MigratoryData Server

Configuration Guide

Version 5.0
March 31, 2023

Copyright Information

Contents

1 Introduction **3**

1.1 Release . . . 3

1.2 Related Documents . . . 3

2 Using Configuration File **4**

3 Using Environment Variables **5**

3.1 MIGRATORYDATA_EXTRA_OPTS . . . 5

3.2 MIGRATORYDATA_JAVA_EXTRA_OPTS . . . 6

3.3 MIGRATORYDATA_JAVA_GC_OPTS . . . 6

3.4 MIGRATORYDATA_JAVA_GC_LOG_OPTS . . . 7

3.5 MIGRATORYDATA_MAX_DESCRIPTORS . . . 7

4 Core Parameters **8**

4.1 LicenseKey . . . 8

4.2 Memory . . . 8

4.3 Listen . . . 9

4.4 ListenEncrypted . . . 10

4.5 KeyStore . . . 10

4.5.1 How to add a certificate to the keystore . . . 11

4.5.1.1 Adding a self-signed certificate of an address to the keystore . . . 12

4.5.1.2 Adding a CA-signed certificate of an address to the keystore . . . 12

4.6 KeyStorePassword . . . 14

4.7 Monitor . . . 15
4.8 MonitorUsername . . . 16
4.9 MonitorPassword . . . 16
4.10 MonitorJMX.Listen . . . 17
4.11 MonitorJMX.Authentication . . . 17
4.12 MonitorJMX.Encryption . . . 18
4.12.1 Secure JMX monitoring over insecure networks . . . 18
4.13 MonitorHTTP.Listen . . . 20
4.13.1 Accessing the HTTP Monitoring Service . . . 20
4.13.1.1 XML and JSON Output Format . . . 21
4.13.1.2 Filters . . . 22
4.13.1.3 Secure HTTP monitoring over insecure networks . . . 24
4.14 MonitorHTTP.Authentication . . . 24
4.15 MonitorHTTP.Encryption . . . 24
4.16 MonitorPUSH.Stats . . . 25
4.17 MonitorPUSH.AccessLog . . . 25
4.18 MonitorPUSH.MessageLog . . . 25
4.19 MonitorPUSH.PublishLog . . . 26
4.20 LogFolder . . . 26
4.21 LogLevel . . . 27
4.22 LogRotateLimit . . . 27
4.23 LogRotateTime . . . 28
4.24 LogRotateFileCount . . . 28
4.25 DocumentRoot . . . 29
4.26 PublishSnapshotMessage . . . 29
4.27 ClusterPassword . . . 30
4.28 ClusterDeliveryMode . . . 30
4.29 ClusterMemberListen . . . 31
4.30 ClusterMembers . . . 31

4.31 ClusterMemberCoordinationFolder . . . 32
4.32 Entitlement . . . 33
4.33 EntitlementAllowToken . . . 33
4.34 RunAsUser . . . 34
4.35 PublishAllowFromAddressList . . . 34

5 Advanced Parameters 35

5.1 ClusterCommunication.NIO . . . 35
5.2 ClusterCommunication.NIO.IoThreads . . . 35
5.3 Native.Ssl . . . 36
5.4 MaxCachedMessagesPerSubject . . . 36
5.5 SnapshotExpireTime . . . 36
5.6 CacheExpireTime . . . 37
5.7 Workgroups . . . 37
5.8 IoThreads . . . 38
5.9 AccessLog . . . 38
5.10 AccessLog.Folder . . . 39
5.11 AccessLog.Compression . . . 39
5.12 AccessLog.RotateLimit . . . 40
5.13 AccessLog.RotateTime . . . 40
5.14 AccessLog.RotateFileCount . . . 41
5.15 MessageLog . . . 41
5.16 MessageLog.Folder . . . 42
5.17 MessageLog.Compression . . . 42
5.18 MessageLog.RotateLimit . . . 43
5.19 MessageLog.RotateTime . . . 43
5.20 MessageLog.RotateFileCount . . . 44
5.21 CacheLog . . . 44
5.22 CacheLog.Folder . . . 45
5.23 CacheLog.Compression . . . 45

5.24 CacheLog.RotateLimit 46
5.25 CacheLog.RotateTime 46
5.26 CacheLog.RotateFileCount 47
5.27 PublishLog 47
5.28 PublishLog.Folder 48
5.29 PublishLog.Compression 48
5.30 PublishLog.RotateLimit 49
5.31 PublishLog.RotateTime 49
5.32 PublishLog.RotateFileCount 50
5.33 PublishLog.SampleUsers 50
5.34 PublishLog.RecordContent 51
5.35 PublishLog.ComputeLatencyStats 51
5.36 PublishLog.LatencyStatsOutputFrequency 51
5.37 PublishLog.LatencyStatsOnly 52
5.38 Stats.LogInterval 52
5.39 MaxBatchingSpace 52
5.40 MaxBatchingTime 53
5.41 CipherListEnabled 53
5.42 CipherListExcluded 54
5.43 SocketBufferLimit 54
5.44 BufferLimit.Send 55
5.45 BufferLimit.Receive 55
5.46 Proxy.Type 56
5.47 Proxy.Host 56
5.48 Proxy.Username 57
5.49 Proxy.Password 57
5.50 Extension.Presence 57
5.51 Extension.Presence.Subject 58
5.52 Extension.Presence.EntitlementToken 58
5.53 MaxBandwidthRate 59

1. Introduction

This guide describes the configuration of MigratoryData Server. It is recommended to read *MigratoryData Architecture Guide* before reading this document for a better understanding of the concepts and to have a more comprehensive background.

1.1 Release

This guide is part of the documentation set for MigratoryData Server version 5.0.

1.2 Related Documents

- *MigratoryData Architecture Guide*
- *MigratoryData Installation Guide*
- *MigratoryData API Developer's Guide* and *Reference Manual* for each client library of MigratoryData Server

2. Using Configuration File

The configuration file of the MigratoryData server has comments and optional parameters besides required parameters. The optional parameters have default values. An optional parameter that is not present in the configuration file will be used with its default value.

The configuration file supports comments. A line that starts with a # character is interpreted as a comment and is ignored. A parameter can be configured with the following syntax:

```
parameter = value
```

The value of a parameter can be defined on multiple lines by postfixing each line with \. For example

```
LicenseKey = aaaabbbbcccc
```

can be written as follows:

```
LicenseKey = aaaa\
             bbbb\
             cccc
```

The configurable parameters are described in Chapter 4 and Chapter 5.

3. Using Environment Variables

You might use one or more of the environment variables defined below to customize various aspects of your MigratoryData deployment. These environment variables should be defined in one of the following files, which is part of the rpm/deb distribution of the MigratoryData server:

Configuration file for extra options	Platform
/etc/default/migratorydata	For deb-based Linux
/etc/sysconfig/migratorydata	For rpm-based Linux

3.1 MIGRATORYDATA_EXTRA_OPTS

Every configurable parameter of the MigratoryData server (see Chapter 4 and Chapter 5) can be either configured in the configuration file as described in Chapter 2, or specified as an extra option using the MIGRATORYDATA_EXTRA_OPTS environment variable using the following syntax:

```
-Dparameter=value
```

The value of a parameter should be defined without spaces and quotes. For example, use -DMemory=512MB, rather than -DMemory='512 MB', or use -DLogFolder=/tmp/somefile, rather than -DLogFolder="/tmp/some file".

Also, the value cannot be defined on multiple lines. For example, if the LicenseKey parameter is configured in your configuration file on multiple lines as follows:

```
LicenseKey = aaaa\
             bbbb\
             cccc
```

then, you could redefine it as an extra option as follows:

```
-DLicenseKey=aaaabbbbcccc
```

A parameter defined using an extra option will override the value of that parameter defined in the configuration file. This is typically the goal of the extra options. For example, an extra option could be used to temporarily test a feature of your deployment without modifying the configuration file. Also, extra options are useful when running the MigratoryData server into a container. In this case, the default configuration file cannot be typically edited inside the container, so customization is done using extra options.

MIGRATORYDATA_EXTRA_OPTS	
Description	Specifies various extra options for the MigratoryData server which can be used to define or override parameters of the configuration file
Default value	""
Required parameter	Optional

For example, to define JMX monitoring on the port 3000 and without using authentication, or to alter the JMX monitoring configured using the configuration file, you could configure the `MIGRATORYDATA_EXTRA_OPTS` environment variable as follows:

```
MIGRATORYDATA_EXTRA_OPTS='-DMonitor=JMX -DMonitorJMX.Listen=*:3000 -DMonitorJMX.Authentication=false'
```

3.2 MIGRATORYDATA_JAVA_EXTRA_OPTS

Use this environment variable to define various Java options.

MIGRATORYDATA_JAVA_EXTRA_OPTS	
Description	Specifies various extra options for Java
Default value	""
Required parameter	Optional

For most cases, you don't need to configure this environment variable. However, there are situations when adding an extra option for Java could be necessary. For example, if you run the MigratoryData server into a docker container or AWS EC2 machine, then, in order to be able to access the JMX monitoring using the `jconsole` tool, you will need to provide extra options for Java as follows:

```
MIGRATORYDATA_JAVA_EXTRA_OPTS='-Djava.net.preferIPv4Stack=true -Djava.rmi.server.hostname=yourhostname'
```

where `yourhostname` is the DNS name of the machine hosting the docker container, or the public DNS name of the AWS EC2 machine.

3.3 MIGRATORYDATA_JAVA_GC_OPTS

Use this environment variable to define the Java options related to Garbage Collection. These options should not include the Java options for Garbage Collection logging which should be provided using the environment variable `MIGRATORYDATA_JAVA_GC_LOG_OPTS`.

MIGRATORYDATA_JAVA_GC_OPTS	
Description	Specifies various options for Java Garbage Collection.
Default value	`"-verbosegc -XX:+UseParallelOldGC"`
Required parameter	Optional

For most cases, it is sufficient to use the default value of this environment variable.

3.4 MIGRATORYDATA_JAVA_GC_LOG_OPTS

Use this environment variable to define the Java options related to Garbage Collection logging.

MIGRATORYDATA_JAVA_GC_LOG_OPTS	
Description	Specifies options for Java Garbage Collection logs
Default value	`"-XX:+PrintGCDetails -XX:+PrintGCTimeStamps -XX:+PrintGCDateStamps -XX:+UseGCLogFileRotation -XX:GCLogFileSize=10M -XX:NumberOfGCLogFiles=5 -Xloggc:YourLogFolder/all/gc-%t.log"` for Java 8 `"-Xlog:gc*:file=YourLogFolder/all/gc-%t.log:time, uptimemillis:filecount=5,filesize=10000"` for Java 9+
Required parameter	Optional

For most cases, it is sufficient to use the default value of this environment variable.

3.5 MIGRATORYDATA_MAX_DESCRIPTORS

Use this environment variable to define the maximum number of socket descriptors to be used by the maximum number of concurrent users connecting to the MigratoryData server.

MIGRATORYDATA_MAX_DESCRIPTORS	
Description	Specifies the maximum number of socket descriptors.
Default value	1000000
Required parameter	Optional

For most cases, it is sufficient to use the default value of this environment variable.

4. Core Parameters

The basic parameters of MigratoryData Server are described below.

4.1 LicenseKey

LicenseKey	
Description	A string representing the license key
Default value	No default value
Required parameter	Required

The license key consists of a sequence of numbers and letters. License keys are provided by MigratoryData to customers either for evaluation, development, or production usage of MigratoryData Server. To obtain a valid license key, contact MigratoryData at `support@migratorydata.com`

4.2 Memory

Memory	
Description	The maximum memory for the Java Virtual Machine (JVM) process running MigratoryData Server. The memory can be also expressed in gigabytes, megabytes, and kilobytes by using the following postfixes: GB for gigabytes, MB for megabytes, or KB for kilobytes. For example to allocate 512 megabytes for MigratoryData Server, the parameter `Memory` should be configured as follows: `Memory = 512 MB`
Default value	No default value
Required parameter	Required

In a production environment is is recommended to use at least 8192 MB memory space or more depending on the load of data on the server and how much simultaneous clients will be connected. Use *MigratoryData Benchmark Kit* to estimate the memory required for your use case.

4.3 Listen

Listen	
Description	A comma separated list of addresses (IP address or DNS domain name and its corresponding port) to listen for client connections
Default value	No default value
Required parameter	Optional

The format of the addresses is `"IP:Port"` (e.g. `192.168.1.1:80`) or `"Name:Port"` (e.g. `push.example.com:80`). IPv6 addresses must be enclosed in square brackets (e.g. `[2001:db8::a00:20ff:fea7:ccea]:80`).

If you specify an address without a port, the default port `80` will be used.

By specifying an IP address, MigratoryData Server will bind only to that IP address. The wildcard address (`*`) will enable MigratoryData Server to bind to all available IP addresses of the machine.

If MigratoryData Server is installed on a machine that has a firewall enabled, then you will need to allow in firewall the access to the addresses and ports configured by this parameter.

> **Note** — For web applications, in a production environment, MigratoryData Server requires a standard web server – such as Apache – to serve the static pages, the images, and the server scripting of the web application. The DNS name of the address provided by this `Listen` parameter must share the same sub-domain with the DNS name used to access the standard web server.
>
> For example, if your Apache web server is accessible at `www.example.com`, then an example of valid configuration for MigratoryData Server will be:
>
> ```
> Listen = push.example.com:80.
> ```
>
> but the following configuration will not be valid:
>
> ```
> Listen = push.example.org:80.
> ```
>
> because the sub-domain `example.com` of Apache is different from `example.org`, which is the sub-domain configured for MigratoryData Server in this example.

4.4 ListenEncrypted

ListenEncrypted	
Description	A comma separated list of addresses (IP address or DNS domain name and its corresponding port) to listen for encrypted client connections
Default value	No default value
Required parameter	Optional

The same conventions applies as for the `Listen` parameter described in Section 4.3, except the fact that if you specify an address without a port, the default port 443 will be used.

> **Note** — In a production deployment, it is recommend to use HTTPS encrypted client connections. In addition to the fact that data will be securely delivered to clients, there is another important advantage. Some security solutions consider HTTP streaming as a non-standard HTTP access and may block streaming to the client having installed such a security solution. By deploying MigratoryData Server with HTTPS encrypted client connections, the security solutions cannot inspect the content of the HTTP messages so that streaming cannot be blocked.

4.5 KeyStore

KeyStore	
Description	The file containing the entries with SSL certificates for the encrypted connections
Default value	No default value
Required parameter	Required if at least one of the following parameters are configured: • `ListenEncrypted` • `MonitorJMX.Encryption` is set on `true` • `MonitorHTTP.Encryption` is set on `true`

The keystore file must be configured using absolute paths. A value example for this parameter is as follows:

Example of KeyStore	Platform
`KeyStore = /some/path/mykeystore.jks`	For Linux/Unix
`KeyStore = C:/some/path/mykeystore.jks`	For Windows

The keystore must contain a SSL certificate for each address used in the configuration of the following parameters:

`ListenEncrypted`	
`MonitorJMX.Listen`	provided that `MonitorJMX.Encryption` is set on `true`
`MonitorHTTP.Listen`	provided that `MonitorHTTP.Encryption` is set on `true`

Table 4.1: Parameters Used to Configure Encrypted Connections

4.5.1 How to add a certificate to the keystore

Suppose the following DNS address `push.example.com` resolves to the IP address `192.168.1.1`, and vice-versa, the IP address `192.168.1.1` resolves to the DNS address `push.example.com`.

If an address appears in the configuration of any of the parameters used to define encrypted connections (see Table 4.1), then the keystore file must contain an SSL certificate for that address. If the address is specified by its DNS name, say `push.example.com`, then its certificate entry in the keystore must have as alias the string `push.example.com`. If the address is specified by its IP address, say `192.168.1.1`, then its certificate entry in the keystore must have as alias the string `192.168.1.1`.

To create a certificate for an address in the keystore, there are two possibilities:

1. Use a self-signed certificate for that address
2. Use a certificate signed by a Certificate Authority (CA)

> **Tip** — Self-signed certificates can be used in development. In a production environment always use certificates signed by a Certificate Authority.

In the next two subsections we suppose the keystore file is named `mykeystore.jks` and the address for which a certificate should be added in the keystore is specified either by DNS name as `push.example.com` or by IP address as `192.168.1.1`.

4.5.1.1 Adding a self-signed certificate of an address to the keystore

Enter one of following commands on a single line without line breaks depending on how was specified the address in the configuration file:

Address is used in configuration
by DNS name as `push.example.com`
	`keytool`
	`-genkeypair`
	`-dname "CN=`*`push.example.com`*`"`
	`-alias` *`push.example.com`*
	`-keyalg RSA`
	`-validity 3600`
	`-keystore` *`mykeystore.jks`*

Address is used in configuration
by IP address as `192.168.1.1`
	`keytool`
	`-genkeypair`
	`-dname "CN=`*`192.168.1.1`*`"`
	`-alias` *`192.168.1.1`*
	`-keyalg RSA`
	`-validity 3600`
	`-keystore` *`mykeystore.jks`*

Address is used in configuration
for encrypted JMX monitoring
	`keytool`
	`-genkeypair`
	`-dname "CN=`**`jmx`**`"`
	`-alias` **`jmx`**
	`-keyalg RSA`
	`-validity 3600`
	`-keystore` *`mykeystore.jks`*

You will be asked to set a password for the keystore if the file `mykeystore.jks` does not exist, or to enter the keystore password if the keystore file already exists and contains other certificate entries. This password must be used to configure the parameter `KeyStorePassword` (see Section 4.6).

4.5.1.2 Adding a CA-signed certificate of an address to the keystore

Suppose you obtained the following signed certificate file `push.example.com.crt` from a Certificate Authority for the domain `push.example.com` and suppose `push.example.com.key` is the file containing its corresponding private key.

In order to add this CA-signed certificate to the keystore, follow these steps:

1. If the Certificate Authority provides you an intermediate certificate in addition to the signed certificate `push.example.com.crt`, then you will have to chain your signed certificate with the intermediary certificate. If you didn't receive an intermediary certificate, just skip this step.

   To chain your signed certificate `push.example.com.crt` with the intermediary certificate, you should first append to your signed certificate the intermediary certificate (say `intermediary.crt`), and then append the certificate signing request sent by you to the CA authority (say `push.example.com.csr`). On a Unix-like operating system, use the following command:

   ```
   cat intermediary.crt push.example.com.csr >> push.example.com.crt
   ```

   On Windows, use the command:

   ```
   type intermediary.crt push.example.com.csr >> push.example.com.crt
   ```

2. Convert the CA-signed certificate to the PKCS#12 standard by entering the following command on a single line without line breaks:

   ```
   openssl
      pkcs12
      -export
      -in push.example.com.crt
      -inkey push.example.com.key
      -out push.example.com.pkcs12
   ```

   This will produce the file *push.example.com.pkcs12*; the password for the this new file must be the same like that used for the keystore.

3. Add the pkcs#12 certificate obtained in the previous step to the keystore by entering the following command on a single line without line breaks:

   ```
   keytool
      -importkeystore
      -srckeystore push.example.com.pkcs12
      -srcstoretype PKCS12
      -deststoretype JKS
      -destkeystore mykeystore.jks
   ```

   This will insert a new entry in the keystore file `mykeystore.jks` with the default alias **1**.

4. Rename the default certificate alias **1** in the keystore by entering one of following commands on a single line without line breaks depending on how was specified the address in the configuration file:

Address is used in configuration by DNS name as `push.example.com`	`keytool`
	`-v`
	`-keystore` *mykeystore.jks*
	`-changealias`
	`-alias 1`
	`-destalias` *push.example.com*

Address is used in configuration by IP address as `192.168.1.1`	`keytool`
	`-v`
	`-keystore` *mykeystore.jks*
	`-changealias`
	`-alias 1`
	`-destalias` *192.168.1.1*

Address is used in configuration for encrypted JMX monitoring	`keytool`
	`-v`
	`-keystore` *mykeystore.jks*
	`-changealias`
	`-alias 1`
	`-destalias` **jmx**

4.6 KeyStorePassword

KeyStorePassword	
Description	The password to access the keystore
Default value	No default value
Required parameter	Required if the parameter `KeyStore` is configured

Configure this parameter with the password used when you created the keystore file. The keystore file is created when adding the first certificate entry to it as explained in Section 4.5.1.

4.7 Monitor

Monitor	
Description	Monitoring type (possible values: JMX, HTTP, PUSH)
Default value	No default value
Required parameter	Optional

You can configure JMX monitoring, HTTP monitoring, PUSH monitoring or any combination of them. Use comma separated values to configure all monitoring types as follows:

```
Monitor = JMX, HTTP, PUSH
```

PUSH monitoring consists in obtaining monitoring information in real-time using any MigratoryData Client API by subscribing to certain special subjects named *meta-subjects*. Access to meta-subjects is subject to accessibility, entitlement, and encryption like for any other subjects. Therefore, unlike for JMX and HTTP monitoring, there are no specific parameters for PUSH monitoring concerning accessibility, entitlement, and encryption.

The following meta-subjects are available for PUSH monitoring provided that the parameter `MonitorPUSH.Stats` is set on `true`:

- $/meta/connected\_sessions$ - the number of connected clients
  - $/meta/connected\_sessions\_web$ - the number of connected web clients
  - $/meta/connected\_sessions\_mobile$ - the number of connected mobile clients
  - $/meta/connected\_sessions\_desktop$ - the number of connected desktop clients
- $/meta/session\_connections\_per\_second$ - the number of connections per second
- $/meta/session\_disconnections\_per\_second$ - the number of disconnections per second
- $/meta/in\_publish\_messages\_per\_second$ - the number of incoming messages per second received from publisher clients
- $/meta/out\_publish\_messages\_per\_second$ - the number of outgoing messages per second sent to subscriber clients
- $/meta/in\_bytes\_per\_second$ - the number of incoming bytes per second received from publisher clients
- $/meta/out\_bytes\_per\_second$ - the number of outgoing bytes per second sent to subscriber clients

Also, the following optional meta-subjects exist:

- $/meta/access$ - if the following optional parameter `MonitorPUSH.AccessLog` is set on `true`
- $/meta/message$ - if the following optional parameter `MonitorPUSH.MessageLog` is set on `true`
- $/meta/publish$ - if the following optional parameter `MonitorPUSH.PublishLog` is set on `true`

4.8 MonitorUsername

MonitorUsername	
Description	User name for monitoring access
Default value	admin
Required parameter	Required if at least one of the following parameters is configured: • `MonitorJMX.Authentication` is set on `true` • `MonitorHTTP.Authentication` is set on `true`

4.9 MonitorPassword

MonitorPassword	
Description	Password for monitoring access
Default value	pass
Required parameter	Required if at least one of the following parameters is configured: • `MonitorJMX.Authentication` is set on `true` • `MonitorHTTP.Authentication` is set on `true`

4.10 MonitorJMX.Listen

MonitorJMX.Listen	
Description	Address used to listen for JMX compatible clients
Default value	No default value
Required parameter	Optional

The format of this addresse is `"IP:Port"` (e.g. `192.168.1.1:3000`) or `"Name:Port"` (e.g. `push.example.com:3000`). IPv6 addresses must be enclosed in square brackets (e.g. `[2001:db8::a00:20ff:fea7:ccea]:3000`).

> **Note** — In order to access the JMX monitoring in MigratoryData Server from a remote machine, please check your firewall settings so that the network traffic from the remote machine is allowed to the address configured by this parameter `MonitorJMX.Listen`.

The `jconsole` utility that is freely available with OpenJDK can be used to connect to the JMX monitoring service of MigratoryData Server. Also there are many JMX commercial tools that provide enhanced functionality like dashboards and database persistence that can be used to connect to the JMX monitoring service of MigratoryData Server.

> **Caution** — The `jconsole` utility has a known issue when connecting remotely to a JMX service running on Linux. To avoid this issue, execute the following command on the Linux machine where MigratoryData Server runs:
>
> ```
> hostname -i
> ```
>
> The command above should return the address used in the configuration of the parameter `MonitorJMX.Listen`. If it reports somthing like `127.0.0.1`, `jconsole` would not be able to connect remotely to MigratoryData Server. To fix this issue, edit `/etc/hosts` so that `hostname` resolves to the the address used in `MonitorJMX.Listen`.

4.11 MonitorJMX.Authentication

MonitorJMX.Authentication	
Description	Decide whether to use authentication when connecting to the JMX monitoring service
Default value	No default value
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true` then, in order to access the monitoring via JMX, you will need to use the user name configured with the parameter `MonitorUsername` and the password configured with the parameter `MonitorPassword`.

4.12 MonitorJMX.Encryption

MonitorJMX.Encryption	
Description	Decide whether to use TLS/SSL encryption when connecting to the JMX monitoring service
Default value	No default value
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true`, then a JMX client will connect to MigratoryData Server through a SSL/TLS encrypted connection. The use of an encrypted connection is especially recommended for the JMX remote monitoring from an insecure network, including Internet.

If `MonitorJMX.Encryption` is set on `true`, then the address used in the configuration of the parameter `MonitorJMX.Listen` must have an certificate entry in the keystore file defined by the parameter `KeyStore`. The certificate entry in the keystore must be created with the alias **jmx** as explained in Section 4.5.1.

4.12.1 Secure JMX monitoring over insecure networks

In this subsection it is assumed that the JMX client used to connect to MigratoryData Server is `jconsole`. For other JMX clients you can also use the steps below, but you may have to adapt the configuration to suit your specific JMX client.

Supposing the file name of the keystore defined by `KeyStore` paramater is `mykeystore.jks`, and supposing the keystore `mykeystore.jks` includes an certificate entry for the address used in the configuration of the parameter `MonitorJMX.Listen` having as alias **jmx** as explained in Section 4.5.1. Then, use the following steps to establish a secure JMX connection:

1. Create a truststore for the JMX client. The truststore is a special keystore that can verify the trusted SSL certificates. The trustsore will be used by the JMX client. Supposing that the file name used for the truststore is `mytruststore.jks`, enter the following two commands each on a single line without line breaks to create the truststore:

```
keytool
   -export
   -alias jmx
   -keystore mykeystore.jks
   -rfc
   -file temp.cer

keytool
   -import
   -alias jmx
   -file temp.cer
   -keystore mytruststore.jks
```

For the first command above use the password of the keystore as defined by the parameter `KeyStorePassword`. For the second command above use a new password, say `mytruststore-password`.

2. Generate a new keystore for the JMX client and add to it a certificate entry with the alias **jmx**. To do so, supposing the file name for the new keystore is `clientkeystore.jks`, enter the following command on a single line without line breaks:

```
keytool
   -genkeypair
   -alias jmx
   -keyalg RSA
   -validity 3600
   -keystore clientkeystore.jks
```

For this command use a new password, say `clientkeystore-password`.

3. To securely connect to the JMX monitoring service of MigratoryData Server, enter the following command on a single line without line breaks:

```
jconsole
   -J-Djavax.net.ssl.keyStore=clientkeystore.jks
   -J-Djavax.net.ssl.keyStorePassword=clientkeystore-password
   -J-Djavax.net.ssl.trustStore=mystruststore.jks
   -J-Djavax.net.ssl.trustStorePassword=mytruststore-password
```

4.13 MonitorHTTP.Listen

MonitorHTTP.Listen	
Description	Address used to listen for HTTP monitoring clients
Default value	No default value
Required parameter	Optional

The format of this addresse is `"IP:Port"` (e.g. `192.168.1.1:8808`) or `"Name:Port"` (e.g. `push.example.com:8808`). IPv6 addresses must be enclosed in square brackets (e.g. `[2001:db8::a00:20ff:fea7:ccea]:8808`).

> **Note** — In order to access the HTTP monitoring in MigratoryData Server from a remote machine, please check your firewall settings so that the network traffic from the remote machine is allowed to the address configured by this parameter `MonitorHTTP.Listen`.

4.13.1 Accessing the HTTP Monitoring Service

Supposing you have the following monitoring related configuration:

```
Monitor = HTTP
MonitorUsername = admin
MonitorPassword = pass
MonitorHTTP.Listen = push.example.com:8808
MonitorHTTP.Authentication = true
MonitorHTTP.Encryption = false
```

Then you can monitor MigratoryData Server by opening the following URL:

`http://push.example.com:8808/stats?user=admin&password=pass`

Opening the URL above will produce an output with the following format:

```
<fieldname1>:<value1> <fieldname2>:<value2> ...  <fieldnameN>:<valueN>
```

where each field correspond to one of the following **statistics**:

- Average
- Standard Deviation
- Maximum

applied to one of the following **indicators**:

- The number of connected clients
- The number of subjects
- The number of connections per second
- The number of disconnections per second
- The number of incoming messages per second received from publishers
- The number of outgoing messages per second sent to clients
- The number of incoming bytes per second received from publishers
- The number of outgoing bytes per second sent to clients

for one of the following **period** of time:

- Current
- Since startup
- Last minute, last 5 minutes, and last 15 minutes
- Last hour, last 5 hours, and last 15 hours
- Last day, last 5 days, and last 15 days
- Last month, last 5 months, and last 15 months

4.13.1.1 XML and JSON Output Format

You can also retrieve data in XML and JSON format. Please append a `view` GET parameter to the URL above with the value `xml` or `json`. For example, to retrieve monitoring data in XML format use a URL as follows:

`http://`*push.example.com:8808*`/stats?user=`*user*`&password=`*pass*`&view=xml`

4.13.1.2 Filters

You can filter the monitoring data by *indicator* and/or *statistic* and/or *period* of time. To do so add to the URL above one or more of the GET parameters listed in Table 4.2.

For example to retrieve the maximum number of concurrent clients in the last 15 minutes, use the following URL written on a single line without line breaks:

```
http://push.example.com:8808/stats?
  user=user&
  password=pass&
  indicator=ConnectedSessions&
  statistic=MAX&
  period=Last.15.Minute
```

Another example. To retrieve the average for all indicators for all periods of time available, use a URL as follows:

```
http://push.example.com:8808/stats?user=user&password=pass&statistic=AVG
```

GET Parameter	Possible Value	Description
indicator	ConnectedSessions	The number of connected clients
	NumberOfSubjects	Number of subjects
	InBytesPerSecond	The number of incoming bytes per second received from publishers
	InPublishMessagesPerSecond	The number of incoming messages per second received from publishers
	OutBytesPerSecond	The number of outgoing bytes per second sent to clients
	OutPublishMessagesPerSecond	The number of outgoing messages per second sent to clients
	SessionConnectionsPerSecond	The number of connections per second
	SessionDisconnectionsPerSecond	The number of disconnections per second
statistic	AVG	Average
	STDEV	Standard Deviation
	MAX	Maximum
period	Current	Current
	Last.1.Minute	Last 1 minute
	Last.5.Minute	Last 5 minutes
	Last.15.Minute	Last 15 minutes
	Last.1.Hour	Last 1 hour
	Last.5.Hour	Last 5 hours
	Last.15.Hour	Last 15 hours
	Last.1.Day	Last 1 day
	Last.5.Day	Last 5 days
	Last.15.Day	Last 15 days
	Last.1.Month	Last 1 month
	Last.5.Month	Last 5 months
	Last.15.Month	Last 15 months
	SinceStartup	Since startup

Table 4.2: Filters

4.13.1.3 Secure HTTP monitoring over insecure networks

To securely monitor MigratoryData Server over an insecure network such as Internet, configure

```
MonitorHTTP.Encryption = true
```

See Section 4.15 for more details. Then use a URL as follows:

`https://`*push.example.com:8808*`/stats?user=`*user*`&password=`*pass*

4.14 MonitorHTTP.Authentication

MonitorHTTP.Authentication	
Description	Decide whether to use authentication when connecting to the HTTP monitoring service
Default value	No default value
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true` then, in order to access the monitoring via HTTP, you will need to use the user name configured with the parameter `MonitorUsername` and the password configured with the parameter `MonitorPassword`.

4.15 MonitorHTTP.Encryption

MonitorHTTP.Encryption	
Description	Decide whether to use SSL/TLS encryption when connecting to the HTTP monitoring service
Default value	No default value
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true`, then a HTTP monitoring client will connect to MigratoryData Server through a SSL/TLS encrypted connection. The use of an encrypted connection is especially recommended for the HTTP remote monitoring from an insecure network, including Internet.

If `MonitorHTTP.Encryption` is set on `true`, then the address used in the configuration of the parameter `MonitorHTTP.Listen` must have an certificate entry in the keystore file defined by the parameter `KeyStore`. See Section 4.5.1 to lean how to add a certificate to the keystore.

4.16 MonitorPUSH.Stats

MonitorPUSH.Stats	
Description	Decide whether to provide stats logs via a meta-subject
Default value	false
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true`, then a PUSH monitoring client will be able to subscribe to the meta-subjects listed above in Section 4.7 to get the stats logs of the MigratoryData server.

4.17 MonitorPUSH.AccessLog

MonitorPUSH.AccessLog	
Description	Decide whether to provide access logs via a meta-subject
Default value	false
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true`, then a PUSH monitoring client will be able to subscribe to the meta-subject $/meta/access$ to get the access logs of the MigratoryData server.

4.18 MonitorPUSH.MessageLog

MonitorPUSH.MessageLog	
Description	Decide whether to provide message logs via a meta-subject
Default value	false
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true`, then a PUSH monitoring client will be able to subscribe to the meta-subject $/meta/message$ to get the message logs of the MigratoryData server.

4.19 MonitorPUSH.PublishLog

MonitorPUSH.PublishLog	
Description	Decide whether to provide publish logs via a meta-subject
Default value	false
Required parameter	Optional

This parameter can have two values: `true` or `false`. If set on `true`, then a PUSH monitoring client will be able to subscribe to the meta-subject $/meta/publish$ to get the publish logs of the MigratoryData server.

4.20 LogFolder

LogFolder	
Description	The folder where the logs will be written
Default value	`logs`
Required parameter	Optional

If not configured, MigratoryData Server will use the default folder `logs` relative to the directory path used to start MigratoryData Server.

The log folder can be configured using absolute paths. A value example for this parameter is as follows:

Example of log folder	Platform
`LogFolder = /some/path/mylogs`	For Linux/Unix
`LogFolder = C:/some/path/mylogs`	For Windows

4.21 LogLevel

LogLevel	
Description	The level of verbosity of the messages recorded in the logs
Default value	INFO
Required parameter	Optional

The following levels are available:

- `TRACE` (most verbose)
- `DEBUG`
- `INFO` (recommended for production use)
- `WARN`
- `ERROR` (least verbose)

4.22 LogRotateLimit

LogRotateLimit	
Description	The maximum capacity of a log file expressed in kilobytes (KB), megabytes (MB), or gigabytes (GB)
Default value	10 MB
Required parameter	Optional

If the log file reaches the capacity provided by this parameter, then MigratoryData Server will automatically create a new log file. The previous log files are preserved on disk up to the number of log files defined by the parameter `LogRotateFileCount`.

4.23 LogRotateTime

LogRotateTime	
Description	The time interval at which a new log file will be created expressed in minutes (m), hours (h), days (D), weeks (W), months (M), or years (Y).
Default value	No default value
Required parameter	Optional

For example, in order to record the logs in a separate file every day use:

```
LogRotateTime = 1 D
```

To record the logs in separate file every 4 hours use:

```
LogRotateTime = 4 h
```

The previous log files are preserved on disk up to the number of log files defined by the parameter `LogRotateFileCount`.

> **Note** — This parameter takes precedence over the parameter `LogRotateLimit`. Therefore, if the parameter `LogRotateTime` is configured, then the configuration of the parameter `LogRotateLimit` is ignored.

4.24 LogRotateFileCount

LogRotateFileCount	
Description	Limit the number of historical log files created by log rotation
Default value	100
Required parameter	Optional

If the number of log files produced by log rotation defined by the parameters `LogRotateTime` or `LogRotateLimit` reaches the value of this parameter, then the oldest log file is removed whenever a new log file is created such that the total number of logs files will not exceed the value of this parameter.

4.25 DocumentRoot

DocumentRoot	
Description	The folder from which MigratoryData Server will serve files
Default value	`html`
Required parameter	Optional

This parameter is optional, if not supplied the default folder `html` relative to the directory path used to start MigratoryData Server is used.

> **Note** — In a production environment, the parameter `DocumentRoot` should be disabled.
>
> In a production environment, the files of a web application built with MigratoryData APIs should be installed in a standard web server such as Apache. MigratoryData Server has a limited capability to serve web pages, which is only offered to facilitate the development process. The purpose of MigratoryData Server is to push real-time data. Thus, use a web server to provide any static resources necessary for a website such as images, text, and server-side scripting.

4.26 PublishSnapshotMessage

PublishSnapshotMessage	
Description	Specify whether to publish the initial snapshot message or not
Default value	true
Required parameter	Optional

When a client subscribes to a subject, the MigratoryData server will first publish the snapshot message for that subject, and subsequently it will publish any new message for that subject to the client.

See *MigratoryData Architecture Guide*, chapter *Concepts* to learn more about the "Snaphot Message" notion.

4.27 ClusterPassword

ClusterPassword	
Description	Each MigratoryData server in a cluster communicates with the other MigratoryData servers in the cluster, including with itself. This parameter defines the password used by each cluster member for connecting to the other cluster members, including to itself.
Default value	No default value
Required parameter	Required

This password is used only by the MigratoryData servers that form a cluster to connect each other. This password is not used for clients. In order to allow or deny subscriptions and publications for clients, you can enable the entitlement service via the parameter `Entitlement`.

4.28 ClusterDeliveryMode

ClusterDeliveryMode	
Description	Specify whether you want to use Stanadard Message Delivery or Guaranteed Message Delivery
Default value	Standard
Required parameter	Optional

Define the quality-of-service level for message delivery that your cluster of MigratoryData servers will use. The possible values are: "Standard" (Standard Message Delivery) and "Guaranteed" (Guaranteed Message Delivery)

Using Standard Message Delivery, in the case of a failover reconnection, the client will get the latest (most recent) message available at the moment of the reconnection for each of its subscribed subjects.

Using Guaranteed Message Delivery, in the case of a failover reconnection, the client will get not only the latest (most recent) message for each of its subscribed subjects, but it will also get all the potential messages published during the failover period for each of its subscribed subjects.

See *MigratoryData Architecture Guide*, chapter *Guaranteed Message Delivery*, for a complete discussion.

4.29 ClusterMemberListen

ClusterMemberListen	
Description	Each MigratoryData server in a cluster communicates with the other MigratoryData servers in the cluster, including with itself. This parameter represents the address used to listen for incoming connections from the other cluster members, including from itself.
Default value	No default value
Required parameter	Required

The format of this listen address is: "IP_Address:Port" (e.g. 192.168.0.1:8801) or "DNS_Name:Port" (e.g. push.example.com:8801) IPv6 addresses must be enclosed in square brackets. For example [2001:db8::a00:20ff:fea7:ccea]:8801.

> **Note** – The port defined by this parameter must be allowed by the firewall for incoming connections from all MigratoryData servers of the cluster.
>
> In addition, the four consecutive port numbers starting with the port number defined by this parameter must be allowed by the firewall for incoming connections from all MigratoryData servers of the cluster.
>
> For example, supposing that this parameter is configured as follows:
>
> ClusterMemberListen = push.example.com:8801
>
> Then the ports 8801, 8802, 8803, 8804, 8805 must be allowed by the firewall for the internal cluster communication between its members.

4.30 ClusterMembers

ClusterMembers	
Description	Define the cluster by specifying its members.
Default value	No default value
Required parameter	Required

Each cluster member is specified by the listen address defined by its parameter `ClusterMemberListen`.

For developing and testing purposes, you can deploy a cluster of MigratoryData servers where its members run on the same machine. For example, to define a cluster of three MigratoryData servers all running on the same machine, use

```
ClusterMembers = 192.168.1.1:7701, 192.168.1.1:8801, 192.168.1.1:9901
```

For production purposes, you should deploy a cluster of MigratoryData servers where each member runs on a different machine. For example, to define a production cluster of three MigratoryData servers, use something like:

```
ClusterMembers = 192.168.1.1:8801, 192.168.1.2:8801, 192.168.1.3:8801
```

> **Note** — The cluster definition must be **identical** in all cluster members (the order of the listen addresses must be preserved in the configuration of each cluster member)

4.31 ClusterMemberCoordinationFolder

ClusterMemberCoordinationFolder	
Description	A folder used to store internal information useful for cluster coordination
Default value	`coordination_logs`
Required parameter	Optional

If not configured, MigratoryData Server will use the default folder `coordination_logs` relative to the directory path used to start the MigratoryData server.

The coordination log folder can be configured using absolute paths. A value example for this parameter is as follows:

Example of transactions folder	Platform
`ClusterMemberCoordinationFolder = /some/path/coordination_logs`	For Linux/Unix
`ClusterMemberCoordinationFolder = C:/some/path/coordination_logs`	For Windows

> **Note** — It is recommended to configure this folder on a local disk instead of a network-attached disk

4.32 Entitlement

Entitlement	
Description	Entitlement type. The possible values are `Basic` and `Custom`
Default value	`Basic`
Required parameter	Optional

- **Basic entitlement** allows any client to subscribe to any subject, however, publication is allowed only from the clients which authenticate with the token defined by the parameter `EntitlementAllowToken`.

  Also, if `PUSH` monitoring is enabled via the parameter `Monitor`, subscription to the monitoring meta-subjects is allowed only from the clients which authenticate with the token defined by the parameter `EntitlementAllowToken`.

- **Custom entitlement** allows you to define your own entitlement rules using a simple extension API. To learn how to build and deploy an extension for MigratoryData Server, please refer to the documentation of MigratoryData Extension API.

See *MigratoryData Architecture Guide*, Chapter *Entitlement*, to learn more about the *Entitlement* feature.

4.33 EntitlementAllowToken

EntitlementAllowToken	
Description	If the Entitlement type is Basic, then only the clients which use the token defined by this parameter are allowed to publish messages
Default value	No default value
Required parameter	Required if the Entitlement type is Basic

If the Entitlement type is Custom, then this parameter is ignored, and the entitlement is done according to your entitlement rules defined by your extension built with MigratoryData Extension API.

4.34 RunAsUser

RunAsUser	
Description	Configure MigratoryData Server to run as an unprivileged (normal) user.
Default value	No default value
Required parameter	Optional

Note — This parameter should be configured only for the Linux platforms.

Supposing "migratorydata" is an existing normal user. Configure MigratoryData Server as follows:

```
RunAsUser = migratorydata
```

Now, start the push server as root (this is necessary to be able to bind on the privileged ports 80 or 443). Please note that while running as root, MigratoryData Server will not accept any client connections. Then, MigratoryData Server will drop the root privileges (using the system call setuid) and will automatically switch to the normal user "migratorydata". Only at this time, MigratoryData Server will start to accept client connections.

4.35 PublishAllowFromAddressList

PublishAllowFromAddressList	
Description	Define the list of IP addresses allowed for message publication.
Default value	No default value
Required parameter	Optional

If this parameter is configured, then the MigratoryData server will accept message publications only from clients running on any of the IP addresses defined by this parameter.

If this parameter is not set, message publication will be allowed from any client provided however that the client is allowed by the entitlement rules you define (see parameter `Entitlement`).

Note — Use only dotted-decimal notation for the IP addresses (no DNS names) to specify the list of IP addresses. For example:

```
PublishAllowFromAddressList = 192.168.1.100, 192.168.1.101
```

5. Advanced Parameters

The advanced parameters of MigratoryData Server are described below.

5.1 ClusterCommunication.NIO

ClusterCommunication.NIO	
Description	Specify whether you want to use a NIO thread model for inter-cluster communication
Default value	false
Required parameter	Optional

This parameter is strictly related to the communication between cluster members (and not between clients and server).

By configuring this parameter on `true`, the MigratoryData server will use non-blocking IO (NIO) operations for inter-cluster communication as well as a simplified thread model (with a configurable number of threads, see the parameter `ClusterCommunication.NIO.IoThreads`).

Both communications methods (NIO and standard IO) are implemented to provide high performance. Therefore, we recommend to use NIO only if your cluster size is larger than 10 servers.

5.2 ClusterCommunication.NIO.IoThreads

ClusterCommunication.NIO.IoThreads	
Description	Define the number of threads to be used for NIO inter-cluster communication
Default value	2
Required parameter	Optional

This parameter applies only if the parameter `ClusterCommunication.NIO` is set on `true`.

5.3 Native.Ssl

Native.Ssl	
Description	Use BoringSSL instead of Java SSL implementation for encrypted connections
Default value	false
Required parameter	Optional

The requirements for this option is to use Linux. Use this option to optimize CPU and memory usage.

5.4 MaxCachedMessagesPerSubject

MaxCachedMessagesPerSubject	
Description	The number of the most recent received messages from publishers to be stored for each subject
Default value	1000
Required parameter	Optional

The number of messages to be stored in memory for each subject.

> **Note** — This parameter applies only if the *Guaranteed Message Delivery* feature is enabled (see parameter `ClusterDeliveryMode`).

5.5 SnapshotExpireTime

SnapshotExpireTime	
Description	The number of seconds to persist a snapshot message
Default value	0
Required parameter	Optional

The MigratoryData server saves a snapshot message in memory for every subject. By default, the snapshot message is persisted in memory for an indefinite period. However, you can specify a limit on the retention time of the snapshot messages using this parameter.

This feature is especially helpful when new subjects are continuously added to the system, but they are not updated after a certain time. Without a time limit, the snapshot messages of these inactive subjects would accumulate in memory, leading to potential memory issues.

> **Note** — The minimum non-null value of this parameter must be higher than the value of the parameter `CacheExpireTime` which defaults to 180 seconds.

See *MigratoryData Architecture Guide*, chapter *Concepts* to learn more about the "Snaphot Message" notion.

5.6 CacheExpireTime

CacheExpireTime	
Description	The number of seconds a message is held in the cache before it expires
Default value	180
Required parameter	Optional

Messages are removed continuously from the cache of each subject, however each message is held in the cache at least the number of seconds defined by this parameter.

> **Note** — This parameter applies only if the *Guaranteed Message Delivery* feature is enabled (see parameter `ClusterDeliveryMode`).

5.7 Workgroups

Workgroups	
Description	The number of groups of clients
Default value	The number of total CPU cores available
Required parameter	optional

In order to better scale on multiprocessor servers the incoming users are separated in groups. This parameter configures the number of groups (every group has a dedicated thread). If not supplied the total CPU cores available is the default value. In most situations it is not recommended to modify the default value.

5.8 IoThreads

IoThreads	
Description	The number of threads used for I/O processing
Default value	The number of total CPU cores available
Required parameter	Optional

If not supplied the number of total CPU cores available is the default value. In most situations it is not recommended to modify the default value.

5.9 AccessLog

AccessLog	
Description	Specify whether to record access logs or not
Default value	false
Required parameter	Optional

The access log records various information about requests to the MigratoryData server, including:

- Date of the access
- Session identifier
- Network address of the client which makes the request
- The requested operation
- Entitlement token used for authentication
- Type of access (normal, recovery, or history)
- Client type (web, mobile, or desktop) and details about its environment (e.g. the browser used etc)

5.10 AccessLog.Folder

AccessLog.Folder	
Description	The folder where the access logs will be written
Default value	`logs`
Required parameter	Optional

If not configured, and the parameter `AccessLog` is set on `true`, then MigratoryData Server will use the default folder `logs` relative to the directory path used to start MigratoryData Server.

The log folder can be configured using absolute paths. A value example for this parameter is as follows:

Example of access log folder	Platform
`AccessLog.Folder = /some/path/mylogs`	For Linux/Unix
`AccessLog.Folder = C:/some/path/mylogs`	For Windows

5.11 AccessLog.Compression

AccessLog.Compression	
Description	Enable compression of access logs
Default value	false
Required parameter	Optional

In order to reduce the amount of access logs, you can set this parameter on `true` to enable on-the-fly access log compression.

5.12 AccessLog.RotateLimit

AccessLog.RotateLimit	
Description	The maximum capacity of an access log file expressed in kilobytes (KB), megabytes (MB), or gigabytes (GB)
Default value	10 MB
Required parameter	Optional

If the access log file reaches the capacity provided by this parameter, then MigratoryData Server will automatically create a new access log file. The previous access log files are preserved on disk up to the number of access log files defined by the parameter `AccessLog.RotateFileCount`.

5.13 AccessLog.RotateTime

AccessLog.RotateTime	
Description	The time interval at which a new access log file will be created expressed in minutes (m), hours (h), days (D), weeks (W), months (M), or years (Y).
Default value	No default value
Required parameter	Optional

For example, in order to record the access logs in a separate file every day use:

```
AccessLog.RotateTime = 1 D
```

To record the access logs in separate file every 4 hours use:

```
AccessLog.RotateTime = 4 h
```

The previous access log files are preserved on disk up to the number of access log files defined by the parameter `AccessLog.RotateFileCount`.

> **Note** — This parameter takes precedence over the parameter `AccessLog.RotateLimit`. Therefore, if the parameter `AccessLog.RotateTime` is configured, then the configuration of the parameter `AccessLog.RotateLimit` is ignored.

5.14 AccessLog.RotateFileCount

AccessLog.RotateFileCount	
Description	Limit the number of historical access log files created by access log rotation
Default value	100
Required parameter	Optional

If the number of access log files produced by access log rotation defined by the parameters `AccessLog.RotateLimit` or `AccessLog.RotateTime` reaches the value of this parameter, then the oldest access log file is removed whenever a new access log file is created such that the total number of access logs files will not exceed the value of this parameter.

5.15 MessageLog

MessageLog	
Description	Specify whether to record message logs or not
Default value	false
Required parameter	Optional

The message log records the messages received by the MigratoryData server from client for publication. For each message received, the following information are recorded:

- Date when the message is received
- Subject of the message
- Content of the message
- Identification of the message (i.e. sequence number + epoch number)
- Network address of the client which sent the message

The message log can be used as input for MigratoryData Replayer to "replay" the recorded messages.

5.16 MessageLog.Folder

MessageLog.Folder	
Description	The folder where the message logs will be written
Default value	`logs`
Required parameter	Optional

If not configured, and the parameter `MessageLog` is set on `true`, then MigratoryData Server will use the default folder `logs` relative to the directory path used to start MigratoryData Server.

The log folder can be configured using absolute paths. A value example for this parameter is as follows:

Example of message log folder	Platform
`MessageLog.Folder = /some/path/mylogs`	For Linux/Unix
`MessageLog.Folder = C:/some/path/mylogs`	For Windows

5.17 MessageLog.Compression

MessageLog.Compression	
Description	Enable compression of message logs
Default value	false
Required parameter	Optional

In order to reduce the amount of message logs, you can set this parameter on `true` to enable on-the-fly message log compression.

5.18 MessageLog.RotateLimit

MessageLog.RotateLimit	
Description	The maximum capacity of an message log file expressed in kilobytes (KB), megabytes (MB), or gigabytes (GB)
Default value	10 MB
Required parameter	Optional

If the message log file reaches the capacity provided by this parameter, then MigratoryData Server will automatically create a new message log file. The previous message log files are preserved on disk up to the number of message log files defined by the parameter `MessageLog.RotateFileCount`.

5.19 MessageLog.RotateTime

MessageLog.RotateTime	
Description	The time interval at which a new message log file will be created expressed in minutes (m), hours (h), days (D), weeks (W), months (M), or years (Y).
Default value	No default value
Required parameter	Optional

For example, in order to record the message logs in a separate file every day use:

```
MessageLog.RotateTime = 1 D
```

To record the message logs in separate file every 4 hours use:

```
MessageLog.RotateTime = 4 h
```

The previous message log files are preserved on disk up to the number of message log files defined by the parameter `MessageLog.RotateFileCount`.

> **Note** — This parameter takes precedence over the parameter `MessageLog.RotateLimit`. Therefore, if the parameter `MessageLog.RotateTime` is configured, then the configuration of the parameter `MessageLog.RotateLimit` is ignored.

5.20 MessageLog.RotateFileCount

MessageLog.RotateFileCount	
Description	Limit the number of historical message log files created by message log rotation
Default value	100
Required parameter	Optional

If the number of message log files produced by message log rotation defined by the parameters `MessageLog.RotateLimit` or `MessageLog.RotateTime` reaches the value of this parameter, then the oldest message log file is removed whenever a new message log file is created such that the total number of message logs files will not exceed the value of this parameter.

5.21 CacheLog

CacheLog	
Description	Specify whether to record cache logs or not
Default value	false
Required parameter	Optional

If Guaranteed Message Delivery feature is enabled for the MigratoryData cluster, then each cluster member maintains – for each subject – an in-memory cache with the latest messages (for the number of seconds defined by the parameter `CacheExpireTime` which defaults to `180` seconds, limited however to the number of messages defined by the parameter `MaxCachedMessagesPerSubject` which defaults to `1000` most recent messages). Moreover, if there is a failure in the cluster, after recovery the cache is automatically synchronized for each subject and cluster member. In this way, for each subject, each cluster member has the most recent messages for that subject.

This cache can be used to retrieve historical messages using the API method `subscribeWithHistory`. But the main purpose of maintaining a cache is to achieve Guaranteed Message Delivery. For example, if the connection between a client and a cluster member is broken, the client will automatically reconnect to another cluster member. Before resubscribing to its subjects, it will retrieve the messages published during the fail-over time from the cache of the new cluster member.

For every subject, whenever a message is added to the cache of the subject (either during normal operation or cluster recovery), an ADD entry is added to Cache Logs. Whenever a message expires and is removed from the cache, a CLEAN entry is added to Cache Logs. Whenever a

client accesses the cache (either via `subscribeWithHistory` or during fail-over recovery), a GET entry is added to Cache Logs, containing the client identification and the identifiers (i.e. sequence number + epoch number) of the messages provided to that client from the cache.

Cache Logs deal only with client identifiers (IP address + port) and message identifiers (sequence number + epoch number). In order to find an actual client or an actual message, you will have to correlate Cache Logs with Access Logs and Message Logs. So, Cache Logs (correlated with Access Logs and Message Logs) enable to reconstruct the cache of a subject at a given time and how recovery performed for each client. In this way, the Cache Logs will act as a more concrete model where it is much easier to audit the correctness of the system.

5.22 CacheLog.Folder

CacheLog.Folder	
Description	The folder where the cache logs will be written
Default value	`logs`
Required parameter	Optional

If not configured, and the parameter `CacheLog` is set on `true`, then MigratoryData Server will use the default folder `logs` relative to the directory path used to start MigratoryData Server.

The log folder can be configured using absolute paths. A value example for this parameter is as follows:

Example of publish log folder	Platform
`CacheLog.Folder = /some/path/mylogs`	For Linux/Unix
`CacheLog.Folder = C:/some/path/mylogs`	For Windows

5.23 CacheLog.Compression

CacheLog.Compression	
Description	Enable compression of cache logs
Default value	false
Required parameter	Optional

In order to reduce the amount of cache logs, you can set this parameter on `true` to enable on-the-fly cache log compression.

5.24 CacheLog.RotateLimit

CacheLog.RotateLimit	
Description	The maximum capacity of an cache log file expressed in kilobytes (KB), megabytes (MB), or gigabytes (GB)
Default value	10 MB
Required parameter	Optional

If the cache log file reaches the capacity provided by this parameter, then MigratoryData Server will automatically create a new cache log file. The previous cache log files are preserved on disk up to the number of cache log files defined by the parameter `CacheLog.RotateFileCount`.

5.25 CacheLog.RotateTime

CacheLog.RotateTime	
Description	The time interval at which a new cache log file will be created expressed in minutes (m), hours (h), days (D), weeks (W), months (M), or years (Y).
Default value	No default value
Required parameter	Optional

For example, in order to record the cache logs in a separate file every day use:

```
CacheLog.RotateTime = 1 D
```

To record the cache logs in separate file every 4 hours use:

```
CacheLog.RotateTime = 4 h
```

The previous cache log files are preserved on disk up to the number of cache log files defined by the parameter `CacheLog.RotateFileCount`.

Note — This parameter takes precedence over the parameter `CacheLog.RotateLimit`. Therefore, if the parameter `CacheLog.RotateTime` is configured, then the configuration of the parameter `CacheLog.RotateLimit` is ignored.

5.26 CacheLog.RotateFileCount

CacheLog.RotateFileCount	
Description	Limit the number of historical cache log files created by log rotation
Default value	100
Required parameter	Optional

If the number of cache log files produced by log rotation defined by the parameters `CacheLog.RotateLimit` or `CacheLog.RotateTime` reaches the value of this parameter, then the oldest cache log file is removed whenever a new cache log file is created such that the total number of cache logs files will not exceed the value of this parameter.

5.27 PublishLog

PublishLog	
Description	Specify whether to record publish logs or not
Default value	false
Required parameter	Optional

If Guaranteed Message Delivery feature is enabled for the MigratoryData cluster, then publish logs record the messages published to and acknowledged by a number of users defined by `PublishLog.SampleUsers` which are randomly selected from the total number of users.

5.28 PublishLog.Folder

PublishLog.Folder	
Description	The folder where the publish logs will be written
Default value	`logs`
Required parameter	Optional

If not configured, and the parameter `PublishLog` is set on `true`, then MigratoryData Server will use the default folder `logs` relative to the directory path used to start MigratoryData Server.

The log folder can be configured using absolute paths. A value example for this parameter is as follows:

Example of cache log folder	Platform
`PublishLog.Folder = /some/path/mylogs`	For Linux/Unix
`PublishLog.Folder = C:/some/path/mylogs`	For Windows

5.29 PublishLog.Compression

PublishLog.Compression	
Description	Enable compression of publish logs
Default value	false
Required parameter	Optional

In order to reduce the amount of publish logs, you can set this parameter on `true` to enable on-the-fly publish log compression.

5.30 PublishLog.RotateLimit

PublishLog.RotateLimit	
Description	The maximum capacity of a publish log file expressed in kilobytes (KB), megabytes (MB), or gigabytes (GB)
Default value	10 MB
Required parameter	Optional

If the publish log file reaches the capacity provided by this parameter, then MigratoryData Server will automatically create a new publish log file. The previous publish log files are preserved on disk up to the number of publish log files defined by the parameter `PublishLog.RotateFileCount`.

5.31 PublishLog.RotateTime

PublishLog.RotateTime	
Description	The time interval at which a new publish log file will be created expressed in minutes (m), hours (h), days (D), weeks (W), months (M), or years (Y).
Default value	No default value
Required parameter	Optional

For example, in order to record the publish logs in a separate file every day use:

```
PublishLog.RotateTime = 1 D
```

To record the publish logs in separate file every 4 hours use:

```
PublishLog.RotateTime = 4 h
```

The previous publish log files are preserved on disk up to the number of publish log files defined by the parameter `PublishLog.RotateFileCount`.

> **Note** — This parameter takes precedence over the parameter `PublishLog.RotateLimit`. Therefore, if the parameter `PublishLog.RotateTime` is configured, then the configuration of the parameter `PublishLog.RotateLimit` is ignored.

5.32 PublishLog.RotateFileCount

PublishLog.RotateFileCount	
Description	Limit the number of historical publish log files created by log rotation
Default value	100
Required parameter	Optional

If the number of publish log files produced by log rotation defined by the parameters `PublishLog.RotateLimi` or `PublishLog.RotateTime` reaches the value of this parameter, then the oldest publish log file is removed whenever a new publish log file is created such that the total number of publish logs files will not exceed the value of this parameter.

5.33 PublishLog.SampleUsers

PublishLog.SampleUsers	
Description	Define the number of sample users for publish logs
Default value	100
Required parameter	Optional

Define the number of users randomly selected from the total number of users for which to record publish logs and compute latency statistics.

> **Note** — By configuring this parameter on `0`, all users of the system will be used for publish logs. However, recording publish logs for all users and computing latency statistics by asking each user to acknowledge the reception of each message is highly discouraged unless the outgoing throughput of messages published by the server to the users is low enough.

5.34 PublishLog.RecordContent

PublishLog.RecordContent	
Description	Define whether to include the content of the message into the publish logs
Default value	false
Required parameter	Optional

5.35 PublishLog.ComputeLatencyStats

PublishLog.ComputeLatencyStats	
Description	Compute various statistics about the round-trip latency of messages published to the sample users
Default value	false
Required parameter	Optional

5.36 PublishLog.LatencyStatsOutputFrequency

PublishLog.LatencyStatsOutputFrequency	
Description	Define the frequency to output latency stats
Default value	200
Required parameter	Optional

Record in the publish logs various statics about the round-trip latency (i.e. the time necessary for a message to be delivered from the server to a client plus the time necessary to get the reception acknowledgment for the client to the server). The latency stats are computed for all messages published to the sample users, but they are output to the logs only after having processed the number of messages defined by this parameter.

5.37 PublishLog.LatencyStatsOnly

PublishLog.LatencyStatsOnly	
Description	Record only the latency stats without recording the publish logs
Default value	false
Required parameter	Optional

If this parameter is configured on `true`, the MigratoryData server will record only the latency stats and will not record the publish logs.

5.38 Stats.LogInterval

Stats.LogInterval	
Description	Various statistics are logged at time intervals (seconds) defined by this parameter
Default value	60
Required parameter	Optional

The minimum value of this parameter is 5 (seconds).

5.39 MaxBatchingSpace

MaxBatchingSpace	
Description	The maximum size of the batching in bytes
Default value	0
Required parameter	Optional

See *MigratoryData Architecture Guide*, Chapter *Batching*, to understand the concept of *batching* before configuring this parameter. If this parameter is not configured or configured with the default `0` value, then that means batching is disabled.

5.40 MaxBatchingTime

MaxBatchingTime	
Description	The maximum time of the batching in milliseconds
Default value	0
Required parameter	Optional

See *MigratoryData Architecture Guide*, Chapter *Batching*, to understand the concept of *batching* before configuring this parameter. If this parameter is not configured or configured with the default 0 value, then that means batching is disabled.

5.41 CipherListEnabled

CipherListEnabled	
Description	Enable one or more SSL ciphers in addition to the default JVM ciphers.
Default value	No default value
Required parameter	Optional

The JVM supports a number of ciphers as listed in:

https://docs.oracle.com/javase/8/docs/technotes/guides/security/SunProviders.html

Some of these ciphers are enabled by default. Use this parameter to enable one or more supported ciphers not enabled by default.

For example, to enable the ciphers

- `TLS_DHE_RSA_WITH_AES_128_CBC_SHA` and
- `TLS_DHE_DSS_WITH_AES_128_CBC_SHA`

configure this parameter as follows:

```
CipherListEnabled = TLS_DHE_RSA_WITH_AES_128_CBC_SHA, TLS_DHE_DSS_WITH_AES_128_CBC_SHA
```

5.42 CipherListExcluded

CipherListExcluded	
Description	Exclude one or more SSL ciphers from the default JVM ciphers.
Default value	No default value
Required parameter	Optional

The JVM supports a number of ciphers as listed in:

https://docs.oracle.com/javase/8/docs/technotes/guides/security/SunProviders.html

Some of these ciphers are enabled by default. Use this parameter to disable one or more ciphers enabled by default.

For example, to disable the ciphers

- `TLS_DHE_RSA_WITH_AES_128_CBC_SHA` and
- `TLS_DHE_DSS_WITH_AES_128_CBC_SHA`

configure this parameter as follows:

```
CipherListExcluded = TLS_DHE_RSA_WITH_AES_128_CBC_SHA, TLS_DHE_DSS_WITH_AES_128_CBC_SHA
```

5.43 SocketBufferLimit

SocketBufferLimit	
Description	The maximum number of bytes of a client request
Default value	65536 (64 KB)
Required parameter	Optional

The value of this parameter determines the maximum size (in bytes) of a client requests that MigratoryData Server will accept.

5.44 BufferLimit.Send

BufferLimit.Send	
Description	The default initial memory size (in bytes) for outgoing messages
Default value	8192
Required parameter	Optional

The default memory size used by MigratoryData Server to create an outgoing message to be sent to the clients is given by the value of this parameter. If the actual size of the message is larger than this default value, then MigratoryData Server will reallocate the necessary memory size.

The default value of `BufferLimit.Send` is 8192 bytes. You could adjust the value of this parameter based on the common size of your messages to either avoid memory reallocation or optimize memory consumption.

5.45 BufferLimit.Receive

BufferLimit.Receive	
Description	The default initial memory size (in bytes) for incoming messages
Default value	8192
Required parameter	Optional

The default memory size used by MigratoryData Server to create memory space for an incoming message is given by the value of this parameter. If the actual size of the message is larger than this default value, then MigratoryData Server will reallocate the necessary memory size.

The default value of `BufferLimit.Receive` is 8192 bytes. You could adjust the value of this parameter based on the common size of your messages to either avoid memory reallocation or optimize memory consumption.

5.46 Proxy.Type

Proxy.Type	
Description	The type of proxy used to connect to the MigratoryData license servers (supported types are SOCKS or HTTP)
Default value	No default value
Required parameter	Optional

If MigratoryData Server is deployed behind a proxy, please configure your proxy type in order to be able to access the MigratoryData licensing servers to validate your evaluation license key. Production license keys do not use remote license verification, thus this parameter should be normally enabled only during the evaluation of MigratoryData Server.

5.47 Proxy.Host

Proxy.Host	
Description	The address of your proxy used to connect to the MigratoryData license servers
Default value	No default value
Required parameter	Optional

If MigratoryData Server is deployed behind a proxy, please configure the address of your proxy in order to be able to access the MigratoryData licensing servers to validate your evaluation license key. Production license keys do not use remote license verification, thus this parameter should be normally enabled only during the evaluation of MigratoryData Server.

The format of the proxy address is "ip_address:port" (e.g. 192.168.0.1:3128) or "dns_name:port" (e.g. push.example.com:3128).

5.48 Proxy.Username

Proxy.Username	
Description	The username to authenticate into your proxy to connect to the MigratoryData license servers
Default value	No default value
Required parameter	Optional

If MigratoryData Server is deployed behind a proxy, please configure the user name to authenticate into the proxy in order to be able to access the MigratoryData licensing servers to validate your evaluation license key. Production license keys do not use remote license verification, thus this parameter should be normally enabled only during the evaluation of MigratoryData Server.

5.49 Proxy.Password

Proxy.Password	
Description	The password to authenticate into your proxy to connect to the MigratoryData license servers
Default value	No default value
Required parameter	Optional

If MigratoryData Server is deployed behind a proxy, please configure the password to authenticate into the proxy in order to be able to access the MigratoryData licensing servers to validate your evaluation license key. Production license keys do not use remote license verification, thus this parameter should be normally enabled only during the evaluation of MigratoryData Server.

5.50 Extension.Presence

Extension.Presence	
Description	Specify whether to enable presence extension or not
Default value	false
Required parameter	Optional

To enable a presence extension built with *MigratoryData Presence API* and deployed under the folder `extensions` of your MigratoryData installation. See *Developer's Guide* and *Reference Manual* of *MigratoryData Presence API* to learn how to build a presence extension.

5.51 Extension.Presence.Subject

Extension.Presence.Subject	
Description	Subject used to replicate presence updates across the cluster
Default value	`/__migratorydata__/presence`
Required parameter	Optional

Whenever a user connects to or disconnects from a cluster member, a presence update is provided to the presence extension of that cluster member. Moreover, this presence update is replicated across the cluster such that the presence update to be provided to the presence extension of each cluster member.

Internally, MigratoryData uses its client library to replicate presence updates across the cluster. Each cluster member subscribes to the subject defined by this parameter. Whenever a user connects to or disconnects from a cluster member, that cluster member publishes a presence update on the subject defined by this parameter.

5.52 Extension.Presence.EntitlementToken

Extension.Presence.EntitlementToken	
Description	Entitlement token used to replicate presence updates across the cluster
Default value	the value of the parameter `EntitlementAllowToken`
Required parameter	Optional

As detailed in Section 5.51 above, MigratoryData uses its standard messaging to replicate presence updates across the cluster.

The default value of this parameter is the value of the parameter `EntitlementAllowToken`. In this way, if the value of the parameter `Entitlement` is `Basic`, then there is no need to configure this parameter, simply use its default value. Otherwise, if the value of the parameter `Entitlement` is `Custom`, then your custom entitlement rules should entitle subscriptions and publications on the subject defined by the parameter `Extension.Presence.Subject` for the clients using the entitlement token defined by this parameter.

5.53 MaxBandwidthRate

MaxBandwidthRate	
Description	Specifies the maximum bandwidth rate - in megabytes per second (MB/s) - that MigratoryData server is allowed to consume from the available network bandwidth capacity to push messages to clients
Default value	0
Required parameter	Optional

This parameter specifies the maximum bandwidth rate – in megabytes per second (MB/s) – that MigratoryData server is allowed to consume from the available network bandwidth capacity to push messages to clients. If zero is specified, the bandwidth rate will not be limited (this is the default value).

This parameter is particularly useful for use cases where, occasionally, the outgoing message throughput is higher than the available network bandwidth capacity. Supposing the MigratoryData server is installed on a machine with a network bandwidth capacity of a 1 Gbps (i.e. 125 MB/s), and supposing there are one or more large messages to be delivered to a large number of users, producing an outgoing data volume of 1250 MB. By configuring `MaxBandwidthRate` at 125 MB/s, the assumed available network capacity, the MigratoryData server will be able to deliver that volume of data during 10 seconds, and then operate normally, with low message latency. Otherwise, without configuring `MaxBandwidthRate`, the system might be severely impacted - as the capacity of the kernel's socket buffers could be exhausted.

Note that configuring `MaxBandwidthRate` could be useful not only for the use cases discussed above. It can be used as a protection against the deliberate or accidental exceeding of the available network bandwidth capacity for any other use case.

For most cases, `MaxBandwidthRate` should be configured to be equal to the available network bandwidth capacity.

> **Note** — With a LAN-based deployment, `MaxBandwidthRate` can be configured at 125 (MB/s) on a 1 GbE network, and respectively at 1250 (MB/s) on a 10 GbE network. However, with an Internet-based deployment, please check with your network provider to determine the bandwidth capacity available for your deployment.