TIBCO FTL®

Release Notes

Version 6.7.1
September 2021

Contents

About this Product

TIBCO® is proud to announce the latest release of TIBCO FTL® 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 FTL 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 FTL in the Community Edition has the following additional limitations and exclusions:

- Excludes transport bridges.
- Excludes the RDMA transport protocol.
- Excludes disaster recovery features.
- Excludes customizable dashboards and monitoring gateway.

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

No new features have been added to Release 6.7.1 of TIBCO FTL software. However, the following new features were added to recent releases of TIBCO FTL software.

6.7.0

Disk-Based Persistence

You can now store FTL messages and metadata to disk when disk is more readily available and cost effective than using memory. These messages and metadata can also then be automatically recovered on a full restart of the persistence cluster.

New GUI Disk Persistence Settings

The administrative GUI now has the following new options, related to the new disk-based persistence feature:

- Message Swapping Mode: Allow the persistence service to store more message data than available memory.
- Disk Persistence Mode: Enable disk-based persistence and select one of two modes, depending on performance requirements.
- Force Quorum Delay: Force a quorum after this amount of delay, even if all members are not present.

The modes are also included in monitoring statistics.

New GUI Cluster Backup Command

The administrative GUI now has the following new cluster backup command, available from the Cluster status display:

- Back up the cluster: Save the state of all persistence servers in this cluster. This option available only if disk persistence is enabled, and can be used while the cluster is running. The backup is intended for disaster recovery, and does not contain the most recent state.

New Log Level Parameter for tibftlserver

You can now specify the log level for `tibftlserver` as a command-line option.

New Group Client API Calls

In the client API, the group facility now has the following calls:

- `tibGroup_GetOrdinal()`, to get the current ordinal for a member of a group.
- `tibGroup_GetMembers()`, to get a list of all members in a group.

Format IDs Preserved

When downloading and then uploading a realm configuration, format IDs are now included in the upload, for more reliable migrations/backups.

Improved Deployment Description Format

In the Administrative GUI, deployment descriptions use a more readable format.

Improved Performance of Message Serialization and De-serialization

The FTL client API library now has a number of performance improvements during message serialization and de-serialization.

6.6.0

Swagger UI support for FTLserver web API

FTL now supports and includes Swagger UI for access to the FTL server web API REST commands.

Password Masking

Password masking by base64 encoding is now available for server and client connections.

In the client API, the `tibRealm_Connect` call accepts masked passwords in the appropriate password property. The password is unmasked before being sent to the realm service.

Where plain-text passwords appear in command line options or configuration files, a masked version of the password is now also accepted. Configuration files like the `tibftlserver` YAML file accept masked passwords where the following options are available:

- `pass:<password>`
- `file:<password file>`
- `env:<password env>`

New command line options for `tibftladmin` let you perform password masking encoding explicitly or on passwords contained within a configuration file.

Robust Monitoring Gateway

The FTL monitoring gateway (`tibmongateway`) now runs witn increased convenience and reliability through the following improvements:

- The `tibmongateway` process now runs as a service under the FTL server.
- Multiple `tibmongateway` processes can now collaborate in an active/standby configuration for increased fault tolerance.

Easier Auto Transport DR Configuration

Externally reachable addresses are no longer required when configuring auto transports for disaster recovery.

For Go API map Function, Simpler No-Lock Option

When using a Go-language API `map` function without a lock, for better code clarity you can now use a constant, `ftl.NoLock`, for the lock value.

FTL Client Prefetch Limit

Now client applications can set a prefetch limit. If the client sets its prefetch to a value greater than the configured value, the client's prefetch is automatically set to, and overrides, the configured value.

Full Satellite Display in GUI

In the GUI, Realm Service Status page, all satellite server URLs are now shown (leader and non-leaders).

Automatic Setup of Default Zone Address

The FTL server can now automatically set an externally reachable address for default zones in the following additional scenarios:

- where a load balancer is placed between the satellite and the primary cluster.
- where there is only one-way connectivity from the satellite to the primary.

Expired Dynamic Durable Logging Improved

When a dynamic durable expires, the log entry now includes the durable name.

Map Remove All Operation

You can now clear a map with one round-trip between client and server (rather than one round-trip per key) (`tibMap_RemoveAll()` in C).

Configuration Parameters for Persistence Service in the Default Cluster

In the FTL server YAML configuration file, you can now specify parameters for the default persistence service.

6.5.0

Save Persistence Cluster State Upon Shutdown

Sometimes a persistence cluster must save its state to disk before shutting down. To do this, you can use `tibftladmin` with a new parameter: `--saveonexit <true or false>`.

This new parameter works only with main commands `-xc --shutdown_cluster` or `-x --shutdown`, and when present, it appends the save state request to the main shutdown command being sent to the FTL server.

Administrative GUI Tooltips

The administrative GUI grid views now include cursor-hover tooltips.

Monitor Gateway Retention Policy Per Measurement

In the `tibmongateway` command you can now specify the InfluxDB retention policy to be used for each measurement (metric, event, log) measurement with the following new options:

- `--influx-log-retention-policy name`
- `--influx-event-retention-policy name`
- `--influx-metric-retention-policy name`

New Client API Call to Explicitly acknowledge a batch of messages

You can now use a durable subscriber call (e.g., tibSubscriber_AcknowledgeMessages in C; similar API calls have been added to java, .NET and Go clients) to acknowledge batches of messages.

New Monitoring Counters

The following new persistence monitoring counters (C) are added:

- TIB_MONITORING_TYPE_STORE_BYTE_LIMIT 1008 - the configured memory limit of a store
- TIB_MONITORING_TYPE_PERSISTENCE_SERVER 1100 - the persistence service name
- TIB_MONITORING_TYPE_PERSISTENCE_CLUSTER 1101 - the cluster name

New Client API Call to Get the FTL Server Version

You can now use a client API call (tibRealm_GetServerVersion in C) to return the version of the FTL server to which the client is connected.

New Client API Call to Get Event Queue and Map Names

You can now use a client API call (tibEventQueue_GetName, tibMap_GetName in C) to return the names of the event queue and key/value map.

New Client API Call to Get a Store-Local Message ID

You can now use a client API call (tibMessage_GetStoreLocalMessageId in C) to return a message monotonically increasing ID as assigned by the local persistence service.

New Message Dispatch API for Go Language Clients

For Go language applications, you can now invoke a dispatch-style call for messages in a event queue.

Check Realm Availability via Command Line

You can now use the `tibftladmin` command to check whether the realm service is available. (This is equivalent to the REST API call `/api/v1/available`.)

New Persistence Dashboard

The monitoring dashboards for Grafana now include a dashboard for persistence service monitoring.

Highlighting of Stores and Durables Approaching Limits

In the administrative GUI, durables and/or persistence stores that are approaching their configured limits are now highlighted.

Inbox Routing via Default Cluster

Inboxes are now able to be reached with routing durables using the default cluster.

6.4.0

Message Swapping Configuration Enhancements

For the message swapping persistence feature, you can now control additional parameters to determine when store message swapping to disk takes place, on a store, zone, or durable basis. This allows for finer tuning of the balance between performance and memory conservation.

Set Maximum Message Size in a Store

You can now configure, per store, a maximum message size, via the Administration GUI or web API. Messages exceeding this size generate an error when `Store Publisher Mode` is set to `Store - Confirm - Send`.

Custom Log Level Support for Client in Administration GUI

You can now set client log level in the Administration GUI. For example:

```
transports:debug;api:debug;msg:off
```

Wide-Area Forwarding of Last-Value Durables

Wide-area stores (routed stores) now support last-value durables.

Additional Samples

New samples (in C, Java, .NET and Go), offer different combinations of multiple subscribers and rate control.

Default Store Forwarding Between the Primary Realm Service and Satellites.

To simplify configuration, a default store-forwarding cluster routes messages between primary and satellite realm services.

Java System Property Log Level.

You can now pass property `com.tibco.ftl.log.level` as a Java system property.

Port Ranges for Dynamic TCP Transports

You can now set a range of ports on a dynamic TCP transport. When the transport is instantiated, it selects an unused port from the given range, and binds to that port. This applies to the listen side in a client-server dynamic TCP transport, or if the `mode` is set to `mesh`.

6.3.0

Targeted Deployments

For the Targeted Deployment feature introduced in FTL 6.2.0, FTL now has a more refined ability to skip deployments for clients that do not need it.

Throughput Improvements

Publishers now batch messages for improved throughput. A new publisher property is available to disable this behavior in favor of latency.

New Send Request and Send Reply APIs

You can now use simplified send-request and send-reply APIs to send a request message and receive a corresponding reply message .

Password Hashing

For user authentication based on text-file user databases and URLs of type `file://`, security is enhanced via UNIX-style password hashing. Hash modes are added to `auth.url` configuration entry.

Enhanced Administration and Monitoring of Durables

The REST API and FTL Server GUI have new functionality:

- Purge and delete multiple durables with a single GUI operation or REST request.
- Query/get durables by type (static or dynamic).

Migration of Stores from One Cluster to Another

The persistence service now supports migrating a store from one cluster to the other using the dump-state-to-disk file.

Pass FTL Properties as Java System Properties

For Java clients, you can now override application settings that are passed during realm connect. Realm properties like connect retries, client label, etc., can be passed to a Java application via `–D<property_name>=<property_value>`.

Filtering for Durables in Monitoring Request

When requesting durables for a store via a REST call, you can now filter the returned durables by durable name for realm services in all languages.

Support for eFTL-Channel Firebase Cloud Messaging Push Notifications

For eFTL Firebase Cloud Messaging (FCM) push notifications you can now set the server/api key via GUI or JSON configuration.

Administration Web API FTL 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 FTL Server or groupserver.

Realm Service Initial Configuration Option startup option to load an initial JSON

A new FTL server parameter, `initial.realm.config` lets you specify a JSON-configuration filename containing an initial realm configuration. The first time the FTL server is run, the FTL server is populated with this configuration. Subsequent runs of the FTL server use the last deployed realm configuration.

Persistence State File Management

A persistence service now moves a state file automatically after successfully loading the state file.

Linux Concurrent Installations

You can now install side-by-side versions of FTL on a Linux computer.

Field for timed out clients in component counts of the ftlserver cluster

The REST API `api/v1/server` now returns the count of timed out clients within the `server_cluster.component_count` section with field name `timed_out_client_count`.

New Persistence Monitoring Statistics

Persistence monitoring has new additional statistics values for the number of messages added to and removed from a store.

New Persistence Quorum Status via Web API

Now, via web APIs, you can get persistence quorum information regarding status and statistics.

6.2.0

Message Swapping

You can now enable persistence services to swap messages to disk when needed. This can be activated via a switch in the GUI or via web API.

.NET Core support

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

Targeted Deployments

During a deployment, clients are no longer targeted if there are no changes that would affect those clients.

Multiple Different Realm Connects From a Single Application

An application can now connect to different realm servers or services and be connected to multiple realms at the same time. Administrative changes made at one realm will not affect the application's connection to the other realms.

Map Iterator Matcher

For persistent stores, you can now create a map iterator that specifies a matcher, thus limiting the scope of the iteration. In addition, you can get the number of key/value pairs present in the map.

Dual-Use http Connection

For persistent http connections, you can now use the same http connection to make eFTL publish/subscribe web API calls and also realm web API calls.

OpenSSL 1.1.1c Support

FTL now supports OpenSSL 1.1.1c.

6.1.0

Performance Enhancements

Improved throughput and scalability of FTL servers.

Wide-Area Forwarding Zones

Enhanced GUI and REST API for zones.

Client Monitoring and Logging Enhancements

Clients that FTL creates internally now set their client labels for improved readability of log and monitoring data.

Log messages incorporate this information.

Grafana Dashboards

Updated Grafana dashboards, and added a new dashboard to monitor eFTL channels.

Server-Defined Transport for Message Broker Client Applications

An application definition can specify a server-defined transport, `Server`, which automatically connects with the persistence services of the message broker. The GUI specifies this behavior by default for new application definitions.

Map Label Property

Map create calls now accept a label property. Monitoring data incorporates the label.

Persistence Enhancement

Standard durables and templates no longer require a direct path.

FTL Server Web API

Added a new web API REST command to shut down only core servers.

Application-Defined Locking for Persistence Methods

A new API property governs retries for methods that request locks.

Docker Image for Log Service

A new Docker image runs `tiblogsvc`.

Go Language API

Go language message objects now support the Marshal and Unmarshal interfaces.

Documentation for Go Language API

Enhanced documentation for Go language API is now available in the book *TIBCO FTL Development*, and through GoDoc.

6.0.0

FTL Server

A new FTL server executable consolidates several services into one convenient executable. The FTL server incorporates the realm server, persistence server, group server, transport bridge process, and eFTL server.

See *TIBCO FTL Administration*.

Wide-Area Durables and Stores

Client applications can publish and subscribe to wide-area durables across network boundaries.

GUI Grid Search

You can search for objects in a GUI grid, or in any column of a grid. You can search using simple strings or regular expressions.

Message Broker

The FTL server can operate as a message broker.

Updated Resources on TIBCO Community

Supplemental resources are now distributed at the TIBCO FTL Community Wiki in the Reference Info tab. You can always find the latest versions of these resources in that location.

Those resources include *TIBCO FTL Quick Start Guide* and *TIBCO FTL Tutorials*. They also include sample FTL server configuration files and sample realm definition files.

Changes in Functionality

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

6.7.1

Security

Changes have been made for improved security.

6.7.0

tibmongateway as a Service

When `tibmongateway` is run as a service under the FTL server, it no longer requires FTL trust parameters to be specified.

Realm Database Filenames

The realm database maintained by the core servers now uses a different file name (*`filename`*`.persist` rather than *`filename`*`.dat`). When migrating to FTL 6.7.x, the old *`filename`*`.dat` file is imported and renamed.

6.6.0

Log Level Changes

Changes to a client or service log level are now always logged.

Monitoring Gateway Startup

Startup options are now printed when tibmongateway starts.

Grafana Persistence Server Count

In the monitoring Grafana page **`FTL Services: Overview`** or **`FTL Applications: Overview`**, the **`Persistence Server Count`** value no longer includes the internal config cluster's persistence services.

Grafana Custom Dashboards

If your 6.5.x (or earlier) FTL monitoring uses FTL-supplied Grafana dashboards, you must delete and re-deploy those dashboards using the new versions supplied with FTL 6.6.0. If those dashboards were customer-modified, you must re-apply those modifications.

Realm Service Monitoring Message Re-publishing

The realm service no longer automatically creates a subscription on the monitoring endpoint to re-publish monitoring messages for the benefit of pre-6.0 clients. For the realm server to re-publish monitoring messages to pre-6.0 clients you must set the `legacy.monitoring` configuration property in the realm service YAML configuration file.

6.5.0

Expanded Persistence Monitoring Statistics

Store monitoring REST API call responses now include the configured memory size limit (bytelimit). Also Durable monitoring REST API call responses now include message limits.

Realm Configuration Deployment

The FTL server now deploys a realm configuration even if there are no changes.

TIBCO Hawk Microagent

TIBCO Hawk Microagent for TIBCO FTL/TIBCO eFTL Software is now available under TIBCO Hawk at eDelivery.tibco.com.

ftlstart Samples Script

The `ftlstart` command in the samples directory no longer loads the `tibrealm.json` configuration file. To use the advanced sample features, load `tibrealm.json` after starting the FTL server.

6.4.0

Port Ranges for Dynamic TCP Transports

In the administrative GUI Transports grid, for a DTCP Transport you can now specify a range of ports.

6.3.0

YAML File Syntax for Windows

For Microsoft Windows-based installations, in the YAML configuration file, the `auth.url` value now requires forward slashes instead of backslashes.

6.2.0

Authentication-Only Security

Running the FTL Server in authentication-only security mode now requires running `tibftlserver --init-auth-only` before launching the server. This command generates security files `ftl-trust.pem` and `ftl-tport.p12`. Copy these files into the data directory of all the core FTL servers.

6.1.0

Persistence Stores Grid

GUI persistence stores grid has changed to simplify configuration of wide-area stores.

The columns *Persistence Cluster* and *Zone* have been replaced by new columns *Scope* and *Cluster/Zone*.

The Scope column indicates the scope of a persistence store, that is, whether the durables of the store are available only within a specific cluster of persistence services, or throughout the clusters of a zone.

The Cluster/Zone column names the cluster or zone where the durables of the store are available.

Persistence Clusters Status Web API

REST API calls to locations that start with

```
http://host:port/api/v1/persistence
```

now begin instead with

```
http://host:port/api/v1/persistence/clusters
```

Calls to old locations are deprecated, and will become obsolete in the November 2019 release. Until then, the old locations automatically map to the new calls.

Note: Update your code appropriately.

Inter-Cluster Transport

The GUI parameter formerly named "Inter-Cluster Transport" is now renamed "Zone Transport." You can set this parameter in the persistence service details panel.

Go Language API

The Go language API (in Releases 6.0.1 and earlier) misspelled the following constants:

- `PublisherPropertyStringLabel` was misspelled as `PublisterPropertyStringLabel`.
- `MonitoringTypeRecordsToCatchUp` was misspelled as `MonitoringTypeRecordsToCatcUp`.
- `MonitoringTypePendingConnectionCount` was misspelled as `MonitoringTypePendingMessageCount`.
- `MonitoringTypeClientDestroyedCount` was misspelled as `MonitoringTypeClientDestrouedCount`.

If you copied the misspelled names of these constants into your Go program code, correct them before migrating to Release 6.1 or later.

Map Property Constants

The following API constant values were incorrect. If your code uses the old literal values of these constants, you must update your code. It is good practice to code using constants rather than values.

Language	Constant	Old Value	New Value
.NET	`FTL.TIBMAP_PROPERTY_STRING_LABEL`	`com.tibco.ftl.client.`**`publisher`**`.label`	`com.tibco.ftl.client.`**`map`**`.label`
Java	`TibMap.PROPERTY_STRING_LABEL`	`com.tibco.ftl.client.`**`publisher`**`.label`	`com.tibco.ftl.client.`**`map`**`.label`

6.0.0

FTL Server

A new FTL server executable consolidates several services that were formerly separate executables. The FTL server incorporates the realm server, persistence server, group server, transport bridge process, and eFTL server.

Those executable processes are now obsolete. However, to facilitate migration to Release 6.0, the new services that replace them (within the FTL server) can interoperate with the older server processes. It is good practice to migrate your enterprise expeditiously.

Docker images for those executables are also obsolete. The release package now includes exactly one docker image, for the FTL server.

Parameter names that previously referred to the realm server now refer to the FTL server instead.

See *TIBCO FTL Administration*.

File-Based Configuration

The FTL server executable accepts a dramatically reduced set of command line parameters. Configure all parameters for the FTL server and its services in a consolidated configuration file. See *TIBCO FTL Administration*.

Realm Service Fault Tolerance Based on Clusters

Release 6.0 features a new fault tolerance mechanism for realm services within FTL core servers. This new mechanism, based on clusters of active FTL servers, completely replaces the old arrangement of backup realm servers.

Backup realm servers are obsolete, along with all configuration parameters related to them, including parameters of the realm service and also of its clients. The authorization group `ftl-backup` is also obsolete.

Realm Server Administration Utility

The new FTL server administration utility (`tibftladmin`) completely replaces the old realm server administration utility (`tibrealmadmin`). The new name indicates additional functionality.

Use only command line parameters to guide the behavior of the FTL administration utility, as it does not support reading parameters from a configuration file.

Persistence Retry Behavior

The default behavior is now `Unlimited`. That is, persistence calls retry the interaction indefinitely. Calls return only upon success. In earlier releases the default behavior was `None`.

Persistence Cluster Transport

It is now illegal to modify the cluster transport of a persistence cluster.

Authorization Groups

The authorization groups `ftl-primary`, `ftl-satellite`, `ftl-backup`, `ftl-dr` are obsolete. For each of these, use `ftl-internal` instead.

GET server REST API

The response output of the REST request `GET api/v1/server` has changed to reflect the new organization of the FTL server providing services. In particular, this request gets information pertaining only to responding FTL server.

New REST requests get information about each FTL core server and the core servers cluster as a whole.

For example, to get the client counts for all core servers, use `GET api/v1/cluster` (rather than `GET api/v1/server`).

Default Application Definition

The behavior of the `default` application definition has changed. In earlier releases the default application specified peer-to-peer message delivery. In Release 6.0 it specifies delivery using a message broker model, using the default non-persistent store and the dynamic standard durable template named `pubsub`.

For the previous behavior, use the `default-peer-to-peer` application definition.

Built-In Dynamic Durable Templates

The names of the built-in dynamic durable templates have changed.

Built-In Clusters

The name of the default cluster has changed.

Agent

The agent component is obsolete. FTL clients and servers running in Docker containers no longer require the agent.

Deprecated and Removed Features

The following tables list any features that have been deprecated or removed as of Release 6.7.1 of TIBCO FTL software.

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.

Platforms

Affected Platform	Migration	Affected Release
Windows Server 2012	Migrate to Windows Server 2016.	6.1.0
Windows Server 2008	Migrate to Windows Server 2016.	Deprecated in Release 5.3.0
Apple macOS 10.14 64-bit, x86-64	Migrate to 10.15.x.	6.6.1
Apple macOS 10.13 64-bit, x86-64	Migrate to 10.14.x, 10.15.x.	6.6.0
Apple macOS 10.12 64-bit, x86-64	Migrate to 10.14.x, 10.15.x.	6.1.0
Apple Mac OS X 10.11 64-bit, x86-64	Migrate to 10.14.x, 10.15.x	6.0.1
Red Hat Enterprise Linux Server 5.x 64-bit, x86-64	Migrate to 6.x or 7.x.	4.3.0
Novell SUSE Linux Enterprise Server	Migrate to 12.x or 15.	6.7.0

Affected Platform	Migration	Affected Release
11.x 64-bit, x86-64		
Novell SUSE Linux Enterprise Server 11.0 64-bit, x86-64	Migrate to 11.4 or 12.	4.2.0

Removed Features

Affected Component	Description	Deprecated Release	Removed Release
Pre-Built Docker Images	Pre-built Docker images are no longer supplied with FTL software distributions.	6.7.1	6.7.1
Administrative GUI	The display of client application statistics is deprecated. It is planned for removal in the next minor release.	6.7.1	
Persistence Service	The `disk_mode` configuration parameter for a persistence cluster in the web API is deprecated. Instead, use `disk_swap` and/or `disk_persistence`.	6.7.0	
RDMA Transport	On the Linux and Windows platforms, the RDMA transport is deprecated.	6.7.0	
FTL Server Monitoring	For the web API call `api/v1/persistence/ftlservers/`*<ftl server name>*`/status` and in the administrative GUI, FTL Server status, the following fields are deprecated: • `current_connection`	6.7.0	

Affected Component	Description	Deprecated Release	Removed Release
	• `max_connections` • `rejected_connections` • `lookup_failures` Also in the administrative GUI, the **`Connections at this server`** section is deprecated.		

Persistence Monitoring Web API	REST API calls to locations that started with
	`api/v1/persistence`
	now begin instead with
	`api/v1/persistence/clusters`
	Until (but not including) the removal release, the old locations automatically map to the new calls.
	Note: Update your code appropriately.
		6.1.0	

Go API	`MsgContent` is deprecated. Instead, use methods of the `Message` object to marshal and unmarshal data between messages and Go structs.	6.1.0	
FTL Server Web API	The REST API call `server {"cmd":"shutdown"}` is obsolete. Instead, use `ftlservers {"cmd":"shutdown"}`	6.1.0	6.1.0
FTL Server	Support for reading parameters from a	6.0.0	6.0.0

Affected Component	Description	Deprecated Release	Removed Release
Administration Utility	configuration file is deprecated. Supply all commands and parameters on the command line.		
Agent	The agent component is obsolete. FTL clients and servers running in Docker containers no longer require the agent.	6.0.0	6.0.0
Prometheus	Support for Prometheus is obsolete.	5.4.0	6.0.0
Monitoring Message Stream	New monitoring message types replace old types, which are deprecated: • 90012 replaces type 2. • 90013 replaces type 1. Use the new types. The old types remain in Release 5.4.0 for backward compatibility, but will become obsolete in a future release.	5.4.0	
Bridge Setup Scripts	The Python scripts `init_bridge.py` and `init_dtcp_bridge.py` are obsolete. Use the realm server web API instead.	5.2.0	5.2.0
Realm Configuration Python Scripts	The realm configuration Python script, `rs_script.py`, and its supporting utility scripts are obsolete. Use the realm server web API instead.	5.2.0	5.2.0
Group Setup	The group facility is automatically enabled. The Python script `init_groups.py` is obsolete and no longer needed.	5.2.0	5.2.0

Affected Component	Description	Deprecated Release	Removed Release
	The web API calls `POST realm/groupserver` and `DELETE realm/groupserver` are obsolete and no longer needed.		
Realm Server Internal JAAS	The realm server now relies on a separate authentication service, rather than an internal JAAS component.	5.2.0	5.2.0
Realm Server Monitoring Interface	The realm server no longer stores historical monitoring data. For replacement functionality, see *TIBCO FTL Monitoring*.	5.0.0	5.0.0
Edit Transport Configuration Manually	The realm server GUI transport definition page no longer support manually editing a transport's JSON definition. To modify the transport definition, use either the GUI or the web API. See "PUT realm/transports/*name*" in *TIBCO FTL Administration*.	5.0.0	5.0.0
Adapter	The adapter converts and forwards messages between TIBCO FTL and TIBCO Rendezvous. This component is obsolete. This functionality is now part of TIBCO Rendezvous Network Server software.	4.2.0	4.3.0
API	API calls that facilitated request/reply interactions between TIBCO FTL and TIBCO eFTL are obsolete.	4.2.0	Deactivated in 4.2.0. Removed in 5.0.0.

Affected Component	Description	Deprecated Release	Removed Release
FTL Server Configuration File	Support for eFTL Service configuration parameters `server.cert`, `private.key`, and `private.key.password` is deprecated. Instead, use FTL Server configuration parameters `custom.cert`, `custom.cert.private.key` , and `custom.cert.private.key.password` .	6.2.0	November 2019

Migration and Compatibility

The section identifies compatibility issues among releases of TIBCO FTL software. Instructions on how to migrate from a previous release to Release 6.7.1 of TIBCO FTL software are included in the product documentation set.

If you are upgrading from 6.3.1 or earlier, and using auth-only or secure mode, you must first upgrade to FTL 6.3.1 HF 4 or FTL 6.4.0 HF1 (if you haven't already).

Update your applications to use the latest FTL client library.

See “Upgrade Migration to a New Release” in *TIBCO FTL Administration*.

Note: If you are using the administrative GUI, after upgrading, clear your browser cache.

Compatibility with Earlier Releases of TIBCO FTL

- If using insecure mode, components of FTL Release 6.7.1 can communicate with those of Release 5.4 or later.
- If using auth-only or secure mode, components of FTL Release 6.7.1 can communicate only with those of Release 6.3.1 HF4, or Release 6.4.0 HF1, or later.
- If using auth-only or secure mode, we recommend upgrading all clients to FTL Release 6.7.1 or later.
- If using auth-only or secure mode, clients developed at FTL Release 6.7.1 or later cannot communicate with FTL servers at any earlier release.
- If you plan to use disk persistence (6.7.0 or later), you must upgrade all clients to 6.7.x or later.

Satellite Default Routing

If a server or cluster is started with no realm database, and if the primary site servers are at 6.7 and the satellite site servers are at 6.5.x or earlier, default routing is not functional. If this issue occurs, migrate the satellite server(s) to 6.7.

Compatibility with Releases of Other TIBCO Products

TIBCO ActiveSpaces®

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

TIBCO Enterprise Message Service

TIBCO FTL 6.7.1 is not compatible with TIBCO Enterprise Message Service (EMS) 8.5.0 or earlier releases. Upgrade to EMS 8.5.1 or later. When starting the TIBCO EMS server, ensure that the TIBCO EMS libraries precede the TIBCO FTL libraries in the `module_path`.

Closed Issues

The following tables list closed issues in recent releases of TIBCO FTL software.

Release 6.7.0

Key	Summary
FTL-12617	In the administrative GUI, durables that were created on the Primary Core servers do not appear upon a start of the Satellite Servers.
FTL-12614	If the realm property `Manage All Formats` is set to `true`, then the `tibPublisher_SendRequest` API call fails to send the request.
FTL-12611	If an API call `tibMap_SetMultiple` or `tibMap_SetMultipleWithLock` has more than 1000 keys, the client could abruptly exit.
FTL-12598	On Windows, attempting to set a transport backlog buffer size of 2GB or larger for a persistence transport causes an error in the persistence service.
FTL-12587	On Linux systems with Transparent Huge Pages enabled, the persistence service might use more memory than expected.
FTL-12546	If an application has an endpoint with a routed store, realm connect could fail following a disruption at the core servers.
FTL-12544	If the inter-cluster connection is interrupted for 20 seconds or more, messages queued for delivery across a route while the connection is down might be lost.
FTL-12543	A subscriber that experiences a lengthy disconnect and uses asynchronous acknowlegements might, after it reconnects, incorrectly acknowledge messages no longer owned.
FTL-12536	Unacked messages on a route might be duplicated if the inter-cluster link goes down.
FTL-12525	A persistence service might not recover from a synchronization failure if

Key	Summary
	last-value durables with message swapping enabled were in use.
FTL-12511	In rare cases, attempts to create publishers in parallel on endpoints configured with a store could deadlock.
FTL-12499	A realm deployment could cause the persistence service **`Server Status`** field in the administrative GUI to erroneously display the service as **`Offline`**.
FTL-12498	A sender discard during syncronization of the persistence services might permanently disrupt the quorum.
FTL-12494	An FTL client might not log anything if it attempts to connect to an unresponsive or unreachable FTL server.
FTL-12492	A persistence service might crash if a store is deleted from, and added again to, the realm definition.
FTL-12464	In some scenarios where FTL servers in a cluster fail, client group members do not receive `GROUP_MEMBER_DISCONNECTED` status advisories, and connected members receive erroneous `GROUP_MEMBER_JOINED` status advisories.
FTL-12463	If a group service becomes inactive due to an FTL server leader restart, the process during which the group service returns to an active state can cause erroneous status messages being sent to group clients.
FTL-12457	In the administrative GUI, some error messages for incorrect numeric field are vague.
FTL-12447	For messages received directly from a publisher, the delivery count returns a value of zero, when it should be -1.
FTL-12445	If a subscriber's endpoint is configured with a store and asynchronous auto acknowledgements, the acknowledgements could erroneously be sent to the persistence service prior to the callback handling of the messages.

Key	Summary
FTL-12436	When restoring the default cluster from a dump file, the quorum might not form automatically unless all members are present.
FTL-12419	On Windows installations, the swap file used by persistence services is unnecessarily large.
FTL-12416	During cluster quorum formation, FTL might issue a warning that `follower replied to a message not yet sent`.
FTL-12410	A persistence service that experiences a network interruption while synchronizing a last-value durable could ultimately fail to synchronize at all.
FTL-12379	See FTL-12464.
FTL-12376	The realm service uses more memory than necessary.
FTL-12349	A client application might abruptly exit when calling `tibRealm_Close`.
FTL-12303	In rare cases data loss in a persistence cluster can occur after a sync failure.
FTL-12296	**Summary:** The `tibftladmin` tool with option `--available` returns an exit code of 0 if the FTL server is unavailable. **Resolution:** The `tibftladmin` tool with option `--available` returns a non-zero exit code if the FTL server is unavailable.
FTL-12281	In rare cases a message in a routed store could be reflected back to the originating cluster.
FTL-12272	The administrative GUI contains an inaccurate Clients Connect Count display.
FTL-12239	If a REST request to the ftl server contains a malformed JSON body, the error response is vague.
FTL-12229	The persistence services ignore the configured weights after the first quorum formation.

Key	Summary
FTL-12228	Go-language client applications using API calls from both the FTL and ActiveSpaces libraries might throw an exception.
FTL-12226	If multiple persistence services at the DR site are syncing, they could experience excessive CPU usage.
FTL-12222	In Windows installations, the `ftlstart.bat` script fails to change directory to the data directory if the data directory of the FTL server is set to a different drive.
FTL-12196	Retry durations specified in the subscriber, publisher, map, and lock create calls were not interpreted correctly for very large positive values.
FTL-12182	When starting an FTL server with invalid password type stdin, the appropriate error message does not always display.
FTL-12144	In rare cases, after syncing, durable TTL might not be calculated correctly.
FTL-12130	A core realm service might fail to sync with the other core realms services after a restart..
FTL-12124	The API call `tibMap_Get` fails to honor the configured retry duration.
FTL-12121	Enabling centralized logging for a process at the INFO level, from the administrative GUI, might result in log messages not being sent to the realm service.
FTL-12114	See FTL-12228.
FTL-12104	Subscriber close may block for an unnecessarily long time for subscribers in non-replicated stores.
FTL-12053	A subscriber on a durable in a non-replicated store might experience an unnecessary reconnect if the quorum is temporarily disrupted.
FTL-11873	For a realm deployment with formats already part of realm configuration, the client library erroneously reports a `needs restart` message

Key	Summary
FTL-11713	In the administrative GUI, the monitoring status of zone stores erroneously displays the status as `Forwarding` instead of `Running` or `Off Line`.
FTL-11306	Large concurrent publishes can cause discards on the persistence service cluster transport.
FTL-10407	If interrupted while forming a quorum for the first time, the persistence service might log a warning about sync failure.

Release 6.6.1

Key	Summary
FTL-12506	A defect in the multicast transport could cause a subscriber to abruptly exit while processing a data packet. This can happen as a result of a race condition that prematurely destroys the multicast receiver object even while data is being processed.
FTL-12493	The multicast transport sometimes causes excessive CPU utilization when `Packet Send Limit` is set to unlimited (zero).
FTL-12480	When delivered by a persistence service, built-in opaque messages with a very small opaque field (less than 30 bytes) contains extra data at the end of the opaque field.
FTL-12477	A defect in the client library prevents a subscriber from FTL version 5.4.x to receive published messages from clients of version 6.5.x or greater.
FTL-12403	In the administrative GUI, the Channel Details page is missing the Save button and durables templates.
FTL-12393	The samples setup script GOPATH setting is incorrect.
FTL-12380	A client repeatedly calling `tib_open/tib_close` can sometimes result in thread local exhaustion.
FTL-12297	If a subscriber matches against a keyed opaque message, it does not receive messages.

Key	Summary
FTL-12273	On occasion, if receiving messages just before a reconnect, a subscriber can stop receiving messages after the reconnect.
FTL-12254	In rare cases, performing a realm deployment while sending messages on a route can cause the persistence service to stop acknowledging messages on the route.
FTL-12237	Persistence store forwarding does not function for built-in formats.

Release 6.6.0

Key	Summary
FTL-12178	Fixed a defect in the FTL golang library where a 'nil' value for the `GroupPropertyMessageMemberDescriptor` could cause the client to abruptly exit.
FTL-12151	In the administrative GUI, made corrections to some tables column headings for client and server statistics.
FTL-12148	Fixed an issue where a long-running persistence client could stop accepting deployments after an extended disconnect or a large number of previous deployments.
FTL-12147	Fixed a GUI issue where removing a transport from an eFTL channel on the eFTL table grid page was unsuccessful.
FTL-12125	Fixed an issue where the realm database could become inconsistent after a particular pattern of partitions and restarts.
FTL-12119	Fixed an issue where a REST API call to collect the latest persistence monitor counters would return unknown counters.
FTL-12078	Fixed an issue where a persistence service could occasionally fail to sync if leadership changed while it was waiting for sync to begin.
FTL-12052	Fixed an issue where, if a client subscribed to a durable using a matcher with duplicate fields, the persistence quorum was permanently disrupted.

Key	Summary
FTL-12033	Fixed an issue where, if the FTL server was configured with a hostname (other than localhost) that resolved to local interfaces only, the FTL server would open ports on external interfaces.
FTL-12032	Fixed an FTLServer defect where an invalid core server address in the YAML configuration file caused the FTLserver to abruptly exit.
FTL-12029	Improved the GUI displays of some FTL Realm pages for client statistics.
FTL-11990	Fixed an issue where some FTL REST API requests would return an HTTP status of 503 for a POST or PUT request with a body conting no content. These requests now return an HTTP status of 400.
FTL-11989	Fixed an issue where, in some instances, an FTL REST API call could return an HTTP status of 200 when the resource was not found (`/api/v1/server/`, `/api/v1/realm/map/`, `/api/v1/realm/deployments/`, `/api/v1/applications/`). These now return a status of 404.
FTL-11985	Fixed an issue where, in environments with high network or disk latency, a persistence service would occasionally be left out of a quorum, and would not attempt to rejoin.
FTL-11983	Fixed an issue where, if a secure ftlserver was started with a default realm configuration, the group server, group client, and eFTL cluster transports were not secure by default.
FTL-11975	Fixed an issue where, if started without `satelliteof /drfor`, an auxiliary `ftlserver` process at a satellite/dr site could initiate a deployment.
FTL-11974	Fixed an issue where, if using DR transports of type auto, disabling DR via realm deployment would cause all persistence services to respond with a `Needs Restart` display.
FTL-11973	Fixed an issue where the group server transport would open a port on an external interface even when the FTL server was listening on a local interface.

Key	Summary
FTL-11966	Fixed an issue where status codes for REST commands to a persistence service would indicate success even if the persistence service didn't actually exist.
FTL-11961	Fixed an issue where, in some disaster recovery (DR) scenarios, FTL servers would sometimes fail to start if FTL server names were reused across the DR event (e.g., if the original primary servers and the new DR servers shared the same names).
FTL-11957	An FTL persistence service now logs the name of the store if the store byte limit is exceeded.
FTL-11956	Fixed a REST API issue where, when calling `/api/v1/reflector` and a timeout occured, the realm would incorrectly log a warning-level error.
FTL-11952	Fixed a client library defect that prevented static and dynamic format messages from being forwarded to a subscriber on an endpoint that has a process transport configured.
FTL-11947	Fixed an issue where monitoring of an FTL server cluster sometimes caused high CPU usage.
FTL-11942	Fixed an issue where, for a single client process with many publishers, a reconnect to the persistence service might fail.
FTL-11940	Fixed an issue where, in rare cases (such as a very specific pattern of network delays), a client would be unable to store messages after reconnecting to a persistence service.
FTL-11935	Fixed an issue where a client calling `tibLock_Return` might cause a reconnect under high-load conditions.
FTL-11932	Fixed a GUI issue where The FTL Server monitoring page showed all zeros for **Connections at this server.**
FTL-11913	Fixed a secure-connection issue with Go language clients. Because FTL 6.6.0 uses Go version 1.15, custom certificates that are specified to the eFTL

Key	Summary
	service must specify the hostname in the Subject Alternates Names section of the certificate. For more information see the Go 1.15 Release Notes at https://golang.org/doc/go1.15.
FTL-11900	Fixed an issue where, if a persistence service lost leadership and then became leader again, it might log an exception such as `violated radix heap invariant`, though it would continue to function normally.
FTL-11890	In the administrative GUI, improved the way that statistics are displayed.
FTL-11885	The FTL server's configuration database no longer has a byte limit.
FTL-11883	Fixed an issue where the realm service was writing to disk even when no new clients were connecting and no deployments were in progress.
FTL-11880	Fixed a UI persistence store issue where the `Used/Max Bytes` and `Consumed Store Limit` value displays were sometimes erratic.
FTL-11875	Fixed a monitoring defect where an incorrect query prevented the FTL Persistence dashboard from displaying persistence service data.
FTL-11874	Fixed an FTL Go API library defect where a nil value specified for the `group.GroupPropertyMessageMemberDescriptor` property could cause the client to abruptly exit.
FTL-11871	Fixed an issue where the persistence servers' monitoring overview page was not displaying history, consistency, or disk status.
FTL-11831	Fixed an issue where the client would leak some memory on `tibRealm_Close`.
FTL-11793	Fixed a GUI issue where eFTL statistics data overlapped other data.
FTL-11789	Fixed an issue where, if `tibSubscriber_Close` was called while the client was disconnected from the persistence service, the call could block for an

Key	Summary
	unnecessarily long time.
FTL-11758	Fixed panic in tibftladmin when the FTL server responded with a compressed HTTP response.
FTL-11705	Fixed erroneous HTTP headers returned by the realm service.
FTL-11255	Fixed an issue where the message TTL for existing messages would reset after a rolling restart of persistence services.
FTL-11044	Fixed an issue where, at steady-state (no messages stored or acknowledged), a DR persistence service would sometimes report its status as syncing despite already being in sync.

Release 6.5.0

Key	Summary
FTL-11818	Fixed a client library defect that could cause the client to abruptly exit. This symptom could occur if the application had connected and then closed the connection to the FTL server numerous times.
FTL-11776	Fixed a Windows issue where `tibPublisher_SendRequest()` may return a timeout if multiple reply messages are received.
FTL-11771	Fixed an issue where, in the administrative GUI, `Cluster Grid`, `Channel`, for `FTL Cluster` and `FTL Store`, the selection `None` was not available.
FTL-11742	Fixed a defect in the FTL server that could cause the FTL server to abruptly exit if the `tls.trust.file yaml` parameter was invalid.
FTL-11738	Fixed an issue where `tibMap_RemoveWithLock` could hang if the lock was concurrently lost.
FTL-11734	Fixed a defect in the realm service that was causing it to report an incorrect satellite FTL server count.
FTL-11702	Fixed an issue where, for shared durables and standard durables with

Key	Summary
	prefetch, if a message limit was configured and the discard policy was old, the persistence service could occasionally discard a recent message.
FTL-11700	Fixed an issue where a format created in the administrative GUI would not display after deployment.
FTL-11679	Fixed a defect in the client library that was preventing the administrator from setting the multicast transport's UDP receive buffer size and UDP send buffer size to a value greater than 16MB.
FTL-11676	Fixed an issue where if the `tibMap_Get` call failed, the error description might be missing or incorrect.
FTL-11675	Fixed a memory leak caused by a failed realm connect operation.
FTL-11674	Fixed an issue where an application created in the administrative GUI would not allow the removal of an automatically configured a persistence cluster or store.
FTL-11671, FTL-6844	Fixed an issue where, during high throughput, a multicast sender could silently discard data. When the discarded data are fragments of a large message, the receiver could receive incomplete messages.
FTL-11662	Fixed an issue where, during reconnect, the FTL URLs would not randomize for load balancing.
FTL-11654	Fixed an issue where sending a reused message with NULL format on a non-inline peer to peer transport could result in message loss. This problem occurred only if a new field was added to the reused message between sends.
FTL-11653	Fixed an issue where a client connected to a persistence service would not fail over if a network partition was introduced between the original leader and the replicas and a blocking API call was in progress (e.g., send message).
FTL-11649	Fixed a defect in the FTL client library that was preventing Actives Spaces

Key	Summary
	clients from invoking `tibdg_Open()/tibdg_Close()` in a loop.
FTL-11643	Fixed a defect that caused the administrative GUI to sometimes display an incorrect number of stores.
FTL-11642	Fixed a defect in the multicast transport where when an FTL 5.4.x receiving client logged a warning message `ParseOpts ignoring unknown option type` when receiving data from a FTL 6.x sender. This message is now only logged once per FTL 6.x sender.
FTL-11636	Fixed an issue where traces for aggregated lvo durable discard advisories reported an incorrect event count.
FTL-11611	Fixed an issue of high memory usage when running the FTL server administrative GUI.
FTL-11610	Fixed an administrative GUI issue where the transport grid Address column was not visible, and where the browser console was displaying errors.
FTL-11602	In the administrative GUI, the eFTL channel's FTL `persistence_duration` configuration ios now displayed in the Channel information page.
FTL-11600	Fixed an issue where deployments could occasionally fail because a change to the default zone triggered a `Needs Restart` response.
FTL-11598	Fixed an issue where any time a message expired based on a TTL timeout, the persistence service could silently expire other messages too early or discard newly published messages.
FTL-11594	Fixed a Linux issue where running FTL servers or clients under a user whose group did not exist or was not mapped to a name could cause a client or the FTL server to abruptly exit.
FTL-11577	Fixed an issue where whenever a message was expired based on TTL, the leader persistence service would log an inexact count. The count is now exact and logged less frequently.

Key	Summary
FTL-11563	In the administrative GUI, the `Logging` button for processes listed under `Other Services` now functions correctly.
FTL-11561	Fixed an issue where a realm service could abruptly exit when requesting monitoring data via the administrative GUI.
FTL-11560	Fixed an issue of occasional excess message iterator contention.
FTL-11538	Fixed an issue where the administrative GUI could display the incorrect transport type (Dynamic TCP instead of Auto) after REST API configuration and deployment.
FTL-11535	Fixed an issue where if a persistence service encountered an exception while storing a message, the message could be lost even if the client received a send ack.
FTL-11532	Fixed an administrative GUI issue where when a new eFTL channel was created the channel's Cluster, Store, and Template were not blank.
FTL-11507	Fixed an issue with the `ftlstart` script where supplying both `-secure` and `--auth` would fail to start the FTL server.
FTL-11496, FTL-11387	Fixed an issue where creating a store and selecting the `Scope` as `Zone` would fail.
FTL-11494	Fixed an issue where, in the administrative GUI, `Application Grid`, for `Cluster` and `Store`, the selection `None` was not available.
FTL-11484	Fixed a defect where many clients connected to a persistence service could cause an internal timer in the persistence service to fire very rapidly.
FTL-11476	Fixed an issue where a persistence quorum could fail to store all messages from a call to `tibPublisher_SendMessages` if the middle of the batch goes missing on the network.
FTL-11466	Fixed a `tibftlserver yaml` file issue that would not allow valid `tls.trust.everyone` values.

Key	Summary
FTL-11459	Fixed a defect in the client library where `tibRealm_RemoveMap` was not honoring the map retry duration property.
FTL-11456	Fixed an issue where if a tibstore of a version older than 6.4 has a non-empty datalog on disk, and a migration to 6.4 or later is attempted, then the new tibstore failed to upload the data log from the disk, and would throw a `_tibMsgProps_Deserialize` exception.
FTL-11434	Fixed an issue where a restarted satellite FTL server would fail to reconnect to a cluster's primary server.
FTL-11427	Fixed a defect in the group client library where in if the observer flag is set to false during group join, the group member is not treated as ordinal member.
FTL-11406	Fixed an issue where a persistence cluster quorum could reform if publishers were sending at the same time that messages were expiring.
FTL-11390	Fixed an issue where the FTL client could abruptly exit when creating event queue timers with an interval of zero.
FTL-11333	Fixed an issue where the FTL server could abruptly exit on a call to `api/v1/support`.
FTL-11315	Corrected an administrative GUI label to indicate that Dynamic TCP port fields must always be a range.
FTL-11314	Fixed an issue where running the FTLserver docker image (`ftl-tibftlserver:6.4.0` and `ftl-tibeftlserver:6.4.0`) resulted in a benign warning (`Cannot assign requested address`) and an exception stack trace being printed on startup.
FTL-11305	A timeout can now be provided when creating a realm configuration workspace. Once the timeout expires, if the workspace has not yet been deployed the workspace is automatically deleted.
FTL-11298	Fixed an issue where, in the Go API, a deadlock could occur under a

Key	Summary
	combination of the following conditions: • An unbuffered message channel is created. • A subscriber is created using the unbuffered message channel on a given event queue. • One or more messages are delivered to the application via the unbuffered message channel, but the Go code does not receive the messages (perhaps the Go routine servicing the unbuffered message channel has exited). • The event queue's `CloseSubscription()` or `Destroy()` method is called.
FTL-11282	Timed-out clients of all types can now be purged.
FTL-11268	Fixed an FTL 6.4 issue where, if a persistence service discarded on its transport to the client, the message could be lost.
FTL-11247	Fixed an administrative GUI issue where, in eFTL Server Details, the last row is shown as `Undefined`.
FTL-11243	Fixed a client library defect where a create subscriber call on an advisory endpoint failed if the properties passed in had durable properties set.
FTL-11226	Fixed an issue where partially connected zones could experience a buildup of acknowledgments for messages exchanged between clusters.
FTL-11129	Fixed an issue where uploading a JSON file with `tibftladmin` could flag an error in the administrative GUI, but without any details available.
FTL-11127	Fixed a defect in the FTL client library where in a call to `tibMessage_ToString` with a small buffer could cause the client to abruptly exit.
FTL-11115	Fixed an issue where if `tibRealm_Unsubscribe` was called while the client had a corresponding subscriber object still open, or if `tibRealm_RemoveMap` was called while the client had a corresponding map object still open, the call was likely to time out.

Key	Summary
FTL-11113	Fixed an issue where eFTL client connections could fail if an HTTP load balancer was placed between the client and the FTL server.
FTL-11024	Fixed an error where, when a durable monitoring call (`GET api/v1/persistence/clusters/<cluster>/durables/<durable>`) was issued, the FTL server REST API could mistakenly return 404 Not Found if the persistence service was unavailable.
FTL-10939	Fixed an issue where, when an FTL server exited, sometimes not all services exited.
FTL-10801	Fixed an issue where store limit overruns or cluster reconnects might not be logged.
FTL-10546	The FTL server now rejects the yaml configuration if the supplied password in the yaml file uses the 'stdin' form.

Release 6.4.0

Key	Summary
FTL-11283	Fixed a defect in the client library that was failing to throw an exception if the 'send' capabilities were not set on the transport.
FTL-11183	Fixed an issue where a REST web request formats output incorrectly included an internal field entry.
FTL-11177	Fixed a realm service defect in which the internal `_inboxEndpoint` was erroneously removed from applications upon migration from 5.4.x.
FTL-11163	Fixed a monitoring issue where a store's `disk_state` is always incorrectly `0` in the response for REST web call `api/v1/persistence/<cluster_name>/servers/<server_name>`. Also, note that a successfully completed dump-to-disk operation for a persistence service is indicated when `disk_state` becomes `2`.
FTL-11155	Fixed an issue that would occasionally, but rarely, cause a persistence

Key	Summary
	service to unexpectedly exit.
FTL-11149	Fixed a client library defect that could cause the client to deadlock when publishing to and receiving from persistence service.
FTL-11144	Fixed an persistence service defect, where under certain conditions, the following warning was continually repeated: `realm <timestamp> warn ftl: Subscriber(1075) purge. Not found!`
FTL-11134	Fixed a defect in the persistence service where the monitoring counters that return the number of messages out of the persistence service were incorrect.
FTL-11126	Fixed an issue where a persistence service would occasionally restart with `disk_mode` set to `swap`.
FTL-11121	Fixed an issue where a persistence service might fail to join its quorum if given an invalid state file to load.
FTL-11082	Fixed an issue where, in rare cases, a persistence service could under-report store message and byte counts after catching up from another persistence service.
FTL-10979	Fixed an issue where, if a store was migrated from 5.4.x, after migration the inbound/outbound message monitoring counters would not function for that store.
FTL-11075	Fixed a persistence service defect that could cause the persistence service to abruptly exit. This symptom could occur if message swapping was enabled.
FTL-11062	Fixed a realm service defect that was preventing the log level to be set in the presence of multiple eFTL services.
FTL-11032	Fixed an issue where a call to `tibRealm_Close` could occasionally stall for clients that received inbox messages via a persistence store.

Key	Summary
FTL-11031	Fixed an issue where a subscriber configured for asynchronous acknowledgments would sometimes reconnect if a follower persistence service rejoined the quorum.
FTL-11027	Fixed a client library defect that could cause the client to abruptly exit. This symptom could occur during a reconnect to the persistence service.
FTL-11023	Fixed an issue where map operations using a lock could timeout if a persistence service concurrently rejoined a quorum.
FTL-11022	If a deployment determines that a client or satellite will not reply, this is now reflected in realm deployment status queries.
FTL-11020	Fixed an issue in the FTL Server in which the FTL Server failed to process boolean arguments.
FTL-11006	Fixed an issue where when starting a set of FTL Servers for the first time, the initial deployment could occasionally time out.
FTL-11005	Fixed an issue where On occasion, a durable subscriber configured for asynchronous acknowledgments could drop an ack when reconnecting to a shared durable.
FTL-10997	Via the web API call for server status (`api/v1/server`), you can now see whether disaster recovery realm services are in sync with the latest deployment (`"dr_in_sync": true`).
FTL-10991	Fixed an issue where, in rare cases, a persistence service could crash if a client makes a call to `tibSubscriber_Create` immediately after calling `tibSubscriber_Close` for a subscriber on the same durable.
FTL-10987	Fixed a client library defect that could cause clients or FTL services to abruptly exit.
FTL-10979	Fixed an issue where, if a store was migrated from 5.4.x, after migration the inbound/outbound message monitoring counters would not function for that store.

Key	Summary
FTL-10978	Fixed an issue where, on rare occasions, a realm service could stall during the initial deployment, preventing its FTL Server from fully initializing.
FTL-10977	Fixed an issue where, in rare cases, if a deployment changed the publisher mode of a store from `store_confirm_send` to `store_send_noconfirm`, a client with existing publishers might not return from a call to `tibPublisher_Send/tibPublisher_SendMessages`.
FTL-10972	Fixed an issue where, in rare cases, a persistence service could stall when a quorum reformed.
FTL-10969	Fixed a client library defect where `tibMessage_ToString` was returning a string containing internal field names.
FTL-10960	Fixed an issue where store names were required to be unique across clusters. Note that while store names no longer need to be unique across clusters, in the presence of multiple zones, stores names must be unique when associated with a zone.
FTL-10957	Fixed an issue where, in rare cases, a persistence service could crash if it started while a deployment was in progress.
FTL-10947	Fixed an issue where, in rare cases, a client publishing to a store configured with `store_confirm_send` could deadlock during a realm deployment.
FTL-10891	Fixed an issue where calls to `tibPublisher_Send` could timeout if a persistence service concurrently rejoined a quorum.
FTL-10783	Fixed an issue where durable message TTLs would reset if the leader of a persistence cluster changed.
FTL-10764	Fixed an issue where occasionally a sync ack durable subscriber would needlessly disconnect from the persistence service if a quorum follower rejoined the quorum.
FTL-10669	Fixed a client library defect that could cause a client application using a shared memory transport to deadlock when an administrator updated the

Key	Summary
	realm in a way that affected the transport.
FTL-10635	Fixed an issue that allowed FTL clients to attempt to reconnect to the same port from which it just lost its connection, when alternate ports were available.
FTL-9550	Fixed a defect in the FTL Server that could cause it to abruptly exit if the certification file is empty.
FTL-7627	In the administrative GUI, added a an icon/check box etc to show all the fields or reset to the default fields.

Release 6.3.1

Key	Summary
FTL-11075	Fixed a persistence service defect that, with message swapping enabled, could cause the persistence service to abruptly exit.
FTL-11051	Fixed a client library defect that prevented the client from accepting a new deployment after a reconnect to the FTL Server.
FTL-11048	Fixed a client library defect in which the application-specified identifier was erroneously being overwritten after a deployment.
FTL-10721	Fixed a defect in the FTL Server GUI in which the log level and centralized logging flags were not displayed correctly after their values were changed.

Release 6.3.0

Key	Summary
FTL-10928	Fixed an issue for when environment variable `ALL_PROXY` was set, the FTL server would not start.
FTL-10915	Corrected usage help text for `ftlserver--init-auth-only` and `--init-security`.

Key	Summary
FTL-10859	Fixed an issue where stores defined in the default cluster did not have a byte limit by default.
FTL-10858	Fixed an issue where an unsubscribe operation would occasionally fail.
FTL-10853	Fixed an Administration GUI issue where the endpoint description was labeled Application Description.
FTL-10847	Fixed an issue where a persistence service storing messages whose formats have no fields defined would fail when syncing another persistence service.
FTL-10830	Fixed an issue where if the leader of the default cluster was shut down, messages would start to accumulate in `ftl.system.mon.store`.
FTL-10826	Fixed an issue where auxiliary FTL servers would fail to initialize if provided a 5.4.x realmserver database file.
FTL-10824	Fixed an issue where automatic re-deployments for out-of-sync clients would take longer than expected.
FTL-10814	For the function `tibRealm_CreateMap` you can now supply NULL for `endpointName`. If NULL is specified for `endpointName` then the client library sets `map` as the `endpointName`.
FTL-10805	Fixed an Administration GUI issue where the Firebase Cloud API key text box (under eFTL channel settings) was too small to display the entire key.
FTL-10795	Fixed an issue where a persistence service configured for store forwarding would not close its connections to other persistence clusters when a zone-scoped store was deleted.
FTL-10789	A structure definition for the FTL advisory message was added to facilitate unmarshaling a received advisory message, for simpler processing.
FTL-10788	For consistency, the function `group.Version()` was added to the FTL golang Group API.

Key	Summary
FTL-10782	Fixed a defect where the FTL golang API did not properly handle properties where the property value was `ftl.Message`.
FTL-10772	Fixed an issue where if an FTL client subscribed with an invalid content matcher, the persistence server could abruptly exit.
FTL-10751	Fixed an issue where if an FTL server was restarted with the same name but a different host/port, connections among FTL servers were sometimes not made properly.
FTL-10743	Fixed an issue where a call to `tibRealm_Connect` would not fail immediately, as it should, if a password was specified without a username.
FTL-10725	Fixed an issue where if a persistence service encountered an invalid dump file, it would not exit.
FTL-10722	Fixed an issue where `ftlserver` would occasionally fail to restart a service correctly after that service became unresponsive.
FTL-10716	Fixed an issue where map iterators using matchers returned incomplete results if there was a large number of key/value pairs in the map.
FTL-10715	Fixed an issue where map operations with a lock could fail unnecessarily if the client was disconnected from the cluster during the request.
FTL-10699	Fixed an issue where, when deleting an eFTL channel, the eFTL server could report an exception following deployment, requiring a restart of. the eFTL server.
FTL-10690	Fixed a defect in which, on rare occasions, DTCP transport connections could be delayed by a minute or more.
FTL-10685	Fixed an issue in the client library to improve how fast a client can connect/reconnect to the persistence service.
FTL-10684	Fixed an issue where deployment status would be reported inconsistently across different FTL servers while a deployment was in progress.

Key	Summary
FTL-10681	Fixed a defect in which an FTL server could fail to continue a deployment while a disaster recovery FTL server was connected.
FTL-10614	Fixed an issue where a `GET api/v1/persistence/clusters/<cluster>/store/<stores>/durables/<durable>` or `GET api/v1/persistence/zones/<zone>/store/<stores>/durables/<durable>` request could return, in addition to `<durable>`, other durables not named `<durable>`.
FTL-10609	Fixed a defect where if a deployment changed the name of the durable template assigned to an endpoint, associated clients did not automatically prompt for a restart.
FTL-10594	Fixed a defect in which, if very high latency existed between a client and a persistence service, the client could fail to add a durable subscriber to an event queue.
FTL-10592	Fixed a defect in which, while sending a message to an inbox via a store, a client could hang.
FTL-10585	Fixed an issue where, on rare occasions, an FTL server that started a deployment could erroneously time out other FTL servers in the same cluster.
FTL-10573	Fixed a defect in which, if a client's application instance had been deleted from the realm configuration, that client could fail to get a deployment.
FTL-10572	Fixed a defect where when making a change to an application definition, the FTL server could fail to maintain the configured ordering of application instances.
FTL-10553	Fixed a defect with .NET clients, when if a call to TibMap.Key(key, lock) failed, the API erroneously returned a non-null message.
FTL-10552	Fixed a defect in which a client with many (i.e., several thousand) standard durable subscribers could experience a delay of several minutes before

Key	Summary
	receiving messages for some subscribers.
FTL-10548	Fixed a defect in which, for a persistence cluster definition, increasing the `pserver_timeout_pserver` value to greater than the default value of 3 seconds would cause the quorum to take longer than expected to form.
FTL-10540	Fixed a defect for while a client was creating an inbox subscriber on an endpoint with a store, that client could experience timeouts.
FTL-10529	Fixed an issue where an eFTL channel's connection and subscription monitoring counters could be incorrect when using the eFTL REST API.
FTL-10524	Fixed a defect where if `client_pserver_heartbeat` was set to less than 2 seconds in a persistence cluster configuration, clients of that cluster were unable to connect.
FTL-10476	Fixed an issue where client transport settings in a default persistence cluster would reset after an FTL server restart.
FTL-10475	Modified Administration GUI numeric field displays to show a consistent and limited number of digits after a decimal.
FTL-10462	Fixed an Administration GUI issue where monitoring data did not show client instant changes.
FTL-10460	Fixed a defect where if any FTL server in a disaster recovery cluster with persistence services using a DTCP cluster transport was restarted, the quorum might not reform.
FTL-10459	Improved exception stack prints in the FTL Golang API and the realm service.
FTL-10458	Fixed a defect where if a `GET /channel/v1/subscribe/<durname>` REST call was immediately followed by a `DELETE /channel/v1/subscribe/<durname>` REST call in a different connection, the DELETE call could fail.

Key	Summary
FTL-10457	Fixed a defect where, on rare occasions, a persistence service in a store-forwarding zone was unable to maintain its connection to another persistence service in another cluster of the zone.
FTL-10452	Fixed a defect in the realm service that was preventing eFTL clients from invoking `KVMap` REST APIs when authentication is enabled on the eFTL server.
FTL-10448	Improved the administration GUI so that when viewing durables, content matcher strings are fully visible without the need to scroll.
FTL-10447	Fixed an issue where you could not always purge applications that were no longer running.
FTL-10442	Fixed a defect where if persistence cluster configuration changes required the quorum to restart (e.g., timeout or heartbeat intervals), the quorum did not always reform.
FTL-10437	Samples have been updated to be more flexible and available in more languages.
FTL-10430	In the samples file `dotnet/Makefile.Linux`, removed a line at the top of the file with `GroupClient`.
FTL-10419	Fixed a defect where when under heavy load, if realm components became unavailable, the realm service or go clients might exit.
FTL-10418	Fixed a defect where if the satellite core FTL servers quorum reforms while the satellite servers are disconnected from the primary cluster, a satellite FTL server could abruptly exit.
FTL-10413	Fixed a defect in which redeploying a disaster recovery cluster could erroneously generate a `read only` error message.
FTL-10404	Fixed a defect in which an FTL server internal timeout could erroneously return an HTTP `404` for certain HTTP requests.

Key	Summary
FTL-10392	Fixed an issue where the DR (Disaster Recovery) realm service was not updated with a deployment in a timely manner.
FTL-10366	Fixed a defect where core-server quorum delays (e.g., a slow disk) prevented some of the FTL servers from initializing properly.
FTL-10363	Fixed a defect in which processes that include the tib library, including tibftlserver and its modules, could on rare occasions crash when exiting.
FTL-10237	Fixed an issue where setting certain http proxy-related environment variables, such as `http_proxy`, could prevent basic FTL functioning.
FTL-10201	Fixed a defect where an auxiliary FTL server that was disconnected from the core FTL servers for an extended period of time could cause a REST API shutdown command to the auxiliary FTL server to fail.
FTL-10185	Fixed an issue where ephemeral durables would not automatically expire.

Release 6.2.0

Key	Summary
FTL-10372	Fixed a defect in the eFTL Service that caused Apple push notifications to fail.
FTL-10347	Fixed a defect where the default cluster did not always reform a quorum following a restart of 2 out of 3 FTL servers.
FTL-10346	Fixed a defect where an FTL Server sometimes generates error message `error processing cluster commit`.
FTL-10333	Fixed a defect where, when authentication is enabled, the persistence service could abruptly exit.
FTL-10332	Fixed a defect where the persistence service would sometimes abruptly exit while printing a message.
FTL-10328	Fixed a defect where a server removed from a cluster sometimes caused

Key	Summary
	the quorum protocol to abruptly exit on heartbeats.
FTL-10286	Fixed a defect where an FTL server in a cluster sometimes fails before a quorum is first formed.
FTL-10258	Fixed a client library defect that caused a client to abruptly exit when reconnecting to a persistence service.
FTL-10257	Fixed a defect that caused the realm service in the core servers to abruptly exit when it loses heartbeats from an affiliate.
FTL-10243	Fixed a defect where restarted servers in a cluster sometimes fail to form a quorum and abruptly exit.
FTL-10180	The `ftlstart` script from the `samples/scripts` directory no longer deletes the auto-generated YAML configuration file.
FTL-10093	Fixed a defect in the realmservice that caused ActiveSpaces tibdg process from getting status of ActiveSpaces nodes and proxies.
FTL-10090	Fixed a defect that caused the persistence service to abruptly exit upon shutdown, instead of a clean shutdown.
FTL-10049	Fixed a defect where an FTL server sometimes did not run required services.
FTL-10047	Fixed a defect where the primary realm service would sometimes stop processing satellite server connect requests.
FTL-9957	Fixed a Windows Settings defect where the `Uninstall` buttons for FTL and eFTL were unresponsive.
FTL-9956	Fixed a memory leak in the client library where, as seen with SubscribeToInbox, excessive subscribers are created.
FTL-9950	Fixed an FTL Server defect that prevented from it parsing the YAML configuration.

Key	Summary
FTL-9941	Fixed a defect where an eFTL server failed to start due to an incorrect max log file size parameter.
FTL-9935	Fixed a defect where sometimes after long quorum disruption, subscribers were timed out early.
FTL-9930	Fixed a defect where sometimes a client would abruptly exit while reconnecting to an FTL server.
FTL-9901	Fixed a defect where persistence servers sometimes reported their status as out-of-sync.
FTL-9814	Removed the occurrence of `???` in all server output logs.
FTL-9770	Fixed a defect in which the UI and the Web API allowed a user to change the durable/template type after it was deployed.
FTL-9745	Fixed a defect where an endpoint with no associated store/template incorrectly used a server based transport, but no validation error was generated
FTL-9684	Improved UI performance when there are large number of clients.
FTL-9360	Fixed a defect where a client used a *persistent* HTTP connection and sent *multiple* REST requests that the FTL server routes to different target services (for example, realm service and eFTL service) and some requests then failed with HTTP error `404 page not found`.
FTL-9138	auxiliary persistence servers do not get the realm server available/unavailable advisory anymore.
FTL-8716	Fixed a defect where durable subscribers could throw an exception while acknowledging messages after an extended network disconnect from the persistence service. This symptom sometimes affected shared durables and standard durables with prefetch.
FTL-8464	Fixed a persistence service defect that could cause it to abruptly exit during

Key	Summary
	shutdown.
FTL-7959	Fixed a defect in the client library that could cause a client using a DTCP transport to abruptly exit.

Release 6.1.0

Key	Summary
FTL-9864	Fixed a memory leak associated with durable subscribers.
FTL-9687	Fixed a defect in which the realm service could erroneously omit clients from dynamic TCP transports.
FTL-9766	Modified a default value for new eFTL clusters to allow unlimited client connections.
FTL-9751	Fixed a defect in which it was possible to circumvent realm validation when defining or modifying a static durable or a dynamic durable template.
FTL-9724	Improved memory performance of the realm service when managing dynamic TCP transport buses.
FTL-9723	Modified the default parameters for new persistence stores and durable templates in `ftl.default.cluster` for improved performance.
FTL-9651	Fixed a defect in which realm validation erroneously allowed hub-and-spoke zones without a hub cluster.
FTL-9627	Improved throughput performance of the client API.
FTL-9626	Fixed a defect in which simultaneous realm deployment operations could interfere with one another.
FTL-9611	Fixed a defect in which the FTL server incorrectly parsed IPv6 literal addresses in its configuration file.
FTL-9607	Improved performance of TCP and dynamic TCP transports.

Key	Summary
FTL-9584	Fixed a client library defect in which creating an inbox subscriber on an endpoint that disables the receive inbox ability could cause the client to abruptly exit.
FTL-9582	Fixed a Go language client library defect in which clients could abruptly exit in the method `Message.String()`.
FTL-9575	Improved error reporting during parsing the FTL server configuration file.
FTL-9568	Fixed a Go language API defect in which calling `Realm.Unsubscribe()` on an uninitialized realm object could cause programs to abruptly exit.
FTL-9562	Fixed a memory leak in `tibmongateway`.
FTL-9557	Improved performance of FTL servers with many clients.
FTL-9544	Fixed a defect in which the FTL server erroneously reported the status of some internal clients.
FTL-9524	Fixed a defect in which the monitoring gateway could abruptly exit while communicating with the Prometheus pushgateway.
FTL-9515	Fixed a client library defect in which the realm server could abruptly exit while attempting to get a non-existent field from a message.
FTL-9512	Fixed a client library defect in which internal clean-up of dynamic formats could interfere with event queue timers.
FTL-9510	Fixed a client library defect in which client monitoring API and GUI could omit dynamic formats that begin with an underscore character.
FTL-9507	Fixed a memory leak in the client library affecting durable subscriptions.
FTL-9500	Fixed a client library defect that prevented 6.x clients from connecting to realm servers from Release 5.4 and earlier.
FTL-9497	Fixed a persistence service defect in which clients did not failover to a new

Key	Summary
	persistence leader after network segmentation.
FTL-9495	Fixed a defect that affected durable discard behavior.
FTL-9493	Fixed a defect in which the modifications to the definition of the default persistence cluster, `ftl.default.cluster`, were not persisted to the realm database. After stopping all of the core FTL servers, then restarting them, those modifications were lost.
FTL-9492	Fixed a defect in which FTL server could exhaust file descriptors and so could not accept client connections nor service REST requests.
FTL-9491	Fixed a defect in which the realm service could abruptly exit while outputing log messages.
FTL-9452	Fixed a defect in which a persistence service could abruptly exit after reconnecting to its cluster following a period of disconnection.
FTL-9418	Fixed a defect in which realm validation erroneously allowed an application endpoint associated with a wide-area store but not associated with a dynamic durable template.
FTL-9356	Fixed a defect in which client programs could erroneously publish to internal stores.
FTL-9338	Fixed Go language API defects in which constants were misspelled. For more detail, see Changes in Functionality. If you copied the incorrect names of these constants into your Go program code, correct them before migrating to Release 6.1 or later.
FTL-9336	Improved memory usage in the realm service.
FTL-8930	The realm connect call now emits warning logs when it cannot connect to an FTL server.
FTL-8724	Fixed a GUI defect in which the zones grid misreported the number of non-

Key	Summary
	forwarding clusters.
FTL-8511	Fixed a defect in which multicast transports could erroneously not transmit backlog outbound message data packets.
FTL-8507	Fixed a defect in which multicast transports could incorrectly sequence outbound message data packets. This symptom could occur if the transport send rate parameter was unlimited.

Release 6.0.1

Key	Summary
FTL-9463 FTL-9430	Improved scalability of client connections.
FTL-9444 FTL-9416	Fixed a defect in which the GUI and REST API did not include wide-area stores in the list of persistence stores.
FTL-9434	Fixed a GUI defect affecting the durable interest field.
FTL-9429 FTL-9428	Fixed a defect in which the FTL server could update heartbeat values but continue to timeout clients according to old values.
FTL-9417	Fixed a memory leak on Windows platforms triggered when clients connected to the FTL server.
FTL-9402	Fixed a defect in which a restarted FTL server could erroneously delete a dynamic durable template.
FTL-9395	Fixed a client library defect in which FTL servers or clients could abruptly exit after persistence services reformed a quorum.
FTL-9394	Fixed a defect in which realm services and eFTL services could not start because of port collisions.

Key	Summary
FTL-9391	Fixed a defect in which the GUI could list the status of a wide-area persistence store as *unknown*.
FTL-9389	Fixed a GUI defect in which purging wide-area durables was ineffective.
FTL-9357	Fixed a defect in which the persistence service did not automatically create the data directory if it was different from the `savedir`.
FTL-9324	Fixed a defect in which the FTL administration utility did not print a label when it prompted for a password.
FTL-9322	Fixed a defect in which the FTL administration utility printed status fields that are no longer relevant.
FTL-9300	Fixed a defect in which FTL servers that could not reach their quorum leader could log many warning messages, such as the following: `warn ftl: Error publishing DTCP Transport Join notification message: unavailable`
FTL-9273	Fixed an issue in which an FTL server that restarted could hold state information about clients that had stopped or disconnected. The monitoring GUI and web API could report incorrect state about those clients.
FTL-9271	Fixed a defect in printing the names of auxiliary servers.
FTL-9259	Fixed a defect in which FTL servers could output log messages such as the following: `warn ftl: bad role ftl-primary` `Invalid message type received from peer 'NULL'`
FTL-9224	Clarified logging output during migration from Release 5.x to Release 6.x.

Key	Summary
FTL-9217	Fixed a defect on Windows platforms in which a cluster of FTL servers could become unstable if the data directory were a network drive.
FTL-9194	Fixed a defect in which an FTL server that restarted could hold state information about clients that had stopped or disconnected. The monitoring GUI and web API could report incorrect state about those clients.
FTL-9078	Fixed a GUI defect in the cluster details panel in which users could appear to remove an externally reachable address even when not in edit mode.
FTL-9060	Fixed a client library defect in which applications and services using dynamic TCP transports could abruptly exit.
FTL-8949	Fixed a defect in which the GUI appeared to permit purging of internal persistence stores.
FTL-8918	Fixed a defect in which the GUI varied the order of services in the server status pane.
FTL-8905	Fixed a misleading warning message when starting an empty data store.
FTL-8541	Fixed a client library defect in which applications and services could abruptly exit after the Close cleanup call.
FTL-8893	Fixed a defect in which the FTL server GUI could display hidden internal endpoints even when the checkbox option "Show hidden FTL Objects" was cleared.
FTL-8308	Fixed an defect in which the web API command to compact the realm server's database was not effective on Windows platforms.

Release 6.0.0

Key	Summary
FTL-9275	Fixed a defect in which clients using dynamic TCP transports could not reconnect to the realm server after the realm server had been restarted. In

Key	Summary
	some instances, clients became unresponsive. This symptom could also affect services that used dynamic TCP transports.
FTL-8551	Fixed a defect in which monitoring data from TIBCO ActiveSpaces was not available for display.
FTL-8521	Clarified the documented behavior of the FTL administration utility.
FTL-8505	Fixed a defect in which group members could miss status updates.
FTL-8491	Fixed a defect in which new DTCP clients could exhibit significant delays in joining the bus, while they attempted to connect with defunct DTCP clients.
FTL-8472	Fixed a defect in which clients for which centralize logging is enabled could abruptly exit.
FTL-8470	Fixed a defect in which a multicast subscriber could erroneously miss initial messages from a sender.
FTL-8410	Fixed a defect in which clients that collect subscription statistics could abruptly exit.
FTL-8403	Fixed a misleading error string that was output when exceeding a message redelivery count.
FTL-8401	Fixed a Java client library defect in which `FTL.setLogHandler` could cause the client to abruptly exit.
FTL-8390	Fixed a defect in which exception strings could contain unreadable characters.
FTL-8374	Fixed a defect in which publisher, subscriber, and map calls rejected the persistence retry value -1.
FTL-8371	Fixed a API defect that affected the persistence retry duration properties in calls to publisher and subscriber methods.

Key	Summary
FTL-8240	Relaxed a constraint: application clients no longer require restart after administrators modify or remove a mapping from a subscriber name to a durable name.
FTL-8171	Fixed a log service defect related to incorrect parameters.
FTL-8142	Fixed a log service defect related to unrecognized parameters.
FTL-8092	Fixed a log service defect related to realm server restart.

Known Issues

The table lists known issues in Release 6.7.1 of TIBCO FTL software.

Key	Summary
FTL-12619	**Summary**: FTL 6.4 clients that have endpoints configured with server-based transports cannot use request/reply calls or request/reply inboxes with FTL 6.5 or later clients. **Workaround**: Upgrade the clients to 6.5 or later.
FTL-12580	**Summary**: Although the API to backup a realm database (`api/v1/server` command `backup`) returns 200 on success, the backup has not necessarily completed when the request returns. **Workaround**: None.
FTL-12526, FTL-12041	**Summary**: When using the administrative GUI to access multiple clusters on the same host name using the same browser, a session might abruptly terminate, or logging in might be inhibited. **Workaround**: Use a different browser type for each FTL cluster. Before accessing the GUI, clear the browser cache on all browsers used.

FTL-12517	**Summary**: On Linux Platforms, if you have *both* FTL 6.7.0 and eFTL 6.7.0 installed and are using yum or zypper package managers to upgrade to FTL/eFTL 6.7.1, the upgrade procedure can fail.
	
	Workaround: When installing, follow these steps:
	
	1. Install FTL/eFTL 6.7.1 together.
	• For yum:
	`yum install -y TIB_ftl_6.7.1/rpm/*.rpm TIB_eftl_6.7.1/rpm/*.rpm`
	• For zypper:
	`zypper install TIB_ftl_6.7.1/rpm/*.rpm TIB_eftl_6.7.1/rpm/*.rpm`
	2. Manually move `/opt/tibco/eftl/6.7/bin/modules/tibeftlserver` to `/opt/tibco/ftl/6.7/bin/modules/tibeftlserver`.

Key	Summary
FTL-12181	**Summary**: For the Administrative GUI, resizing the window sometimes hides the vertical scrollbar. **Workaround**: Resize to a larger window or avoid resizing.
FTL-11597	**Summary**: If an FTL client creates many subscribers on durables using async acks and a low ack batch time, the client experiences high CPU usage. **Workaround**: None.
FTL-11185	**Summary**: If the cluster message swapping setting `disk_mode` is set to `swap` and the disk is being accessed via NFS, client or quorum timeouts of up to a few seconds can occur. **Workaround**: Use a local filesystem for the location of swap files, via the new cluster message swapping setting `swapdir`.
FTL-10631	**Summary**: Inbox subscribers do not explicitly acknowledge message delivery if the subscriber's endpoint has no direct path transports associated. **Workaround**: None.
FTL-10393	**Summary**: FTL 6.3.x or later versions are not compatible with EMS 8.5.0 or earlier releases. **Workaround**: Upgrade to EMS 8.5.1.
FTL-10335	**Summary**: When 6.2 or later clients connect to a 5.4 realm server, the realm server logs a panic. The 5.4 realm server does not crash, and is functional after it logs a panic. 6.2 clients are able to successfully connect. **Workaround**: Upgrade all FTL servers to 6.7.1 before upgrading clients to 6.7.1.
FTL-9499	**Summary**: Importing realm definition data into Release 6.0.1 that had been output in JSON format from Release 6.0.0 could cause the FTL server to reject the deployment even though it did not report validation errors. **Workaround**: Remove the definitions of `_clientEndpoint`, `_inboxEndpoint`, `_loggingEndpoint`, and `_monitoringEndpoint` from `_GroupServer` application.

Key	Summary
FTL-9293	**Summary**: During migration from Release 5.4 to Release 6.x, it is possible that the old realm server and the new FTL server could both assign ordinals to group members. This could result in thrashing behavior by the group members. **Workaround**: Immediately stop the old 5.4 realm server.
FTL-9281	**Summary**: The REST command to compact an FTL server database is not functional. **Workaround**: None.
FTL-9231	**Summary**: When FTL servers are not in a quorum, the GUI displays incorrect monitoring data. **Workaround**: None
FTL-8319	**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.
FTL-8280	**Summary**: Subscribers on the monitoring endpoint could miss the final monitoring metrics from a closing client. **Workaround**: None.
FTL-7718	**Summary**: Debian Linux Uninstall After uninstalling Debian Linux packages, the command **`dpkg -query`** reports that the package `tibco-ftl-thirdparty` is still installed (even though it has, in fact, been successfully uninstalled). **Workaround**: This command resolves this issue by removing package information from the database. `sudo dpkg --purge tibco-ftl-thirdparty`
FTL-7161	**Summary**: The realm server GUI does not support Internet Explorer (IE) 11 and earlier.

Key	Summary
	However, it does support Edge (Windows 10). **Workaround**: Use any supported browser as listed in the file `readme.txt`.
FTL-6708	**Summary**: On Windows 2008 platforms, the realm server can exit abruptly during a deployment. This symptom can occur only if both a primary and a backup realm server are running on Windows 2008 host computers, and only if the realm server data directory is on a network mounted drive. **Workaround**: Choose either of two workarounds: • Upgrade the host computers to Windows Server 2012, Windows Server 2016, or Windows 10. • Move the data directory to a local drive.
FTL-5909	**Summary**: IPv4 and IPv6 The realm server interprets wildcard addresses narrowly in its command line parameters. That is, it interprets `[::]` to mean any IPv6 address, and `0.0.0.0` to mean any IPv4 address. This interpretation could change in a later release, so do not depend on it. **Workaround**: Specify `*` as the wildcard to include any IPv4 or IPv6 address.
FTL-5630	**Summary**: Changing a Persistence Cluster If you change the name of a persistence cluster, all running persistence servers in the cluster require restart. However, the realm server GUI does not detect this condition. **Workaround**: Ensure that you restart persistence servers after changing their cluster's name.
FTL-4976	**Summary**: IPv4 Overrides IPv6, or Vice Versa If the realm server command line specifies `--http` *`hostname`*`:`*`port`*, and *`hostname`* resolves to both IPv4 and IPv6 addresses, then the realm server binds to only the computer's preferred address. For example, if the preferred address

Key	Summary
	is IPv4, then clients cannot connect to the IPv6 address. If the preferred address is IPv6, then clients cannot connect to the IPv4 address. **Workaround**: Specify `--http *:`*`port`*, using `*` to cover the two interfaces.
FTL-4386	**Summary**: Chrome and Safari browsers can no longer access TIBCO HTML documentation from a file system, that is, using the `file://` protocol. **Workaround**: Access using a different browser, or access the HTML documentation through the web, that is, using the `http://` protocol.
FTL-496	**Summary**: On Microsoft Windows platforms (only), `tcp` and `shm` transports support at most 60 simultaneous connections. For example, when 60 `tcp` transports are connected to one listening transport in an application program (for example, a server hub application), the 61st cannot connect, and FTL logs an error. Similarly, when `shm` transports in 60 application processes on the same host computer are connected to the same shared memory segment, the 61st cannot connect, and FTL logs an error. When inline mode is disabled, this limitation of 60 connections applies separately to each transport. However, when inline mode is enabled - by using the property `TIB_EVENTQUEUE_PROPERTY_BOOL_INLINE_MODE` when creating an event queue - then this limitation applies cumulatively across all the transports of all the endpoints (that is, subscribers) associated with that queue. The sum of all the connections to those transports cannot exceed 60. **Workaround**: None.

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.

Product-Specific Documentation

Documentation for TIBCO FTL® is available on the TIBCO FTL® Product Documentation page.

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

`TIBCO_HOME/release_notes/TIB_ftl_6.7.1_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`.

TIBCO FTL® Documentation Set

- *TIBCO FTL Concepts* This booklet presents an intuitive introduction to the fundamental concepts of FTL software.
- *TIBCO FTL Development* Application developers and architects read this manual to understand concepts relevant in any supported programming language.
- *TIBCO FTL API Reference* Application developers use this HTML documentation to learn the details of the FTL API in specific programming languages.
- *TIBCO FTL Administration* Administrators read this manual to learn how to use the FTL server, its interfaces, and its services, and how to define a realm. Developers can also benefit from understanding FTL software from an administrator’s perspective.
- *TIBCO FTL Security* This manual contains security-related tasks for administrators and security tips for application developers.

- *TIBCO FTL Monitoring* Administrators read this manual to learn about monitoring and metrics. Developers read this manual to learn how an application can subscribe to the stream of monitoring data.
- *TIBCO FTL Shifting to FTL* This manual contrasts TIBCO FTL with TIBCO Enterprise Message Service™, and offers suggestions to smooth your transition to TIBCO FTL. Application developers, architects, and administrators familiar with TIBCO Enterprise Message Service read this manual.
- *TIBCO FTL Installation* Read this manual before installing or uninstalling the product.
- *TIBCO FTL Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

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.

Updated Resources on TIBCO Community

Supplemental resources are now distributed at the TIBCO FTL Community Wiki in the Reference Info tab. You can always find the latest versions of these resources in that location.

Those resources include *TIBCO FTL Quick Start Guide* and *TIBCO FTL Tutorials*. They also include sample FTL server configuration files and sample realm definition files.

TIBCO eFTL™ Documentation Set

TIBCO eFTL software is documented separately. Administrators use the FTL server GUI to configure and monitor the eFTL service. For information about these GUI pages, see the documentation set for TIBCO eFTL software.

How to Contact TIBCO Support

Get an overview of TIBCO Support. You can contact TIBCO Support in the following ways:

- For accessing the Support Knowledge Base and getting personalized content about products you are interested in, visit the TIBCO Support website.
- 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 TIBCO Support website. 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 TIBCO Community.

Legal and Third-Party Notices