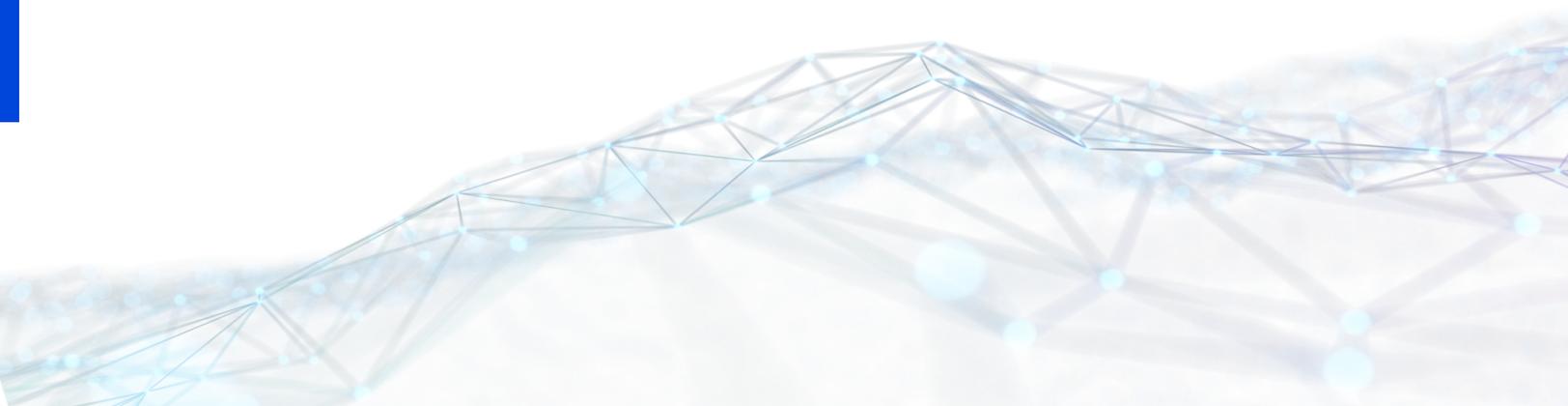




TIBCO Enterprise Message Service™

Release Notes

Version 10.3.0 | February 2024



Contents

Contents	2
About this Product	4
New Features	6
Release 10.3	6
Release 10.2	8
Release 10.1	9
Release 10.0	10
Changes in Functionality	11
Release 10.3	11
Release 10.2	11
Release 10.1	13
Deprecated and Removed Features	15
Deprecated Features	15
Removed Features	16
Platform Support	18
Migration and Compatibility	20
Migrating from Release 10.2	20
Migrating from Release 10.1	23
Migrating from Release 8.5	27
Closed Issues	29
Known Issues	38
TIBCO Documentation and Support Services	41

Legal and Third-Party Notices **43**

About this Product

TIBCO is proud to announce the latest release of TIBCO Enterprise Message Service™ software.

This release is the latest in a long history of TIBCO products that leverage the power of the Information Bus® technology to enable truly event-driven IT environments. To find out more about how TIBCO Enterprise Message Service software and other TIBCO products are powered by TIB® technology, please visit us at www.tibco.com.

TIBCO Enterprise Message Service software lets application programs send and receive messages according to the Jakarta Messaging protocol. It also integrates with TIBCO FTL and TIBCO Rendezvous.

TIBCO EMS software is part of TIBCO® Messaging.

Product Editions

TIBCO Messaging is available in a community edition and an enterprise edition.

TIBCO Messaging - Community Edition is ideal for getting started with TIBCO Messaging, for implementing application projects (including proof of concept efforts), for testing, and for deploying applications in a production environment. Although the community license limits the number of production clients, you can easily upgrade to the enterprise edition as your use of TIBCO Messaging expands.

The community edition is available free of charge. It is a full installation of the TIBCO Messaging software, with the following limitations and exclusions:

- Users may run up to 100 connections in a production environment. A connection is as defined in the Jakarta Messaging Specification and established between an application built with the TIBCO Enterprise Message Service Client Libraries and an instance of the TIBCO Enterprise Message Service Server.
- Users do not have access to TIBCO Support, but you can use TIBCO Community as a resource (<https://community.tibco.com>).
- Available on Red Hat Enterprise Linux Server, Microsoft Windows & Windows Server and Apple macOS.

TIBCO Messaging - Community Edition has the following additional limitations and exclusions:

- Excludes Fault Tolerance of the server.
- Excludes Unshared State Failover.
- Excludes Routing of messages between servers.
- Excludes JSON configuration files.
- Excludes EMS OSGi bundle.
- Excludes grid store and FTL store types.

TIBCO Messaging - Enterprise Edition is ideal for all application development projects, and for deploying and managing applications in an enterprise production environment. It includes all features presented in this documentation set, as well as access to TIBCO Support.

New Features

This section lists features added since the last major (10.0.0) release of TIBCO Enterprise Message Service.

Release 10.3

OAuth 2.0 Authentication

This release adds support for OAuth 2.0 authentication in the EMS server and clients.

In the OAuth 2.0 authentication model, all incoming connections to the EMS server are expected to present an OAuth 2.0 access token (in the form of a JSON Web Token) that contains the user and group information to be associated with the connection. The EMS server accepts or rejects connections based on whether the access token can be successfully validated using a configured public key or JSON Web Key Set. Once authenticated, a connection is granted the permissions corresponding to the user and groups obtained from the access token.

Connection objects in the EMS clients can be configured to request the access tokens required for authentication directly from an OAuth 2.0 authorization server using supported OAuth 2.0 grants, or to accept externally-obtained access tokens supplied by the client application. New C API functions, as well as equivalent Java system properties and environment variables are provided for configuring this behavior. Existing client applications that are not amenable to code changes can make use of the latter two options to enable and configure OAuth 2.0 authentication.

OAuth 2.0 support only involves connection factory objects created dynamically by client applications. Connection factories stored as administered objects are not supported and JNDI lookups do not take in OAuth 2.0 properties.

Access token procurement behavior can also be specified for EMS servers configured for fault-tolerance or routing. This can be done through newly added server and route properties.

Prometheus Metrics

This release introduces support for Prometheus-formatted monitoring metrics in the EMS server. Requests for metrics are received and serviced through a dedicated HTTP(S) listen port. Metrics are available for the following.

- Overall server state as well as the state of all destinations.
- Overall server state.
- State of all queues.
- State of all topics.
- State of an individual destination or groups of destinations.

Rendezvous Transports

This release reintroduces support for Rendezvous Transports, which had previously been dropped with release 10.1.0.

FTL Store Enhancements

- With past releases, the memory and disk space claimed by a server configured with FTL stores was subsequently reused by the server but would never be released back to the system. It is now released periodically as the backlog of messages decreases. Reclaiming disk space can also be triggered via the admin tool `compact` command.
- A new parameter called `-preferred_active` can be added to the YAML configuration file to designate one of the two EMS servers in the FT pair as the preferred active server. In scenarios where either EMS server in the FT pair could potentially activate, such as a full FTL server cluster restart, the EMS server configured with `-preferred_active` will always be the one to activate.
- A new server property called `ftl_disk_preallocation` allows for specifying the amount of disk space to reserve for the database underlying FTL store persistence. While this space will not be released even when not in use by the server, reserving it improves the performance of FTL stores. This is relevant when using EMS with FTL 6.10.1-HF01 (or higher) but note that it should no longer be necessary with the next major release of FTL.
- FTL stores now support asynchronous persistence of data to disk. Asynchronous disk persistence offers higher performance at the cost of guaranteed data persistence on

disk. The mode store property can be used to configure each FTL store for either synchronous or asynchronous disk persistence.

General Enhancements

- With past releases, the server would authenticate all JAAS incoming connections synchronously in its network threads. It now does it asynchronously in dedicated authentication threads, the number of which is controlled by the new `auth_thread_count` server property.
- The Health Check Listen port, which has been combined with the new Monitor Listen port, can now be secured through TLS. See the *HTTPS Server Parameters* section in the *User Guide* for details.
- The Java and .NET admin APIs have been extended with functions to get the counts of temporary and dynamic queues and topics.

Release 10.2

Support for Jakarta Messaging 3.0

This release adds support for the Jakarta Messaging 3.0 specification.

Jakarta EE, formerly Java EE, has renamed the `javax.jms` package into `jakarta.jms` with the Jakarta EE 9 release. The Jakarta Messaging 3.0 specification reflects the corresponding change of namespace. EMS implements both Jakarta Messaging 2.0 (`javax.jms` namespace) and 3.0 (`jakarta.jms` namespace) in the form of two sets of jar files – see the *TIBCO Enterprise Message Service User Guide* for details.

You must keep using the original set of jar files to run applications based on Jakarta Messaging 2.0 (also note that the JMS interface file has been renamed – see in the [Changes in Functionality](#) section) and switch to the new set to run applications based on Jakarta Messaging 3.0, primarily for use in the Jakarta EE 9+ world.

General Enhancement

The new `ServerInfo.getInboundMessageSize` and `ServerInfo.getOutboundMessageSize` Java admin API methods, the equivalent new .NET methods, and the admin tool show

server command output now report the total size of inbound or outbound messages handled by the server.

Release 10.1

New Store Types

TIBCO Enterprise Message Service now offers two additional store types that rely on non-shared storage solutions: grid stores and FTL stores. You can configure the server to store messages, state information, and configuration information in supported versions of TIBCO ActiveSpaces or TIBCO FTL, respectively. This is supported only on Linux.

Since both ActiveSpaces and FTL natively provide distributed clustered persistence, the new store types remove the requirement for EMS to interact directly with the underlying file system. In that context, EMS no longer has to rely on meeting the four shared state support criteria for a fault-tolerant setup required by the traditional file-based store, effectively removing the need for the associated physical network attached storage (SAN, NAS, etc.) and distributed file system (NFSv4, GFS2, etc.).

Grid Stores

Grid stores are designed to achieve a minimal EMS server memory footprint and quick EMS server recovery time upon failover. When configured to use grid stores, the majority of server data is stored in an ActiveSpaces data grid and is read into memory only on-demand. This approach decouples the EMS server's memory usage and failover time from the size of its stores and lends itself to quick server start-up times.

The latency costs in communicating with ActiveSpaces can make grid stores slower than file-based stores or FTL stores. The strength of grid stores lies with their scalability and consistent recovery time regardless of store size.

FTL Stores

The behavior of FTL stores is very similar to that of file-based stores. When using FTL stores, all pending persistent message data and state information is maintained in both server memory and in an FTL server cluster. Keeping this information in memory reduces the amount of communication needed with FTL and facilitates faster message processing. As with file-based stores, this approach requires that the EMS server reads the contents of all stores into memory upon start-up or failover to achieve this.

Communicating with FTL for message processing involves more overhead than the simple file reads and writes performed by file-based stores. Performance may vary depending on the usage and environment, but in general FTL stores are likely to be slower than file-based stores for that reason. The main differentiating factor between FTL stores and file-based stores lies in their underlying storage solutions and the implications of those for fault-tolerance. When making the choice between the two store types, in most cases the primary deciding factor should be whether you have access to a shared storage mechanism or not.

TLS Enhancements

- This release now supports TLSv1.3.
- This release supports a simplified way of selecting cipher suites based the OpenSSL SECLEVEL directive. We recommend using this new form.
- The Java, C, and .NET clients now allow server certificates with wildcard hostnames.
- The Java and C clients now allow hostnames in the SAN section of the certificate. Note that the .NET client doesn't.
- The server now disables TLS client side renegotiation.

General Enhancement

The trace statement shown by the server when a client ID or a durable name is too long has been enhanced (effective in EMS 8.6.0).

Release 10.0

TIBCO Enterprise Message Service 10.0 was released only on TIBCO Cloud™ Messaging.

Changes in Functionality

This section lists changes in functionality since the last major release of TIBCO Enterprise Message Service.

Release 10.3

OpenSSL

TIBCO Enterprise Message Service 10.3.0 operates with OpenSSL version 3.0.12.

This release prepares for the reintroduction of FIPS but does not support it yet.

General Changes

The Health Check Listen feature introduced with EMS 8.5 has been combined with the new Monitor Listen feature. It is now configured through the `monitor_listen` server property instead of `health_check_listen`.

Release 10.2

FTL Stores

TIBCO Enterprise Message Service 10.2.0 introduces a redesigned version of FTL stores that offers improved performance, more robust fault-tolerance capabilities, and built-in disaster recovery features.

The configuration and deployment model for FTL stores has been optimized in this new design. Instead of connecting to a separate TIBCO FTL deployment, the EMS server now runs as a service under an FTL server umbrella process. Integration of EMS and FTL in this manner reduces latency in communication between the EMS server and FTL backend, while

also allowing the EMS server to leverage FTL's fault-tolerance and disaster recovery capabilities.

This new version of FTL stores has been designed to allow users to deploy an EMS server with FTL stores with minimal prior knowledge of TIBCO FTL. All information required for configuring and deploying FTL stores is available in the *TIBCO Enterprise Message Service User Guide* for release 10.2.0.

OpenSSL

TIBCO Enterprise Message Service 10.2.1 operates with OpenSSL version 3.0.7.

TIBCO Enterprise Message Service 10.2.0 operates with OpenSSL version 3.0.5.

The server now handles TLS handshakes in an asynchronous manner. This minimizes the impact of slow or misbehaving TLS clients on the server's ability to respond to incoming TLS connection requests in a timely manner.

This release prepares for the reintroduction of FIPS but does not support it yet.

New JMS Interface File

The *Java Message Service (JMS)* specification has been renamed *Jakarta Messaging* as part of Jakarta EE, which has been moved to the Eclipse Foundation. This release of EMS ships the Eclipse Foundation version of the JMS interface file in lieu of the Oracle version shipped previously. The new file is called `jakarta.jms-api-2.0.3.jar`. As a convenience, a symbolic link using the former file name is provided:

```
jms-2.0.jar -> jakarta.jms-api-2.0.3.jar.
```

Native macOS Installer

The macOS EMS distribution now comes in the form of a pkg installer file rather than a set of tar.gz files.

.NET Support

Platform Requirements for .NET Core:

- Red Hat Enterprise Linux Server
- Microsoft Windows

- Microsoft Windows Server

Tool Requirements:

- To run .NET Core programs: .NET Core 6.0 (LTS)
- To run .NET Framework programs: .NET Framework 4.8.0 (Windows only)

Package Requirements:

- Programs that involve LDAP JNDI lookups need to reference the `System.DirectoryServices.Protocols` package. This can be done in Visual Studio or in C# project files (*.csproj).
- An example is provided in the `EMS_ROOT/samples/cs` folder.

Limitations:

- .NET Core 6.0 does not support distributed transactions.
- LDAP JNDI lookups are supported for .NET Core only on Windows.

Release 10.1

OpenSSL

TIBCO Enterprise Message Service 10.1.0 operates with OpenSSL version 1.1.1l.

The introduction of support for TLSv1.3 has changed the way in which cipher suite lists work for the server and C client. See the *Syntax for Cipher Suites* section in the *TIBCO Enterprise Message Service User Guide* for details.

Where appropriate, mentions of 'SSL' have been replaced with 'TLS' throughout the product and documentation.

.NET Support

Platform Requirements for .NET Core:

- Red Hat Enterprise Linux Server
- Microsoft Windows
- Microsoft Windows Server

Tool Requirements:

- To run .NET Core programs: .NET Core 3.1 (LTS)
- To run .NET Framework programs: .NET Framework 4.8.0 (Windows only)

Package Requirements:

- Programs that involve LDAP JNDI lookups need to reference the `System.DirectoryServices.Protocols` package. This can be done in Visual Studio or in C# project files (*.csproj).
- An example is provided in the `EMS_ROOT/samples/cs` folder.

Limitations:

- .NET Core 3.1 does not support distributed transactions.
- LDAP JNDI lookups are supported for .NET Core only on Windows.

General Changes

- The server now attempts to activate instead of exiting when it cannot resolve the hostname of its FT peer.
- A new high performance memory allocator has been added to the server on Linux and macOS.
- The `module_path` server property now only applies to the TIBCO FTL or ActiveSpaces libraries. The location of the OpenSSL and compression libraries is no longer configurable. Also, the `-module_path tibemsadmin` command line parameter has been removed.
- The `tibemsadmin`, `tibemsmonitor` tools and sample client programs no longer automatically disable verification of the server host's certificate if trusted certificates are not provided. Command line parameters are available to explicitly disable server host certificate verification if required.

Deprecated and Removed Features

The following tables list any features that have been deprecated or removed for version 10.3 of TIBCO Enterprise Message Service.

For deprecated features, if relevant, useful alternatives to the deprecated features are listed. Any use of a deprecated feature should be discontinued as it may be removed in a future release. You should avoid becoming dependent on deprecated features and become familiar with the suggested alternative features.

Deprecated Features

Affected Component	Description	Deprecated in Release
Grid Stores	This release deprecates support of stores of type <code>as</code> (grid stores). Support for grid stores will be removed in a future release.	10.3.0
Server Properties	This release deprecates the <code>health_check_listen</code> server property name. It is replaced with the <code>monitor_listen</code> property, which provides the same server health information in addition to server metrics.	10.3.0
Administration Tool	This release deprecates support for the Administration Tool (tibemsadmin).	10.1.0
Server Properties	This release deprecates support for the <code>processor_ids</code> server property.	8.6.0
Store Properties	This release deprecates support for the <code>processor_id</code> store property.	8.6.0
Server Properties	This release deprecates support for the	8.6.0

Affected Component	Description	Deprecated in Release
	<code>ftl_url_secondary</code> server property. Supply the <code>ftl_url</code> property with a pipe-separated list of URLs of FTL servers that provide realm services instead.	
JAAS and JACI	This release deprecates support of the JAAS and JACI features on macOS. Support of these features on macOS will be removed in a future release. This does not affect the support of the JAAS and JACI features on other platforms.	8.6.0
Flow Control	The <code>flow_control_only_with_active_consumer</code> property to revert to the pre-8.4 behavior has been deprecated and will be removed in a future release.	8.4.0
JAAS and JACI	This release deprecates the <code>jaas_classpath</code> and <code>jaci_classpath</code> parameters. Users should migrate to the new <code>security_classpath</code> parameter.	8.1.0

Removed Features

Affected Component	Description	Deprecated in Release	Removed in Release
TLS Communication	The <code>ssl_dh_size</code> property is no longer supported. The EMS server now configures OpenSSL to use its default built-in DH parameters.	N/A	10.2.0
FTL Stores	The <code>FTLStoreInfo.getFTLURL</code> and <code>FTLStoreInfo.getFTLAppName</code> Java API Admin methods and the equivalent .NET methods are no longer relevant. The corresponding administration tool output has been removed.	N/A	10.2.0

Affected Component	Description	Deprecated in Release	Removed in Release
LDAP Authentication	User authentication through LDAP can be implemented either by setting the EMS server properties starting in <code>ldap_</code> or by using the LDAP JAAS authentication modules. Support of the former feature that relies on <code>ldap_</code> properties has been removed. We recommend using the newer LDAP JAAS authentication modules instead, which remain fully supported.	8.5.0	10.1.0
LDAP JNDI Lookups in C	Support by the C client of JNDI Lookups in an LDAP server has been removed. That feature is still supported by the Java and .NET clients.	8.5.0	10.1.0
Client Libraries	Support of the static C client libraries has been removed. The dynamic C client libraries remain fully supported.	8.5.1	10.1.0
Database Stores	Support of stores of type <code>dbstore</code> has been removed.	N/A	10.1.0
mstores	Support of stores of type <code>mstore</code> has been removed.	N/A	10.1.0
File Stores	The <code>file_crc</code> file-based store property has been removed. The EMS server now always uses CRC to validate data integrity when reading file-based stores.	N/A	10.1.0
Rendezvous Transports	Support for the Rendezvous transports has been removed. Pure Java Rendezvous programs are still supported.	N/A	10.1.0

Affected Component	Description	Deprecated in Release	Removed in Release
	Note that Rendezvous transports have been reintroduced with EMS 10.3.0.		
Central Administration	Support of Central Administration has been removed.	N/A	10.1.0
TLS Communication	The TLSv1.1 protocol is no longer supported.	N/A	10.1.0
64-bit Symbolic Links	Upon removing support of the 32-bit server executables and C client libraries on Linux and macOS, the -64 suffix present in the corresponding 64-bit file names had been removed and convenience symbolic links provided (such as <code>tibemsd64</code> -> <code>tibemsd</code>). These symbolic links have now been removed as well.	N/A	10.1.0

Platform Support

Platform	Status	As of Release	Notes
Apple macOS 12	Removed	10.3.0	This release supports Apple macOS 13 and 14.
Apple macOS 11	Obsolete	10.3.0	This release supports Apple macOS 13 and 14.
Apple macOS on x86-64	Deprecated	10.3.0	Support on Apple macOS x86-64 is deprecated and will be removed in a future release.

Platform	Status	As of Release	Notes
Microsoft Windows Server 2019	Obsolete	10.3.0	This release supports Microsoft Windows Server 2022.
Apple macOS 10.15	Obsolete	10.2.0	This release supports Apple macOS 11 and 12.
Apple macOS 10.14	Obsolete	10.1.0	This release supports Apple macOS 10.15 and 11.
Microsoft Windows Server 2016	Removed	10.1.0	This release supports Microsoft Windows Server 2019.

Migration and Compatibility

The following are instructions on how to migrate from a previous release to version 10.3.0 of TIBCO Enterprise Message Service.

Order of Upgrade

Upon upgrading EMS software already installed on separate machines to a newer version of EMS, it is recommended to upgrade and restart in the following order:

1. Upgrade and restart all EMS servers.
2. Upgrade and restart EMS clients.

Compatibility with TIBCO FTL, TIBCO ActiveSpaces, and TIBCO Rendezvous

TIBCO Enterprise Message Service release 10.3.0 is compatible with:

- TIBCO FTL 6.10.1-HF01 or later
- TIBCO ActiveSpaces 4.8.0 or later
- TIBCO Rendezvous 8.6.1 or later

We recommend running EMS with the latest versions of these products.

Migrating from Release 10.2

Migrating FTL Stores

Release 10.3 of EMS introduces a number of enhancements and fixes to the FTL stores feature. It is highly recommended that all existing EMS deployments with FTL stores be migrated to EMS 10.3. The migration should be performed as per the instructions in this section to prevent any chance of data loss.

i Note: Once a deployment is upgraded to EMS 10.3, it is not possible to roll it back to a prior release of EMS.

EMS 10.3 is only compatible with TIBCO FTL 6.10.1-HF01 or newer. Once a deployment is upgraded to EMS 10.3, it is NOT possible to roll it back to a release of FTL earlier than 6.10.1- HF01.

Migrating from Release 10.2.1-HF05

Procedure

1. Back up the FTL server cluster state by following the procedure documented in the TIBCO FTL 6.10.1 Release Notes. Note the following FTL terminology:
 - An FTL server's realm data directory refers to its general data directory.
 - FTL server's `_embedded_tibemsd` persistence cluster data directory refers to its FTL store-specific data directory. Steps relating to all persistence clusters other than `_embedded_tibemsd` can be skipped.
2. Steps 3 to 8 must be repeated for each FTL server in the cluster, in the following order of servers:
 - FTL server associated with the standby-only EMS server.
 - FTL server associated with the current standby EMS server.
 - FTL server associated with the current active EMS server.
3. Shut down the FTL server.
4. If applicable, wait for any EMS client applications connected to the associated EMS server to fail over.
5. Remove the corresponding EMS 10.2 installation.
6. Install EMS 10.3.
7. Upgrade the corresponding FTL installation to TIBCO FTL 6.10.1-HF01 or newer. Refer to the TIBCO FTL documentation for instructions.
8. Restart the FTL server.

Migrating from Release 10.2.1-HF04 or Earlier

Procedure

1. Back up the FTL server cluster state by following the procedure documented in the TIBCO FTL 6.10.1 Release Notes. Note the following FTL terminology:
 - An FTL server's realm data directory refers to its general data directory.
 - An FTL server's `_embedded_tibemsd` persistence cluster data directory refers to its FTL store-specific data directory. Steps relating to all persistence clusters other than `_embedded_tibemsd` can be skipped.
2. Shut down the FTL server cluster.
3. Repeat steps 4 to 7 for each FTL server in the cluster.
4. Remove the EMS 10.2 installation.
5. Install EMS 10.3.
6. Upgrade the corresponding FTL installation to TIBCO FTL 6.10.1-HF01 or newer. Refer to the FTL 6.10.1 Release Notes for instructions.
7. Restart the FTL server.

i Note: Upon upgrading from EMS 10.2.1-HF04 or earlier to EMS 10.3, all data previously held in the `$sys.nonfailsafe` store is moved to a new store named `$sys.tmp.nonfailsafe` for compatibility reasons. Any data persisted after the upgrade is stored in `$sys.nonfailsafe` as usual. The EMS server ensures that the messages in `$sys.tmp.nonfailsafe` are delivered to consumers on the relevant destinations as expected. No user action is needed in regards to this change.

Restoring Pre-Migration State

This section provides instructions for restoring a deployment to its pre-migration state, if needed. These instructions are applicable for both of the migration procedures detailed above.

Procedure

1. Shut down the FTL server cluster.

2. Restore the FTL server cluster state from the backups by following the procedure documented in the TIBCO FTL 6.10.1 Release Notes.
3. Remove all corresponding EMS 10.3 installations and re-install the release of EMS 10.2 that was in use prior to migration.
4. Remove all associated FTL installations and reinstall the release of FTL that was in use prior to migration.
5. Restart the FTL server cluster.

Migrating from Release 10.1

Migrating FTL Stores

Release 10.2 of EMS is not directly backward compatible with FTL stores from release 10.1. To migrate FTL store data from an EMS 10.1 deployment to 10.2, follow the instructions in this section.

The migration procedure requires the EMS server to be offline while obtaining state information from the FTL server cluster. As such, a rolling upgrade from EMS 10.1 with FTL stores to EMS 10.2 is not possible.

i Note: These instructions are provided for migrating a typical EMS 10.1 deployment with FTL stores. If you have any questions or concerns about applying these steps to your particular deployment, please contact TIBCO Support.

1. Download the configuration for the EMS 10.1 deployment from the FTL server cluster.

```
/opt/tibco/ems/10.1/bin/tibemsjson2ftl -url <URL of any FTL server in the cluster> -key <key used while uploading config> -name <FTL app name used while uploading config> -json tibemsd.json -download
```

2. Shut down the 10.1 EMS server.
3. Save the EMS server state that is persisted in the FTL server cluster by requesting cluster shutdown with save state via the tibftladmin tool.

```
tibftladmin -ftls <URL of any FTL server in the cluster> -xc
--savestate true
```

This command will generate a state file with name <FTL server name>.state in the FTL store-specific data directory of each FTL server in the cluster.

4. The YAML configuration file of the EMS 10.1 deployment's FTL server cluster will need to be modified to be usable with EMS 10.2. Before moving on to the next step, please refer to the *Configuring and Deploying FTL Stores* section of the *TIBCO Enterprise Message Service User Guide* and make sure your YAML configuration contains all required sections and parameters.

i Note: If you are migrating an EMS 10.1 deployment that uses in-memory persistence for FTL, you will need to add the `in_memory_replication` parameter to the EMS server configuration. Refer to the *TIBCO Enterprise Message Service User Guide* for more information.

5. For each FTL server in the EMS 10.2 deployment, make sure that the general data directory and the FTL store-specific data directory to be used by the server are empty. These directories are specified in the `servers` section of the YAML configuration, via the `realm: data` parameter and the `tibemspd: -store` parameter respectively. Make sure that the directory from which the FTL server process will be launched is empty as well.
6. For each FTL server in the EMS 10.2 deployment, add the `-config_wait` and `load` parameters to the corresponding server entry in the YAML configuration. Each FTL server's `load` parameter must be supplied with the absolute path to the matching EMS 10.1 deployment .state file that was generated in step 3.

```
servers:
  # ...
  <name of FTL server #1>:
  # ...
  - tibemspd:
    # ...
    -config_wait:
      load: <path to state file of 10.1 deployment's FTL server #1>
  <name of FTL server #2>
  # ...
  - tibemspd:
    # ...
```

```

    -config_wait:
      load: <path to state file of 10.1 deployment's FTL server #2>
    <name of FTL server #3>:
    # ...
  - tibemsd:
    # ...
    -config_wait:
      load: <path to state file of 10.1 deployment's FTL server #3>

```

7. Start the FTL server cluster for the EMS 10.2 deployment using the YAML configuration file that was modified in the above steps.
8. Upload the EMS server configuration that was downloaded in step 1 to the EMS 10.2 deployment's FTL server cluster.

```

/opt/tibco/ems/10.2/bin/tibemsjson2ftl -url <URL of any FTL server
in the cluster> -json <path to downloaded EMS server configuration>

```

TLS Protocol

Industry security guidelines are now recommending that certificates, ciphers, and keys originally created using older protocols be upgraded to newer, stronger implementations as soon as possible to prevent unauthorized access to applications and systems. The present release of TIBCO Enterprise Message Service requires strengthening ciphers and certificates, and removing older, exploitable protocols. It introduces a new set of minimum requirements that will affect the backward compatibility of older certificates, ciphers, and keys. EMS clients and servers using encrypted connections (SSL/TLS) may be affected.

- Certificates, whether used in EMS PKCS#12 files or copied elsewhere, may need to be updated. Refer to the *TIBCO Enterprise Message Service User Guide* under *Digital Certificates* for more information on certificates.
- PKCS#12 files specified in EMS configurations may need to be converted as specified below.

Changes in OpenSSL 3.0

As part of this strengthening of security, EMS is transitioning from OpenSSL 1.1.1 to OpenSSL 3.0. The new version attempts to simplify such things as cipher suite selection and key length choices using a security level setting (SECLEVEL) from 0 to 5. The default SECLEVEL is 1, and includes the following restrictions, as documented on the [OpenSSL site](#):

- RSA, DSA and DH keys shorter than 1024 bits and ECC keys shorter than 160 bits are prohibited.
- All export cipher suites are prohibited.
- SSL version 2 is prohibited.
- Any cipher suite using MD5 for the MAC is also prohibited.
- Signatures using SHA1 and MD5 are also forbidden.

EMS imposes additional restrictions beyond these:

- SSL version 3 is disabled.
- TLS versions 1.0 and 1.1 are disabled.

Additional restrictions may be applied in the future as best practices evolve. Because of these restrictions, many of the cipher suites available in OpenSSL 1.1.1 are, by default, disabled. Certificates and keys that do not meet these criteria will fail.

The first consequence of this that may be noticed is that PKCS#12 files that were encrypted with older ciphers will no longer be readable. This is because, by default, older utilities produced files that use RC2 encryption to protect the private key. RC2 is considered a legacy algorithm in OpenSSL 3.0. Not only does it not meet the criteria of SECLEVEL 1, it is not actually compiled into the main library. See below for instructions on converting older PKCS#12 files to a format acceptable to OpenSSL 3.0.

Converting the PKCS#12 file to newer ciphers will, of course, introduce the requirement that all consumers of the file must support the new ciphers. In practice, this means that EMS Java clients should be running with the specified minimum builds (or later) of the following versions of Java: 8u301 (Oracle), 8u342 (OpenJDK), 11.0.12, or any 17.x build.

Even after the file is converted, if the key algorithm or key size is not acceptable by modern standards, OpenSSL will reject any certificate based on that key. Customers will need to replace existing certificates with new ones that meet the requirements of SECLEVEL 1.

There are other reasons that OpenSSL 3.0 may reject a customer generated certificate. It generally enforces the rules specified in the applicable RFCs much more strictly. For instance, the following errors may come up:

- Path length given without key usage
- Missing Authority Key Identifier
- Missing Subject Key Identifier

- Basic Constraints of CA cert not marked critical
- CA cert does not include key usage extension

This is not an exhaustive list, but represents a few of the errors we have seen in practice.

The restrictions on actual TLS cipher suite selection should be benign, since virtually all clients support at least one cipher mode that meets the criteria.

Converting PKCS#12 Files

To convert a legacy PKCS#12 file to newer algorithms using OpenSSL 1.1.1, use the following commands:

```
openssl pkcs12 -in sample.p12 -passin pass:password -nodes > tmp.txt  
openssl pkcs12 -in tmp.txt -out fixed_sample.p12 -macalg SHA256 -keypbe AES-256-CBC -certpbe AES-256-CBC -export -passout pass:password
```

The corresponding commands for OpenSSL 3.0:

```
openssl pkcs12 -in sample.p12 -passin pass:password -noenc -legacy > tmp.txt  
openssl pkcs12 -in tmp.txt -out fixed_sample.p12 -export -passout pass:password
```

The result can be verified with the following command:

```
openssl pkcs12 -in fixed_sample.p12 -info -noenc -noout -passin pass:password
```

If the output contains "RC2" in any of the text, then the .p12 file is incompatible with OpenSSL 3.0.

New JMS Interface File

As described in the [Changes in Functionality](#) section, the JMS interface file `jms-2.0.jar` has been replaced with `jakarta.jms-api-2.0.3.jar`. As a convenience, a symbolic link using the former file name is provided. However, we recommend that you switch to the new name in your environments.

Migrating from Release 8.5

FTL Transports

Release 8.6.0 of TIBCO Enterprise Message Service introduced changes of behavior for FTL transports. These changes are not backward compatible in that existing FTL transport configurations in EMS and configurations in FTL are likely to require adjustments. For details, refer to the "Interoperation with TIBCO FTL > Configuration > Destination" section in the *TIBCO Enterprise Message Service User Guide* and to the FTL documentation.

Closed Issues

The table lists issues closed in the listed version of TIBCO Enterprise Message Service.

Closed in Release	Key	Summary
Issues Closed in Release 10.3.0		
10.3.0	EMS-9023 EMS-8746	The Java client does not restrict referencing an XML eXternal Entity (XXE) properly upon storing an EMS administered object in an LDAP server or performing a JNDI lookup in such an LDAP server.
10.3.0	EMS-8921	Servers configured with FTL stores could encounter a race condition during state recovery that could result in memory corruption and the interruption of the startup process. This issue affects versions 10.2.1-HF04 and 10.2.1-HF05 of TIBCO Enterprise Message Service.
10.3.0	EMS-8914	The memory used by the server could significantly exceed the value of its <code>max_message_memory</code> property in situations where a particular topic has no publisher and one or more consumers are lagging significantly behind the others.
10.3.0	EMS-8891	Repeatedly deleting and recreating a consumer on a durable subscription in quick succession could result in a slow but unbounded increase in EMS server memory and disk usage when using FTL stores.
10.3.0	EMS-8887	When connected to an EMS server configured with FTL stores, the administration tool could show an irrelevant temporary store under particular circumstances. This issue affects version 10.2.1-HF05 of TIBCO Enterprise Message Service.
10.3.0	EMS-8838	On UNIX platforms, the server may attempt to resolve the

Closed in Release	Key	Summary
		user and group information of the system user who initially logged into the machine, rather than the current system user.
10.3.0	EMS-8820	The performance of FTL stores deployments could be impacted by high client connection rates and/or substantial use of <code>\$sys.nonfailsafe</code> destinations. This issue could manifest to a higher degree in environments with slow network and/or disk.
10.3.0	EMS-8819	The administration tool <code>compact</code> command requires the <code>store-name</code> and <code>max-time</code> arguments for FTL stores even though these are not applicable to this particular store type.
10.3.0	EMS-8818	The sample producer and consumer programs do not distribute messages across threads properly when the thread count does not evenly divide the message count.
10.3.0	EMS-8808	Large backlogs of messages in servers configured with FTL stores could result in excessive failover time. As this is addressed by maintaining data in the memory of the standby server, one should expect its memory usage to increase accordingly.
10.3.0	EMS-8800	The way the prebuilt <code>LDAPAuthentication</code> and <code>LDAPGroupUserAuthentication</code> JAAS modules (and any other class that implements either of those) handled concurrent login calls was not optimal. During an outage of the LDAP server, only one authentication thread was able to attempt reconnecting at a time, which would leave any other login attempt hanging without being subject to the <code>tibems.ldap.operation_timeout</code> LDAP JAAS module parameter.
10.3.0	EMS-8768	A server configured with FTL stores handling a large backlog could get a "Lock owned by another client" error

Closed in Release	Key	Summary
		upon store recovery, which would prevent it from activating.
10.3.0	EMS-8767	A disconnected client reconnecting and then committing an XA transaction during the lifespan of an XA session can lead to a server crash.
10.3.0	EMS-8763	On a JAAS authentication call, message handling can be held up until the call completes. For JAAS LDAP authentication, this means that login attempts can disrupt message handling during an LDAP server outage.
10.3.0	EMS-8757	The ungraceful shutdown (e.g. killing) of an active server configured with FTL stores and with <code>max_client_msg_size</code> set to a value larger than the 10 MB default could result in an excessive failover time.
10.3.0	EMS-8740	When configured with FTL stores, the server's memory and disk usage would grow without bound in proportion to the number of connections, sessions, consumers, and producers being created and destroyed.
10.3.0	EMS-8738	In a disaster recovery setup involving FTL stores, the FTL server cluster of the primary site would not activate if the FTL server cluster at the disaster recovery site was down or unreachable.
10.3.0	EMS-8735	The server could abruptly exit if the grid store background scan discards purged or expired message in a way that results in a nearly empty grid store. In that situation, it would output the following log statement: "WARNING: Grid store failure: cannot rollback on write failure. Exiting."
10.3.0	EMS-8730	The grid store background scan might not complete in a timely fashion. This could result in unconsumed, expired messages being incorrectly reported as pending, as well as increased use of storage space.

Closed in Release	Key	Summary
10.3.0	EMS-8729	The graceful shutdown of an active server configured with FTL stores could result in an excessive failover time.
10.3.0	EMS-8727	During recovery, the server can display 'Destination growing very large' warnings for destinations whose pending message count and size do not exceed the backlog threshold values.
10.3.0	EMS-8724	The server log format for factory-related admin actions does not follow the format of other admin actions.
10.3.0	EMS-8718	The EMS client API includes an undocumented and unsupported method to set a byte array property value into a message. Passing an empty array to that method could cause the server to exit unexpectedly.
10.3.0	EMS-8700	Messages expired through the expiration destination override may still get delivered to consumers when using grid stores.
10.3.0	EMS-8680	A timing issue in the server can cause recently connected clients to experience an invalid connection timeout, resulting in a forced disconnection.
10.3.0	EMS-8657	A leftover FTL workspace created through the FTL REST API or by enabling the FTL User Interface "Edit Mode" could prevent a server configured with FTL stores from starting up.
10.3.0	EMS-8647	A server configured with one of the prebuilt JAAS LDAP Authentication modules does not reconnect to the LDAP server if disconnected by a third party such as a firewall.
10.3.0	EMS-8620	The C client outputs additional debug information upon connecting or reconnecting through TLS, which also manifests in the administration tool.

Closed in Release	Key	Summary
10.3.0	EMS-8601	Sending or receiving large messages using the C client isn't performant.
10.3.0	EMS-8584	The <code>-url</code> option of the <code>tibemjson2ftl</code> tool accepts a single FTL server URL rather than a pipe-separated list of FTL server URLs.
10.3.0	EMS-8462	Memory is leaked upon parsing a JSON configuration file.
10.3.0	EMS-8449	The presence of an FTL transport in the configuration in the absence of the FTL client libraries in the <code>module_path</code> property causes the server to crash upon shutdown. Also, assigning an FTL transport to a destination without pointing the <code>module_path</code> to the FTL client libraries causes the server to crash.
10.3.0	EMS-8423	The presence of an FTL transport in the configuration in the absence of the <code>ftl_url</code> server property causes the server to crash.
10.3.0	EMS-8398	For file stores, partial disk writes are not treated as non-retryable errors.
10.3.0	EMS-8263	The combination of a client consuming from a queue with <code>prefetch</code> set to "none" and calling <code>receive</code> with a short timeout in a loop can cause the memory utilization of the server to grow significantly. This can happen when the receive timeout is so short that the server doesn't have a chance to deliver a message to the consumer before being asked again, causing a backup of receive requests in the server. To prevent this from happening, set the <code>prefetch_none_timeout_request_reply</code> server property to enabled.
10.3.0	EMS-8053	Configuring a <code>redeliveryDelay</code> of 8hour on a destination results in an actual <code>redeliveryDelay</code> of 15sec.

Closed in Release	Key	Summary
10.3.0	EMS-8010	A failure to export a message through an FTL transport results in the full body of the message being traced in the server log.
10.3.0	EMS-7132	Sample programs <code>tibjmsPerfController</code> and <code>tibjmsPerfWorker</code> do not correctly exit when a timeout is reached.
10.3.0	EMS-6685	The FTL and JNDI server tracing options cannot be altered through the admin API.
Issues Closed in Release 10.2.0		
10.2.0	EMS-8566	The presence of Rendezvous transports in the configuration prevents the EMS server from starting, even if the <code>tibrv_transports</code> property is absent or disabled.
10.2.0	EMS-8451	When configured to use either grid stores or FTL stores, the EMS server fails to execute TIBCO Messaging Manager (MSGMX) commands that apply to transports and stores.
10.2.0	EMS-8408	The EMS server could log 'Sender has discarded data' errors and exit upon accumulating a large backlog of messages in FTL stores.
10.2.0	EMS-8400	Exporting messages through FTL transports could result in duplicate messages, messages with an empty body, or server error traces, in particular while recovering messages upon a fault-tolerant failover or server restart. This issue affects versions 8.6.0 through 10.1.0 of TIBCO Enterprise Message Service.
10.2.0	EMS-8328	A disruption in the FTL server cluster used by an EMS server configured with FTL stores would cause the EMS server to exit. Such a disruption included a rolling upgrade of the FTL infrastructure. This issue is not relevant now that the FTL store feature no longer relies on an external FTL cluster.

Closed in Release	Key	Summary
10.2.0	EMS-8320	The tibemsjson2ftl and tibemsjson2grid tools return 0 even in case of unsuccessful completion.
10.2.0	EMS-8307	Getting to a point when the size of a store of type ftl exceeds the configured byte limit of the corresponding store in FTL could result in undefined behavior, possibly including the EMS server becoming unresponsive.
10.2.0	EMS-8306	When using stores of type ftl, sending messages or transactions that exceeded the configured maximum message size for the corresponding store in FTL would result in undefined behavior, possibly including the EMS server exiting.
10.2.0	EMS-7131	On Linux, macOS, and Windows, attempting to synchronize the current server with a JSON configuration through the admin API fails if the JSON configuration holds UTF-16 characters.

Issues Closed in Release 10.1.0

10.1.0	EMS-8312	The server could crash upon sending messages recovered from disk across an FTL transport. This issue affects only version 8.6.0 of TIBCO Enterprise Message Service.
10.1.0	EMS-8121	The Java and .NET implementations of the client could leak a thread upon automatically reconnecting to the server.
10.1.0	EMS-8109	The bin/post-install.sh script that ships on Linux and macOS reports an error on particular platforms because it is pointing to the wrong shell.
10.1.0	EMS-8048	(Windows only) A change of behavior in particular versions of the Microsoft Windows SDK results in slower console tracing in the server. This affects the overall performance of the server, depending on your console_trace settings. This issue affects versions 8.6.0, 8.5.1, and 8.5.0 of TIBCO

Closed in Release	Key	Summary
		Enterprise Message Service.
10.1.0	EMS-8018	(Windows only) The mechanism used by the EMS server to time out incoming TLS connection attempts is not effective if an incoming connection fails to initiate the TLS handshake. This failure can prevent the server from processing other incoming connections until the initial connection is timed out by the TCP/IP stack. This issue affects versions 8.6.0, 8.5.1, 8.5.0, and 8.4.1 of TIBCO Enterprise Message Service.
10.1.0	EMS-8002	If the <code>server_timeout_server_connection</code> or <code>server_timeout_client_connection</code> server properties are set, the server can trace an incorrect timeout value when timing out either the TLS handshake for an incoming connection request or an outgoing connection attempt.
10.1.0 (fixed in 8.6.0)	EMS-7960	The UFO implementation of the <code>tibemsConnectionFactory_PrintToBuffer</code> C API function is incorrect.
10.1.0 (fixed in 8.6.0)	EMS-7952	If the Java or .NET UFO client libraries are explicitly set to invoke the exception listener in the event of a fault-tolerant failover, they will never attempt to perform the failover process. In the case of the C UFO client library, this instead results in a crash during the failover process.
10.1.0 (fixed in 8.6.0)	EMS-7716	(Windows only) The Uninstall button for the EMS entry in the "App & features" list in Windows Setting is grayed out.
Issues Closed in Release 10.0.0		
10.0.0	EMS-8137	The EMS route protocol is now optimized to reduce the number of protocol messages sent under particular circumstances when a route hub-and-spoke architecture involves a large number of spokes.

Closed in Release	Key	Summary
10.0.0	EMS-8058	When a Java client enables server certificate verification but does not explicitly specify trusted server certificates, the client fails to automatically load the system-level trust file. This issue affects versions 8.6.0, 8.5.1, and 8.5.0 of TIBCO Enterprise Message Service.
10.0.0	EMS-8049	If a topic is configured to both import from and export to TIBCO FTL, the server crashes when messages are imported on that topic. This issue affects only version 8.6.0 of TIBCO Enterprise Message Service.
10.0.0	EMS-8024	The server cannot load DER format certificates.
10.0.0 (fixed in 8.6.0)	EMS-7713	If a connection attempt is refused due to bad credentials, the server may incorrectly warn that the connection was timed out.
10.0.0	EMS-7331	The server could leak memory when processing incoming connection requests. This issue affects versions 8.3.0-HF06 through 8.6.0-HF01 of TIBCO Enterprise Message Service.

Known Issues

The table lists known issues in the listed version of TIBCO Enterprise Message Service

Key	Summary/Workaround
EMS-9065	<p>Summary: If started with Java 8, the Java client could leak memory upon creating and closing JMS connections configured for OAuth 2.0 authentication with the client credentials grant type.</p> <p>Workaround: If using the client credentials grant type for OAuth 2.0 authentication, start the client with Java 11 or 17.</p>
EMS-9040	<p>Summary: OAuth 2.0 properties affecting routes are not exposed through the show, create, setprop, addprop, and removeprop administration tool commands. The corresponding admin API calls are not supported.</p> <p>Workaround: None. This will be addressed in a future release.</p>
EMS-9039	<p>Summary: OAuth 2.0 authentication is not supported in the EMS .NET client. It is supported in the EMS Java and C clients.</p> <p>Workaround: None. This will be addressed in a future release.</p>
EMS-9002	<p>Summary: When using the default admin user, tibemsmonitor may fail to connect to a server configured with only OAuth 2.0 authentication.</p> <p>Workaround: Add local to the list of authentication methods in the user_auth server property and then explicitly configure a local admin user.</p>
EMS-8948	<p>Summary: OAuth 2.0 authentication is not supported in FTL transports and FTL stores.</p> <p>Workaround: None. This will be addressed in a future release.</p>
EMS-8651	<p>Summary: When using FTL stores, the EMS server uses the value of its max_client_msg_size, in_memory_replication, and ftl_disk_preallocation properties, if set, to adjust specific FTL server</p>

Key	Summary/Workaround
	<p>properties accordingly. However, those FTL properties are only set the very first time the EMS server is started. Subsequent changes of the above EMS properties are not reflected in FTL. This issue affects versions 10.2.0 through 10.3.0 of TIBCO Enterprise Message Service for <code>max_client_msg_size</code> and <code>in_memory_replication</code> and version 10.3.0 alone for <code>ftl_disk_preallocation</code>.</p> <p>Workaround: Set the <code>max_client_msg_size</code>, <code>in_memory_replication</code>, and <code>ftl_disk_preallocation</code> properties as needed before the first run of the EMS server and do not change them afterwards.</p>
EMS-8574	<p>Summary: The FTL store feature of EMS does not support the custom cert feature of FTL. FTL stores will not work if the corresponding set of FTL servers is configured with a custom cert.</p> <p>Workaround: None.</p>
EMS-8572	<p>Summary: On macOS, when the installation package is downloaded through a Web browser, it may get labeled as quarantined by the operating system. Installation may result in a system prompt stating that the package cannot be opened.</p> <p>Workaround: Remove the quarantine flag from the package before installing it. For example:</p> <pre>xattr -d com.apple.quarantine TIB_ems_10.3.0_macos_x86_64.pkg</pre>
EMS-8405	<p>Summary: A disruption in the ActiveSpaces grid used by an EMS server configured with grid stores may cause the EMS server to exit. Such a disruption includes a rolling upgrade of the ActiveSpaces infrastructure.</p> <p>Workaround: Make sure to configure fault-tolerance for a standby EMS server to take over when the active server exits.</p>
EMS-8286	<p>Summary: Deleted dynamic topics may reappear after a fault-tolerance failover in a pair of servers configured with grid stores and remain even after the first scan has been completed. Note that no message is being redelivered.</p> <p>Workaround: None.</p>

Key	Summary/Workaround
EMS- 7993	<p>Summary: Since EMS 8.6.0, the server creates a single default FTL durable per FTL transport when the <code>import_subscriber_name</code> transport property is not set. However, this FTL durable is reported as an FTL subscriber in the administration tool and through the corresponding Admin API calls.</p> <p>Workaround: None.</p>
EMS-7694	<p>Summary: Installing EMS using <code>zypper install</code> on Novell SUSE Linux Enterprise Server on x86-64 may result in the installer signaling that the corresponding EMS packages are not signed, which is expected.</p> <p>Workaround: Since the EMS RPM packages are not signed, ignore the corresponding warning.</p>
EMS-7189	<p>Summary: On certain 7.x releases of Red Hat Enterprise Linux (and all Linux distributions that are materially equivalent) older than 7.3, an EMS C client using an FT URL with its second or further member containing either <code>localhost</code> or the hostname of the local machine may not be able to connect to the EMS server on the local machine because of an issue with the hostname resolution.</p> <p>Workaround: In place of <code>localhost</code> or the local machine's hostname, provide the C client with the local machine's IP address or its loopback IP address.</p>
EMS-7061	<p>Summary: Expired messages that have been moved to the <code>\$sys.undelivered</code> queue from a topic and that are consumed from <code>\$sys.undelivered</code> right before stopping the server may be redelivered when the server is restarted.</p> <p>Workaround: None.</p>
EMS-6401	<p>Summary: Starting with EMS 8.2.2, the presence of a valid CRL file that is empty of revoked certificates in the <code>ssl_crl_path</code> directory will trigger a warning. Such a warning encountered at startup time will cause the EMS server to abort if the <code>startup_abort_list</code> holds the SSL condition.</p> <p>Workaround: If the <code>startup_abort_list</code> holds the SSL condition, make sure that no valid CRL file that is empty of revoked certificates is placed in the <code>ssl_crl_path</code> directory.</p>

TIBCO Documentation and Support Services

For information about this product, you can read the documentation, contact TIBCO Support, and join TIBCO Community.

How to Access TIBCO Documentation

Documentation for TIBCO products is available on the [Product Documentation website](#), mainly in HTML and PDF formats.

The [Product Documentation website](#) is updated frequently and is more current than any other documentation included with the product.

Product-Specific Documentation

The following documentation for this product is available on the [TIBCO Enterprise Message Service™ Product Documentation](#) page:

- *TIBCO Enterprise Message Service™ Release Notes*
- *TIBCO Enterprise Message Service™ Installation*
- *TIBCO Enterprise Message Service™ User Guide*
- *TIBCO Enterprise Message Service™ C and COBOL Reference*
- *TIBCO Enterprise Message Service™ Java API Reference*
- *TIBCO Enterprise Message Service™ .NET API Reference*

Other TIBCO Product Documentation

When working with TIBCO Enterprise Message Service™, you may find it useful to read the documentation of the following TIBCO products:

- TIBCO® Messaging Manager
- TIBCO FTL®
- TIBCO ActiveSpaces®

- TIBCO Rendezvous®
- TIBCO® EMS Client for z/OS (CICS)
- TIBCO® EMS Client for z/OS (MVS)
- TIBCO® EMS Client for IBM i

How to Access Related Third-Party Documentation

When working with TIBCO Enterprise Message Service™, you may find it useful to read the documentation of the following third-party products:

- Jakarta Messaging™ Message specification, available through <https://jakarta.ee/specifications/messaging/2.0>.
- *Java™ Message Service* by Richard Monson-Haefel and David A. Chappell, O'Reilly and Associates, Sebastopol, California, 2001.
- Java™ Authentication and Authorization Service (JAAS) LoginModule Developer's Guide and Reference Guide, available through <http://www.oracle.com/technetwork/java/javase/jaas/index.html>.

How to Contact Support for TIBCO Products

You can contact the Support team in the following ways:

- To access the Support Knowledge Base and getting personalized content about products you are interested in, visit our [product Support website](#).
- To create a Support case, you must have a valid maintenance or support contract with a Cloud Software Group entity. You also need a username and password to log in to the our [product Support website](#). If you do not have a username, you can request one by clicking **Register** on the website.

How to Join TIBCO Community

TIBCO Community is the official channel for TIBCO customers, partners, and employee subject matter experts to share and access their collective experience. TIBCO Community offers access to Q&A forums, product wikis, and best practices. It also offers access to extensions, adapters, solution accelerators, and tools that extend and enable customers to gain full value from TIBCO products. In addition, users can submit and vote on feature requests from within the [TIBCO Ideas Portal](#). For a free registration, go to [TIBCO Community](#).

Legal and Third-Party Notices

SOME CLOUD SOFTWARE GROUP, INC. (“CLOUD SG”) SOFTWARE AND CLOUD SERVICES EMBED, BUNDLE, OR OTHERWISE INCLUDE OTHER SOFTWARE, INCLUDING OTHER CLOUD SG SOFTWARE (COLLECTIVELY, “INCLUDED SOFTWARE”). USE OF INCLUDED SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED CLOUD SG SOFTWARE AND/OR CLOUD SERVICES. THE INCLUDED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER CLOUD SG SOFTWARE AND/OR CLOUD SERVICES OR FOR ANY OTHER PURPOSE.

USE OF CLOUD SG SOFTWARE AND CLOUD SERVICES IS SUBJECT TO THE TERMS AND CONDITIONS OF AN AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER AGREEMENT WHICH IS DISPLAYED WHEN ACCESSING, DOWNLOADING, OR INSTALLING THE SOFTWARE OR CLOUD SERVICES (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH LICENSE AGREEMENT OR CLICKWRAP END USER AGREEMENT, THE LICENSE(S) LOCATED IN THE “LICENSE” FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE SAME TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of Cloud Software Group, Inc.

TIBCO, the TIBCO logo, the TIBCO O logo, TIBCO Cloud Integration, TIBCO Flogo Apps, TIBCO Flogo, TIB, Information Bus, TIBCO Enterprise Message Service, Rendezvous, and TIBCO Rendezvous are either registered trademarks or trademarks of Cloud Software Group, Inc. in the United States and/or other countries.

Java Platform Enterprise Edition (Java EE), Java 2 Platform Enterprise Edition (J2EE), and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle Corporation in the U.S. and other countries.

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only. You acknowledge that all rights to these third party marks are the exclusive property of their respective owners. Please refer to Cloud SG’s Third Party Trademark Notices (<https://www.cloud.com/legal>) for more information.

This document includes fonts that are licensed under the SIL Open Font License, Version 1.1, which is available at: <https://scripts.sil.org/OFL>

Copyright (c) Paul D. Hunt, with Reserved Font Name Source Sans Pro and Source Code Pro.

Cloud SG software may be available on multiple operating systems. However, not all operating system platforms for a specific software version are released at the same time. See the “readme” file for the availability of a specific version of Cloud SG software on a specific operating system platform.

THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT.

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. CLOUD SG MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S), THE PROGRAM(S), AND/OR THE SERVICES DESCRIBED IN THIS DOCUMENT AT ANY TIME WITHOUT NOTICE.

THE CONTENTS OF THIS DOCUMENT MAY BE MODIFIED AND/OR QUALIFIED, DIRECTLY OR INDIRECTLY, BY OTHER DOCUMENTATION WHICH ACCOMPANIES THIS SOFTWARE, INCLUDING BUT NOT LIMITED TO ANY RELEASE NOTES AND "README" FILES.

This and other products of Cloud SG may be covered by registered patents. For details, please refer to the Virtual Patent Marking document located at <https://www.tibco.com/patents>.

Copyright © 1997-2024. Cloud Software Group, Inc. All Rights Reserved.