TIBCO eFTL™

Release Notes

Version 6.6.0
November 2020

Contents

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 TIBCO Product Documentation website, mainly in HTML and PDF formats.

The TIBCO Product Documentation website is updated frequently and is more current than any other documentation included with the product. To access the latest documentation, visit https://docs.tibco.com.

Product-Specific Documentation

Documentation for TIBCO eFTL™ is available on the TIBCO eFTL™ Product Documentation page.

To directly access documentation for this product, double-click the following file:

TIBCO_HOME`/release_notes/TIB_eftl_6.6.0_docinfo.html` where *TIBCO_HOME* is the top-level directory in which TIBCO products are installed. On Windows, the default *TIBCO_HOME* is `C:\tibco`. On UNIX systems, the default *TIBCO_HOME* is `/opt/tibco`.

The following documents for this product can be found on the TIBCO Documentation site:

- *TIBCO eFTL Concepts*
- *TIBCO eFTL Administration*
- *TIBCO eFTL Development*
- *TIBCO eFTL Installation*
- *TIBCO eFTL API Reference* (HTML only)
- *TIBCO eFTL Release Notes*

Additional information resources can be found, after file extraction, in the samples directory. These include a Quick Start Guide, tutorials, readme.txt files, and sample applications.

How to Contact TIBCO Support

You can contact TIBCO Support in the following ways:

- For an overview of TIBCO Support, visit http://www.tibco.com/services/support.
- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support portal at https://support.tibco.com.
- For creating a Support case, you must have a valid maintenance or support contract with TIBCO. You also need a user name and password to log in to https://support.tibco.com. If you do not have a user name, 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 https://community.tibco.com.

About this Product

TIBCO® is proud to announce the latest release of TIBCO eFTL ™ software.

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

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 processes, 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 application instances or 1000 web/mobile instances in a production environment.
- Users do not have access to TIBCO Support, but you can use TIBCO Community as a resource (community.tibco.com).

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.

Bundling

The enterprise edition of TIBCO ActiveSpaces® uses the enterprise edition of TIBCO Messaging and includes a license for it. The community editions of those related products are compatible with both the enterprise and community editions of TIBCO Messaging.

New Features

The following new features were introduced in Release 6.6.0 and recent releases of TIBCO eFTL software.

6.6.0

FTL Message Delivery Count for eFTL Clients.
FTL message delivery count is now available to eFTL clients. This is useful for duplicate message detection. See the eFTL Client API documentation for more information.

Ability for an eFTL Client to Stop Message Delivery
With a new client API call, an eFTL client can stop message delivery from a subscription without removing the durable subscription. See the eFTL Client API documentation for more information.

Monitor Session Counters
The eFTL server now provides a session and max session monitor counter.

State Change Callbacks
JavaScript, Python and Go clients now support state change callbacks like C clients.

Python Client Connection via Websockets
You can now program Python clients to connect to an eFTL Service using websockets. For more information, see the eFTL Python Client API documentation and eFTL Python samples.

6.5.0

Client Maximum Pending Unacknowledged Message Count
An eFTL client is now able to set its maximum pending unacknowledged message count, overriding the value configured in the eFTL server. The maximum pending unacknowledged message count specifies the number of unacknowledged messages a client is allowed to accumulate before the eFTL server stops sending messages to the client.

Explicit Client Acknowledgments
The eFTL REST API `/<channel>/v1/subscribe/<durable-name>` now supports explicit client acknowledgment in conjunction with the `/<channel>/v1/subscribe/<durable-name>/acknowledge` API.

Store Local Message ID

The new eFTL client APIs now support getting the messages' local store message identifier.

Request/Reply REST API Support

Between eFTL clients, between eFTL and FTL clients, and between eFTL and EMS clients, you can now post a request message and receive a response in the form of a reply message.

6.4.0

eFTL Channel REST API Get Messages Number Limit

You can now limit the number of messages returned for an eFTL channel REST API when getting messages from subscriptions. The new parameter, for `GET <chan_name>/v1/subscribe/<dur_name>`, is `max=`*`number`*. See *eFTL Development* for details.

Go API Options with Default Values

For Go client applications, you can now use an options constructor method, `eftl.DefaultOptions()`, to set option default values.

Remove Key-Value Map Call

You can now call a function/method to remove a key-value map. The client API (RemoveKVMap) for this is available in C, Java, .NET, JavaScript, and Go. You can also create or destroy a key-value map via the following REST API calls:

```
POST <chan_name>/v1/kvmap/<map_name>
```

```
DELETE <chan_name>/v1/kvmap/<map_name>
```

See *eFTL Development* for details.

6.3.0

EMS Queues Support

eFTL clients can now access EMS queues using the eFTL client API and eFTL REST API, by prefixing the destination name with `QUEUE:` .

EMS Shared Durables Support

Clients connected to EMS can now create shared durables by subscribing with property `type` set to `shared`.

Message Acknowledgment Modes

eFTL now supports multiple message acknowledgment modes, which can be set via new API subscription properties. These modes are:

- Auto Acknowledge (default mode): Messages are automatically acknowledged once they have been successfully handled by the user. This is the mode in previous releases.
- Client Acknowledge: Messages are manually acknowledged by the user.
- No Acknowledgment: The acknowledgment protocol is not used.

Firebase Cloud Messaging Push Notifications

As an update from Google Cloud Messaging, eFTL can now use Firebase Cloud Messaging for push notifications.

Administration Web API eFTL Server Running Status Check

The new administration web REST API, `/api/v1/available`, returns a status code to quickly tell the health and availability of an eFTL Server and all configured channels.

6.2.0

eFTL-to-EMS Message Consumer ID

An eFTL subscriber client consumer ID is now provided for EMS monitoring, via logging (`consid=n`) and REST API requests (`consumer_id=n`).

Additional eFTL-to-EMS Message Headers and Properties

FTL subscribing clients now receive EMS messages populated with, additionally, EMS-JMS message headers and properties. These message headers have a field name prefix of `_emshdr:` while EMS message properties have a field name prefix of `_emsprop:`. For example:

```
{
  long=1,
  text=This is a sample eFTL message,
  time=Fri Jun 21 2019 14:16:39 GMT-0700 (Pacific Daylight Time),
  _dest=sample,
  _emshdr:JMSDeliveryMode=1,
  _emshdr:JMSPriority=4,
  _emshdr:JMSMessageID=ID:E4EMS-SERVER.6745D0C18A010:1,
  _emshdr:JMSTimestamp=1561151799772,
  _emshdr:JMSDeliveryTime=1561151799772,
  _emshdr:JMSDestination=TOPIC:sample,
  _emsprop:JMSXDeliveryCount=1
}
```

For more information about these headers and properties, see the TIBCO Enterprise Message Service (EMS) documentation.

Multiple URL Connect Tries

When connecting to an eFTL client you can now supply multiple URLs, separated by the "|" character. The eFTL service randomly selects and passes each URL to the eFTL client until a connection is made. If the list is exhausted without a successful connection, the service raises an error.

This feature applies to an initial connection and also a reconnection following a disconnect.

Subscriber Acknowledgment Mode Setting

You can now set acknowledgment mode of an EMS-channel subscriber via the administrative UI or web API.

.NET Core support

.NET Core is now supported, in addition to .NET Framework.

OpenSSL 1.1.1c Support

eFTL now supports OpenSSL 1.1.1c.

6.1.0

New Release Numbering

Release 6.0 brings the eFTL release numbering in line with FTL release numbering (skipping from 3.4.0 directly to 6.0).

Key/Value Maps

eFTL clients can access FTL key/value maps through a persistent FTL channel. Programs can set, get, and remove key/value pairs in a map. These calls are available in the client API of all supported languages except Objective C. REST web API calls are also available.

Changes in Functionality

The following changes in functionality were introduced in Release 6.6.0 and recent releases of TIBCO FTL software.

6.6.0

REST API Last-Value Durable

When calling the REST API `/{channel}/v1/subscribe/{durable}` on a shared or last-value durable, the durable type query string parameter, `?type=shared or ?type=last-value`, is no longer required.

Client Auto-Reconnect Default.

The default number of eFTL client default auto-reconnect attempts is increased from 5 to 256.

6.5.0

Increased Go Client Synchronous Operation timeout Interval Default

The eFTL Go language client's default synchronous operation default timeout is now 10 seconds.

6.4.0

Reduced Channel Default Configuration Values for Heartbeat and Timeout

The eFTL channel configuration default heartbeat value has been changed from 240 seconds to 60 seconds. The eFTL channel configuration default timeout value has been changed from 600 seconds to 150 seconds.

6.0.0

eFTL Service

In earlier releases the eFTL server was a standalone executable. In Release 6.0 the FTL server provides and manages an eFTL *service*. Configure the parameters of the new eFTL service in the FTL server configuration file.

Installation Prerequisite

Installing eFTL software requires prior installation of a compatible release of FTL software.

Deprecated and Removed Features

The following tables list any features that have been deprecated or removed as of Release 6.6.0 of TIBCO eFTL software.

For deprecated features, if relevant, useful alternatives are listed. Any use of a deprecated feature should be discontinued because it might be removed in a future release. To avoid becoming dependent on deprecated features, ensure that you become familiar with the suggested alternative features.

Platforms

Affected Platform	Description	Affected Release
Windows Server 2012	Migrate to Windows Server 2016.	6.1.0
Windows Server 2008	Migrate to Windows Server 2016.	3.4.0
Windows Phone	TIBCO eFTL no longer supports this device.	3.2.0
Red Hat Enterprise Linux Server 5.x 64-bit, x86-64	Migrate to 6.x or 7.x.	2.0.0
Novell SUSE Linux Enterprise Server 11.0 64-bit, x86-64	Migrate to 11.4 or 12.	2.0.0

Removed Features

Affected Component	Description	Deprecated Release	Removed Release
API	TIBCO eFTL no longer supports request/reply interactions. API calls related to this functionality are obsolete.	2.0.0	3.0.0

Affected Component	Description	Deprecated Release	Removed Release
	Method names related to this functionality have been removed.		
Push Notifications	Google has deprecated Google Cloud Messaging, effective April 10, 2019.	6.1.0	6.1.0
API	Support for Objective C clients is deprecated.	6.6.0	TBD

Migration and Compatibility

The following are instructions on how to migrate from a previous release to Release 6.6.0 of TIBCO eFTL software.

Migrating to Release 6.6.0

You can upgrade to Release 6.6.0 directly from Release 6.5, 6.4, 6.3, 3.4, 3.3, 3.2, 3.1, 3.0, or 2.0.

First upgrade to TIBCO FTL 6.6.0. Then upgrade to TIBCO eFTL 6.6.0.

Update your applications to use the latest eFTL client library.

Compatibility with Releases of Other TIBCO Products

TIBCO FTL
: TIBCO eFTL 6.6.0 is compatible with the realm server of TIBCO FTL 6.6.0.

TIBCO Enterprise Message Service
: TIBCO eFTL 6.6.0 is not compatible with TIBCO Enterprise Message Service (EMS) 8.5.0 or earlier releases. Upgrade to EMS 8.5.1.

Closed Issues

The following tables list closed issues in Release 6.6.0 and recent releases of TIBCO eFTL software.

Release 6.6.0

Key	Summary
EFTL-1744	Fixed a defect in the eFTL client library, where the client was not republishing messages upon a reconnect to the eFTL service.
EFTL-1663	You can now use the `publish.user` and `publish.client.id` FTL server YAML configuration properties with the eFTL `publish` REST requests.

Release 6.5.0

Key	Summary
EFTL-1700	Fixed issue where publishing an HTTP message with an empty array could result in a crash of the eFTL server.
EFTL-1699	Fixed an issue in the eFTL server where when a subscription was using explicit client acknowledgment and messages were acknowledged out of order, acknowledged messages could be redelivered.
EFTL-1698	Fixed an issue where `tibeftl_Acknowledge()` was acknowledging all messages up to and including the message being acknowledged rather than just the single message being acknowledged.
EFTL-1692	Fixed an issue where the C eFTL client would stop auto-reconnecting.
EFTL-1675	Updating the eFTL channel's persistence duration or max queue size no longer requires a restart of the eFTL server.
EFTL-1668	If a connection's subscription fails following a successful reconnect, the

Key	Summary
	subscription is no longer removed. This allows for subsequent reconnects to retry the subscription.
EFTL-1657	Fixed an issue where the eFTL server was unable to use the FTL server as an authentication service.
EFTL-1656	If the eFTL server could not authorize an eFTL client connection due to an error or timeout while accessing the authentication service, the eFTL server could incorrectly return an HTTP status code of `401 (Unauthorized)`. This issue has been fixed so that now this error/timeout returns an HTTP status code of `5xx` will be returned.
EFTL-1655	Fixed an issue where an eFTL client would sometimes fail to be authenticated during a connect.
EFTL-1650	Fixed an issue where an eFTL Go-language client would not disconnect when the channel to which it was connected is administratively removed.
EFTL-1634	Fixed a shared durable subscription issue that could occur when an eFTL C client reconnects to the server. The reconnected C eFTL client would incorrectly create a standard durable subscription rather than a shared durable subscription.
EFTL-1632	Fixed issues that occurred when eFTL clients reconnected to the server after the server lost connection to the persistence service.
EFTL-1626	Fixed issue where the eFTL Java `CompletionListener onError()` callback may have an error code of 0 rather than the actual error code.
EFTL-1623	Fixed an issue where an eFTL Go-language client might not receive messages following a reconnect to the server after the server lost connection to the persistence service.
EFTL-1593	Fixed an issue in the eFTL client library where an eFTL client connection could erroneously return the error "in progress".
EFTL-1587	The `/api/v1/available/eftl` REST request now returns an HTTP status

Key	Summary
	code of 503 (Service Unavailable) if the one or more eFTL channels lose connection to the persistence service.
EFTL-1586	Fixed an issue in the eFTL server. If an eFTL channel loses its connection to the persistence service, the eFTL channel will now stop, disconnecting the current clients and preventing new clients from connecting. When the eFTL channel has reestablished its connection to the persistence server, the eFTL channel then restarts.
EFTL-1580	An HTTP GET request for an eFTL channel subscription that doesn't exist now returns an HTTP status of 404 (instead of 400).

Release 6.4.0

Key	Summary
EFTL-1546	Fixed an issue where the eFTL server was returning a `store byte limit reached` error when the cluster was unavailable.
EFTL-1538	Fixed an issue where monitoring of a channel's outbound map messages was incorrect.
EFTL-1537	Fixed an issue where Go clients would return an error if a message contained a bool or nil field.
EFTL-1528	Fixed an issue where when an eFTL server was cleanly restarted (e.g., CTRL-C), the Objective-C eFTL client would not automatically reconnect.
EFTL-1503	Fixed an issue where a disconnected Go sender experienced a `write: broken pipe` error. eFTL now buffers publishes and subscribes when a Go client reconnects, instead of returning an error.
EFTL-1075	C-language clients now use the system default CA certificates to authenticate the server's certificate when the user has not configured their own trust store.

Release 6.3.1

Key	Summary
EFTL-1502, EFTL-1485	Fixed a client library defect (C, .NET, Java, JavaScript, Golang, and Objective-C) that prevented clients from automatically attempting to reconnect after the eFTL service was shutdown gracefully.
EFTL-1498	Fixed an eFTL service defect that caused the eFTL service to report incorrect connection count.
EFTL-1495	Fixed an issue where, if a persistence store's byte limit was exceeded, the associated client would hang until the number of bytes fell below the limit. Now a client `publish` returns immediately with an error if a persistence store's byte limit is exceeded.
EFTL-1492	Fixed an eFTL Golang client library defect that prevented the eFTL Golang client from reconnecting to the eFTL service.
EFTL-1481	Fixed an eFTL Golang client library defect that prevented the eFTL Golang client from detecting missing heartbeats from the eFTL service.

Release 6.3.0

Key	Summary
EFTL-1456	Fixed an issue where eFTL service configuration changes could occur even when the deployment was canceled.
EFTL-1443	For responses to REST requests, updated JSON message body formatting to be more consistent.
EFTL-1442	Prevent a Key-Value Map REST request from hanging if an invalid map is provided in the REST request URL.
EFTL-1433	Fixed an issue where calling the eFTL Java method `Message.getDouble()` could throw a `cast cast` exception if the value was a whole number.

Release 6.2.0

Key	Summary
EFTL-1306	Fixed a defect in the eFTL service where invoking the REST API `/api/v1/eftl/<cluster>/clients` could cause the eFTL service to abruptly exit.
EFTL-1286	Fixed a defect in the eFTL service where the eFTL REST API was introducing unnecessary delays when retrieving messages from subscriptions.
EFTL-1278	Fixed an issue where a durable subscription might not be removed when an eFTL client unsubscribes.
EFTL-1276	Corrected the field type for opaque message fields. A field type of `Opaque` is now returned instead of `String`.
EFTL-1067	The eFTL Go SDK now supports message double field types with the values `NaN`, `+Inf`, and `-Inf`.

Release 6.1.0

Key	Summary
EFTL-1253	Fixed a memory leak associated with durable subscribers.
EFTL-1236	Fixed a defect in which the eFTL service could abruptly exit if two clients with the same client identifier simultaneously connect and subscribe.
EFTL-1226	Fixed a defect in the eFTL service in which REST subscribe requests with timeout value zero erroneously returned an empty list of messages, even when messages were waiting for the subscriber.
EFTL-1225	Fixed an eFTL service file descriptor leak triggered by REST subscribe requests.
EFTL-1215	Fixed a memory leak in the eFTL service associated with erroneous durable subscriptions on non-persistent EMS channels. (Non-persistent channels do not support durable subscriptions.)
EFTL-1211	Fixed a memory leak in the eFTL service affecting durable subscriptions.

Release 6.0.1

Key	Summary
EFTL-1205	Fixed a memory leak in the eFTL service associated with clients disconnecting.

Release 6.0.0

Key	Summary
EFTL-1158	Fixed a defect in which the eFTL service running on Windows platforms could miss detecting disconnected clients.
EFTL-1143	Fixed a defect in which the eFTL service could stop delivering messages to one or more subscribers within a client process.
EFTL-1136	Fixed a defect in which the eFTL service running on Windows platforms could abruptly exit while processing messages sent from FTL or EMS.
EFTL-1133	Fixed an API library defect in which client apps could abruptly exit while disconnecting from an eFTL service.
EFTL-1118	Fixed a defect in which the eFTL service could abruptly exit if a REST request included an invalid authentication token.
EFTL-1117	Corrected examples of eFTL REST API calls in *TIBCO eFTL Development*.
EFTL-1116	Fixed a defect in which FTL monitoring reported incorrect connection count for the eFTL service.
EFTL-1113	Fixed a defect in which the eFTL service could disconnect from the FTL persistence store each time an eFTL client unsubscribed from a durable subscription.
EFTL-1112	Fixed a defect in which the C client library used an incorrect default port.
EFTL-1110	Fixed a Golang API library defect in which client apps could abruptly exit while attempting to reconnect to an eFTL service.

Known Issues

The table lists known issues in Release 6.6.0 of TIBCO eFTL software.

Key	Summary
EFTL-1372	**Summary**: The `trust_all` property is currently not supported in the .NET API with regard to WebSockets.
EFTL-1303	**Summary:** Installing eFTL using zypper install on version 15 of Novell SUSE Linux Enterprise Server on x86-64 may result in the installer signaling that the corresponding eFTL packages are not signed, which is expected. **Workaround:** Since the eFTL RPM packages are not signed, ignore the corresponding warning
EFTL-1299	**Summary**: eFTL 6.2.0 or later is not compatible with EMS 8.5.0 or earlier releases. **Workaround**: Upgrade to EMS 8.5.1.
EFTL-1107	**Summary**: On Windows platforms, after silent installation of one installation type, subsequently installing with a different installation is ineffective. **Workaround**: Completely uninstall the previous installation type, and reinstall with a new installation type.
EFTL-980	**Summary**: Linux Uninstall Uninstalling the product leaves empty directories after uninstallation completes. **Workaround**: Manually remove the empty directories `/opt/tibco/eftl/`*`release_num`*.
EFTL-317	**Summary**: Chrome and Safari browsers can no longer access the HTML documentation from a file system (`file://` protocol). **Workaround**: Access using a different browser, or access the HTML documentation through the web (`http://` protocol).

Legal and Third-Party Notices