes follow, each separated with a comma.

- `type` is the type of object. Values are `Clients`, `Executors`, `Maps`, `Members`, `MultiMaps`, `Queues`, `Counters`, `Services`, and `Topics`.
- `name` is the name of object.
- `member` is the member address of object (only required if the statistics are local to the member).

A sample bean is shown below.

```
ManagementCenter[dev]:type=Services,name=OperationService,member="192.168.2.79:5701"
```

Here is the list of attributes that are exposed from the Clustered JMX interface.

- **ManagementCenter**
- ManagementCenter
  - ClusterType
  - LicenseExpirationTime
  - Clusters
- **ManagementCenter[<ClusterName>]**
- <ClusterName>
  - MasterAddress
  - LicenseExpirationTime
- ClientStats
  - <Client UUID>
    - UsedHeapMemory
    - FreeHeapMemory
    - MaxHeapMemory
    - TotalHeapMemory
    - ClientName
    - AvailableProcessors
    - Uptime
    - Enterprise
    - MemberConnection
    - ClusterConnectionTimestamp
    - LastStatisticsCollectionTime
    - CommittedVirtualMemorySize
    - FreePhysicalMemorySize
    - FreeSwapSpaceSize
    - MaxFileDescriptorCount
    - OpenFileDescriptorCount
    - ProcessCpuTime
    - SystemLoadAverage
    - TotalPhysicalMemorySize
    - TotalSwapSpaceSize

        - Version
        - Address
        - Type
        - NearCacheStats
            - CACHE
                - `<Cache Name>`
                    - Evictions
                    - Expirations
                    - Hits
                    - Misses
                    - OwnedEntryCount
                    - OwnedEntryMemoryCost
                    - LastPersistenceDuration
                    - LastPersistenceKeyCount
                    - LastPersistenceTime
                    - LastPersistenceWrittenBytes
                    - CreationTime
            - MAP
                - `<Map Name>`
                    - Evictions
                    - Expirations
                    - Hits
                    - Misses
                    - OwnedEntryCount
                    - OwnedEntryMemoryCost
                    - LastPersistenceDuration
                    - LastPersistenceKeyCount
                    - LastPersistenceTime
                    - LastPersistenceWrittenBytes
                    - CreationTime
- Clients
    - `<Client Address>`
        - Address
        - CanonicalHostName
        - ClientName

        - ClientType
        - IpAddress
        - Labels
        - Uuid
- Executors
    - `<Executor Name>`
        - Cluster
        - Name
        - StartedTaskCount
        - CompletedTaskCount
        - CancelledTaskCount
        - PendingTaskCount
        - TotalExecutionTime
        - TotalStartLatency
- Maps
    - `<Map Name>`
        - Cluster
        - Name
        - BackupEntryCount
        - BackupEntryMemoryCost
        - CreationTime
        - DirtyEntryCount
        - Events
        - GetOperationCount
        - HeapCost
        - Hits
        - LastAccessTime
        - LastUpdateTime
        - LockedEntryCount
        - MaxGetLatency
        - MaxPutLatency
        - MaxRemoveLatency
        - OtherOperationCount
        - OwnedEntryCount
        - PutOperationCount

        - RemoveOperationCount
        - TotalPutLatency
        - TotalGetLatency
        - TotalRemoveLatency
- ReplicatedMaps
    - `<Replicated Map Name>`
        - Cluster
        - Name
        - BackupEntryCount
        - BackupEntryMemoryCost
        - CreationTime
        - DirtyEntryCount
        - Events
        - GetOperationCount
        - HeapCost
        - Hits
        - LastAccessTime
        - LastUpdateTime
        - LockedEntryCount
        - MaxGetLatency
        - MaxPutLatency
        - MaxRemoveLatency
        - OtherOperationCount
        - OwnedEntryCount
        - PutOperationCount
        - RemoveOperationCount
        - TotalPutLatency
        - TotalGetLatency
        - TotalRemoveLatency
- Caches
    - `<Cache Name>`
        - Cluster
        - Name
        - CreationTime
        - Hits

        - Misses
        - GetOperationCount
        - PutOperationCount
        - RemoveOperationCount
        - Evictions
        - AverageGetTime
        - AveragePutTime
        - AverageRemoveTime
        - LastAccessTime
        - LastUpdateTime
        - OwnedEntryCount
- Members
    - `<Member Address>`
        - Uuid
        - Address
        - CpMemberUuid
        - ConnectedClientCount
        - FreeHeapMemory
        - MaxHeapMemory
        - CommittedHeapMemory
        - UsedHeapMemory
        - IsMaster
        - OwnedPartitionCount
        - MaxNativeMemory
        - CommittedNativeMemory
        - UsedNativeMemory
        - FreeNativeMemory
- MultiMaps
    - `<MultiMap Name>`
        - Cluster
        - Name
        - BackupEntryCount
        - BackupEntryMemoryCost
        - CreationTime
        - DirtyEntryCount

        - Events
        - GetOperationCount
        - HeapCost
        - Hits
        - LastAccessTime
        - LastUpdateTime
        - LockedEntryCount
        - MaxGetLatency
        - MaxPutLatency
        - MaxRemoveLatency
        - OtherOperationCount
        - OwnedEntryCount
        - PutOperationCount
        - RemoveOperationCount
        - TotalPutLatency
        - TotalGetLatency
        - TotalRemoveLatency
- Queues
    - `<Queue Name>`
        - Cluster
        - Name
        - MinAge
        - MaxAge
        - AverageAge
        - OwnedItemCount
        - BackupItemCount
        - OfferOperationCount
        - RejectedOffers
        - PollOperationCount
        - EmptyPolls
        - OtherOperationsCount
        - Events
        - CreationTime
- Counters
    - `<Counter Name>`

        - Cluster
        - Name
        - CreationTime
        - StatsPerMember
            - <Member Address>
                - Value
                - IncOperationCount
                - DecOperationCount
- Services
    - ConnectionManager
        - ActiveConnectionCount
        - ClientConnectionCount
        - ConnectionCount
    - EventService
        - EventQueueCapacity
        - EventQueueSize
        - EventThreadCount
    - OperationService
        - ExecutedOperationCount
        - OperationExecutorQueueSize
        - OperationThreadCount
        - RemoteOperationCount
        - ResponseQueueSize
        - RunningOperationsCount
    - PartitionService
        - ActivePartitionCount
        - PartitionCount
    - ProxyService
        - ProxyCount
        - CreatedCount
        - DestroyedCount
    - ManagedExecutor[<Managed Executor Name>]
        - Name
        - CompletedTaskCount
        - MaximumPoolSize

        - PoolSize
        - QueueSize
        - RemainingQueueCapacity
- Topics
    - `<Topic Name>`
        - Cluster
        - Name
        - CreationTime
        - PublishOperationCount
        - ReceiveOperationCount
- ReliableTopics
    - `<Reliable Topic Name>`
        - Cluster
        - Name
        - CreationTime
        - PublishOperationCount
        - ReceiveOperationCount
- FlakeIdGenerators
    - `<Generator Name>`
        - Cluster
        - Name
        - CreationTime
        - StatsPerMember
            - `<Member Address>`
                - BatchRequests
                - IdCount
- WanConfigs
    - `<Wan Replication Config>[<Publisher ID>]`
        - Cluster
        - ConfigName
        - PublisherId
        - OutboundQueueSize
        - TotalPublishedEventCount
        - TotalPublishLatency
- Jobs (present only if it's a Jet cluster)

- `<Job ID>`
  - CompletionTime
  - ConfigJson
  - Edges
    - [0…n]
      - FromOrdinal
      - LastMinRecordsFlow
      - SourceVertex
      - TargetVertex
      - ToOrdinal
      - TotalRecordsFlow
  - Failure
  - Id
  - JobName
  - LastExecutionId
  - LastMinIn
  - LastMinOut
  - LastSnapshotDuration
  - LastSnapshotKeys
  - LastSnapshotSize
  - LastSnapshotTime
  - ProcessingGuarantee
  - RunningNodeCount
  - SnapshotIntervalMillis
  - Status
  - SubmissionTime
  - TotalIn
  - TotalNodeCount
  - TotalOut
  - Vertices
    - [0..n]
      - GlobalParallelism
      - Id
      - Incoming
      - LastMinIn

        - LastMinOut
        - MaxLatencyToRealTime
        - Outgoing
            - [0..n]
                - Info
                    - LastMin
                    - Ordinal
                    - Total
                - TargetVertex
        - Parallelism
        - Processors
            - [0..n]
                - Info
                    - CapPercentage
                    - EmittedCount
                    - LastForwardedWmLatency
                    - QueueCapacity
                    - QueueSize
                    - ReceivedCount
                - ProcessorId
        - Skew
        - TotalIn
        - TotalOut
- ExportedJobSnapshots (present only if it's a Jet cluster)
    - `<Snapshot Name>`
        - CreationTime
        - JobId
        - JobName
        - Name
        - PayloadSize

12.3.3. Integrating with New Relic

Use the Clustered JMX interface to integrate the Hazelcast Management Center with New Relic. To perform this integration, attach the New Relic Java agent and provide an extension file that describes which metrics will be sent to New Relic.

See Custom JMX instrumentation by YAML on the New Relic webpage.

The following is an example Map monitoring `.yml` file for New Relic:

```
name: Clustered JMX
version: 1.0
enabled: true

jmx:
- object_name: ManagementCenter[clustername]:type=Maps,name=mapname
  metrics:
  - attributes: PutOperationCount, GetOperationCount, RemoveOperationCount, Hits,
BackupEntryCount, OwnedEntryCount, LastAccessTime, LastUpdateTime
  - type: simple
- object_name: ManagementCenter[clustername]:type=Members,name="member address in
double quotes"
  metrics:
  - attributes: OwnedPartitionCount
  - type: simple
```

Put the `.yml` file in the `extensions` directory in your New Relic installation. If an `extensions` directory does not exist there, create one.

After you set your extension, attach the New Relic Java agent and start the Management Center as shown below.

```
java -javaagent:/path/to/newrelic.jar -Dhazelcast.mc.jmx.enabled=true\
    -Dhazelcast.mc.jmx.port=9999 -jar hazelcast-management-center-4.2020.08.jar
```

If your logging level is set to `FINER`, you should see the log listing in the file `newrelic_agent.log`, which is located in the `logs` directory in your New Relic installation. The following is an example log listing:

```
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINE:
    JMX Service : querying MBeans (1)
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    JMX Service : MBeans query ManagementCenter[dev]:type=Members,
    name="192.168.2.79:5701", matches 1
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric OwnedPartitionCount : 68
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    JMX Service : MBeans query ManagementCenter[dev]:type=Maps,name=orders,
    matches 1
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric Hits : 46,593
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric BackupEntryCount : 1,100
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric OwnedEntryCount : 1,100
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric RemoveOperationCount : 0
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric PutOperationCount : 118,962
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric GetOperationCount : 0
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric LastUpdateTime : 1,401,962,426,811
Jun 5, 2014 14:18:43 +0300 [72696 62] com.newrelic.agent.jmx.JmxService FINER:
    Recording JMX metric LastAccessTime : 1,401,962,426,811
```

Then you can navigate to your New Relic account and create Custom Dashboards. See Creating custom dashboards.

While you are creating the dashboard, you should see the metrics that you are sending to New Relic from the Management Center in the **Metrics** section under the JMX directory.

12.3.4. Integrating with AppDynamics

Use the Clustered JMX interface to integrate the Hazelcast Management Center with **AppDynamics**. To perform this integration, attach the AppDynamics Java agent to the Management Center.

For agent installation, see the Install the App Agent for Java page.

For monitoring on AppDynamics, see the Using AppDynamics for JMX Monitoring page.

After installing AppDynamics agent, you can start the Management Center as shown below:

```
java -javaagent:/path/to/javaagent.jar \
     -Dhazelcast.mc.jmx.enabled=true \
     -Dhazelcast.mc.jmx.port=9999 -jar hazelcast-management-center-4.2020.08.jar
```

When the Management Center starts, you should see the logs below:

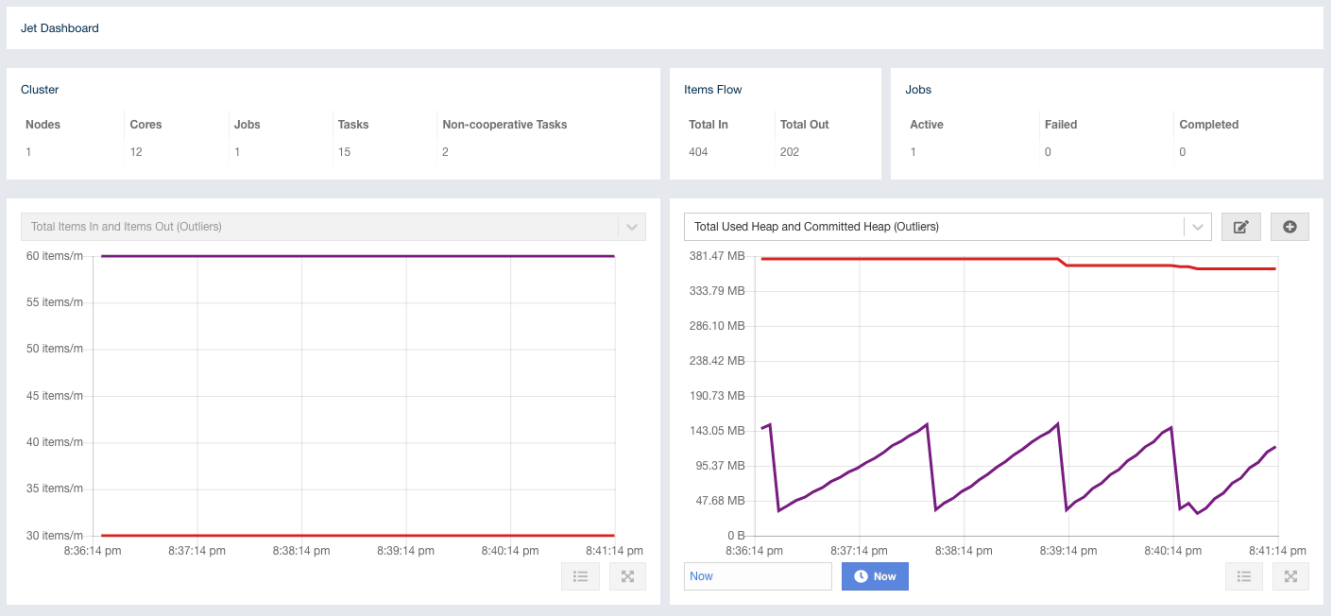
```
Started AppDynamics Java Agent Successfully.
Hazelcast Management Center starting on port 8080 at path : /
```

13. Monitoring a Jet Cluster

This chapter details the monitoring and administering of a Jet cluster using Management Center.

13.1. Dashboard Page

This page gives an overview of the connected Jet cluster, as shown below:



The following subsections describe each portion of the page.

13.1.1. Cluster

Shows a summary of the cluster by providing the following metrics:

- **Nodes:** Number of cluster members.
- **Cores:** Number of available CPU cores in the cluster reported by the JVM.
- **Jobs:** Number of jobs in the cluster.
- **Tasks:** Number of cooperative tasks in the cluster. See https://jet-start.sh/docs/concepts/dag#tasks-concurrency-is-cooperative for more detailed explanation.
- **Non-cooperative Tasks:** Number of non-cooperative tasks in the cluster. See https://jet-start.sh/docs/concepts/dag#tasks-concurrency-is-cooperative for more detailed explanation.

13.1.2. Items Flow

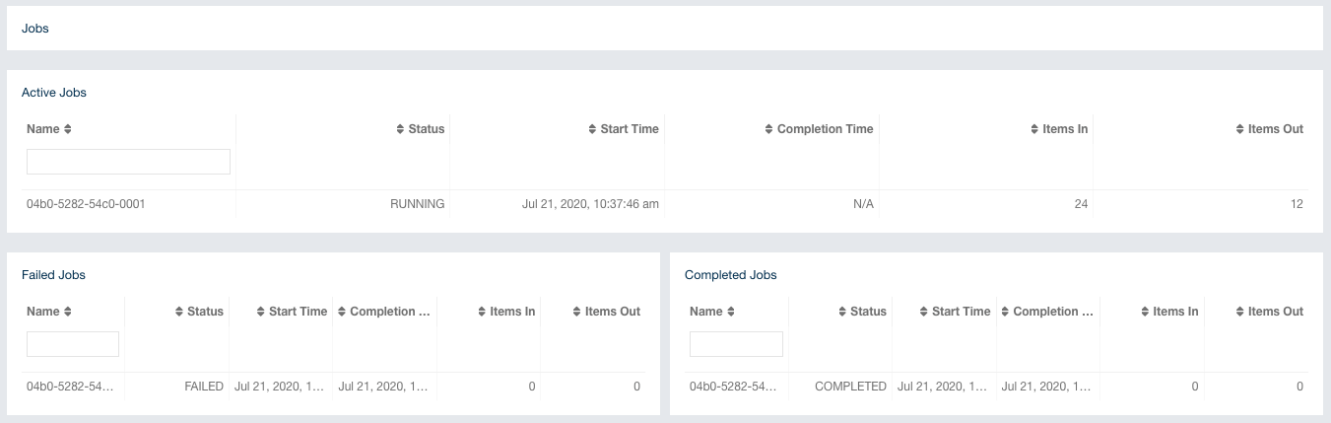
- **Total In:** Total number of items read from the sources of all jobs.
- **Total Out:** Total number of items written to the sinks of all jobs.

13.1.3. Jobs

- **Active:** Number of active jobs in the cluster.
- **Failed:** Number of failed jobs in the cluster.
- **Completed:** Number of completed jobs in the cluster.

13.2. Jobs

This page provides a general overview of the jobs in a Hazelcast Jet cluster.



13.2.1. Active Jobs

It shows the list of actively running jobs in the cluster:

- **Name:** Name/ID of the job.
- **Status:** Current status of the job.
- **Start Time:** Start time of the job.
- **Completion Time:** Completion time of the job.
- **Items In:** Total number of items read from the sources of the jobs.
- **Items Out:** Total number of items written to the sinks of the jobs.

13.2.2. Failed Jobs

It shows a list of failed jobs in the cluster with the following information:

- **Name:** Name/ID of the job.
- **Status:** Current status of the job.
- **Start Time:** Start time of the job.

- **Completion Time:** Completion time of the job.
- **Items In:** Total number of items read from the sources of the jobs.
- **Items Out:** Total number of items written to the sinks of the jobs.

13.2.3. Completed Jobs

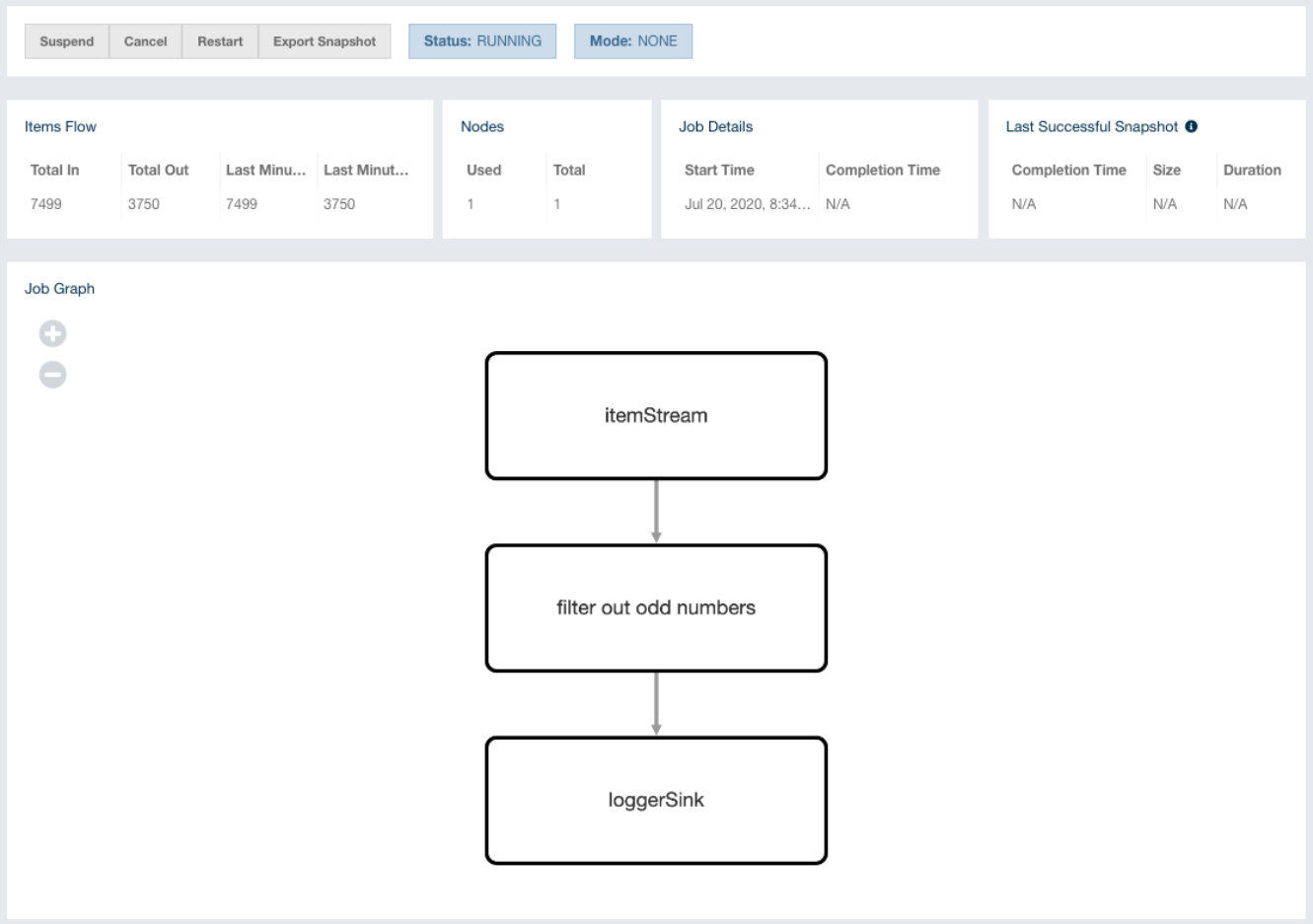
It shows a list of completed jobs in the cluster with the following information:

- **Name:** Name/ID of the job.
- **Status:** Current status of the job.
- **Start Time:** Start time of the job.
- **Completion Time:** Completion time of the job.
- **Items In:** Total number of items read from the sources of the jobs.
- **Items Out:** Total number of items written to the sinks of the jobs.

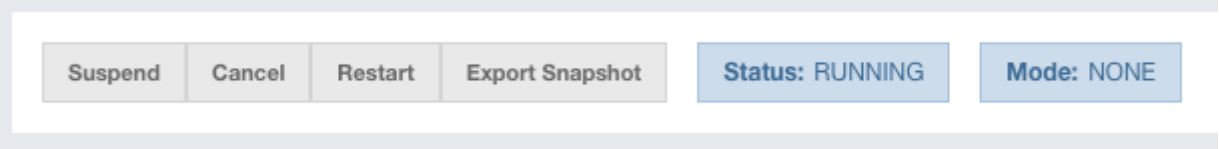
You can click on a job to see a detailed view of it.

13.2.4. Job Details

This page offers a tool for diagnosing data flow within the job. It provides graphical visualization of the stages, ability to manage the lifecycle of the job and allows you to peek into dataflow stats across the DAG. You can diagnose bottlenecks this way.



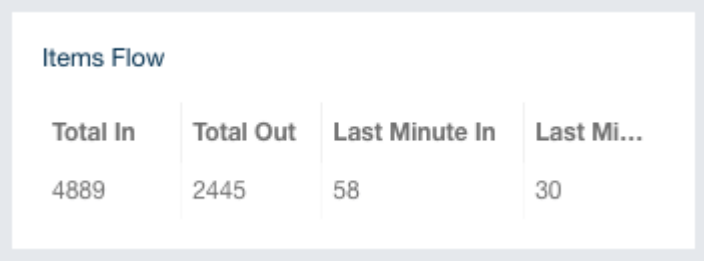
Job Management:



You need to be an admin user to execute any of the actions below.

- **Suspend:** Suspends the running job, only visible when the job is in the `RUNNING` state.
- **Resume:** Resumes the suspended job, only visible when the job is in the `SUSPENDED` state.
- **Cancel:** Stops the execution of the job.
- **Restart:** Stops the execution of the job and starts a new execution for it.
- **Export Snapshot:** Initiates a named snapshot export; exported snapshots can be managed via snapshots view.
- **Status:** Current status of the job.
- **Mode:** Processing guarantee mode of the job; either `None`, `At Least Once` or `Exactly Once`.

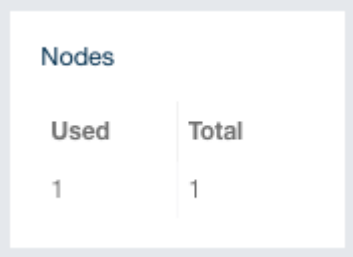
Items Flow:



It shows the dataflow metrics for the cluster:

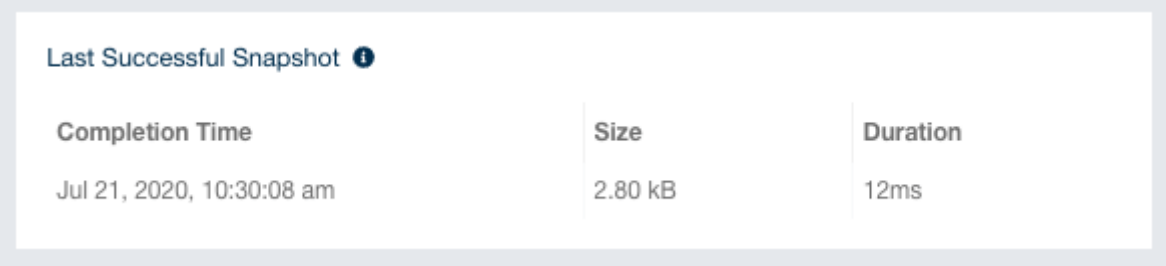
- **Total In:** Total number of items read from the source of the job.
- **Total Out:** Total number of items written to the sink of the job.
- **Last Min In:** Number of items read from the source of the job in the last minute.
- **Last Min Out:** Number of items written to the sink of the job in the last minute.

Nodes:



- **Used:** Number of members used by the job.
- **Total:** Number of total members in the cluster.

Last Successful Snapshot:

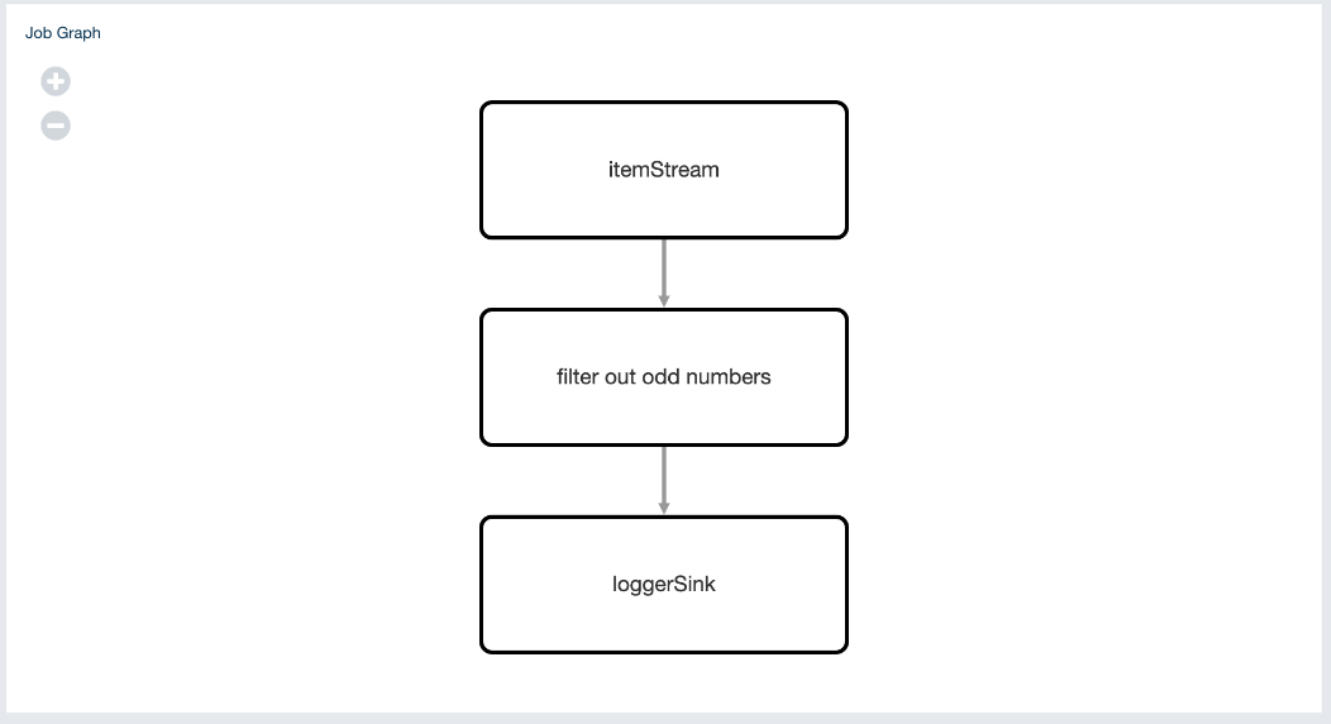


- **Completion Time:** Latest successful snapshot completion time.
- **Size:** Size of the snapshot.
- **Duration:** Duration of the snapshot creation.

These metrics are for only non-exported snapshots.

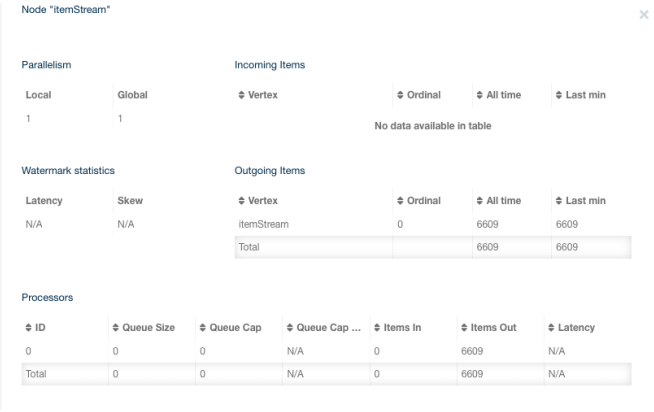
Job Visualization:

It is the graphical representation of the job topology.



Vertex Details:

It shows information about the selected vertex on the Job Visualization section.



Parallelism::

- **Local:** Number of processors running for that vertex on each member.
- **Global:** Total number of processors running for that vertex on the cluster.

Incoming Items:

It lists all the incoming edges by their source vertices and shows the following information and totals for each of them.

- **All Time:** Total number of items received by this vertex.
- **Last Min:** Number of items received by this vertex in the last minute.

Outgoing Items:

It lists all the outgoing edges by their target vertices and shows the following information and totals for each of them.

- **All Time:** Total number of items sent by this vertex.
- **Last Min:** Number of items sent by this vertex in the last minute.

Watermark Statistics:

- **Latency:** This is the time difference between wall-clock time and the last forwarded watermark ("event time, time of the stream"). Multiple factors contribute to the total latency, such as the latency in the external system, allowed lag (which is always included), clock drift and also long event-to-event intervals in any partition (this one is the trickiest). See here for more information.
- **Skew:** This is the difference between latencies of the processor with the highest and lowest latencies. Most common cause is a long event-to-event interval in some source partition or an idle partition (until the idle timeout elapses). Overload of events in one partition can also cause it.

Processors:

It lists all the processors this vertex has in the cluster and shows the following information for each of them.

- **Queue Size:** Current size of the processor inbox queue.
- **Queue Cap:** Capacity of the processor inbox queue.
- **Queue Cap Usage:** Queue utilization percentage.
- **Items In:** Total number of items received by this processor.
- **Items Out:** Total number of items sent by this processor.
- **latency:** Time difference between the wall-clock time and the last forwarded watermark ("event time, time of the stream"). Multiple factors contribute to the total latency:
  - **latency in the external system**; events arrive already delayed to Jet source
  - **allowed lag**; if you allow for some time to wait for delayed events, watermarks will always be delayed by this lag. Note that the actual output might not be delayed.
  - **event-to-event interval**; if there is a time period between two events, the event time "stops" for that time. In other words, until a new event comes, Jet thinks the current time is the time of the last event. As "current event time" is tracked independently for each partition, this can be the major source of skew. If your events are irregular, you might consider adding heartbeat events. This factor also applies if you use `withIngestionTimestamps` since a new wall-clock time is assigned only if new event arrives.
  - **time to execute map/filter stages**; they contribute with the latency of the async call or with the time to execute CPU-heavy sync call.
  - **internal processing latency of Jet**; typically very low: 1 or 2 milliseconds. It can be higher if the network is slow, system is overloaded, if there are many vertices in the job or many jobs, which causes lot of switching, etc.
  - **clock drift**; since we're comparing to the real time, latency can be caused by a clock drift between the machine where event time is assigned (which can be also be an end user's device). It can even be negative. Always use NTP to keep wall-clock precise and avoid using timestamps from devices out of your control as event time.

Edge Details:

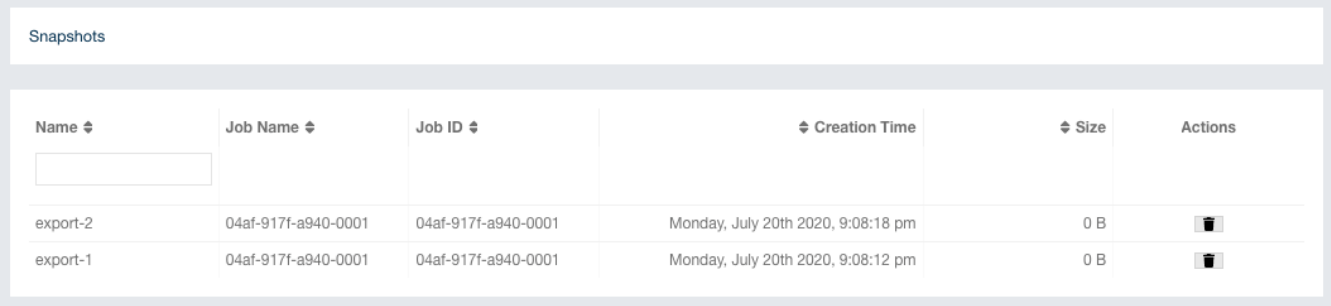
It shows information about the selected edge on the Job Visualization section.

Items Flow:

- **Total:** Total number of items passed through this edge.
- **Last Min:** Number of items passed through this edge in the last minute.

13.3. Snapshots

This page provides an overview of the exported snapshots of the Hazelcast Jet jobs. To create a new export, please refer to the Job Management section.



Here are the descriptions of parameters:

- **Name:** Given name of the exported snapshot.
- **Job Name:** Name of the job.
- **Job ID:** ID of the job.
- **Creation Time:** Creation time of the export.
- **Size:** Size of the exported snapshot.

You can click on the trash icon to delete any exported snapshots. Note that you need to be an admin user to execute this action.

14. Prometheus Exporter

Hazelcast Management Center can expose the metrics collected from cluster members to Prometheus. This feature can be turned on by setting the `hazelcast.mc.prometheusExporter.enabled` system property to `true`.

Prometheus can be configured to scrape Management Center in `prometheus.yml` as follows:

```
scrape_configs:
  - job_name: 'HZ MC'
    # scheme defaults to 'http'.
    static_configs:
    - targets: ['localhost:8080'] # replace this address with the network address of
Hazelcast Management Center
```

After starting Prometheus with this configuration, all metrics will be exported to Prometheus with the `hz_` prefix. The metrics are also available via the member JMX API. With the default configuration, Management Center will export all metrics reported by the cluster members. Since it can be overly verbose for some usecases, the metrics can be filtered with the `hazelcast.mc.prometheusExporter.filter.metrics.included` or the `hazelcast.mc.prometheusExporter.filter.metrics.excluded` system properties, both being comma-separated lists of metric names.

Example of starting Management Center with specifying the metrics exported to Prometheus:

```
java -jar -Dhazelcast.mc.prometheusExporter.enabled=true \
  -Dhazelcast.mc.prometheusExporter.filter.metrics.included
=hz_topic_totalReceivedMessages,hz_map_totalPutLatency \
  -jar {mc-jar-file}
```

Example of starting Management Center with specifying the metrics to be excluded from the Prometheus export:

```
java -jar -Dhazelcast.mc.prometheusExporter.enabled=true \
  -Dhazelcast.mc.prometheusExporter.filter.metrics.excluded=hz_os_systemLoadAverage
,hz_memory_freeHeap \
  -jar {mc-jar-file}
```

Prometheus connects via the same IP and port as the Management Center web interface.

15. Management Center Configuration Tool

The Management Center Configuration Tool (MC-Conf) is a command line tool that allows you to update certain parts of the Management Center configuration by using its built-in tasks. You can use the `mc-conf.sh` or `mc-conf.bat` script to run the MC-Conf tool.

You must run the MC-Conf tool on the same machine where the Management Center web application is deployed.

The Management Center must not be running when changes are made via MC-Conf.

15.1. Built-In Help

In order to see all available commands, run the MC-Conf script with no arguments as shown below.

```
./mc-conf.sh
```

As the result, you should see an output similar to below.

```
Hazelcast Management Center Configuration Tool 4.0
Usage: mc-conf [-hV] COMMAND TASK
Command line tool for interacting with Hazelcast Management Center
configuration.

Global options are:
  -h, --help      Show this help message and exit.
  -V, --version   Print version information and exit.
Commands:
  cluster           Manage Cluster Connection Configs
  user              Manage Default Security Provider Users
  ldap              Manage LDAP Security Provider
  active-directory  Manage Active Directory Security Provider
  jaas              Manage JAAS Security Provider
  security          General Security Provider management
  set               Change MC settings
  dev-mode          Manage DevMode Security Provider
```

When you choose a specific subcommand from the list above, you can see all tasks available for it. See the following example:

```
$ ./mc-conf.sh user
Usage: mc-conf user [-hV] TASK
Manage Default Security Provider Users
  -h, --help      Show this help message and exit.
  -V, --version   Print version information and exit.
Commands:
  create           Create a new user record in the default security provider.
                   *Important notice* Make sure that Management Center web
                     application is stopped (offline) before starting this
                     task.

  update-password  Change password for the given user record in the default
                     security provider.
                   *Important notice* Make sure that Management Center web
                     application is stopped (offline) before starting this
                     task.
```

You can also get specific help for any task by using the `-h` (or `--help`) command line option. See the following example:

```
$ ./mc-conf.sh user create -h
Usage: mc-conf user create [-hvV] [-p[=<password>]] [-H=<homedir>]
                           -n=<username> -r=<role>
Create a new user record in the default security provider.
*Important notice* Make sure that Management Center web application is stopped
(offline) before starting this task.

  -H, --home=<homedir>   Optional path to Management Center home directory. By
                           default ~/hazelcast-mc/ is used.
  -n, --username=<username>
                         Username for the user record.
  -p, --password[=<password>]
                         Password for the user record. Provide value directly
                           or use without value to enter securely with
                           interactive prompt.
  -r, --role=<role>      Roles for the user record. Valid values: readonly,
                           readwrite, metricsonly, admin.
  -h, --help             Show this help message and exit.
  -V, --version          Print version information and exit.
  -v, --verbose          Enable full logging output. Use this option to see
                           full stack traces.
```

15.2. Configuring Cluster Connection

The `cluster add` task adds a new connection configuration for a cluster. Note that you must stop the Management Center web application before running this task.

You can use this task for various scripting purposes, and automatically configuring Management Center, without the need for a manual cluster connection configuration through UI.

If you have used a non-default Management Center home directory location, you must provide the path to the home directory with the `-H` (or `--home`) option.

15.3. Creating Users

The `user create` task creates a new user in the default security provider. Note that you must stop the Management Center web application before running this task.

You can use this task for various scripting purposes. See the Hazelcast Docker Code Samples repository for an example of Docker image for the Management Center container with a built-in user account.

If you have used a non-default Management Center home directory location, then you must provide the path to the home directory with the `-H` (or `--home`) option.

If you're on Linux or MacOS devices and provide value directly to `mc-conf`, please enclose password in single quotes like: `-p='mysecr3tp@s$word'`

15.4. Changing User Password

The `user update-password` task resets the password of a specified user in the default security provider. Note that you must stop the Management Center web application before running this task.

You can use this task as a recovery mechanism for the Management Center's administrator user account.

If you have used a non-default Management Center home directory location, you must provide the path to the home directory with the `-H` (or `--home`) option.

If you're on Linux or MacOS devices and provide value directly to `mc-conf`, please enclose password in single quotes like: `-p='mysecr3tp@s$word'`

15.5. Configuring LDAP Security Provider

The `ldap configure` task configures the LDAP security provider. Note that you must stop the Management Center web application, before running this task.

You can use this task for various scripting purposes and automatically configuring Management Center without the need for a manual security provider configuration through UI.

You can encrypt the LDAP password before saving with this task. See the Variable Replacers section for more information.

As with the UI based LDAP configuration, you can also use keystore for secure password storage, by using the optional `--key-store-*` options, as shown in the examples below.

If you want to use the built-in Management Center managed keystore, you can add the following options: `--ks-create --key-store=<hazelcast-mc directory>/mc.jceks --key-store-password=<password>`. This creates a keystore in the default Management Center directory, and saves the LDAP password in it. If you want to customize the keystore name or Management Center directory when starting Management Center, you need to reflect that with the `--key-store=<path>` option.

If you want to use the existing externally managed keystore, you can use the following options: `--key-store=<keystore path> --key-store-password=<password> [--key-store-type=<type> --key-store-provider=<provider>]`. Note that if the keystore with such path doesn't exist, task fails.

You still need to properly configure Management Center to use keystore. See LDAP Authentication section for details on using the built-in and existing keystores.

If you have used a non-default Management Center home directory location, then you must provide the path to the home directory with the `-H` (or `--home`) option.

15.6. Updating LDAP Password

The `ldap update-password` task updates the encrypted LDAP password stored in the keystore. It expects information about the keystore such as its location and password and the new LDAP password that you want to use. See the LDAP Authentication section for more information on the encrypted LDAP passwords. After updating the LDAP password, you need to click on the **Reload Security Config** button on the login page.

15.7. Configuring Active Directory Security Provider

The `active-directory configure` task configures the Active Directory security provider. Note that you must stop the Management Center web application before running this task.

You can use this task for various scripting purposes, and automatically configuring Management Center, without the need for a manual security provider configuration through UI.

If you have used a non-default Management Center home directory location, then you must provide the path to the home directory with the `-H` (or `--home`) option.

15.8. Configuring JAAS Security Provider

The `jaas configure` task configures the JAAS security provider. Note that you must stop the Management Center web application before running this task.

You can use this task for various scripting purposes, and automatically configuring Management Center, without the need for a manual security provider configuration through UI.

If you have used a non-default Management Center home directory location, then you must provide the path to the home directory with the `-H` (or `--home`) option.

15.9. Configuring Dev Mode Security Provider

The `dev-mode` configure task configures the Dev Mode security provider. Note that you must stop the Management Center web application before running this task.

You can use this task for various scripting purposes, and automatically configuring Management Center, without the need for a manual security provider configuration through UI.

If you have used a non-default Management Center home directory location, then you must provide the path to the home directory with the `-H` (or `--home`) option.

15.10. Resetting Security Provider

The `security reset` task resets current security provider used in the Management Center. For the default security provider it also deletes all built-in user accounts. Note that you must stop the Management Center web application before running this task.

You can use this task as a recovery mechanism for the Management Center deployment in case if a non-default security provider is configured. In case of the default security provider, you can also use the `user create` or `user update-password` task as the recovery mechanism.

If you have used a non-default Management Center home directory location, then you must provide the path to the home directory with the `-H` (or `--home`) option.

15.11. Enabling/Disabling Metrics Persistence

The `set metrics-persistence-enabled` task lets you choose whether metrics should be persisted to disk or not. Note that you must stop the Management Center web application before running this task.

15.12. Advanced Features

MC-Conf supports interactive options for secure processing of passwords. To use it, you need to use the password option without providing a value, i.e., instead of `--password=<password>` use `--password`. When you use this option without providing a value, you will get a prompt to enter a value on the console. An example of the interactive option usage is shown below.

```
$ ./mc-conf.sh user update-password --username=admin --password
Enter value for --password (Password for the user record. Provide value directly,
or use without value to enter securely with interactive prompt.): ********
Successfully changed password for user 'admin'.
```

As you see in the above example, the password input is not echoed to the console since it is provided with the secure interactive mode.

Another advanced feature of MC-Conf is the support for argument files. When an argument beginning with the character `@` is encountered, it is treated as a path leading to a text file. The contents of that file are automatically expanded into the current task. An example of the argument file usage is shown below.

```
$ ./mc-conf.sh user update-password @arg-file.txt
Successfully changed password for user 'admin'.
$ cat arg-file.txt
--username=admin --password=mnb3c4s0
```

16. Phone Home

Hazelcast uses phone home data to learn about usage of Hazelcast Management Center.

Hazelcast Management Center instances initially call our phone home server 30 minutes after they are started and once every 24 hours thereafter.

What is sent in?

The following information is sent in a phone home:

- Hazelcast Management Center version
- Authentication provider used (Default, LDAP, ActiveDirectory, JAAS)
- Whether clustered REST is enabled or not
- Whether clustered JMX is enabled or not
- Whether TLS is enabled or not
  - If TLS is enabled, whether mutual authentication is enabled or not
- Whether Management Center is deployed on an application server or used in standalone mode
  - If not in standalone mode, type of the application server
- Number of users (if the default security provider is used)
- Number of clusters
- Management Center uptime
- Minimum and maximum cluster sizes
- Minimum and maximum cluster versions
- Total number of members
- Size of the Hazelcast Management Center home directory
- Hash value of Hazelcast Management Center license key
- Environment Information:
  - Name of operating system
  - Version of installed Java

For each user login, we store the following information and send it in a phone home:

- Browser (Chrome, Firefox, IE etc.)
- Browser major version
- Operating system
- Operating system version
- Screen height and width
- Window height and width

Disabling Phone Homes

Set the `hazelcast.mc.phone.home.enabled` system property to false on the Java command line.

Phone Home URL

http://phonehome.hazelcast.com/pingMc

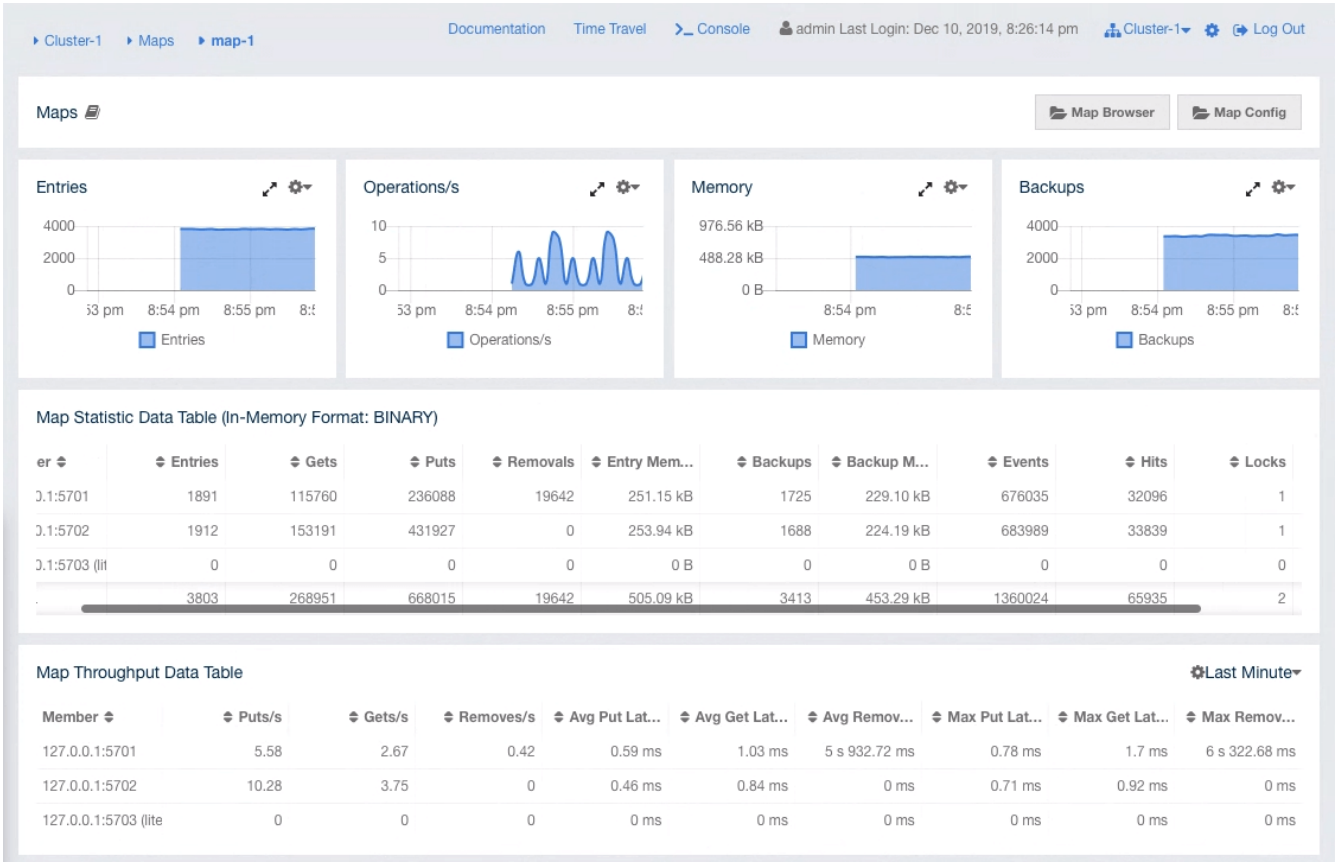
17. Management Center Documentation

To see the Management Center documentation (this Reference Manual), click on the **Documentation** button located at the toolbar. This Management Center manual appears as a tab.

18. Troubleshooting

18.1. Horizontal Scrollbar in Tables when Using macOS

MacOS automatically shows a horizontal scrollbar in the status tables of Management Center, when you scroll through a table content. See below for an example view:



The scrollbar should hide shortly after you stop scrolling. This is the default behavior and it shouldn't cause any inconvenience. However, if you still want to change it, address "Show scroll bars" section of the "Change General preferences on Mac" guide.

Appendix A: Migration Guides

This appendix provides information on compatibility related changes for Hazelcast Management Center releases.

A.1. Hazelcast Management Center 3.12.x

- Default home directory location has been changed from `<user-home>/hazelcast-mancenter-`

`<version>` to `<user-home>/hazelcast-mc`.

- Parameter to change home directory location has been changed from `hazelcast.mancenter.home` to `hazelcast.mc.home`.
- The `UpdateLdapPassword` utility (available via `updateLdapPassword.sh` or `updateLdapPassword.bat` scripts) has been merged into the MC Conf tool (available via the `mc-conf.sh` or `mc-conf.bat` scripts).

A.2. Hazelcast Management Center 3.10.x

- Hazelcast Management Center's default URL has been changed from `localhost:8080/mancenter` to `localhost:8080/hazelcast-mancenter`.
- Default home directory location has been changed from `<user-home>/mancenter-<version>` to `<user-home>/hazelcast-mancenter-<version>`.
- Name of the WAR file has been changed from `mancenter-{version}.war` to `hazelcast-mancenter-{version}.war`.

A.3. Hazelcast Management Center 3.8.x

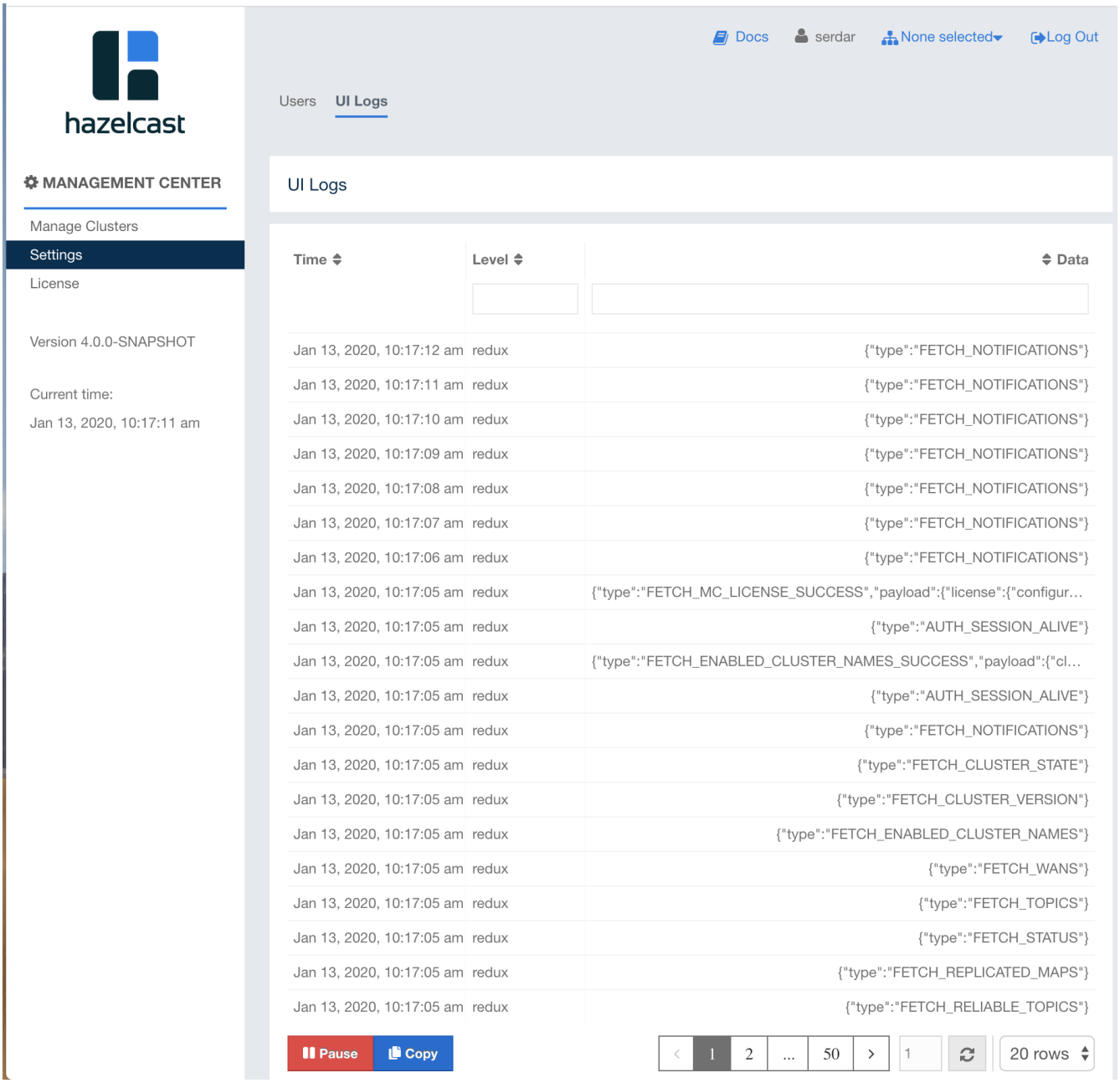
Starting with Management Center 3.8.4, you can use the following system properties for Clustered JMX via Management Center:

- `-Dhazelcast.mc.jmx.rmi.port=9001`
- `-Dhazelcast.mc.jmx.host=localhost`

See the Clustered JMX via Management Center chapter.

Appendix B: User Interface Logs

Using the "UI Logs" page of the "Settings" menu, you can see the log entries related to the Management Center user interface. This page looks like the following:



This is basically useful to make the troubleshooting easier related to the issues in the user interface. You can see the timestamp and type of each log entry.

You can reorder the entries by their timestamps any types, and also filter them by giving a keyword to the `Data` field such as "auth", "map" and "session".

You can pause the log flow using the `Pause` button, copy the log entries to the clipboard using the `Copy` button (so that the entries can be examined in detail) and resume the flow using the `Resume` button on the page.

You may only need to share the information in these logs if requested by the Hazelcast's support team.