TIBCO® RTView® for TIBCO ActiveSpaces® User's Guide

Version 7.1.2

TIBCO Software Inc.
3303 Hillview Avenue
Palo Alto, CA 94304-1213

CUSTOMER SUPPORT
For an overview of TIBCO Support Services, and information about getting started with TIBCO Product Support, visit this site:

http://www.tibco.com/services/support/default.jsp

If you already have a valid maintenance or support contract, visit this site:

http://support.tibco.com

Entry to this site requires a username and password. If you do not have a username, you can request one.

Preface

Welcome to the TIBCO® RTView® for TIBCO ActiveSpaces® User's Guide

Document Conventions

This guide uses the following standard set of typographical conventions.

Convention	**Meaning**
italics	Within text, new terms and emphasized words appear in italic typeface.
boldface	Within text, directory paths, file names, commands and GUI controls appear in bold typeface.

Courier	Code examples appear in Courier font:
	`amnesiac > enable`
	`amnesiac # configure terminal`

< >	Values that you specify appear in angle brackets: **interface <ipaddress>**

Third Party Notices

Please refer to the **LICENSES_thirdparty.txt** file from your product installation.

CHAPTER 1 Quick Start

Whether you want to evaluate TIBCO® RTView® for TIBCO ActiveSpaces® (also referred to as the *ActiveSpaces Monitor* or, the *Monitor*) for purchase, or you TIBCO RTView for TIBCO ActiveSpaces already purchased it and want to install and set it up--This chapter is intended for you.

These instructions describe the minimum steps needed to get the Monitor up and running (using default settings, and Eclipse Jetty as the application server which is delivered with the Monitor). Most of the configurations are defined using the RTView Configuration Application.

This step must be performed before running any deployment of the Monitor.

After you complete these instructions, see Configuration to optionally modify your setup or take advantage of additional features.

This chapter contains:

- Prerequisites
- UNIX/Linux Quick Start Steps
- Windows Quick Start Steps

Prerequisites

- Supported Java JDK (1.8+, see **rtvapm/README_sysreq.txt** for full list)
- TIBCO ActiveSpaces 4.1+
- TIBCO FTL 6.1+
- Application Server (for example, Eclipse Jetty which is delivered with the Monitor, or Tomcat 8.5+)

UNIX/Linux Quick Start Steps

Do the following to download and install ActiveSpaces Monitor and define the TIBCO ActiveSpaces Realms to be monitored. Note that TIBCO ActiveSpaces is monitored via TIBCO FTL which means that you configure TIBCO FTL.

1. Download **TIB_rtview-as_<version>.zip** to your local UNIX/Linux server.

   Note: If using UNIX, do not include spaces in your installation directory path. The scripts will not function properly if spaces are included in the installation directory path.

2. Extract the files:

   unzip -a TIB_rtview-as_<version>.zip

   You should see a top level directory, **TIB_rtview-as** containing two subdirectories: **projects** and **rtvapm**.

3. Set the **JAVA_HOME** environment variable to point to your Java installation. For example:

   export JAVA_HOME=/opt/Java/jdk1.9.0

4. Navigate to the **TIB_rtview-as** directory and type:

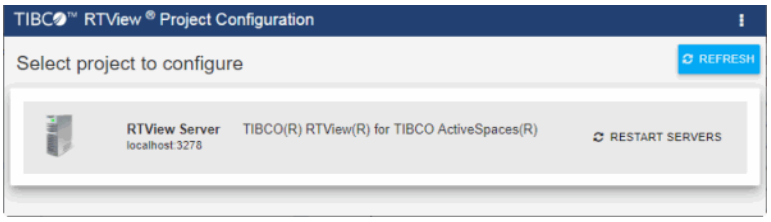
   start_server.sh

5. Open a browser and type the following URL to open the RTView Configuration Application:

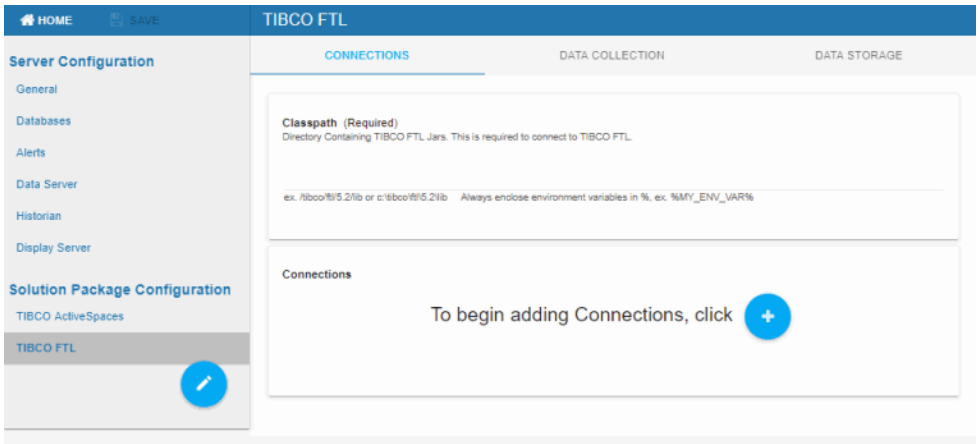
   http://localhost:3270/rtview-tdgmon-rtvadmin

   Use rtvadmin/rtvadmin for the username/password.

   The RTView Configuration Application Select Project page opens.



6. Select the **RTView Server TIBCO ActiveSpaces Monitor** project to open the main configuration page.
7. Select **TIBCO FTL** in the navigation tree. If TIBCO FTL is NOT in the navigation tree, click (pencil icon) to open the **Solution Packages** dialog, add it to your project and then select it.



8. In the TIBCO FTL **CONNECTIONS** tab, enter the following:

   Classpath: Provide the full path to the directory containing the TIBCO FTL jar files in the Classpath field. Use forward slashes in path name. Enclose environment variables with **%%** (even on UNIX). This is required to connect to TIBCO FTL. For example:

   /tibco/ftl/6.1/lib

9. Click SAVE (in title bar) to save your changes.
10. Click to open the **Add Connection** dialog and enter the following:

    Specify the connection information for your TIBCO FTL realm server, where:

    Name: The name for the connection. This entry is required. Use a semicolon-separated list format for multiple connections.

    Primary URL: The primary URL for the connection (for example, http://myhost:8080).

    Backup URL: The failover URL for the primary connection (for example, http://myhost:8090).

    Primary Cores: The number of primary cores.

Backup Cores: The number of backup cores.

Username: The username is used when creating a connection to a secure realm server. For details about connecting to a secure server, see Security Configuration.

Password: This password is also used when creating a connection to a secure realm server. By default, the password entered is hidden. For details about connecting to a secure server, see Security Configuration.

The newly created connection displays in the Connections section.

Repeat this Step for each TIBCO FTL realm server to be monitored.

Click SAVE to close the dialog and SAVE (in title bar) to save your changes.

11. Click RESTART SERVERS (which is visible in the upper right-hand corner after clicking SAVE) to apply your changes.

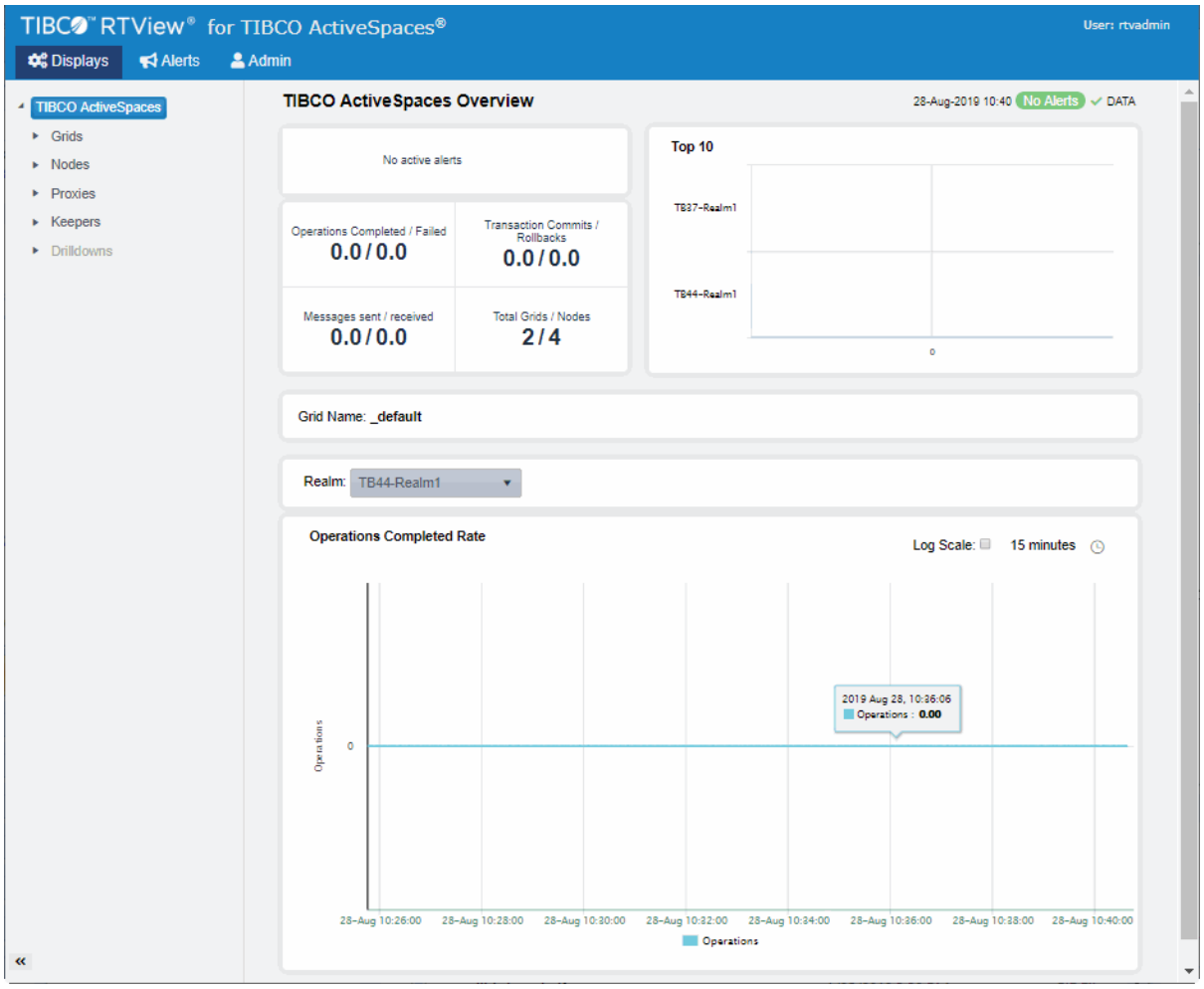
    The Select Project page reopens with the **RESTARTING DATASERVER...** message. Once the data server has restarted, the message disappears and you can click your project and resume making changes (if desired). You can also:

    - check the log files in the **TIB_rtview-as/projects/rtview-server/logs** directory for errors.
    - verify that your caches are collecting data by browsing to the RTView Cache Viewer application URL:

    http(s)://localhost:3270/common

    The RTView Cache Viewer allows you to view the details for the caches that are collecting data.

12. Open the ActiveSpaces Monitor by browsing to **http://localhost:3270/rtview-tdgmon**. Login as rtvadmin/rtvadmin.



Congrats! You have completed the Quick Start. See Configuration to optionally modify your setup or take advantage of additional features.

Windows Quick Start Steps

Do the following to download and install ActiveSpaces Monitor and define the TIBCO ActiveSpaces Realms to be monitored. Note that TIBCO ActiveSpaces is monitored via TIBCO FTL which means that you configure TIBCO FTL.

1. Download **TIB_rtview-as_<version>.zip** to your local Windows server.
2. Extract the files in **TIB_rtview-as_<version>.zip** using right mouse-click >"**Extract All...**"

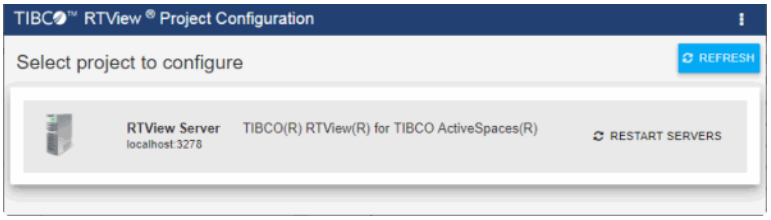
   You should see a top level directory, **TIB_rtview-as** containing two subdirectories: **projects** and **rtvapm**.
3. Set the **JAVA_HOME** environment variable to point to your Java installation. For example:

   set JAVA_HOME=C:\Java\jdk1.9.0
4. Execute the **start_server** script, located in the **TIB_rtview-as** directory.
5. Open a browser and type the following URL to open the RTView Configuration Application:

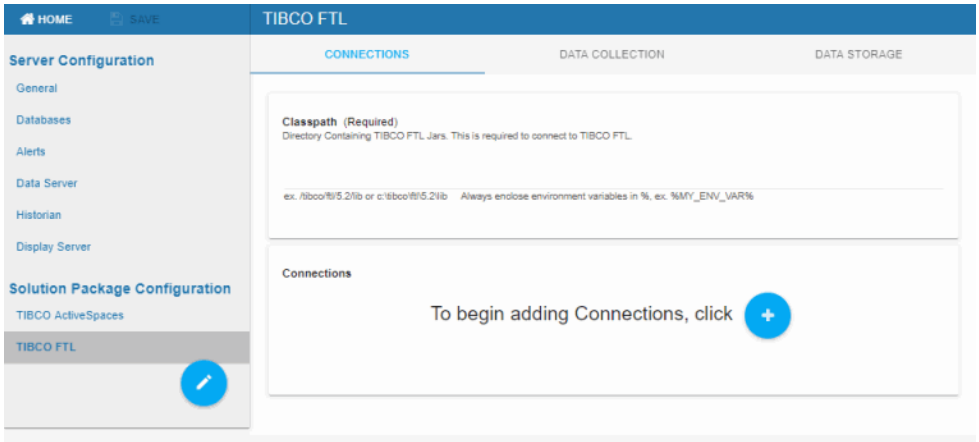
   http://localhost:3270/rtview-tdgmon-rtvadmin

   Use rtvadmin/rtvadmin for the username/password.

The RTView Configuration Application Select Project page opens.



6. Select the **RTView Server TIBCO ActiveSpaces Monitor** project to open the main configuration page.
7. Select **TIBCO FTL** in the navigation tree. If TIBCO FTL is NOT in the navigation tree, click (pencil icon) to open the **Solution Packages** dialog, add it to your project and then select it.



8. In the TIBCO FTL **CONNECTIONS** tab, enter the following:

   Classpath: Provide the full path to the directory containing the TIBCO FTL jar files in the Classpath field. Use forward slashes in path name. Enclose environment variables with **%%** (even on UNIX). This is required to connect to TIBCO FTL. For example:

   c:\tibco\ftl\6.1\lib

9. Click SAVE (in title bar) to save your changes..
10. Click to open the **Add Connection** dialog and enter the following:

    Specify the connection information for your TIBCO FTL realm server, where:

    Name: The name for the connection. This entry is required. Use a semicolon-separated list format for multiple connections.

    Primary URL: The primary URL for the connection (for example, http://myhost:8080).

    Backup URL: The failover URL for the primary connection (for example, http://myhost:8090).

    Primary Cores: The number of primary cores.

    Backup Cores: The number of backup cores.

    Username: The username is used when creating a connection to a secure realm server. For details about connecting to a secure server, see Security Configuration.

Password: This password is also used when creating a connection to a secure realm server. By default, the password entered is hidden. For details about connecting to a secure server, see Security Configuration.

The newly created connection displays in the Connections section.

Repeat this Step for each TIBCO FTL realm server to be monitored.

Click SAVE to close the dialog and SAVE (in title bar) to save your changes.

11. Click RESTART SERVERS (which is visible in the upper right-hand corner after clicking SAVE to apply your changes.

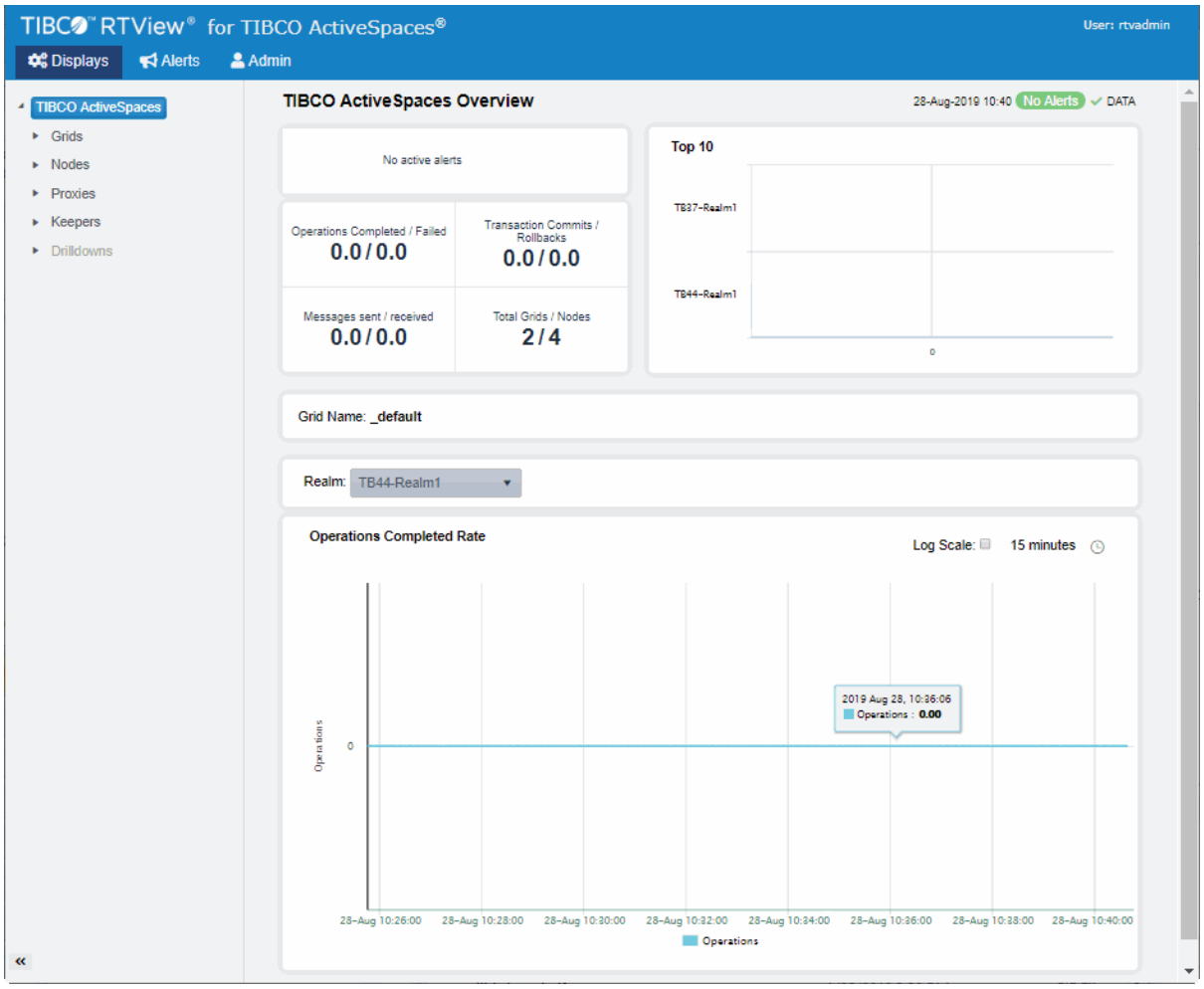
    The Select Project page reopens with the **RESTARTING DATASERVER...** message. Once the data server has restarted, the message disappears and you can click your project and resume making changes (if desired). You can also:

    - check the log files in the **TIB_rtview-as\projects\rtview-server\logs** directory for errors.
    - verify that your caches are collecting data by browsing to the RTView Cache Viewer application URL:

    http(s)://localhost:3270/common

    The RTView Cache Viewer allows you to view the details for the caches that are collecting data.

12. Open the ActiveSpaces Monitor by browsing to **http://localhost:3270/rtview-tdgmon**. Login as rtvadmin/rtvadmin.



Congrats! You have completed the Quick Start. See Configuration to optionally modify your setup or take advantage of additional features.

CHAPTER 2 Introduction

This section contains the following:

Overview

The Monitor takes the time and guesswork out of monitoring and troubleshooting ActiveSpaces deployments, providing a centralized view of both real-time and historical performance metrics across multiple ActiveSpaces data grids.

The Monitor enables TIBCO users to continually assess and analyze the health and performance of their ActiveSpaces infrastructure, gain early warning of issues with historical context, and effectively plan for capacity of their ActiveSpaces data grids. It does so by aggregating and analyzing key performance metrics across all realms, nodes and proxies, and presents the results, in real time, through meaningful dashboards as data is collected.

Users also benefit from pre-defined rules and alerts that pin-point critical areas to monitor in most ActiveSpaces environments and allow for customization of thresholds to let users fine-tune when alert events should be activated.

The Monitor also contains alert management features so that the life cycle of an alert event can be managed to proper resolution. All of these features allow you to know exactly what is going on at any given point, analyze the historical trends of the key metrics, and respond to issues before they can degrade service levels in high-volume, high-transaction environments.

The Monitor can be deployed as a stand-alone desktop client or as a Web application run in a browser.

System Requirements

Please refer to the **README_sysreq.txt** file from your product installation. A copy of this file is also available on the product download page.

Upgrade Notes

This section describes the steps necessary to upgrade existing Monitor applications.

Follow the steps for each version between the version you are upgrading from and the version to which you are upgrading:

- 7.1.2 -- See these steps to upgrade to version 7.1.2

7.1.2

Log4j2

The syntax used in a Log4j properties file was changed completely by Apache in version 2. The **sl.log4j.properties** file distributed with RTView has been changed to use the version 2 syntax. If, in previous RTView versions, you customized that file or specified your own custom log4j properties file (e.g. using the "log4jprops" option), you'll need to remake those customization using the version 2 syntax.

Customization changes should be made to the copy of **sl.log4j.properties** in the **projects** directory, instead of the copy under **rtvapm\common\conf** in order to make it easier toupgrade to future releases.

Note that the default logging behavior has been changed: In this release by default messages are appended to the existing **logs/X.log** file (where X is "dataserver", or "historian", etc depending on the name of the server) until it reaches a size of 50MB. Then it is renamed to X.log.N (where N = 1 - 9) and a new empty X.log file is created. So at any time the logs directory may contain X.log (newest, up to 50MB in size), and X.log.N where N = 1 - 9, each approx 50 MB, where 1 is the oldest and 9 is the newest. Once N = 9 is reached, on the next rollover X.log.1 is deleted and each remaining X.log.N is renamed to X.log.N-1.

SNMP Notifications

If you are upgrading from a previous release that sent SNMP notifications, you need to update the MIB in your SNMP receiver. The MIB definition in **rtvapm\common\lib\SL-RTVIEW-EM-MIB.txt** has changed to include a new field for this.

Admin Displays (HTML UI)

For improved security the following displays have been moved from **assets/packages/common** to **assets/packages/admin**:

- Alert Administration (rtv_alerts_admin_table.html)
- Alert Overrides Admin (rtv_alerts_admin_overrides.html)
- Component Alert Configuration (rtv_alerts_admin_detail.html)

Any existing browser bookmarks to those displays should be updated or recreated.

Installation

ActiveSpaces Monitor can be used as a standalone monitoring system for technical support teams. To install ActiveSpaces Monitor, download the **TIB_rtview-as_<VERSION>.zip** archive, and unzip the **TIB_rtview-as_<VERSION>.zip** file into a directory of your choosing. See Quick Start for more information.

File Extraction Considerations

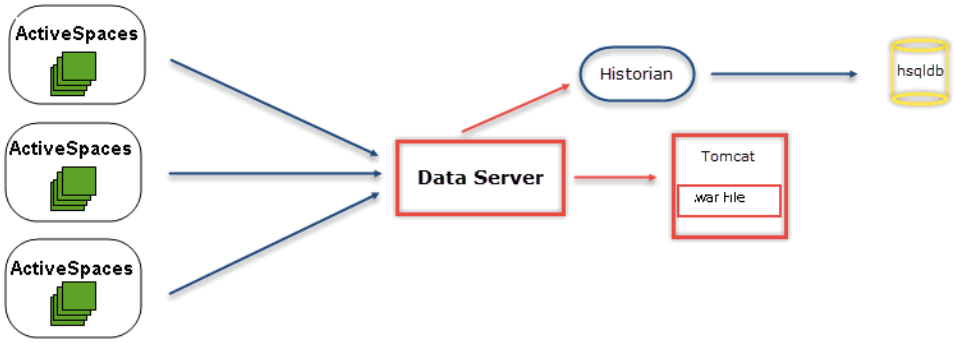
On Windows systems, using the extraction wizard of some compression utilities might result in an extra top-level directory level based on the name of the **.zip** file. The additional directory is not needed because the **.zip** files already contain top-level directory. This extra directory must be removed before clicking the **Next** button that performs the final decompression.

To convert text files on UNIX/Linux systems to the native format, use the **-a** option with unzip to properly extract text files.

If you are using Tomcat as your application server, copy the **TIB_rtview-as/projects/rtview-server/rtview-tdgmon*.war** file to the Tomcat **webapps** directory. If you are using Eclipse Jetty as your application server (which is delivered with ActiveSpaces Monitor), there are no further steps.

Architecture

The typical TIBCO ActiveSpaces deployment involves a Data Server collecting data from ActiveSpaces Servers, storing the data in internal memory caches, and then providing the data to the Historian and to the HTML UI for use in the Monitor. The basic TIBCO ActiveSpaces deployment diagram looks like the image below.



Listed below are some basic definitions for the various components in ActiveSpaces Monitor:

- **Data Server**: This Java process is responsible for accessing metrics from ActiveSpaces Servers, storing data into internal memory caches, providing data to the HTML UI and the Historian, and running the alert rules.
- **Data Historian**: This Java process stores and compacts data from the Data Server into a relational database for archival purposes. The default database used is HSQLDB.

CHAPTER 3 Configuration

These instructions assume that you completed Quick Start instructions, in which you defined the TIBCO ActiveSpaces Realms to monitor, and got the Monitor up and running using default settings.

This section describes how to (optionally) modify those default settings (such as data compaction rules for storing data), as well as how to configure high availability, alert notification, a production database and more.

You configure ActiveSpaces Monitor using the RTView Configuration Application to define properties. Property files are located in your project directory. Example default settings are provided in the **TIB_rtview-as/projects/rtview-server** directory.

This section describes how to configure the Monitor as a standalone application.

This section includes:

- Modify Data Update Rate: Modify the default polling rate.
- Modify Data Storage Settings: Modify the number of history rows to store in memory, compaction rules, the duration before metrics are expired and deleted, and the types of metrics that you want the Historian to store.
- Configure the Database: Configure a production database. The Monitor is delivered with a default memory resident HSQLDB database, which is suitable for evaluation purposes. However, in production deployments, we recommend that you deploy one of our supported databases. For details, see the RTView Core® User's Guide.
- Configure Alert Notification: Configure alerts to execute an automated action (for example, to send an email alert).
- Configure High Availability : Configure redundant system components with failover capability.

Modify Data Update Rate

You can modify the default data update rates for caches for:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

If you want to modify the default values for the update rates for the TIBCO FTL caches, you can update the default polling rates in **Solution Package Configuration > TIBCO FTL > DATA COLLECTION tab > Poll Rates**.

Modify the value for the **Poll Rate** field to modify the default polling rate for the TftlClient, TftlMetrics, TftlServer, TftlSatellite, TftlGroupServer, and TftlGroupServerGroup caches.

TIBCO ActiveSpaces

If you want to modify the default values for the update rates for the TIBCO ActiveSpaces caches, you can update the default polling rates in **Solution Package Configuration > TIBCO ActiveSpaces > DATA COLLECTION tab > Poll Rates**.

Modify Data Storage Settings

You can modify the default settings for the number of history rows to store in memory, compaction rules, the duration before metrics are expired and deleted and the types of metrics that you want the Historian to store.

This sections contains:

- Modify In Memory History Storage Settings
- Modify Compaction Rules
- Modify Expiration and Deletion Duration for Metrics
- Enable/Disable Storage of Historical Data
- Add a Prefix to All History Table Names for Metrics

Modify In Memory History Storage Settings

You can modify the maximum number of history rows to store in memory for:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

You can modify the maximum number of history rows to store in memory in the DATA STORAGE tab. The **History Rows** property defines the maximum number of rows to store for the TftlClient, TftlServer, TftlEvent, TftlAdvisory, and TftlMetrics caches.

To modify these settings:

- Navigate to the **Solution Package > TIBCO FTL > DATA STORAGE** tab.
- In the **Size** region, click the **History Rows** field and specify the desired number of rows.

TIBCO ActiveSpaces

You can modify the maximum number of history rows to store in memory in the DATA STORAGE tab. The **History Rows** property defines the maximum number of rows to store for the TdgRealm, TdgNode, TdgProxy and TdgKeeper caches.

To modify these settings:

- Navigate to the **Solution Package > TIBCO ActiveSpaces > DATA STORAGE** tab.
- In the **Size** region, click the **History Rows** field and specify the desired number of rows.

Modify Compaction Rules

You can reduce the amount of data to store for:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

Data compaction, essentially, is taking large quantities of data and condensing it using a defined rule so that you store a reasonably sized sample of data instead of all of your data, thus preventing you from potentially overloading your database. The available fields are:

- **Condense Interval** -- The time interval at which the cache history is condensed. The default is 60 seconds. The following caches are impacted by this setting: TftlClient, TftlServer, and TftlMetrics.
- **Condense Raw Time** -- The time span of raw data kept in the cache history table. The default is 1200 seconds. The following caches are impacted by this setting: TftlClient, TftlServer, and TftlMetrics.
- **Compaction Rules** -- This field defines the rules used to condense your historical data in the database. By default, the columns kept in history will be aggregated by averaging rows with the following rule 1h -;1d 5m;2w 15m, which means the data from 1 hour will not be aggregated (1h - rule), the data over a period of 1 day will be aggregated every 5 minutes (1d 5m rule), and the data over a period of 2 weeks old will be aggregated every 15 minutes (2w 15m rule). The following caches are impacted by this setting: TftlClient, TftlServer, TftlMetrics, and TftlGroupServer.
- **History Time Span** – The duration of time to retain a row of cached data based on its date received timestamp. The cache trims its History table by removing rows with timestamps that are older than the limit specified here. Specify the duration in seconds or specify a number followed by a single character indicating the desired time interval (e.g. 15m for 15 minutes). The format is a number followed by one of the following valid characters:

y - years (365 days)

M - months (31 days)

w - weeks (7 days)

d - days

h - hours

m - minutes

s - seconds

Example: 1M

Note that this setting only determines the duration of rows kept in the History table by the cache data source. It does not affect database storage, if any, associated with the cache.

The following caches are impacted by this field: TftlClient, TftlServer, TftlEvent, TftlAdvisory and TftlMetrics.

To modify these settings:

- Navigate to the **Solution Package Configuration > TIBCO FTL > DATA STORAGE** tab.
- In the **Compaction** region, click the **Condense Interval**, **Condense Raw Time**, **Compaction Rules**, and **History Time Span** fields and specify the desired settings.

TIBCO ActiveSpaces

Data compaction, essentially, is taking large quantities of data and condensing it using a defined rule so that you store a reasonably sized sample of data instead of all of your data, thus preventing you from potentially overloading your database. The available fields are:

- **Condense Interval** -- The time interval at which the cache history is condensed. The default is 60 seconds. The following caches are impacted by this setting: TdgRealm, TdgNode, TdgProxy and TdgKeeper.

- **Condense Raw Time** -- The time span of raw data kept in the cache history table. The default is 1200 seconds. The following caches are impacted by this setting: TdgRealm, TdgNode, TdgProxy and TdgKeeper.
- **Compaction Rules** -- This field defines the rules used to condense your historical data in the database. By default, the columns kept in history will be aggregated by averaging rows with the following rule 1h -;1d 5m;2w 15m, which means the data from 1 hour will not be aggregated (1h - rule), the data over a period of 1 day will be aggregated every 5 minutes (1d 5m rule), and the data over a period of 2 weeks old will be aggregated every 15 minutes (2w 15m rule). The following caches are impacted by this setting: TdgRealm, TdgNode, TdgProxy and TdgKeeper.
- **History Time Span** – The duration of time to retain a row of cached data based on its date received timestamp. The cache trims its History table by removing rows with timestamps that are older than the limit specified here. Specify the duration in seconds or specify a number followed by a single character indicating the desired time interval (e.g. 15m for 15 minutes). The format is a number followed by one of the following valid characters:

y - years (365 days)

M - months (31 days)

w - weeks (7 days)

d - days

h - hours

m - minutes

s - seconds

Example: 1M

Note that this setting only determines the duration of rows kept in the History table by the cache data source. It does not affect database storage, if any, associated with the cache.

The following caches are impacted by this field: TdgRealm, TdgNode, TdgProxy and TdgKeeper.

To modify these settings:

- Navigate to the **Solution Package Configuration > TIBCO ActiveSpaces > DATA STORAGE** tab.
- In the **Compaction** region, click the **Condense Interval**, **Condense Raw Time**, **Compaction Rules**, and **History Time Span** fields and specify the desired settings.

Modify Expiration and Deletion Duration for Metrics

You can reduce the amount of data to store for:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

The data for each metric is stored in a specific cache and, when the data is not updated in a certain period of time, that data will either be marked as expired or, if it has been an extended period of time, it will be deleted from the cache altogether. The **Expire Time** field, which sets the expire time for the TftlClient, TftlMetrics, TftlSatellite, TftlGroupServer, TftlGroupServerGroup, TftlClientAvailability and TftlServerAvailability caches, defaults to 120

seconds. The **Server Expire Time** field, which sets the expire time for TftlServer cache, defaults to 10 seconds. The **Delete Time** field, which sets the expire time for the TftlClient, TftlAdvisory and TftlMetrics caches, defaults to 3600 seconds.

To modify these settings:

- Navigate to the **Solution Package Configuration > TIBCO FTL > DATA STORAGE** tab.
- In the **Duration** region, click the **Expire Time**, **Server Expire Time**, and **Delete Time** fields and specify the desired settings.

TIBCO ActiveSpaces

The data for each metric is stored in a specific cache and, when the data is not updated in a certain period of time, that data will either be marked as expired or, if it has been an extended period of time, it will be deleted from the cache altogether. The **Expire Time** field, which sets the expire time for the TdgRealm, TdgNode, TdgProxy and TdgKeeper caches, defaults to 120 seconds. The **Server Expire Time** field, which sets the expire time for TftlServer cache, defaults to 10 seconds. The **Delete Time** field, which sets the expire time for the TftlClient, TftlAdvisory and TftlMetrics caches, defaults to 3600 seconds.

To modify these settings:

- Navigate to the **Solution Package Configuration > TIBCO ActiveSpaces > DATA STORAGE** tab.
- In the **Duration** region, click the **Expire Time**, **Server Expire Time**, and **Delete Time** fields and specify the desired settings.

The following caches are impacted by this field: TdgRealm, TdgNode, TdgProxy and TdgKeeper.

Enable/Disable Storage of Historical Data

You can enable and disable the storage of historical data to store for:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

The **History Storage** region allows you to select which metrics you want the Historian to store in the history database. To enable/disable the collection of historical data, perform the following steps:

- Navigate to the **Solution Package Configuration > TIBCO FTL > DATA STORAGE** tab.
- In the **History Storage** region, select the toggle for the FTL metrics if you want to collect/deselect for the FTL metrics if you do not want to collect. Blue is enabled, gray is disabled.

TIBCO ActiveSpaces

The **History Storage** region allows you to select which metrics you want the Historian to store in the history database. To enable/disable the collection of historical data, perform the following steps:

- Navigate to the **Solution Package Configuration > TIBCO ActiveSpaces > DATA STORAGE** tab.

- In the **History Storage** region, select the toggle for the ActiveSpaces metrics if you want to collect/deselect for the ActiveSpaces metrics if you do not want to collect. Blue is enabled, gray is disabled.

The following caches are impacted by this setting: TdgRealm, TdgNode, TdgProxy and TdgKeeper.

Add a Prefix to All History Table Names for Metrics

You can add a prefix to table names for:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

The **History Table Name Prefix** field allows you to define a prefix that will be added to the database table names so that the Monitor can differentiate history data between data servers when you have multiple data servers with corresponding Historians using the same solution package(s) and database. In this case, each Historian needs to save to a different table, otherwise the corresponding data server will load metrics from both Historians on startup. Once you have defined the **History Table Name Prefix**, you will need to create the corresponding tables in your database as follows:

Locate the .sql template for your database under **RTVAPM_HOME/tdgmon/dbconfig** and make a copy of it.

Add the value you entered for the History Table Name Prefix to the beginning of all table names in the copied .sql template.

Use the copied .sql template to create the tables in your database.

Note: If you are using Oracle for your Historian Database, you must limit the History Table Name Prefix to 2 characters because Oracle does not allow table names greater than 30 characters (and the longest table name for the solution package is 28 characters).

To add a prefix:

- Navigate to the **Solution Package Configuration > TIBCO FTL > DATA STORAGE** tab.
- Click on the **History Table Name Prefix** field and enter the desired prefix name.

TIBCO ActiveSpaces

The **History Table Name Prefix** field allows you to define a prefix that will be added to the database table names so that the Monitor can differentiate history data between data servers when you have multiple data servers with corresponding Historians using the same solution package(s) and database. In this case, each Historian needs to save to a different table, otherwise the corresponding data server will load metrics from both Historians on startup. Once you have defined the **History Table Name Prefix**, you will need to create the corresponding tables in your database as follows:

Locate the .sql template for your database under **RTVAPM_HOME/tdgmon/dbconfig** and make a copy of it.

Add the value you entered for the History Table Name Prefix to the beginning of all table names in the copied .sql template.

Use the copied .sql template to create the tables in your database.

Note: If you are using Oracle for your Historian Database, you must limit the History Table Name Prefix to 2 characters because Oracle does not allow table names greater than 30 characters (and the longest table name for the solution package is 28 characters).

To add a prefix:

- Navigate to the **Solution Package Configuration > TIBCO ActiveSpaces > DATA STORAGE** tab.
- Click on the **History Table Name Prefix** field and enter the desired prefix name.

Configure the Database

The Monitor is delivered with a default memory resident HSQLDB database, which is suitable for evaluation purposes. However, in production deployments, we recommend that you deploy one of our supported databases. For details, see the TIBCO© RTView© Standard Monitor User's Guide.

This section describes how to setup an alternate (and supported) database.

Database Requirements

The Monitor requires two database connections that provide access to the following information:

- **Alert Settings**

  The ALERTDEFS database contains alert administration and alert auditing information. The values in the database are used by the alert engine at runtime. If this database is not available, the Self-Service Alerts Framework under which alerts are executed will not work correctly.

- **Historical Data**

  The RTVHISTORY database contains the historical monitoring data to track system behavior for future analysis, and to show historical data in displays.

To Configure the Monitor Database:

You configure the database by defining database configurations in the RTView Configuration Application. You will also copy portions of the **database.properties** template file (located in the **common/dbconfig** directory) into the RTView Configuration Application.

1. Install a database engine of your choice. Supported database engines are Oracle, Microsoft SQL Server, MySQL, and DB2.

   NOTE: The default page size of DB2 is 4k. It is required that you create a DB2 database with a page size of 8k. Otherwise, table indexes will not work.

2. Open the **database.properties** template file, which is located in the **common/dbconfig** directory, and find the line that corresponds to your supported database from the "Define the ALERTDEFS DB" section.

3. Navigate to the RTView Configuration Application > **(Project Name)** > **Server Configuration** > **Databases** > **Connections** tab, click the Edit icon in the **Alert Threshold Database Connection** region.

   The **Edit Connection** dialog displays.

4. Enter the information from the **database.properties** template file "Define the ALERTDEFS DB" section into the **Edit Connection** dialog as follows:

   URL - Enter the full database URL to use when connecting to this database using the specified JDBC driver.

   Driver - Enter the fully qualified name of the JDBC driver class to use when connecting to this database.

   Classpath - Enter the classpath for the JDBC driver file.

   Username - Enter the username to enter into this database when making a connection.

Password - Enter the password to enter into this database when making a connection. If there is no password, use "**-**".

Run Queries Concurrently - Select this check box to run database queries concurrently.

5. Click SAVE to close the dialog and SAVE (in title bar) to save your changes.

6. Return to the **database.properties** template file, which is located in the **common/dbconfig** directory, and find the line that corresponds to your supported database from the "Define the RTVHISTORY DB" section.

7. Navigate to the RTView Configuration Application > **(Project Name)** > **Server Configuration** > **Databases**, and click the Edit icon in the **Historian Database Connection** region.

   The **Edit Connection** dialog displays.

8. Enter the information from he **database.properties** template file "Define the RTVHISTORY DB" section into the **Edit Connection** dialog as follows:

   URL - Enter the full database URL to use when connecting to this database using the specified JDBC driver.

   Driver - Enter the fully qualified name of the JDBC driver class to use when connecting to this database.

   Classpath - Enter the classpath for the JDBC driver file.

   Username - Enter the username to enter into this database when making a connection.

   Password - Enter the password to enter into this database when making a connection. If there is no password, use "**-**".

   Run Queries Concurrently - Select this check box to run database queries concurrently.

9. Click SAVE to close the dialog and SAVE (in title bar) to save your changes.

10. Manually create database tables. If your configured database user has table creation permissions, then you only need to create the Alerts tables. If your configured database user does not have table creation permission, then you must create both the Alert tables and the History tables.

    To create tables for your database, use the **.sql** template files provided for each supported database platform, which is located in the **dbconfig** directory of the **common**, **tdgmon** and **tftlmon** directories:

    - **Alerts**
      rtvapm/common/dbconfig/create_common_alertdefs_tables_<db>.sql
    - **History**
      rtvapm/tdgmon/dbconfig/create_tdgmon_history_tables_<db>.sql
      rtvapm/tftlmon/dbconfig/create_rtvmgr_history_tables_<db>.sql
      where **<db>** ={**db2**, **mysql**, **oracle**, **sqlserver**}

NOTE: The standard SQL syntax is provided for each database, but requirements can vary depending on database configuration. If you require assistance, consult with your database administrator.

The most effective method to load the **.sql** files to create the database tables depends on your database and how the database is configured. Some possible mechanisms are:

- **Interactive SQL Tool**
  Some database applications provide an interface where you can directly type SQL commands. Copy/paste the contents of the appropriate **.sql** file into this tool.
- **Import Interface**
  Some database applications allow you to specify a **.sql** file containing SQL commands. You can use the **.sql** file for this purpose.

Before loading the **.sql** file, you should create the database and declare the database name in the command line of your SQL client. For example, on MySQL 5.5 Command Line Client, to create the tables for the Alert Settings you should first create the database:

create database myDBName;

before loading the **.sql** file:

mysql -u myusername -mypassword myDBName < create_common_alertdefs_tables_mysql.sql;

If you need to manually create the Historical Data tables, repeat the same process. In some cases it might also be necessary to split each of the table creation statements in the **.sql** file into individual files.

Third Party Application

If your database does not have either of the two above capabilities, a third party tool can be used to enter SQL commands or import **.sql** files. Third party tools are available for connecting to a variety of databases (RazorSQL, SQLMaestro, Toad, for example).

You have finished configuring the databases. Proceed to Configure Alert Notification.

Configure Alert Notification

This section describes how to configure alerts to execute an automated action (such as sending an email alert). To setup an alert notification, select the event you want to notify on and then select the action to execute.

You set alerts to execute notifications based on the following events:

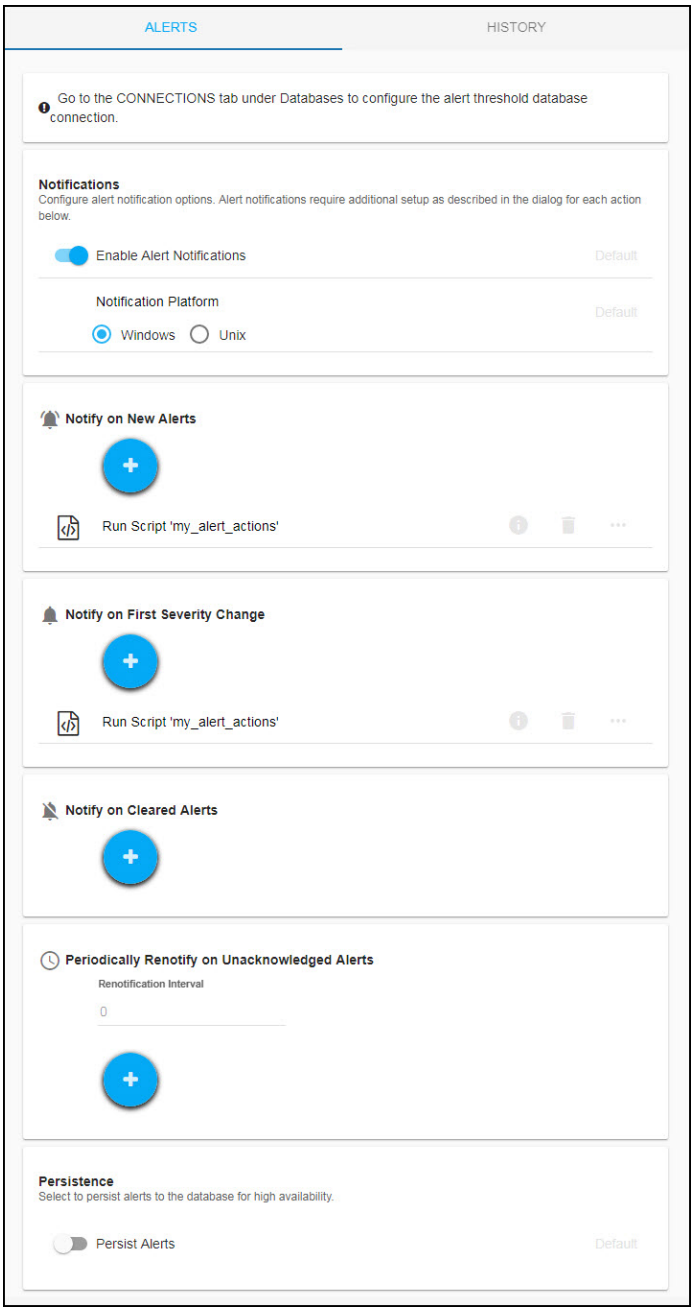
- when a new alert is created
- the first time the **Severity** level on an alert changes
- when an alert is cleared
- periodically renotify unacknowledged alerts

By default, a **.bat** script is executed for new alerts and on the first severity change for an alert. The script, by default, is not configured to execute an automated action. However, you can uncomment a line in the script that prints alert data to standard output. Or, you can modify the script to execute an automated action (such as sending an email alert). The following is a sample output from the alert command script:

----- Alert command script executed: DOMAINNAME=MYMON-1, ALERTNAME=someAlert, ALERTINDEX=alertIndex1~alertIndex2, ALERTID=1075, ALERTSEVERITY=2, ALERTTEXT=High Alert Limit exceeded current value: 100.0 limit: 80.0 #####

To configure Alert Notification:

1. Open the RTView Configuration Application, select **Alerts** (in the navigation tree) and then the **Alerts** tab.



2. Toggle on **Enable Alert Notfications** and select the **Notification Platform** type (**Windows or Unix**).

3. Select an alert event that you want to notify on by clicking  next to the option.

Alert Event Options

- **Notify on New Alerts**: A notification is executed every time a new alert is created.
- **Notify on First Severity Change**: A notification is executed the first time the **Severity** changes for each alert.
- **Notify on Cleared Alerts**: A notification is executed every time an alert is cleared.

- **Periodically Renotify on Unacknowledged Alerts**: Enter the **Renotification Interval** (number of seconds). A notification is executed for each unacknowledged alert per the interval you specify here. If the Renotification Interval is greater than **0** and no actions are defined, the **New Alerts** action will be used for renotifications.

4. Select the alert action(s) you want to execute.

Alert Action Options


Run a Script


Execute Java Code

Add Email Notification

Send SNMP Trap


Run Command String


Conditional Filter

You can choose multiple actions.

5. Click 

to close the dialog and 

(in title bar) to save your changes.

6. Some alert notification actions require additional setup as described in the dialog for each action. See the descriptions of each action below for details on the dialogs and additional setup for each action.

7. Click 

to apply changes.

Run a Script

This alert notification action executes the following script in the **TIB_rtview-ems/projects/rtview-server** directory:

- **my_alert_actions.bat/sh** – New and First Severity Change
- **my_alert_actions.cleared.bat/sh** – Cleared
- **my_alert_actions.renotify.bat/sh** – Periodically Renotify

This action can only be added once per notification type. In addition to selecting this action in the Configuration Application, you must also modify the appropriate script to execute the actions for your notification. This script has access to the following fields from the alert: **Alert Name**, **Alert Index**, **ID**, **Alert Text** and **Severity**.

Return to Alert Event Options.

Execute Java Code

This alert notification action allows you to implement your alert notification actions using Java code. It executes the **my_alert_notification.$domainName.$alertNotifyType.$alertNotifyCol** command in your Custom Command Handler and passes the row from the alert table that corresponds to the alert.

This action can only be added once per notification type. In addition to selecting this action the Configuration Application you must also modify the custom command handler to execute the actions for your notification. A sample custom command handler is included under **projects/custom**. It prints the alert notification to the console. You will modify this command handler to implement your own notification actions.

Make the following entries:

- **Custom Command Handler Class Name**: Enter the fully qualified name of the Custom Command Handler class. This defaults to the sample Custom Command Handler in the **emsmon/projects/custom** directory.
- **Custom Command Handler Jar**: Enter the path and name of the jar containing the Custom Command Handler class. The path may be absolute or relative to the location of data server. This defaults to the sample Custom Command Handler in the **emsmon/projects/custom** directory.

Note that if you can only have one custom command handler per Data Server, so changing these settings for one notification event will change them for the rest of the notification events.

Customizing the Custom Command Handler

The source for the Custom Command handler is provided in the **RtvApmCommandHandler.java** file, located in the **projects\custom\src\com\sl\rtvapm\custom** directory. By default, the handler prints the alert data to standard output. To change this behavior perform the following steps:

1. Open the **RtvApmCommandHandler.java** file.
2. Modify the **OutputAlertString** method as needed. You can replace this method with your own if you modify the **invokeCommand** method to call it, and your method accepts the same arguments as **OutputAlertString**.
3. Save the **RtvApmCommandHandler.java** file.
4. Compile **RtvApmCommandHandler.java** and rebuild **rtvapm_custom.jar** using the supplied script (**make_all.bat** or **make_all.sh**) in **projects\custom\src** directory.

Return to Alert Event Options.

Send Email

This alert notification action sends an email. This action can be added multiple times per notification type. No additional setup is required beyond filling in the **Add Email Notification** dialog in the Configuration Application.

Make the following entries:

- **SMTP Host**: The SMTP host address. This is required. Consult your administrator.
- **SMTP Port**: The SMTP port number. This is required. Consult your administrator.
- **User**: The user name for the account from which you are sending the email. This is optional.
- **Password**: The password for the account from which you are sending the email. This is optional.
- **From**: The email address to which to send the email. This is required.

- **To**: The email address to which to send the email. This is required and may contain multiple entries.
- **Subject**: The subject for the email. This is required. You can include the value from any column in the alert table in your subject. Click the **Show More** link at the bottom of the dialog to see the alert column values you can use in the **Subject**.
- **Body**: The body of the email. This is optional. Click the **Show More** link at the bottom of the dialog to see the alert column values you can use in the **Subject**.

Return to Configure Alert Notification.

Send SNMP Trap

This alert notification action sends an SNMP Trap as described in **rtvapm/common/lib/SL-RTVIEW-EM-MIB.txt**. This action can be added multiple times per notification type. No additional setup is required beyond filling in the **Add SNMP Trap Notification** dialog in the Configuration Application

Make the following entries:

- **Trap Type**: Select the SNMP version of the trap. This is required.
- **Destination Address**: The system name or IP address of the receiving system. This is required.
- **Destination Port**: The UDP port on the receiving system. This is required.
- **Community Name**: (This field is visible when **Trap Type v2/v3** is selected.) The SNMP v2 Community Name string. This is required.

Return to Alert Event Options.

Run Command String

This alert notification action executes a specified command. This action can be added multiple times per notification type. Make the following entry:

Command String: Enter the command string for any command supported by RTView Classic. To enter a command string, you must know the correct syntax for the command. Contact Technical Support for assistance on syntax. You can include the value from any column in the alert table using the syntax in the Show More link at the bottom of the dialog.

Return to Alert Event Options.

Conditional Filter

This alert notification action alert allows you to execute different actions for different alerts based on information in the alert. For example, you can configure ActiveSpaces alerts to send emails to your ActiveSpaces team and FTL alerts to send emails to your FTL team. This action can be added multiple times per notification type.

To create a condition, make the following entries:

- **Alert Field**: Select an alert field: **Alert Name**, **Alert Index**, **Category, Owner, Package,** or **Severity**.
- **Operator**: Select one - **EQUALS**, **DOES NOT EQUAL**, **STARTS WITH**, **ENDS WITH** or **CONTAINS**. This is required.
- **Value**: Enter the value to which to compare the Alert Field. Cannot contain wildcard characters. This is required.
- **Action(s)**: Select one or more actions to execute when this condition is met - Run a Script , Execute Java Code , Send SNMP Trap , Send Email , Run Command String.

Return to Alert Event Options.

Configure High Availability

High Availability (HA) mitigates single point of failure within ActiveSpaces Monitor by providing a means of defining redundant system components, together with failover capability, for users of those components.

When using HA, components are designated **PRIMARY** and **BACKUP**. If the **PRIMARY** component fails, failover occurs to the **BACKUP** component. If the **PRIMARY** component is subsequently restarted, the **BACKUP** component allows the newly restarted component to take the primary role and return to its backup role.

This section contains the following:

- Overview of High Availability Architecture
- Requirements for Configuring High Availability
- Steps to Configure High Availability
- Verifying the High Availability Configuration

Overview of High Availability Architecture

Data Server High Availability

The primary and backup data servers connect to each other via socket. If the primary data server stops, then the backup server takes over. If the primary then comes back online, then the primary takes over again and the backup returns to standby mode. The data client connections will move between the two servers accordingly.

NOTE: Be aware that data clients can connect to the standby server using a non-fault tolerant URL and still get data because of a proxy feature where the standby server forwards data requests to the primary server. This can be confusing when you use the HTML Cache Viewer (**http://localhost:3270/common**) on the standby server to view cache contents because it looks like the standby server caches are updating, but you are really viewing the data in the primary server and not in the standby server.

HTML User Interface High Availability

The HMTL UI client connects to the data server via an HA configured rtvquery servlet.

Historian High Availability

The primary and backup historian connect to each other via socket. If the primary historian stops, then the backup takes over. If the primary historian comes back online, then the primary takes over again and the backup returns to standby mode. Only the active historian writes to the database.

The historian is a data client of the data server and connects to it via a fault tolerant URL (socket only), which means that the data servers and historians can fail over separately or together.

Requirements for Configuring High Availability

The following are minimum requirements for High Availability:

- Two host machines, one for the primary host and one for the backup host.
- Both hosts must be configured such that the RTView processes on each host can connect to each other via socket.
- Both hosts must be able to access:

  - the same data connections
  - the same historian database
  - the alert threshold database
- The RTView processes on both hosts must be able to run against identical properties files. In the case where drivers or other third party jars are located in different directories on the two hosts, create a directory in the same location in each host, copy the jar files into and reference that directory in your properties.
- Tomcat or other Application Server
  - The HTML UI and rtv servlets must be deployed on an application server other than the internal Jetty server. Note that this requires extra configuration of the servlet **.war** files in the application server.

Steps to Configure High Availability

To Configure High Availability:

1. On both the primary and backup hosts, define the following environment variables:
   - **PRIMARYHOST** - the IP Address or hostname of the host running the primary servers (for example, **set PRIMARYHOST=MyHost**).
   - **BACKUPHOST** - the IP Address or hostname of the host running the backup servers (for example, **set BACKUPHOST=OtherHost**).
2. Install the Monitor on both the primary host and the backup host.
3. Configure your servlets to be HA and deploy them to your application server:
   - **cd projects\rtview-server**
   - In a text editor, open **update_wars**(**.bat** or **.sh**) and fill in the values for **HOST** and **HA_HOST** as described in the script.
   - Run the **update_wars**(**.sh** or **.bat**) script.
   - Copy the generated war files to the **webapps** directory of your application server.
4. To run High Availability, you must run the following from the command line:

   Windows
   - From the command line on the primary host, **cd** to **TIB_rtview-as** and type **start_server -haprimary**.
   - From the command line on the backup host, **cd** to **TIB_rtview-as** and type **start_server -habackup**.

   Unix
   - From the command line on the primary host, **cd** to **TIB_rtview-as** and type **start_server.sh -haprimary**.
   - From the command line on the backup host, **cd** to **TIB_rtview-as** and type **start_server.sh -habackup**.
5. Configure the Monitor on the primary host using the RTView Configuration Application (see Quick Start). Make sure to configure data collection, configure server options and databases, and enable alert persistence.

Note that the RTView Configuration Application must be able to connect both the primary and backup servers in order to enable editing. The same properties are saved to both servers. The **RESTART SERVERS** button (in the RTView Configuration Application) restarts both the primary and backup servers at the same time. If you want to stagger the restarts, use the scripts under **TIB_rtview-as** to stop and then start your servers after making changes in the RTView Configuration Application.

Note: Jetty does not have to be disabled, but data clients will not be able to make high availability connections to the data server using the Jetty URL. However, the Jetty URL can still be used to configure the application.

Verifying the High Availability Configuration

Verify failover and failback configurations by looking for the following in the log files.

Note: If the PRIMARYHOST and/or BACKUPHOST environment variable(s) is/are not set, you will get the following error in the log files and HA will be disabled:

```
ERROR: Disabling HA because the PRIMARYHOST and/or BACKUPHOST environment
variable is not set.
```

The following log files are available:

- Primary Data Server Log File
- Backup Data Server Log File
- Primary Historian Log File
- Backup Historian Log File

Primary Data Server Log File

```
startup

[rtview] Starting as primary HA data server accessible via
//primaryhostname:3278,//backuphostname:3278

[rtview] DataServerHA: connected to backuphostname:3278

[rtview] DataServerHA: run as primary server, backuphostname:3278 has lower
priority than this server

[rtview] leaving standby mode
```

Backup Data Server Log File

```
startup

[rtview] Starting as backup HA data server accessible via
//primaryhostname:3278,//backuphostname:3278

rtview] entering standby mode

after failover (primary data server exits)

[rtview] DataServerHA: error receiving message: java.net.SocketException:
Connection reset (primaryhostname:3278)

[rtview] DataServerHA: becoming primary server, lost connection to primary server
primaryhostname:3278

[rtview] leaving standby mode

after failback (primary data server comes back up)
```

```
[rtview] DataServerHA: resigning as primary server, got standby directive from
other server primaryhostname:3278

[rtview] connected to primaryhostname:3278

[rtview] entering standby mode
```

Primary Historian Log File

```
[rtview] Starting as primary HA historian paired with backup historian at
<backuphostname>:3222

[rtview] ServerGroup: status of member <backuphostname>:3222: primary, priority= 1,
started=Wed Nov 14 12:56:01 PST 2018

[rtview] ServerGroup: primary server = local

[rtview] ServerGroup: becoming primary server
```

Backup Historian Log File

```
[rtview] Starting as backaup HA histoiran paired with primary historian at
<primaryhostname>:3222

[rtview] ServerGroup: status of member <primaryhostname>:3222: primary, priority= ,
started=Wed Nov 14 12:56:01 PST 2018

[rtview] ServerGroup: primary server = <primaryhostname>:3222

after failover (primary historian exits):

[rtview] error receiving message: java.io.EOFException (primaryhostname:3222)

[rtview] ServerGroup: disconnected from primaryhostname:3222

[rtview] ServerGroup: primary server = local

after failback (primary historian starts back up):

[rtview] ServerGroup: status of member primaryhostname:3222: primary, priority= 2,
started= Tue Nov 20 09:12:43 PST 2018

[rtview] ServerGroup: connected to primaryhostname:3222

[rtview] ServerGroup: primary server = primaryhostname:3222
```

CHAPTER 4 Deployment

This section describes how to deploy the Monitor components. This section includes:

Overview

The Monitor is deployed as a web application that runs in a browser. Evaluation environments can use the provided HSQLDB database. Production environments require a supported JDBC- or ODBC-enabled relational database to store historical information. Supported databases are MySql, Oracle, SqlServer and DB2.

The RTView Historian and RTView Data Server are typically deployed on the same host. However, these processes can optionally be configured on separate hosts. Doing so can increase performance in deployments that need to support many end users or systems with large TIBCO servers.

To deploy the Monitor as a Web Application:

- Web Application Deployment: Clients need only a browser installed. The RTView Data Server, RTView Historian and Application Server are typically installed on the same host.

To configure the RTView process to run as a Windows Service:

- RTView Server Components as Windows Services: The RTView Data Server abd Historian can optionally be run as a Windows Service.

Web Application Deployment

This section describes how to deploy the Monitor as a web application. You start the Monitor using the **start_server** script (and stop the Monitor using the **stop_server** script). The following processes are started: the RTView Data Server, the Historian and the database.

This section contains:

- Windows
- UNIX/Linux

Windows

> **Note:** You can skip Step 1 and Step 2 if you are using Eclipse Jetty, which is delivered with the Monitor, as your application server.

1. Copy the **.war** files, located in the **TIB_rtview-as\projects\rtview-server** directory, and deploy them to your Application Server.

2. Start your Application Server if using Tomcat or an application server other than Eclipse Jetty.

3. You can skip this step if you are using Eclipse Jetty. The RTView Configuration Application uses digest authentication for security, and only allows access to users with the "rtvadmin" role. In order to allow access to the RTView Configuration Application in your application server, you need to add a user with the "rtvadmin" role. For example, if using Tomcat, follow the instructions below. For other application servers, refer to their documentation for adding users.
   - Edit **<Tomcat installation directory>\conf\tomcat-user.xml**
   - Add the following lines inside the tomcat-users tag:

```
<role rolename="rtvquery"/>
<user username="rtvquery" password="rtvadmin" roles="rtvquery"/>
```

4. Change directory (**cd**) to the **TIB_rtview-as** directory.

5. Start the Monitor applications by typing:

   start_server

NOTE: The **start_server** command starts all the Monitor applications at once. Use the **stop_server** script to stop Monitor applications.

6. Open a Web browser and access the following URL to open the Monitor:

   If using Eclipse Jetty as your application server: **http://localhost:3270/rtview-tdgmon**

   or

   If using your own application server: **http://host:port/rtview-tdgmon**

   Where **host** is the IP or host name where your Application Server is running, **port** is the port used by your Application Server and **rtview-tdgmon** is the Monitor you are deploying. The login display opens in the Web browser.

   Login. The default user name and password are:

   User Name: **rtvadmin**

Password: **rtvadmin**

The main Monitor display opens.

UNIX/Linux

Note: You can skip Step 1 and Step 2 if you are using Eclipse Jetty, which is delivered with the Monitor, as your application server.

1. Copy the **.war** files, located in the **TIB_rtview-as\projects\rtview-server** directory, and deploy them to your Application Server.

   Note: You can skip this step if you are using Eclipse Jetty, which is delivered with the Monitor, as your application server.

2. Start your Application Server if using Tomcat or an application server other than Eclipse Jetty, which is delivered with the Monitor.

3. You can skip this step if you are using Eclipse Jetty. The RTView Configuration Application uses digest authentication for security, and only allows access to users with the "rtvadmin" role. In order to allow access to the RTView Configuration Application in your application server, you need to add a user with the "rtvadmin" role. For example, if using Tomcat, follow the instructions below. For other application servers, refer to their documentation for adding users.

   - Edit **<Tomcat installation directory>\conf\tomcat-user.xml**
   - Add the following lines inside the tomcat-users tag:

```
<role rolename="rtvquery"/>
<user username="rtvquery" password="rtvadmin" roles="rtvquery"/>
```

4. Change directory (**cd**) to the **TIB_rtview-as** directory.

5. Start the Monitor applications by typing:

   start_server.sh

NOTE: The **start_server.sh** command starts all the Monitor applications at once. Use the **stop_server.sh** script to stop Monitor applications.

6. Open a Web browser and access the following URL to open the Monitor:

   If using Eclipse Jetty as your application server: **http://localhost:3270/rtview-tdgmon**

   or

   If using your own application server: **http://host:port/rtview-tdgmon**

   Where **host** is the IP or host name where your Application Server is running, **port** is the port used by your Application Server and **rtview-tdgmon** is the Monitor you are deploying. The login display opens in the Web browser.

   Login. The default user name and password are:

   User Name: **rtvadmin**

   Password: **rtvadmin**

   The main Monitor display opens.

   See "Quick Start" for a more detailed example.

RTView Server Components as Windows Services

This section describes how to configure an RTView process (Data Server, Historian) to run as a Windows service.

To Configure the Data Server or Historian to run as a Windows Service

1. Navigate to the RTView Configuration Application > **(Project Name)** > **Server Configuration** > **General** > **Custom Properties** tab.

2. Click the icon.
   The **Add Property** dialog displays.

3. Define the values for each of the following properties:

   Name: sl.rtview.cmd_line
   Value: -install_service
   Filter: installservice
   Comment: (description of the filter)

   Name: sl.rtview.cmd_line
   Value: -dir:%RTVAPM_STARTUP%
   Filter: installservice
   Comment: (description of the filter)

   Name: sl.rtview.cmd_line
   Value: -uninstall_service
   Filter: uninstallservice
   Comment: (description of the filter)

   Note: The environment variable %RTVAPM_STARTUP% is set by run script to the directory where the script was started.

4. For each Windows service you want to create, add the following property and replace ServiceName in the value and filter fields with a name you choose for the service:

   Name: sl.rtview.cmd_line
   Value: -service:ServiceName
   Filter: ServiceName

   For example, choose ActiveSpacesMonData as the name for starting a Data Server as a Windows service and ActiveSpacesMonHist to indicate a name for starting a Historian as a Windows service.

   Name: sl.rtview.cmd_line
   Value: -service:ActiveSpacesMonData

Filter: ActiveSpacesMonData

Name: sl.rtview.cmd_line

Value: -service:ActiveSpacesMonHist

Filter: ActiveSpacesMonHist

> **Note:** Each service must have a unique name and the beginning of the property entered must match the name of the service.

Once all your properties have been added, click SAVE to close the dialog and SAVE (in title bar) to save your changes.

1. Click RESTART SERVERS (which is visible in the upper right-hand corner after clicking SAVE) to apply your changes.

To install and run

5. Execute the following scripts to start the service:

NOTE: These scripts must be run in an initialized command window.

rundata -propfilter:installservice -propfilter:ActiveSpacesMonData

runhist -propfilter:installservice -propfilter:ActiveSpacesMonHist

To uninstall

6. Execute the following scripts to uninstall the services:

NOTE: These scripts must be run in an initialized command window.

runhist -propfilter:uninstallservice -propfilter:ActiveSpacesMonHist

rundata -propfilter:uninstallservice -propfilter:ActiveSpacesMonData

Troubleshooting

This section includes:

- Log Files
- JAVA_HOME
- Permissions
- Network/DNS
- Verify Data Received from Data Server
- Restarting the Data Server

Log Files

When a Monitor component encounters an error, an error message is output to the console and/or to the corresponding log file. If you encounter issues, look for errors in the following log files, located in the **TIB_rtview-as/projects/rtview-server/logs** directory:

- **dataserver.log**
- **historian.log**

Logging is enabled by default. If you encounter issues with log files, verify the **logs** directory exists in the **TIB_rtview-as/projects/rtview-server/logs** directory.

JAVA_HOME

If the terminal window closes after executing the **start_server** command, verify that **JAVA_HOME** is set correctly.

Linux users: JAVA_HOME is required for Tomcat.

Permissions

If there are permissions-related errors in the response from the **start_server** command, check ownership of the directory structure.

Network/DNS

If any log file shows reference to an invalid URL, check your system's hosts file and check with your Network Administrator that your access to the remote system is not being blocked.

Verify Data Received from Data Server

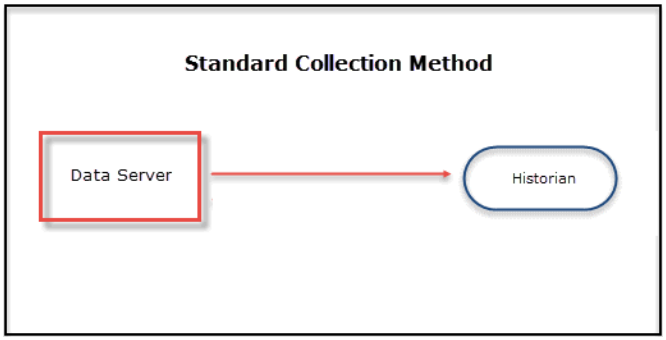
Open the **Cache Viewer Display** to verify data is arriving correctly from the Data Server. To access the **Cache Viewer Display**, choose **Administration** in the navigation tree, then choose **RTView Cache Tables** display or the **RTView Cache Overview** display. You should see all caches being populated with monitoring data (number of rows > 0). Otherwise, there are problems with the connection to the Data Server.

Restarting the Data Server

If the HTML UI or the Historian fails to connect to the Data Server or receives no data, verify the ports are assigned correctly in your properties files and then restart the Data Server.

Sender/Receiver: Distributing the Load of Data Collection

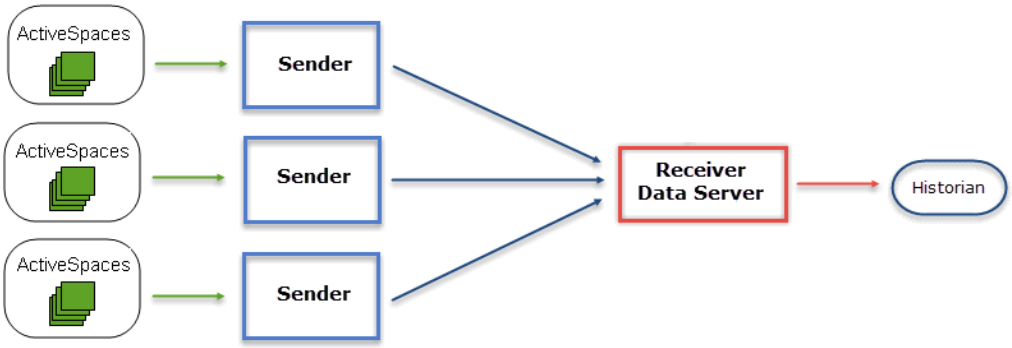
The standard method of collecting data involves one Data Server that sends the data to the HTML UI and the Historian. For example:



This method is optimized to deliver data efficiently when large tables and high data volumes are involved. There is, however, an alternative method of collecting data: the Sender/Receiver Data Collection Method. This collection method allows you to configure ActiveSpaces Monitor so that you have a Data Server (Receiver) that collects data from one or more remote Senders. This type of configuration could be useful in the following scenarios:

1. When dividing the collection load across different machines is more efficient

   In the Sender/Receiver Data Collection Method, the Senders are configured as lightweight Data Servers without history being configured and whose primary purpose is to collect and aggregate data from their respective local ActiveSpaces Servers that they then send to the full-featured Data Server (Receiver). The benefit of this type of configuration comes from balancing the load of the data collection. The Senders collect data exclusively from the ActiveSpaces Servers in their network and send the data to the Receiver, which collects the data and sends it to the ActiveSpaces Monitor HTML UI and the Historian. The following illustration provides one configuration example:

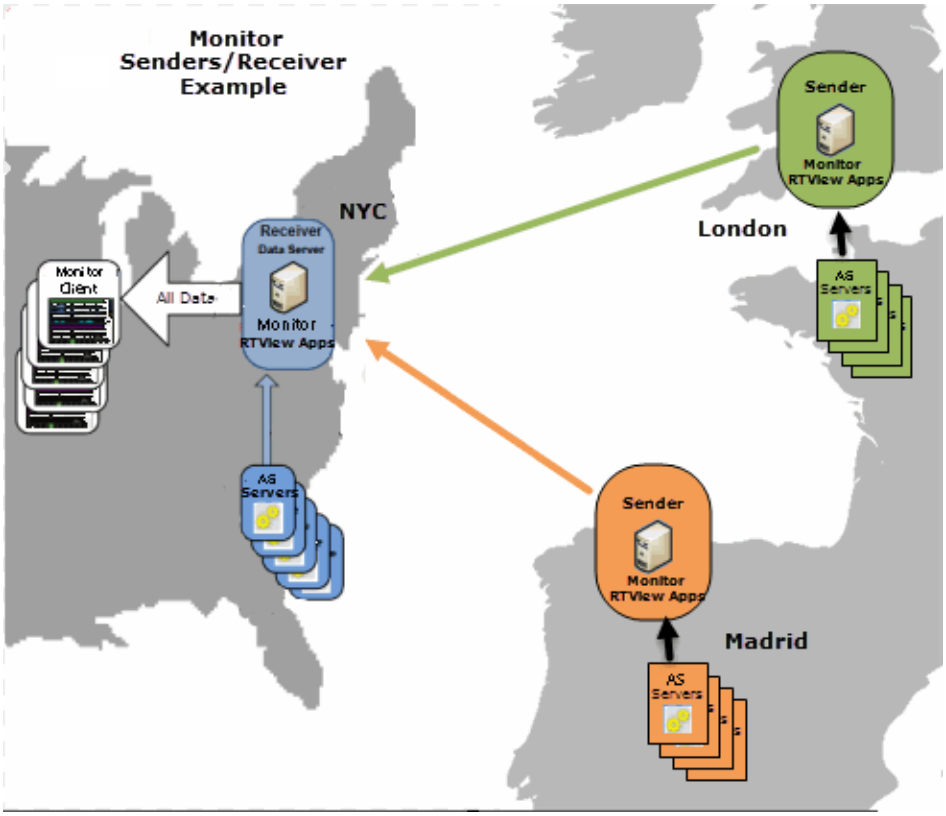


2. When firewall limitations prevent the Receiver Data Server from receiving data directly, Senders behind the firewall can be configured to send data to the Receiver

In the Standard Collection Method, the client must specify the network address of the Data Server to which it wants to connect, which might not be allowed due to security restrictions. In these situations, the Sender/Receiver Collection Method could be considered since the Receiver does not need to know the network addresses of the Senders because it simply opens the port and passively receives data from any defined Sender.

Example

The following example contains Senders in London and Madrid that collect data from their associated ActiveSpaces Servers and send the data to a Receiver Data Server in New York City. The Receiver takes the collected data from London and Madrid along with data collected from its own associated ActiveSpaces Servers and sends it to the ActiveSpaces Monitor displays. In the following figure, TIBCO ActiveSpaces is referred to as *AS* and the ActiveSpaces Monitor is referred to as *Monitor*.



Receiver Data Server -- NYC	Sender -- London	Sender -- Madrid
• Automatically detects and gathers data from its local ActiveSpaces Servers. • Receives data from London and Madrid Senders. • Aggregates data. • Provides data to the ActiveSpaces Monitor displays.	• Automatically detects and gathers data from its local ActiveSpaces servers. • Sends data to the NYC Data Server.	• Automatically detects and gathers data from its local ActiveSpaces Servers. • Sends data to the NYC Data Server.

Setting Up the Sender/Receiver Configuration

The following steps outline the workflow for setting up a Sender/Receiver configuration:

Receiver Configuration

This section assumes you have already installed the Monitor on the system where you will be running the receiver. See Installation for information on installing the Monitor and Quick Start to configure the Monitor.

Set up your receiver data server

1. Start the project using **start_server**.

2. By default, the receiver is setup to receive data on port 3272. If your senders cannot access the system on which the receiver is running, they can send data to the rtvagent servlet instead, which will forward the data to the receiver. To deploy the rtvagent servlet:

   If you are using Eclipse Jetty (the default application server):

   There are no required steps.

   If you are using Tomcat/a different application server:

   Copy the **rtview-tdgmon_rtvagent.war** files located in the **projects/rtview-server** directory to the Tomcat **webapps** directory.

3. Remove any connections that will be serviced by a sender in the RTView Configuration Application > (**PROJECT NAME) > Solution Package Configuration** > **TIBCO ActiveSpaces** > **CONNECTIONS**. If all connections will be serviced by senders, any connections created in the **CONNECTIONS** tab need to be removed.

4. Restart the project using **stop_server** and **start_server**. See Quick Start for more information.

Sender Configuration

This section assumes you have already installed the Monitor on the system where you will be running the sender, and also that you have created a project directory. See Installation for information on installing the Monitor and Quick Start for how to configure the Monitor. You can run as many senders on as many systems as needed.

1. In the **rtvservers.dat** file located in the **projects/rtview-server** directory, add **-propfilter:sender** to the end of the dataserver line and comment out the historian and database processes as follows (since they are not used by sender data servers):

   ```
   default . dataserver rundata -propfilter:sender
   #default . historian runhist -ds
   #default . database rundb
   ```

2. Start the sender project using **start_server**. See Quick Start for more information.

   Note: If you are running multiple senders on the same system or running the sender on the same system as the receiver, you need to change the port prefix for the sender so that you do not get a port conflict. To do so, use the following on the command line as follows: **-portprefix:XX** where XX is the port prefix. To save this to your properties file so you do not

need to specify it on the command line, add the **-saveportprefix** command line option. For example: **-portprefix:55 -saveportprefix**

3. Open the RTView Configuration Application > (**Project Name**) > **Server Configuration** > **Data Servers** > **COLLECTOR** tab.

4. In the **Targets** region, click the icon to add a target as follows:

   ID: A unique name for the target.

   URL: Specify the URL for the receiver. The url can be **host:port** (for example, somehost:3272) or an **http url** for the rtvagent servlet on the receiver. For example, if you are using Tomcat, you would use **http://somehost:8068/rtview-tdgmon-rtvagent**. If you are using Jetty, you would use **http://somehost:3270/rtvagent**.

   Targets: Select the **All solution packages** option.

   Enabled: Select this check box to enable the target.

5. Click **Save** to exit the **Add Target** dialog.

6. Fill in a unique value for this sender in the **Identifier** field on the **COLLECTOR** tab. This should be unique across all senders.

7. Click on the RTView Configuration Application > (**Project Name**) > **Solution Package Configuration** > **TIBCO ActiveSpaces** > **CONNECTIONS** tab and verify that this sender is configured to collect only from its local connections.

8. If you changed the port prefix, click on the RTView Configuration Application > (**Project Name**) > **Server Configuration** > **General** > **GENERAL** tab and confirm the port prefix is set to the correct value. If not, modify it accordingly.

9. Click **Save** in the RTView Configuration Application and restart your project using **stop_server** and **start_server**. See Quick Start for more information.

CHAPTER 5 Using the Monitor

This section describes Monitor features, graphs and behavior as well as the Monitor displays. This section includes:

- Overview: Describes the Monitor navigation, layout, graphic objects and functionality.
- Displays: Describes the displays available for TIBCO ActiveSpaces data.
- Alerts: Describes the displays available for alerts.
- Admin: Describes the displays available for administering the Monitor.

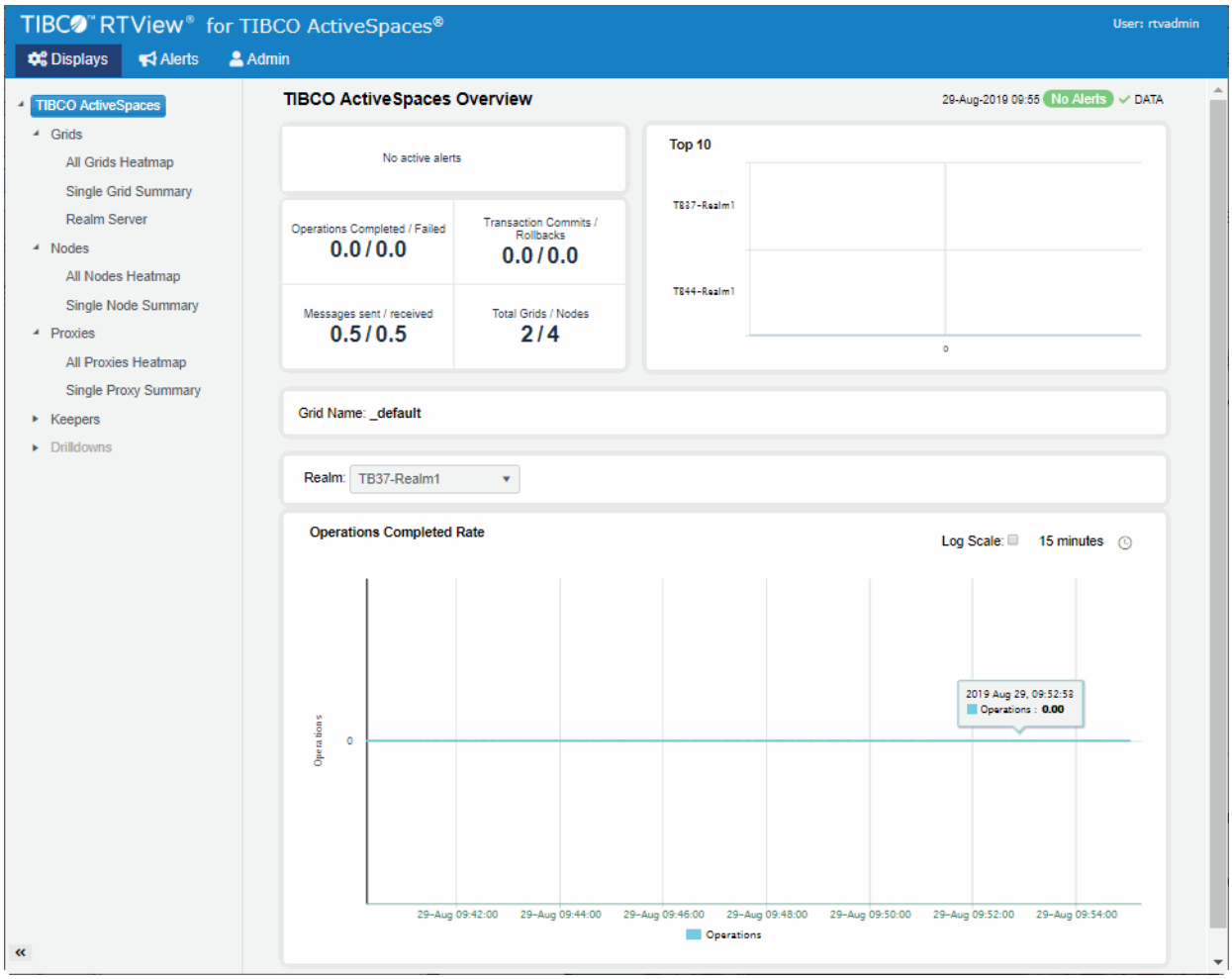
Overview

This section describes the Monitor console layout, navigation, GUI functionality and how to read and use graphic objects. This section includes:

- Login: Describes how to login to the Monitor and the main console from which you navigate and manage your monitoring system..
- User Permissions: Describes Monitor user roles/access.
- Navigation Tree: Describes the navigation tree.
- Heatmaps: Describes how to read heatmaps and heatmap functionality.
- Tables: Describes how to read tables and table functionality.
- Trend Graphs: Describes how to read trend graphs and trend graph functionality.
- Icons and Buttons: Describes the behavior of graphic icons shared by Monitor displays, such as the title bar.

Login

To access the Monitor, browse to **http://localhost:3270/rtview-tdgmon**. Login as rtvadmin/rtvadmin.



User Permissions

There are three types of users:

- **End-users** use rtvuser/rtvuser as their username/password which permits read-only access to all displays except for **Admin** tab displays.
- **End-user with alert management privileges** use rtvalertmgr/rtvalertmgr as their username/password which permits the same access as the end-user. Additionally, you can use the **Own**, **Ack**, **Unack** and **Comment** functions in the **Alerts Table**.
- **Administrators** use rtvadmin/rtvadmin as their username/password which permits read-only access to all displays as well as **Admin** tab displays. You can also enable and administer alerts, view cache contents and use the **Own**, **Ack**, **Unack** and **Comment** functions in the **Alerts Table**.

The following figure illustrates the main Monitor console which features the TIBCO ActiveSpaces Overview display. The tabs in the title bar, **Displays**, **Alerts** displays and **Admin**, take you to displays with ActiveSpaces data, displays that show alerts and displays for administering your monitoring system (respectively).

The navigation tree in the left panel takes you to various displays.

Note: Typically, it takes about 30 seconds after a server is started to appear in a Monitor display.

On larger screens the page contains a horizontal menu bar with three tabs:

- **Displays** contains the screens for PubSub+ performance data which you select from the navigation tree in the left panel.
- **Alerts** is used for viewing and managing alerts.
- **Admin** is used for administering alerts and viewing cache contents directly. This tab is only accessible to users with administrator privileges (user accounts with the rtvadmin role). You can hide the navigation tree by clicking **<<** (on the lower left).

Navigation through the displays is recorded in the browser history and you can use the browser's back and next buttons to traverse that history. You can hide the navigation tree in the **Displays** and **Admin** tabs by clicking **<<** (on the lower left).

On smaller screens, the horizontal menu bar is replaced by a vertical menu whose visibility is toggled by clicking the menu icon in the upper right corner of the page.

Once a user is logged in, that user remains logged in until the browser window is closed. Closing just the browser tab that contains the user interface does not log out the user, the browser itself must be closed.

See Displays for details about displays for TIBCO ActiveSpaces.

By default, data is collected every 15 seconds, and the display is refreshed 15 seconds after that.

Navigation Tree

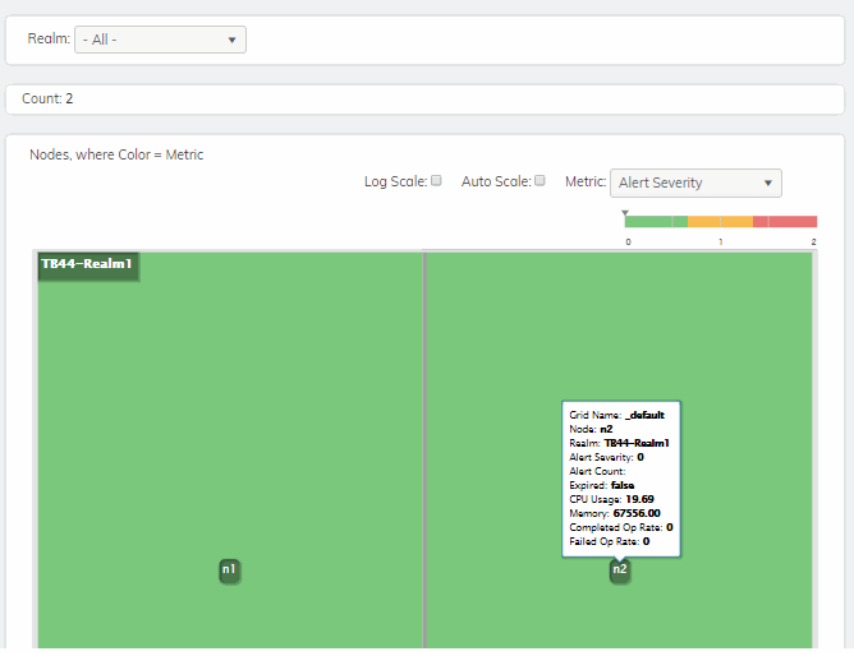
The Monitor navigation tabs are organized by *Views.* Each View features performance data for a type of system resource. Typically, the performance data is shown in a tabular, heatmap, and summary display for each View.

> **Note:** It takes about 60 seconds after the Monitor Data Server is started for data to initially appear in Monitor displays. By default, data is collected every 20 seconds and displays are refreshed every 2 seconds.

Heatmaps

Heatmaps organize your TIBCO ActiveSpaces resources (realms, nodes, proxies and keepers) into rectangles and use color to highlight the most critical values in each. Heatmaps enable you to view various metrics in the same heatmap using drop-down menus. Each metric has a color gradient bar that maps relative values to colors. In most heatmaps, the rectangle size represents the number of resources in the rectangle. Heatmaps include drop-down menus by which to filter data. The filtering options vary among heatmaps.

For example, each rectangle in the following heatmap represents a realm, where color is representative of the selected **Metric**.



The **Metric** drop-down menu in this heatmap contains options to show **Alert Severity**, **Alert Count**, as well as other metrics. Menu options vary according to the data populating the heatmap. **Alert Severity** is selected and its corresponding color gradient bar is shown. **Alert Severity** is the maximum level of alerts in the heatmap rectangle. Values range from **0** - **2**, as indicated in the color gradient bar, where **2** is the highest **Alert Severity**:

Red indicates that one or more resources associated with that application currently has an alert in an alarm state.

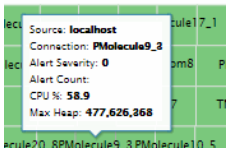
Yellow indicates that one or more resources associated with that application currently have an alert in a warning state.

Green indicates that no resources associated with that application have alerts in a warning or alarm state.

In most heatmaps, you can also drill-down to a *Summary* display containing detailed data for the resource. You can also open a new window and then drill-down. The drill-down opens a display that contains relevant and more detailed data.

Mouse-over

The mouse-over functionality provides additional detailed data in a tool tip when you mouse-over a heatmap. The following figure illustrates mouse-over functionality in a heatmap object.



Log Scale

Typically, heat maps provide the **Log Scale** option, which enables visualization on a logarithmic scale. This option should be used when the range in your data is very broad. For example, if you have data that ranges from the tens to the thousands, then data in the range of tens will be neglected visually if you do not check this option. This option makes data on both extreme ranges visible by using the logarithmic of the values rather than the actual values.

Tables

Monitor tables contain the same data that is shown in the heatmap in the same View. Tables provide you a text and numeric view of the data shown in that heatmap, plus additional data not included the heatmap.

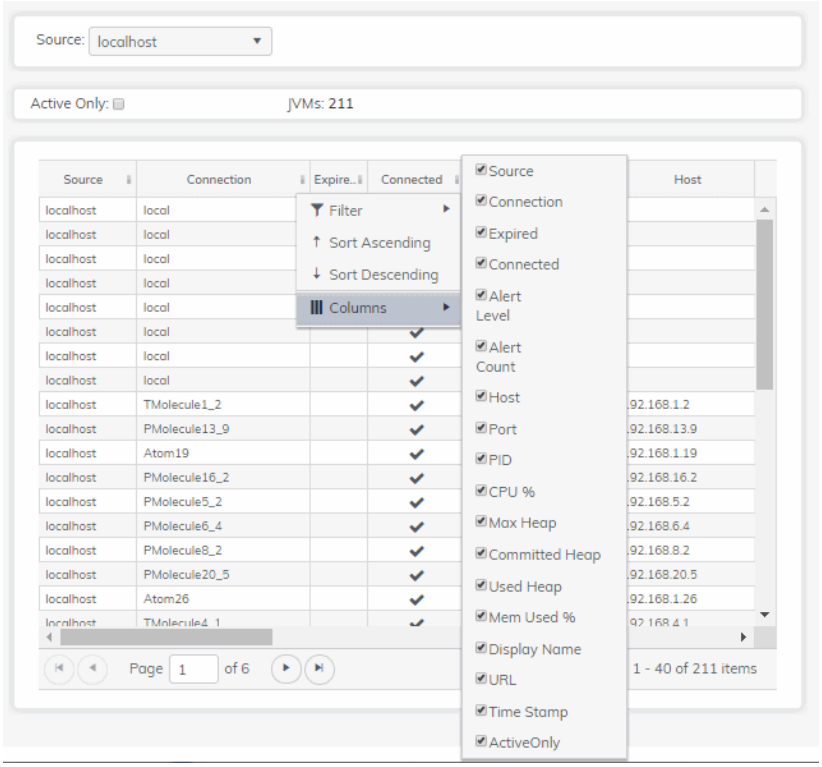


Table rows also sometimes use color to indicate the current most critical alert state for all resources associated with a given row. For example, the color coding is typically as follows:

Red indicates that one or more resources associated with that node process currently has an alert in an alarm state.

Yellow indicates that one or more resources associated with that node process currently have an alert in a warning state.

Green indicates that no resources associated with that node process currently have an alert in a warning or alarm state.

Gray indicates that the resource is in an **Expired** state.

Tables support advanced HTML interactive features such as sorting on multiple columns, filtering on multiple columns, column resizing, column reordering, and hiding columns. Many of these features are accessed from the column menu, shown in the screen shot above, which you open by clicking on the menu icon in a column's header.

Some tables in the **Components** tab gray out rows when they’re in an expired state. A row is expired when data has not been received within the time specified in the solution package that is hosting the data.

Also see:

- Multiple Column Sorting
- Column Visibility
- Column Filtering
- Column Locking
- Column Reordering
- Saving Settings
- Row Paging

Multiple Column Sorting

Click on a column header to sort the table by that column. On the first click, the column is sorted in ascending order (smallest value at the top), on the second click the sort is in descending order, and on the third click, the column is returned to its original unsorted state. A sort on a string column is case-insensitive.

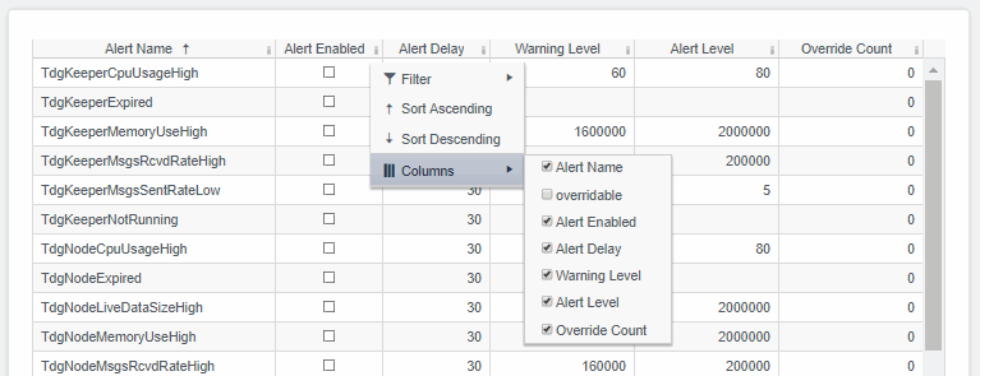
To sort multiple columns, click on the column header for each column you want to sort. The sorting is performed in the order that the column headers were clicked. Multiple column sorting is a very useful feature, but can also cause confusion if you intend to sort on a single column, but forget to "unsort" any previously selected sort columns first. You should check for the up/down sort icon in other column headers if a sort gives unexpected results.

The row selection is cleared if the sort is changed or if columns are resized or reordered.

Column sorting is reflected in an export to HTML and Excel.

Column Visibility

You can hide or show columns in the table by clicking on any column's menu icon, and choosing **Columns** from the menu. This opens a submenu with a check box for each column that toggles the visibility of the column. All columns in the data table appear in the Columns menu, even those that are initially hidden.



The leftmost column (the row header column) cannot be hidden.

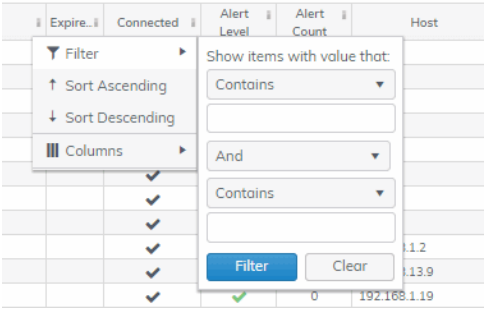
Column visibility changes are NOT reflected in an export to HTML and Excel.

Column Filtering

You can create a filter on any column. If filters are created on multiple columns, then only the rows that pass all of the filters are displayed. That is, if there are multiple filters they are logically "ANDed" together to produce the final result.

The background of a column's menu icon changes to white to indicate that a filter is defined on that column. This is intended to remind you which columns are filtered.

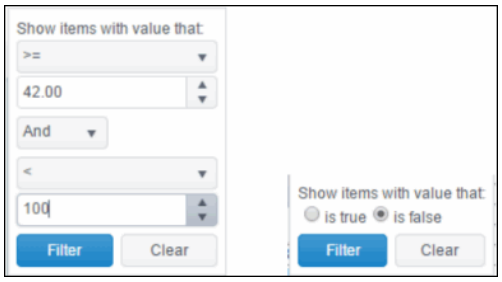
You can configure a filter on any column by clicking on the column's menu icon and choosing **Filter** from the menu. This opens the **Column Filter** dialog:



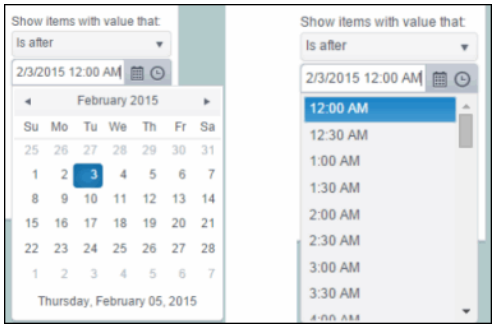
Options in the **Column Filter** dialog vary according to the data type of the selected column:

- **String columns**: You can enter a filter string such as "abc" and, from the dropdown list, select the operator (equal to, not equal to, starts with, contains, etc) to be used when comparing the filter string to each string in the column. All of the filter comparisons on strings are case-insensitive. You can optionally enter a second filter string (e.g. "xyz") and specify if an AND or OR combination should be used to combine the first and second filter results on the column.
- **Numeric columns**: You can enter numeric filter values and select arithmetic comparison operators, (=, !=, >, >=, <, <=). You can optionally enter a second filter value and comparison operator, and specify if an AND or OR combination should be used to combine the first and second filter results.
- **Boolean columns**: You simply select whether matching items should be true or false.

The numeric and boolean filter dialogs are shown below.



- **Date columns**: You can select a date and time and choose whether matching items should have a timestamp that is the same as, before, or after the filter time. The date is selected by clicking on the calendar icon and picking a date from a calendar dialog. The time is selected by clicking on the time icon and picking a time from a dropdown list:



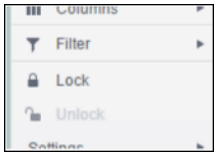
Alternatively, a date and time can be typed into the edit box. The strings shown in a date column are formatted by the Display Server using its time zone. But if a filter is specified on a date column, the date and time for the filter are computed using the client system's time zone. This can be confusing if the Display Server and client are in different time zones.

Data updates to the table are suspended while the filter menu is opened. The updates are applied when the menu is closed.

Column filtering is reflected in an export to HTML and Excel.

Column Locking

The leftmost column is "locked" in position, meaning that it does not scroll horizontally with the other columns in the table. If the row header is enabled, then two items labeled **Lock** and **Unlock** appear in the column menu. These can be used to add or remove additional columns from the non-scrolling row header area.



If the row header is enabled, at least one column must remain locked.

Column locking is NOT reflected in an export to HTML and Excel.

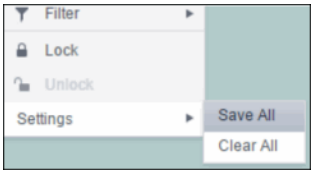
Column Reordering

You can reorder the table columns by dragging and dropping a column's header into another position. Dragging a column into or out of the row header area (the leftmost columns) is equivalent to locking or unlocking the column.

Column reordering is NOT reflected in an export to HTML and Excel.

Saving Settings

You can permanently save all of the custom settings made to the table, including filtering, sorting, column size (width), column order, column visibility, and column locking. This is done by opening any column menu, clicking **Settings**, and then clicking **Save All**:



The table's settings are written as an item in the browser's local storage. The item's value is a string containing the table's settings. The item uses a unique key comprised of the URL path name, the display name, and the table's RTView object name. If the Thin Client's login feature is enabled, the key will also include the username and role, so different settings can be saved for each user and role for a table on any given display, in the same browser and host.

If you save the table settings and navigate away from the display or close the browser, then the next time you return to the display in the same browser the settings are retrieved from the browser's local storage and applied to the table. The browser's local storage items are persistent, so the table settings are preserved if the browser is closed and reopened or if the host system is restarted.

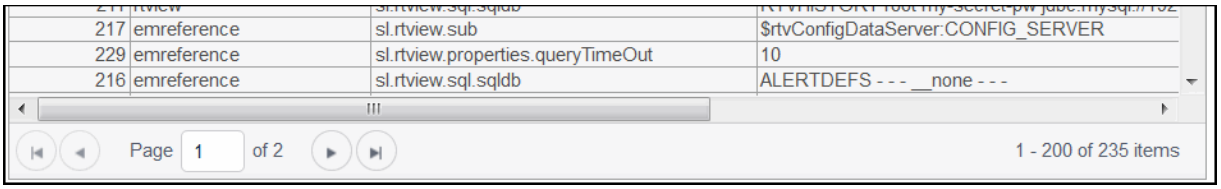
Note that each browser has its own local storage on each host. The local storage items are not shared between browsers on the same host or on different hosts. So, if a user logs in as Joe with **role = admin**, in Internet Explorer on host H1, and saves table settings for display X, then those table settings are restored each time a user logs in as Joe, role admin, on host H1 and opens display X in Internet Explorer. But if all the same is true except that the browser is Chrome, then the settings saved in Internet Explorer are not applied. Or if the user is Joe and role is admin and the browser is IE and the display is X, but the host system is H2 not H1, then the table settings saved on H1 are not applied.

Revert Table Settings

You can delete the table's item from local storage by clicking **Settings> Clear All** in any column menu. This permanently deletes the saved settings for the table and returns the table to the state defined in the display file.

Row Paging

If the data table contains more than one 200 rows, page controls appear at the bottom of the table.



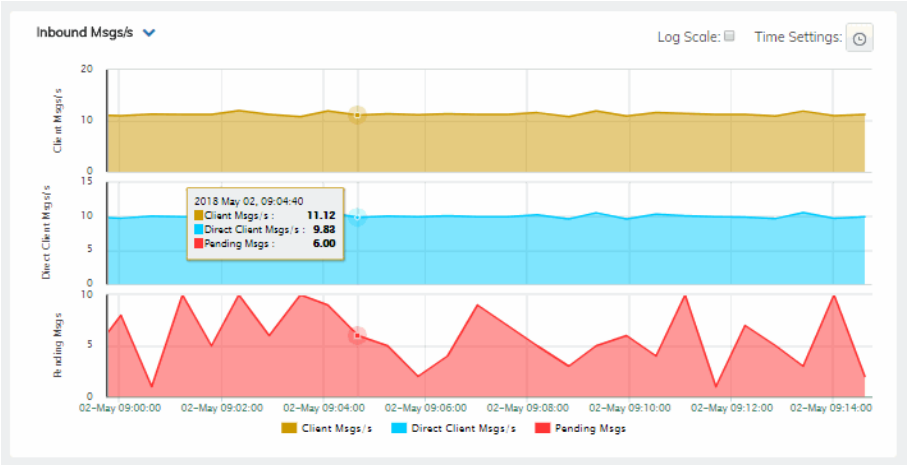
217	emreference	sl.rtview.sub	$rtvConfigDataServer:CONFIG_SERVER
229	emreference	sl.rtview.properties.queryTimeOut	10
216	emreference	sl.rtview.sql.sqldb	ALERTDEFS - - - __none - - -

Page 1 of 2 1 - 200 of 235 items

Trend Graphs

Monitor trend graphs enable you to view and compare performance metrics over time. You can use trend graphs to assess utilization and performance trends.

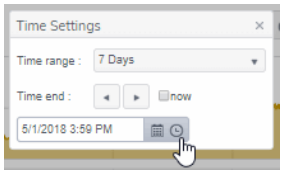
For example, the following figure illustrates a typical Monitor trend graph.



Time Settings

By default, the time range end point is the current time. To change the time range, click the **Time Settings** and either:

- choose a **Time range** from 5 Minutes to 7 Days in the drop-down menu.
- specify begin/end dates using the calendar ..
- specify begin/end time using the clock .



Toggle forward/backward in the trend graph per the period you choose (from the **Time range** drop-down menu) using arrows .

Restore settings to current time by selecting **now** .

Mouse-over

The mouse-over functionality provides additional detailed data in an over imposed pop-up window when you mouse-over trend graphs.

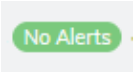
Log Scale

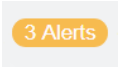
The Log Scale option enables visualization on a logarithmic scale. This option should be used when the range in your data is very broad. For example, if you have data that ranges from the tens to the thousands, then data in the range of tens will be neglected visually if you do not check this option. This option makes data on both extreme ranges visible by using the logarithmic of the values rather than the actual values.

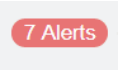
Icons and Buttons

The following describes GUI icons and behavior in the title bar.

The current local date and time. If the time is incorrect, this might indicate that the monitor stopped running. When the date and time is correct and the **Data** indicator is green, this is a strong indication that the platform is receiving current and valid data.



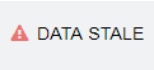




ALERTS: Opens the **Alerts Table**, shows the total number of alerts associated with items currently in the display as well as the maximum alert severity of these, where:

- Green indicates that no metrics have exceeded their alert thresholds.
- Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold.
- Red indicates that one or more metrics exceeded their ALARM LEVEL threshold.

DATA: The data source is currently connected. When the date and time is correct and the **DATA** indicator is green, this is a strong indication that the platform is receiving current and valid data.



DATA STALE: The data source is currently disconnected. There has been no response from the Data Server for 31+ seconds.

Log Scale

Select to enable a logarithmic scale. Use **Log Scale** to see usage correlations for data with a wide range of values. For example, if a minority of your data is on a scale of tens, and a majority of your data is on a scale of thousands, the minority of your data is typically not visible in non-log scale graphs.

Log Scale makes data on both scales visible by applying logarithmic values rather than actual values to the data.

Drop-down menus filter the item/s you want to view. Options differ among displays.

Displays

This section describes displays in the **Displays** tab.

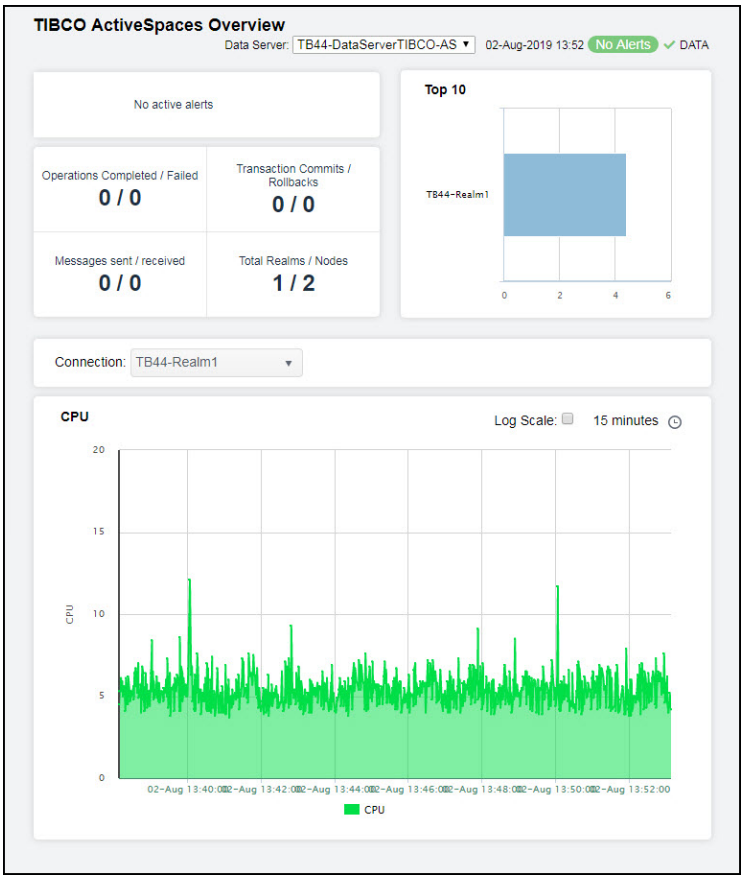
TIBCO ActiveSpaces Overview

The **TIBCO ActiveSpaces Overview** is the top-level display for the TIBCO ActiveSpaces Monitor, which provides a good starting point for immediately getting the status of all your operations, transactions, messages, and realms on your Data Server. You can select the RTView DataServer for which you want to see data and easily view the current data for that DataServer including:

- The total number of active alerts for the selected DataServer, including the total number of critical and warning alerts.
- The current number of operations completed and failed.
- The number of transactions committed and rolled back.
- The number of messages sent and received.
- The total number of realms and nodes.
- A visual list of the top 10 realms containing the total operations/messages/transactions/realms on your connected DataServer.

You can hover over each region in the upper half of the Overview to see more detail. You can also drill down to see even more detail by clicking on each respective region in the Overview. For example, clicking on the alerts in the **CRITICAL** and **WARNING** alerts region opens the **Alerts Table by Components** display.

The bottom half of the display provide a **CPU** trend graph representing CPU usage percentage for a selected connection. You can hover over the trend graph to see the values at a particular time. You can specify the time range for the trend graph and view data based on a log scale, which enables visualization on a logarithmic scale and should be used when the range in your data is very broad.



TIBCO ActiveSpaces HTML Views

The following TIBCO ActiveSpaces Views can be found under **Components** tab > **Middleware** > **TIBCO ActiveSpaces**:

- Grids Views - HTML: The displays in this View provide detailed data for all grids in a heatmap and tabular format, or for a particular grid in tabular and trend graph format.
- Nodes Views - HTML: The displays in this View provide detailed data for all nodes in a heatmap or tabular format.
- Proxies Views - HTML: The displays in this View provide detailed data for all proxies in a heatmap and tabular format.
- Keepers Views - HTML: The displays in this View provide detailed data for all keepers in a heatmap or tabular format.

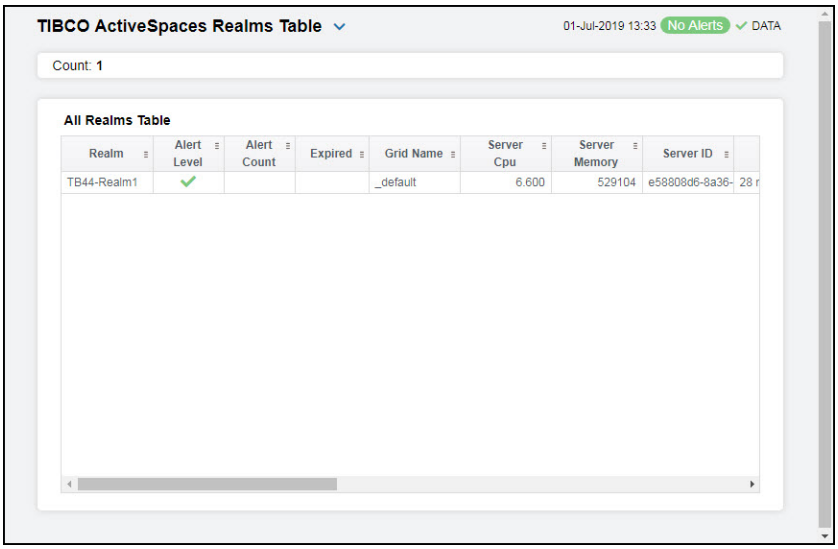
Grids Views - HTML

These displays provide detailed data for all grids in a heatmap and tabular format. Clicking **Grids** in the left/navigation menu opens the TIBCO ActiveSpaces Grids Table - HTML, which provides a tabular view of your grids and their associated metrics. Displays in this View are:

- **All Grids Heatmap**: Opens the TIBCO ActiveSpaces Grids Heatmap - HTML display, which provides a heatmap view of all grids.
- **Single Grid Summary:** Opens the TIBCO ActiveSpaces Grid Summary - HTML display, which provides a view of the current and historical metrics for a single grid.
- **Realm Servers**: Opens the TIBCO ActiveSpaces Realm Server - HTML display, which provides a view of the server CPU percent usage and the server memory used by the realm server manging the grid (in KBs) in a trend graph format.

TIBCO ActiveSpaces Grids Table - HTML

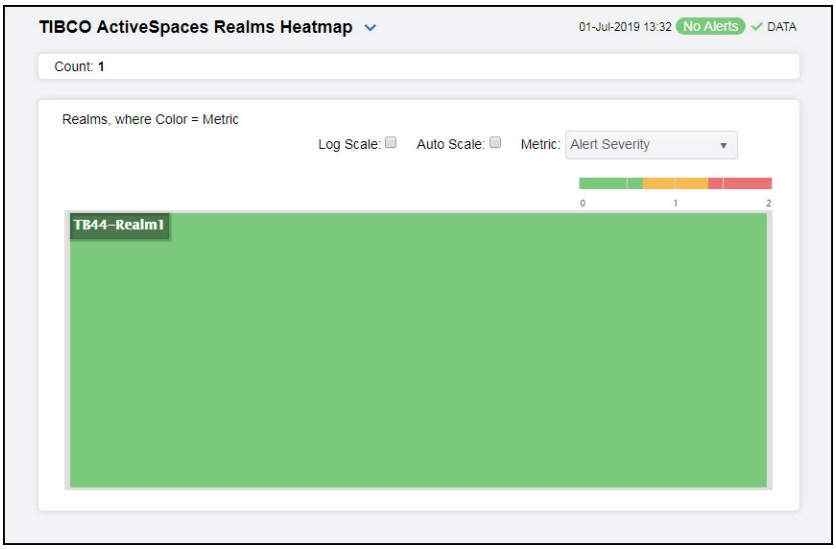
The table in this display provides a view of all of your realms and their associated metric data including alert level, alert count, and the current value of each gathered metric. You can click a column header to sort column data in numerical or alphabetical order, and drill-down and investigate by double-clicking a row to view details for the selected realm in the "TIBCO ActiveSpaces Realm Summary - HTML" display.



TIBCO ActiveSpaces Grids Heatmap - HTML

Clicking **All Realms Heatmap** in the left/navigation menu opens the **TIBCO ActiveSpaces Realms Heatmap**, which provides an easy-to-view interface that allows you to quickly identify the current status of each of your realms for each available metric. You can view the realms in the heatmap based on the following metrics: current alert severity, alert count, CPU usage, memory usage, operations completed, and operations failed. By default, this display shows the heatmap based on the **Alert Severity** metric.

The heatmap is organized so that each rectangle represents a space. The rectangle color indicates the most critical alert state. Click on a node to drill-down to the "TIBCO ActiveSpaces Realm Summary - HTML" display and view metrics for a particular realm. Toggle between the commonly accessed displays by clicking the drop down list on the display title. Mouse-over rectangles to view more details about realm performance and status.



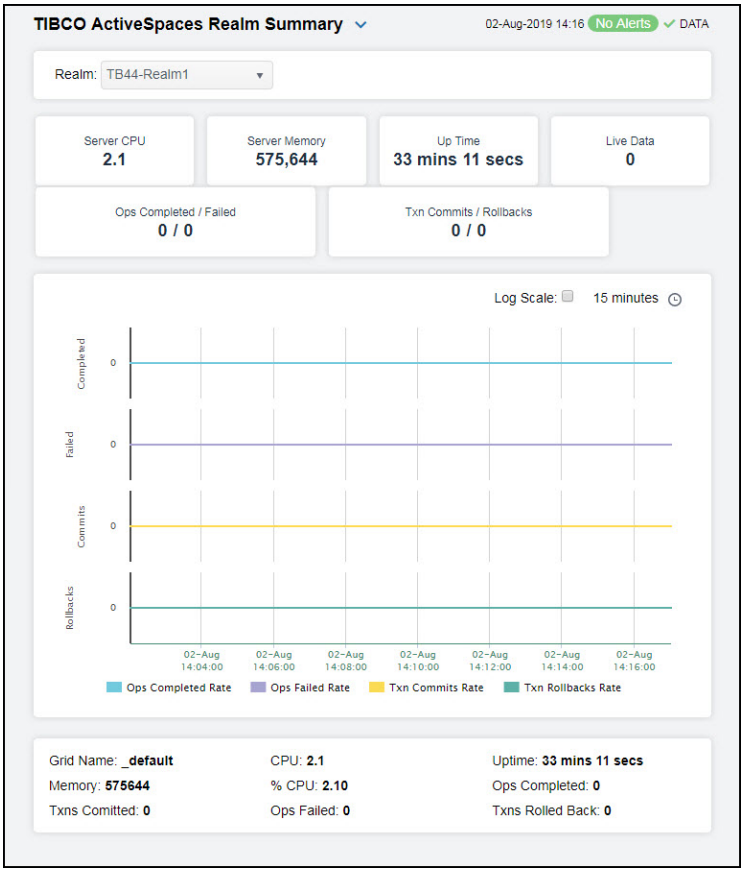
Available Metrics

Select the metric driving the heatmap display. The default is Alert Severity. Each Metric has a color gradient bar that maps values to colors. The heatmap is organized by realms, where each rectangle represents a realm. Mouse-over any rectangle to display the current values of the metrics for the realm. Click on a rectangle to drill-down to the associated "TIBCO ActiveSpaces Realm Summary - HTML" **display for a detailed view of metrics for that particular realm.**

Alert Severity	The current alert severity. Values range from **0** - **2**, as indicated in the color gradient bar, where **2** is the highest Alert Severity: Red indicates that one or more metrics exceeded their ALARM LEVEL threshold. Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold. Green indicates that no metrics have exceeded their alert thresholds.
Alert Count	The total number of alarm and warning alerts in a given item (index) associated with the rectangle. The color gradient bar shows the range of the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of alerts in the heatmap. The middle value in the gradient bar indicates the middle value of the range.
CPU Usage	The milliseconds of CPU time accumulated by the process after each update interval. The color gradient bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgRealmServerCpuUsageHigh**. The middle value in the gradient bar indicates the middle value of the range.
Memory	The amount of memory used in the realm. The color gradient bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgRealmServerMemoryUseHigh**. The middle value in the gradient bar indicates the middle value of the range.
Ops Completed	The number of operations completed in the realm.The color gradient bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of operations in the heatmap. The middle value in the gradient bar indicates the middle value of the range.
Ops Failed	The number of failed operations in the realm. The color gradient bar, populated by the current heatmap, shows the range of the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of operations that have failed in the heatmap. The middle value in the gradient bar indicates the middle value of the range.

TIBCO ActiveSpaces Grid Summary - HTML

Clicking **Single Grid Summary** in the left/navigation menu opens the **TIBCO ActiveSpaces Grid Summary** display, which provides a view of the current and historical metrics for a single grid. Hover over the boxes at the top of the display to view additional information. In the trend graph region, you can view the rate of completed operations, the rate of failed operations, the rate of transactions that are committed, and the rate of transactions that are rolled back over a selected time range.



Filter By:

The display might include these filtering options:

Realm	Select the realm for which you want to show data in the display.

Fields and Data:

Grid Name	The name of the grid.
Server Version	The version of the realm server.
Up Time	The amount of time since the realm server was started.
Live Data	The size of the live data.

Ops Completed/Failed The number of operations completed and the number of operations failed in the grid.*

Txn Commits/Rollbacks The total number of transactions committed and the number of transactions rolled back in the grid.*

Trends

Traces the following:

Ops Completed Rate -- traces the number of operations completed per second.

Ops Failed Rate-- traces the number of operations failed per second.

Txn Commits Rate -- traces the number of transactions committed per second.

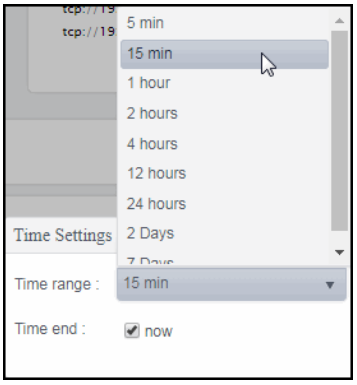
Txn Rollbacks Rate -- traces the number of transactions rolled back per second.

Log Scale

Select to enable a logarithmic scale. Use **Log Scale** to see usage correlations for data with a wide range of values. For example, if a minority of your data is on a scale of tens, and a majority of your data is on a scale of thousands, the minority of your data is typically not visible in non-log scale graphs. **Log Scale** makes data on both scales visible by applying logarithmic values rather than actual values to the data.

Time Settings

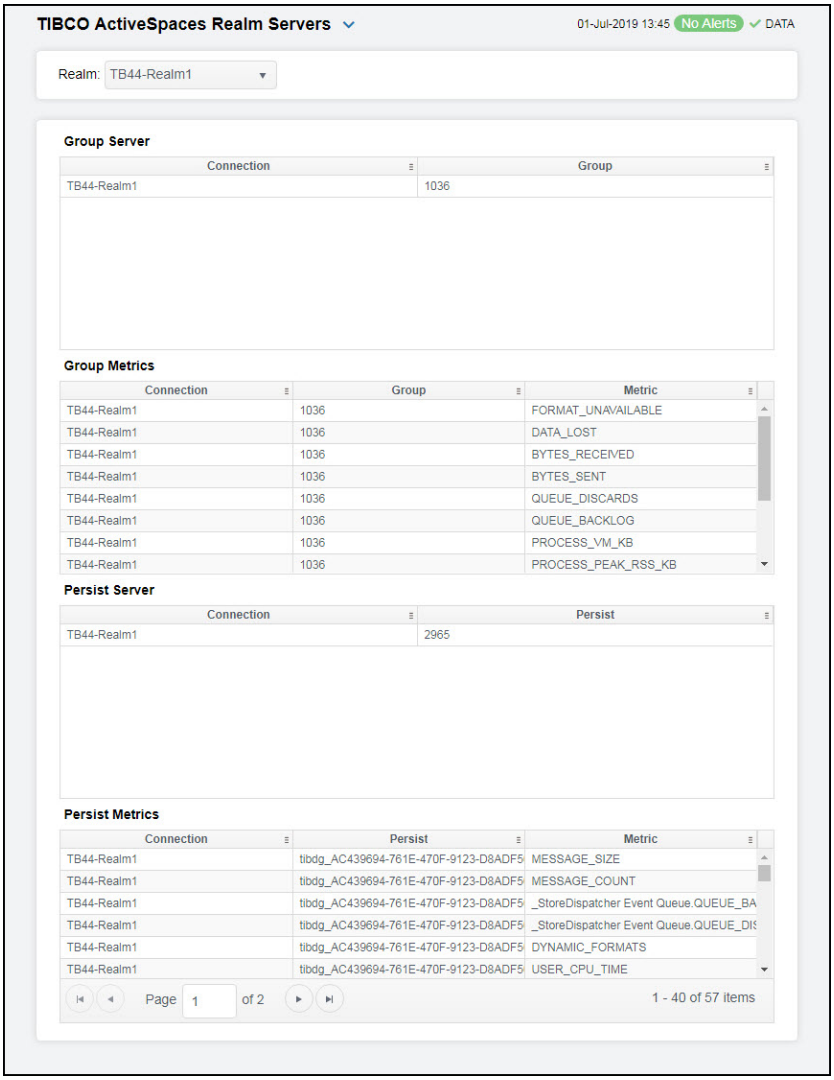
Select a time range from the drop down menu varying from **5 Minutes** to **Last 7 Days**. By default, the time range end point is the current time.



To change the time range, deselect the **now** toggle, which displays some additional date fields. You can click the left and right arrow buttons to decrease the end time by one time period (the time selected in the **Time range** drop down) per click, or you can choose the date and time from the associated calendar and clock icons. You can also enter the date and time in the text field using the following format: **MMM dd, YYYY HH:MM:ss**. For example, Aug 21, 2018 12:24 PM. Click the **now** toggle to reset the time range end point to the current time.

TIBCO ActiveSpaces Realm Server - HTML

Clicking **Realm Server** in the left/navigation menu opens the **TIBCO ActiveSpaces Realm Server** display, which provides a view of the current and historical metrics for the realm server of a single grid. Hover over the boxes at the top of the display to view additional information. In the trend graph region, you can view the server CPU percent usage and the server memory used (in KBs).



Filter By:

The display might include these filtering options:

Realm	Select the realm for which you want to show data in the display.

Fields and Data:

Grid Name	The name of the grid.
Server CPU	The server's CPU usage percentage.
Server Memory	The used memory on the server, in kilobytes.

Up Time The amount of time since the server was started.

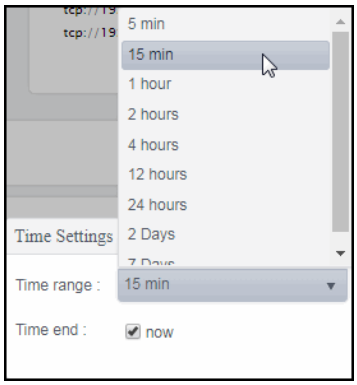
Version The version of the server.

Trends Traces the following:

- **Server CPU** -- traces the server's CPU usage percentage.
- **Server Memory** -- traces the used memory on the server.

Log Scale Select to enable a logarithmic scale. Use **Log Scale** to see usage correlations for data with a wide range of values. For example, if a minority of your data is on a scale of tens, and a majority of your data is on a scale of thousands, the minority of your data is typically not visible in non-log scale graphs. **Log Scale** makes data on both scales visible by applying logarithmic values rather than actual values to the data.

Time Settings Select a time range from the drop down menu varying from **5 Minutes** to **Last 7 Days**. By default, the time range end point is the current time.



To change the time range, deselect the **now** toggle, which displays some additional date fields. You can click the left and right arrow buttons to decrease the end time by one time period (the time selected in the **Time range** drop down) per click, or you can choose the date and time from the associated calendar and clock icons. You can also enter the date and time in the text field using the following format: **MMM dd, YYYY HH:MM:ss**. For example, Aug 21, 2018 12:24 PM. Click the **now** toggle to reset the time range end point to the current time.

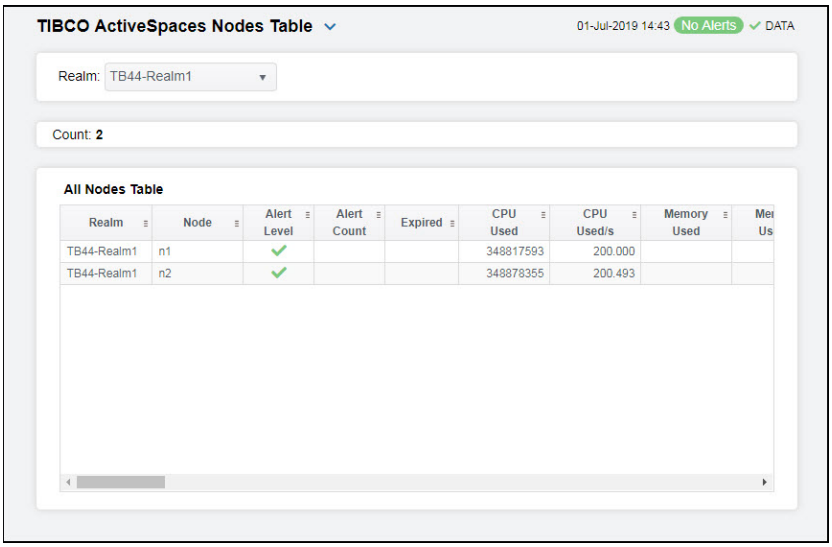
Nodes Views - HTML

These displays provide detailed data for all nodes (in a specific realm) in a heatmap or tabular format. Clicking **Nodes** in the left/navigation menu opens the TIBCO ActiveSpaces Nodes Table - HTML, which provides a tabular view of all nodes (contained within a particular grid) and their associated metrics. Displays in this View are:

- **All Nodes Heatmap**: Opens the TIBCO ActiveSpaces Nodes Heatmap - HTML display, which is a heatmap view of all nodes contained within a particular grid.
- **Single Node Summary**: Opens the TIBCO ActiveSpaces Node Summary - HTML display, which allows you to view metrics and trend data for a particular node.

TIBCO ActiveSpaces Nodes Table - HTML

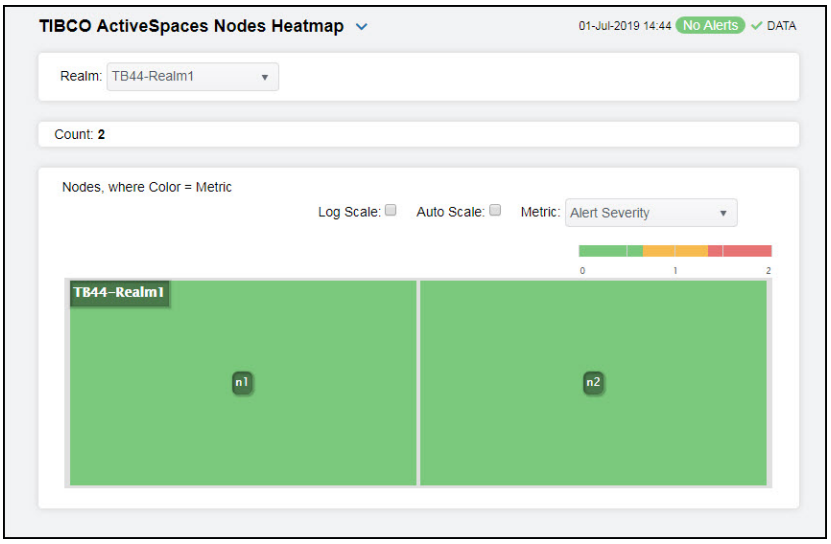
The table in this display provides a view of all nodes and their associated metric data in a specific realm. You can click a column header to sort column data in numerical or alphabetical order, and drill-down and investigate by double-clicking a row to view details for the selected node in the "TIBCO ActiveSpaces Node Summary - HTML" display.



TIBCO ActiveSpaces Nodes Heatmap - HTML

Clicking **All Nodes Heatmap** in the left/navigation menu opens the **TIBCO ActiveSpaces Nodes Heatmap** display, which provides an easy-to-view interface that allows you to quickly identify the current status of each of your nodes for each available metric. You can view the nodes in the heatmap based on the following metrics: current alert severity, alert count, CPU usage, memory usage, rate of failed operations, and rate of completed operations. By default, this display shows the heatmap based on the **Alert Severity** metric.

You can mouse over a rectangle to see additional metrics for a node. Clicking one of the rectangles in the heatmap opens the TIBCO ActiveSpaces Node Summary - HTML display, which allows you to see additional details for the selected node.



Filter By:

Realm Select the realm for which you want to see data.

Fields and Data:

Count The number of nodes listed in the heatmap.

Log Scale Select this check box to use a logarithmic scale, rather than a linear scale, to map from the selected metric value for a cell to the color for the cell. **Log** provides another way to distribute and differentiate values that you might not be able to see on a linear scale due to the dominant nature of large values in a linear scale.

Auto Scale When checked, the values of the selected metric are auto-scaled to its highest defined value. When unchecked, the values of the selected metric display based on the threshold defined for the alert associated with the selected metric. Selecting Auto helps to visualize the range of the values currently present for the selected metric instead of the threshold of the alert that has been associated with the metric. All metrics that have not been associated in the heatmap defaults with alerts use a monochromatic color gradient bar (whites and greens). All metrics that have been associated in the heatmap defaults with alerts use a multi-chromatic color gradient bar (reds, yellows, white, and greens).

Metric

Select the metric driving the heatmap display. The default is **Alert Severity**. Each **Metric** has a color gradient bar that maps values to colors. The heatmap is organized by nodes, where each rectangle represents a node. Mouse-over any rectangle to display the current values of the metrics for the node. Click on a rectangle to drill-down to the associated TIBCO ActiveSpaces Node Summary - HTML display for a detailed view of metrics for that particular node.

Alert Severity The current alert severity. Values range from **0** - **2**, as indicated in the color gradient bar, where **2** is the highest Alert Severity:

- Red indicates that one or more metrics exceeded their ALARM LEVEL threshold.
- Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold.
- Green indicates that no metrics have exceeded their alert thresholds.

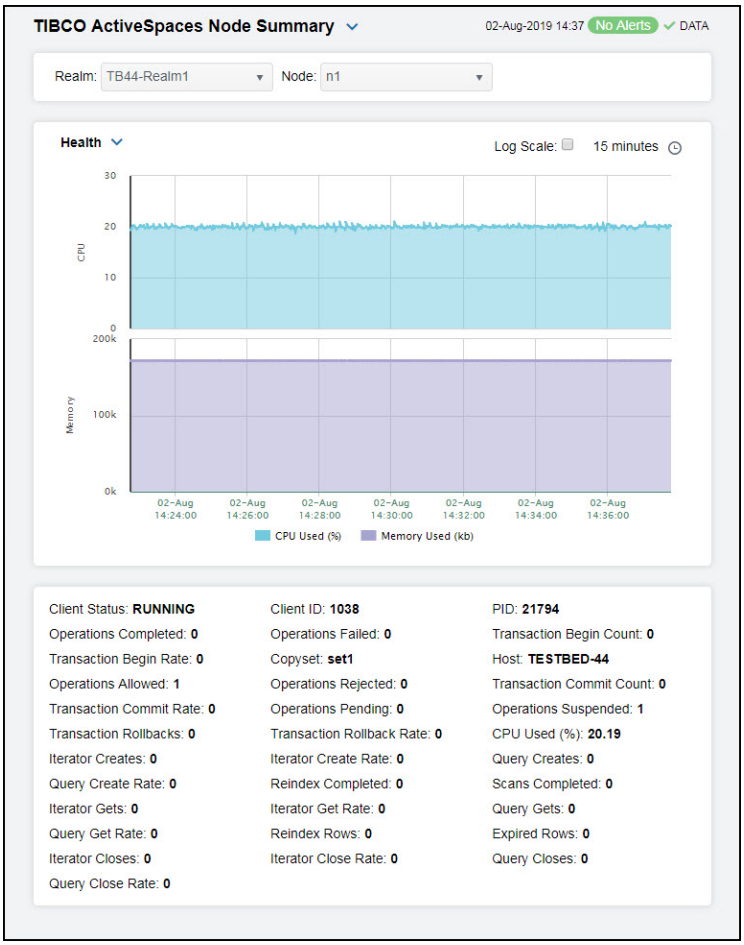
Alert Count The total number of alarm and warning alerts in a given item (index) associated with the rectangle. The color gradient bar shows the range of the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of alerts in the heatmap. The middle value in the gradient bar indicates the middle value of the range.

CPU Usage The milliseconds of CPU time accumulated by the process after the last update interval. The color gradient bar, populated by the current heatmap,

	shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgNodeCpuUsageHigh**. The middle value in the gradient bar indicates the middle value of the range.
Memory	The memory usage for the node. The color gradient bar  , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgNodeMemoryUseHigh**. The middle value in the gradient bar indicates the middle value of the range.
Failed Op Rate	The rate of failed operations.The color gradient bar  , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgNodeOpsFailedRateHigh**. The middle value in the gradient bar indicates the middle value of the range.
Completed Op Rate	The rate of completed operations. The color gradient bar  , populated by the current heatmap, shows the range of the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgNodeOpsCompletedRateLow**. The middle value in the gradient bar indicates the middle value of the range.

TIBCO ActiveSpaces Node Summary - HTML

Clicking **Single Node Summary** in the left/navigation menu opens the **TIBCO ActiveSpaces Node Summary** display, which provides a view of the current and historical metrics for a single node. The trend graph in the bottom half of the display has three options: **Health**, **Live Data**, and **Operations**. **Health** traces the current and historical CPU usage and memory usage over a selected time range. **Live Data** traces the live data size over a selected time range. **Operations** traces the rate of completed operations and the rate of failed operations for the node over a selected time range.



Filter By:

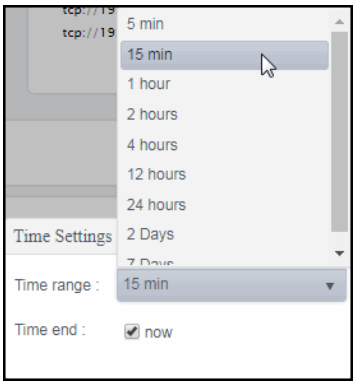
The display might include these filtering options:

Realm	Select the realm (containing the node) for which you want to show data in the display.
Node	Select the node for which you want to show data in the display.
Performance Metric Trends	**Health** Traces the following: **CPU Usage (%)** -- traces the CPU usage percentage for the node. **Memory Used (kb)**-- traces the amount of memory used, in kilobytes. **Live Data** Traces the following: **Live Data Size**-- traces the Live Data Size. **Operations** Traces the following: **Operations Completed Rate** -- traces the rate of completed operations. **Operations Failed Rate** -- traces the rate of failed operations.
Log Scale	Select to enable a logarithmic scale. Use **Log Scale** to see usage correlations for data with a wide range of values. For example, if a minority of your data is on a scale of tens, and a majority of your data is on a scale of thousands, the minority of your data is typically not visible in non-log scale graphs. **Log Scale** makes data on both scales visible by applying logarithmic

values rather than actual values to the data.

Base at Zero Select to use zero (**0**) as the Y axis minimum for all graph traces.

Time Settings Select a time range from the drop down menu varying from **5 Minutes** to **Last 7 Days**. By default, the time range end point is the current time.



To change the time range, deselect the **now** toggle, which displays some additional date fields. You can click the left and right arrow buttons to decrease the end time by one time period (the time selected in the **Time range** drop down) per click, or you can choose the date and time from the associated calendar and clock icons. You can also enter the date and time in the text field using the following format: **MMM dd, YYYY HH:MM:ss**. For example, Aug 21, 2018 12:24 PM. Click the **now** toggle to reset the time range end point to the current time.

Fields and Data:

Grid Name	The name of the grid.*
PID	The process ID of the node.
Transaction Begin Count	The number of transactions started on the node.
Host	The name of the host.
Transaction Commit Count	The number of transactions committed on the node.
Operations Suspended	The number of suspended operations on the node.
CPU Used (%)	The percentage of CPU used
Query Creates	The number of created queries on the node.
Scans Completed	The number of scans completed.
Query Gets	The number of “get” operations on the node.
Expired Rows	The number of expired rows on the node.
Query Closes	The number of closed queries on the node.
Client Status	The current status of the node.

Operations Completed	The number of completed operations on the node.
Transaction Begin Rate	The rate of transactions started on the node.
Operations Allowed	The number of allowed operations on the node.
Transaction Commit Rate	The rate of transactions committed on the node.
Transaction Rollbacks	The number of transactions that have been rolled back on the node.
Iterator Creates	The number of iterator operations on the node.
Query Create Rate	The rate of created queries on the node.
Iterator Gets	The number of "get" iterator operations on the node.
Query Get Rate	The rate of "get" operations on the node.
Iterator Closes	The number of closed iterator operations on the node.
Query Close Rate	The rate of closed queries on the node.
Client ID	The ID of the node.*
Operations Failed	The number of failed operations on the node.
Copyset	The name of the copyset.*
Operations Rejected	The number of rejected operations on the node.
Operations Pending	The number of pending operations on the node.
Transaction Rollback Rate	The rate of transactions that have been rolled back on the node.
Iterator Create Rate	The rate of iterator operations on the node.
Reindex Completed	The number of "reindex" scans on the node.
Iterator Get Rate	The rate of "get" iterator operations on the node.
Reindex Rows	The number of "reindex" rows on the node.
Iterator Close Rate	The rate of closed iterator operations on the node.

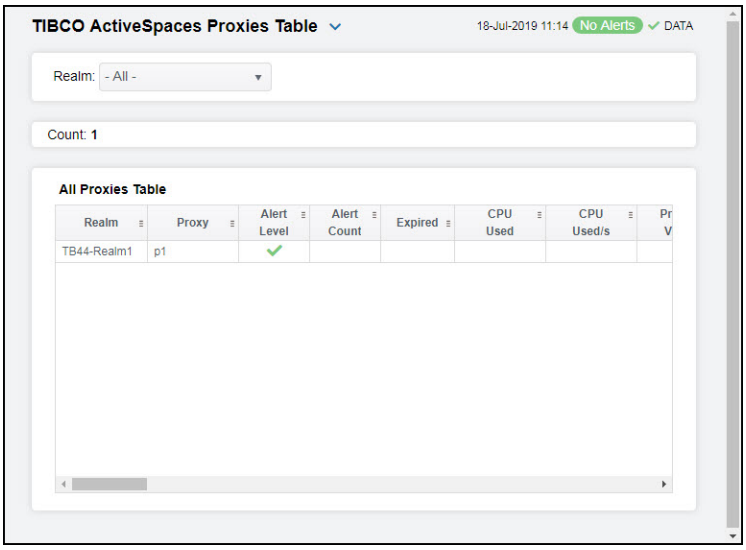
Proxies Views - HTML

These displays provide detailed data for all proxies (in a specific realm) in a heatmap or tabular format. Clicking **Proxies** in the left/navigation menu opens the TIBCO ActiveSpaces Proxies Table - HTML display, which provides a tabular view of your proxies and their associated metrics within a particular realm. Displays in this View are:

- **All Proxies Heatmap**: Opens the TIBCO ActiveSpaces Proxies Heatmap - HTML display, which provides a heatmap view of all proxies contained within a particular realm.
- **Single Proxy Summary**: Opens the TIBCO ActiveSpaces Proxy Summary - HTML display, which allows you to view metrics and trend data for a particular proxy.

TIBCO ActiveSpaces Proxies Table - HTML

The table in this display provides a view of all proxies and their associated metric data in a selected realm. You can click a column header to sort column data in numerical or alphabetical order, and drill-down and investigate by double-clicking a row to view details for the selected proxy in the TIBCO ActiveSpaces Proxy Summary - HTML display



Filter By:

Realm	Select the realm for which you want to view data.
Count	The total number of proxies found for the realm selected in the **Realm** dropdown, which are displayed in the **All Proxies Table**.
All Proxies Table	
Grid Name	The name of the grid.
Proxy	The name of the proxy.
Realm	The name of the realm.
Alert Level	The current alert severity. Red indicates that one or more metrics exceeded their ALARM LEVEL threshold. Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold. Green indicates that no metrics have exceeded their alert thresholds.
Alert Count	The total number of alerts for the proxy.
Expired	When checked, performance data has not been received within the time specified (in seconds) in the **Expire Time** field in the **Duration** region in the RTView Configuration Application > (Project Name) > **Solution Package Configuration** > **TIBCO Active Spaces** > **DATA STORAGE** tab. The **Delete Time** field (also in the **Duration** region) allows you to define the amount of time (in seconds) in which the row will be removed from the table if there is no response.
CPU Used (%)	The percentage of CPU used on the proxy.*
Memory Used (kb)	The amount of memory used, in kilobytes.*
Bytes Received	The number of bytes received.
Bytes	The rate of bytes received.

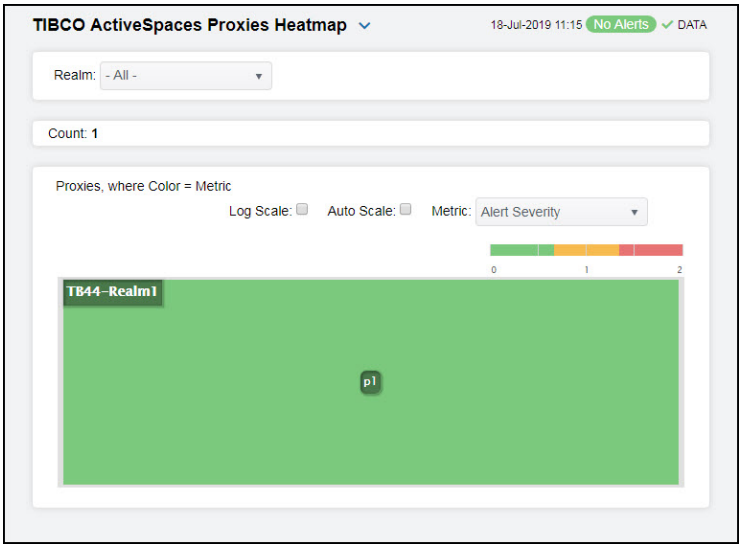
Received/s	
Bytes Sent	The number of bytes sent.
Bytes Sent/s	The rate of bytes sent.
Messages Received	The number of messages received.
Messages Received Rate	The rate of messages received.
Messages Sent	The number of messages sent.
Messages Sent Rate	The rate of messages sent.
Put Rate	The rate of "put" operations (per second) performed on the proxy.*
Get Rate	The rate of "get" operations (per second) performed on the proxy.*
Remove Rate	The rate of "remove" operations (per second) performed on the proxy.*
Transaction Begin Rate	The rate of transactions being started on the proxy.*
Transaction Commit Rate	The rate of transactions being committed on the proxy.*
Transaction Rollback Rate	The rate of transactions rolled back on the proxy.*
Iterator Create Rate	The rate of iterator operations being created on the proxy.*
Iterator Get Rate	The rate of "get" iterator operations on the proxy.*
Iterator Close Rate	The rate of iterator operations being closed on the proxy.*
Query Create Rate	The rate of created queries on the proxy.*
Query Get Rate	The rate of "get" queries on the proxy.*
Query Close Rate	The rate of closed queries on the proxy.*
Queries	The number of queries on the proxy.*
Statements	The number of statements on the proxy.*
Iterators	The number of Iterators on the proxy.*
Listeners	The number of listeners on the proxy.*
Client Status	The status of the client.*
Client ID	The ID of the client.*
PID	The process ID of the host.*
Gets	The total number of "get" operations performed on the proxy.*
Transaction Begins	The number of transactions started on the proxy.*
Client Connected	The number of clients connected.*
Host	The name of the host.*
Puts	The total number of "put" operations performed on the proxy.*
Transaction Commits	The number of commit transactions on the proxy.*
Removes	The total number of "remove" operations performed on the proxy.*
Transaction Rollbacks	The number of transactions rolled back on the proxy.*
Iterator Creates	The number of iterator operations on the proxy.*
Query Creates	The number of created queries on the proxy.*
Iterator Gets	The number of "get" iterator operations on the proxy.*
Query Gets	The number of "get" queries on the proxy..*

Iterator Closes	The number of iterator operations being closed on the proxy.*
Query Closes	The number of closed queries on the proxy.*
Time Stamp	The date and time the row data was last updated.

TIBCO ActiveSpaces Proxies Heatmap - HTML

Clicking **All Proxies Heatmap** in the left/navigation menu opens the **TIBCO ActiveSpaces Proxies Heatmap**, which provides an easy-to-view interface that allows you to quickly identify the current status of each proxy for each available metric. You can view the proxies in the heatmap based on the following metrics: current alert severity, alert count, CPU usage, memory used, iterator count, listener count, query count, and statement count. By default, this display shows the heatmap based on the **Alert Severity** metric.

You can mouse over a rectangle to see additional metrics for a proxy. Clicking one of the rectangles in the heatmap opens the TIBCO ActiveSpaces Proxy Summary - HTML display, which allows you to see additional details for the selected proxy.



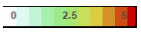
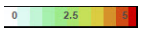
Available Metrics

Select the metric driving the heatmap display. The default is **Alert Severity**. Each **Metric** has a color gradient bar that maps values to colors. The heatmap is organized by proxies, where each rectangle represents a proxy. Mouse-over any rectangle to display the current values of the metrics for the proxy. Click on a rectangle to drill-down to the associated "TIBCO ActiveSpaces Proxy Summary - HTML" display for a detailed view of metrics for that particular proxy.

Alert Severity

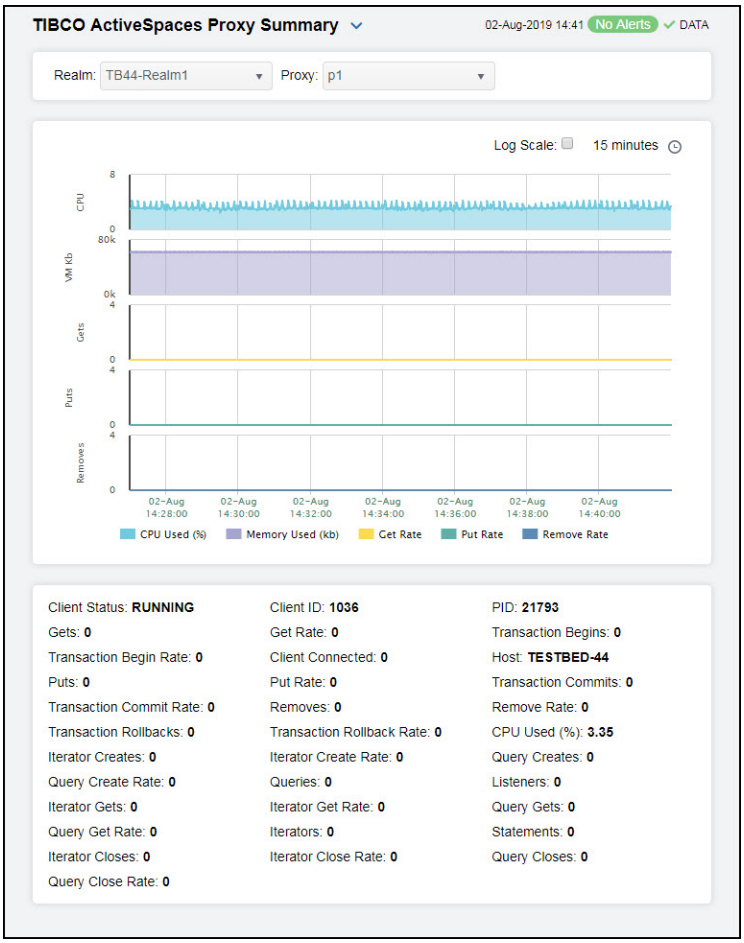
The current alert severity. Values range from **0** - **2**, as indicated in the color gradient bar, where **2** is the highest Alert Severity:

- Red indicates that one or more metrics exceeded their ALARM LEVEL threshold.
- Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold.
- Green indicates that no metrics have exceeded their alert thresholds.

Alert Count	The total number of alarm and warning alerts in a given item (index) associated with the rectangle. The color gradient bar  shows the range of the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of alerts in the heatmap. The middle value in the gradient bar indicates the middle value of the range.
CPU Usage	The CPU usage rate for the proxy. The color gradient bar  , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgProxyCpuUsageHigh**. The middle value in the gradient bar indicates the middle value of the range.
Memory	The memory usage for the proxy. The color gradient bar  , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgProxyMemoryUseHigh**. The middle value in the gradient bar indicates the middle value of the range.
Iterator Count	The number of iterators on the proxy. The color gradient  bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of iterators in the proxy. The middle value in the gradient bar indicates the middle value of the range.
Listener Count	The number of listeners on the proxy. The color gradient  bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of listeners in the proxy. The middle value in the gradient bar indicates the middle value of the range.
Query Count	The number of queries on the proxy. The color gradient  bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of queries in the proxy. The middle value in the gradient bar indicates the middle value of the range.
Statement Count	The number of statements on the proxy. The color gradient  bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of statements in the proxy. The middle value in the gradient bar indicates the middle value of the range.

TIBCO ActiveSpaces Proxy Summary - HTML

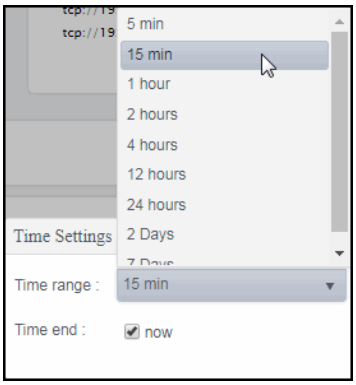
Clicking **Single Proxy Summary** in the left/navigation menu opens the **TIBCO ActiveSpaces Proxy Summary**, which provides a view of the current and historical metrics for a single proxy. The trend graph in the display traces the current and historical rate of CPU usage, process virtual memory usage, rate of get operations, rate of put operations, and the rate of remove operations.



Filter By:

The display might include these filtering options:

Realm	Select the realm (containing the proxy) for which you want to show data in the display.
Proxy	Select the proxy for which you want to show data in the display.
Performance Metric Trends	Traces the following: **CPU Usage (%)** -- traces the percentage of CPU used for the node. **Memory Used (kb)**-- traces the amount of memory used, in kilobytes. **Get Ops/sec** -- traces the rate of "get" operations on the proxy. **Put Ops/sec**-- traces the rate of "put" operations on the proxy. **Remove Ops/sec** -- traces the rate of "remove" operations on the proxy.
Log Scale	Select to enable a logarithmic scale. Use **Log Scale** to see usage correlations for data with a wide range of values. For example, if a minority of your data is on a scale of tens, and a majority of your data is on a scale of thousands, the minority of your data is typically not visible in non-log scale graphs. **Log Scale** makes data on both scales visible by applying logarithmic values rather than actual values to the data.
Base at Zero	Select to use zero (**0**) as the Y axis minimum for all graph traces.
Time Settings	Select a time range from the drop down menu varying from **5 Minutes** to **Last 7 Days**. By default, the time range end point is the current time.



To change the time range, deselect the **now** toggle, which displays some additional date fields. You can click the left and right arrow buttons to decrease the end time by one time period (the time selected in the **Time range** drop down) per click, or you can choose the date and time from the associated calendar and clock icons. You can also enter the date and time in the text field using the following format: **MMM dd, YYYY HH:MM:ss**. For example, Aug 21, 2018 12:24 PM. Click the **now** toggle to reset the time range end point to the current time.

Fields and Data:

Grid Name	The name of the grid.*
PID	The process ID of the proxy.
Transaction Begins	The number of transactions started on the proxy.
Host	The name of the host.
Transaction Commits	The number of transactions committed on the proxy.
Remove Rate	The rate of "remove" operations on the proxy.
CPU Used (%)	The percentage of CPU used.
Query Creates	The number of queries created on the proxy.
Listeners	The total number of listeners on the proxy.
Query Gets	The number of "get" queries on the proxy.
Statements	The total number of statements on the proxy.
Query Closes	The number of queries closed on the proxy.
Client Status	The current status of the proxy.
Gets	The number of "get" operations on the proxy.
Transaction Begin Rate	The rate of transactions being started on the proxy.
Puts	The number of "put" operations on the proxy.
Transaction Commit Rate	The rate of transactions being committed on the proxy.
Transaction Rollbacks	The number of transactions rolled back on the proxy.
Iterator	The number of iterator operations created on the proxy.

Creates	
Query Create Rate	The rate of queries being created on the proxy.
Iterator Gets	The number of "get" iterator operations on the proxy.
Query Get Rate	The rate of "get" queries being created on the proxy.
Iterator Closes	The number of closed iterator operations on the proxy.
Query Close Rate	The rate of queries being closed on the proxy.
Client ID	The ID of the proxy.*
Get Rate	The rate of "get" operations on the proxy.
Client Connected	The number of clients connected.
Put Rate	The rate of "put" operations on the proxy.
Removes	The number of "remove" operations on the proxy.
Transaction Rollback Rate	The rate of transactions being rolled back on the proxy.
Iterator Create Rate	The rate of iterator operations being created on the proxy.
Queries	The number of queries created on the proxy.
Iterator Get Rate	The rate of "get" iterator operations being created on the proxy.
Iterators	The number of iterator operations created on the proxy.
Iterator Close Rate	The rate of iterator operations being closed on the proxy.

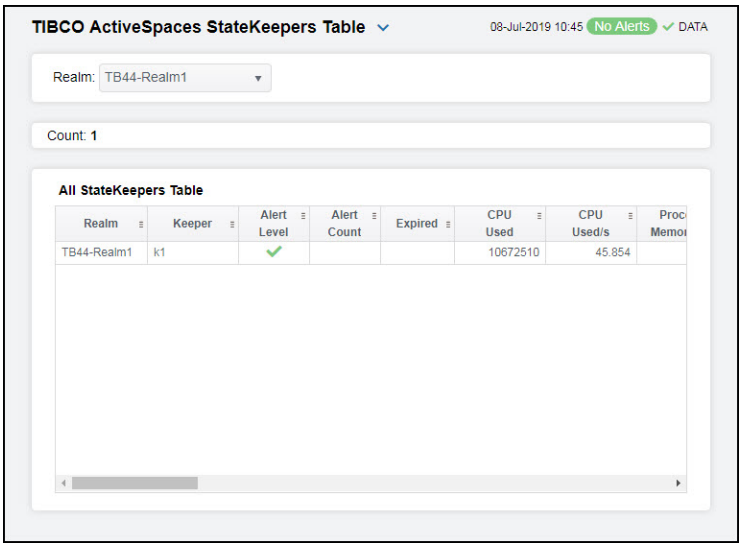
Keepers Views - HTML

These displays provide detailed data for all keepers in a heatmap or tabular format, as well as metrics and trend data for a particular keeper. Clicking **Keepers** in the left/navigation menu opens the TIBCO ActiveSpaces StateKeepers Table - HTML display, which provides a tabular view of all keepers and their associated metrics within a particular realm. Displays in this View are:

- **All Keepers Heatmap**: Opens the TIBCO ActiveSpaces StateKeepers Heatmap - HTML display, which is a heatmap view of all keepers contained within a particular realm.
- **Single Keeper Summary**: Opens the TIBCO ActiveSpaces Keeper Summary - HTML display, which allows you to view metrics and trend data for a particular keeper.

TIBCO ActiveSpaces StateKeepers Table - HTML

The table in this display provides a view of all keepers and their associated metric data for a specific realm. You can click a column header to sort column data in numerical or alphabetical order, and drill-down and investigate by double-clicking a row to view details for the selected keeper in the TIBCO ActiveSpaces Keeper Summary - HTML display



Filter By:

Realm	Select the realm for which you want to view data.
Count	The total number of keepers found for the realm selected in the **Realm** dropdown, which are displayed in the **All StateKeepers Table**.

All StateKeepers Table

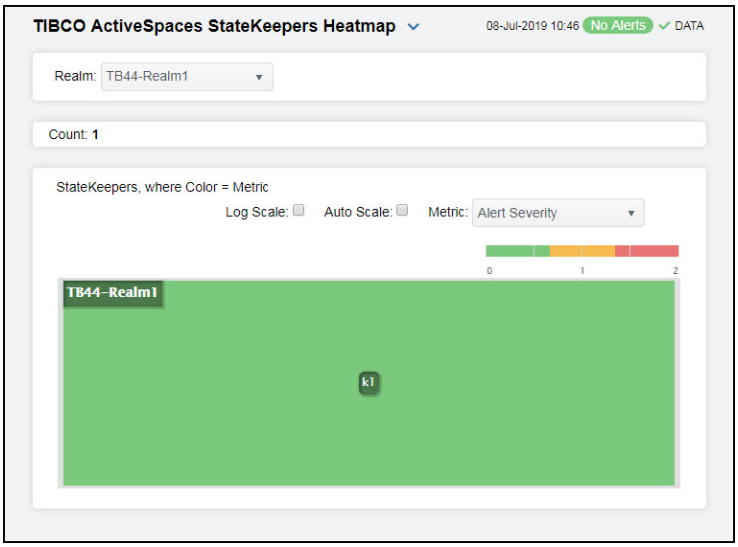
Grid Name	The name of the grid.
Keeper	The name of the keeper.
Realm	The name of the realm.
Alert Level	The current alert severity. Red indicates that one or more metrics exceeded their ALARM LEVEL threshold. Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold. Green indicates that no metrics have exceeded their alert thresholds.
Alert Count	The total number of alerts for the host.
Expired	When checked, performance data has not been received within the time specified (in seconds) in the **Expire Time** field in the **Duration** region in the RTView Configuration Application > (Project Name) > **Solution Package Configuration** > **TIBCO Active Spaces** > **DATA STORAGE** tab. The **Delete Time** field (also in the **Duration** region) allows you to define the amount of time (in seconds) in which the row will be removed from the table if there is no response.
CPU Used (%)	The percentage of CPU memory used by the keeper.
Memory Used (kb)	The memory used by the keeper, in kilobytes.
Bytes Received	The number of bytes received.
Bytes Received/s	The rate of bytes received.
Bytes Sent	The number of bytes sent.
Bytes Sent/s	The rate of bytes sent.
Messages Received	The number of messages received.
Messages Rcvd Rate	The rate of messages received.
Message Sent	The rate of messages sent.

Rate	
Messages Sent	The number of messages sent.
Client Status	The current status of the client on which the keeper resides.*
Client ID	The ID of the client.*
PID	The process ID of the StateKeeper process.*
Host	The name of the host.*
Ready	When checked, the keeper is operational.*
Started	When checked, the keeper has been started and is up and running.*
Copyset Epoch Updated	Any value greater than 0 denotes that a disaster recovery failover to another data grid has occurred.
Time Stamp	The date and time the row data was last updated.

TIBCO ActiveSpaces StateKeepers Heatmap - HTML

Clicking **All Keepers Heatmap** in the left/navigation menu opens the **TIBCO ActiveSpaces StateKeeper Heatmap**, which provides an easy-to-view interface that allows you to quickly identify the current status of each of your keepers for each available metric. You can view the keepers in the heatmap based on the following metrics: current alert severity, alert count, CPU usage, memory usage, rate of messages received, and rate of messages sent. By default, this display shows the heatmap based on the **Alert Severity** metric.

You can mouse over a rectangle to see additional metrics for a keeper. Clicking one of the rectangles in the heatmap opens the TIBCO ActiveSpaces Keeper Summary - HTML display, which allows you to see additional details for the selected keeper.



Filter By:

Realm	Select the realm for which you want to see data.

Fields and Data:

Count	The total number of keepers found for the selected realm.
Log Scale	Select this check box to use a logarithmic scale, rather than a linear scale, to map

from the selected metric value for a cell to the color for the cell. **Log Scale** provides another way to distribute and differentiate values that you might not be able to see on a linear scale due to the dominant nature of large values in a linear scale.

Auto Scale

When checked, the values of the selected metric are auto-scaled to its highest defined value. When unchecked, the values of the selected metric display based on the threshold defined for the alert associated with the selected metric. Selecting Auto helps to visualize the range of the values currently present for the selected metric instead of the threshold of the alert that has been associated with the metric. All metrics that have not been associated in the heatmap defaults with alerts use a monochromatic color gradient bar (whites and greens). All metrics that have been associated in the heatmap defaults with alerts use a multi-chromatic color gradient bar (reds, yellows, white, and greens).

Metric

Select the metric driving the heatmap display. The default is **Alert Severity**. Each **Metric** has a color gradient bar that maps values to colors. The heatmap is organized by keepers, where each rectangle represents a keeper. Mouse-over any rectangle to display the current values of the metrics for the keeper. Click on a rectangle to drill-down to the associated TIBCO ActiveSpaces Keeper Summary - HTML display for a detailed view of metrics for that particular keeper.

Alert Severity

The current alert severity. Values range from **0** - **2**, as indicated in the color gradient bar, where **2** is the highest Alert Severity:

- Red indicates that one or more metrics exceeded their ALARM LEVEL threshold.
- Yellow indicates that one or more metrics exceeded their WARNING LEVEL threshold.
- Green indicates that no metrics have exceeded their alert thresholds.

Alert Count

The total number of alarm and warning alerts in a given item (index) associated with the rectangle. The color gradient bar shows the range of the value/color mapping. The numerical values in the gradient bar range from 0 to the maximum count of alerts in the heatmap. The middle value in the gradient bar indicates the middle value of the range.

CPU Usage

The CPU usage rate for the keeper. The color gradient bar, populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgKeeperCpuUsageHigh**. The middle value in the gradient bar indicates the middle value of the range.

Memory

The usage memory for the keeper. The color gradient bar , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgKeeperMemoryUseHigh**. The middle value in the gradient bar indicates the middle value of the range.

Msgs Rcvd/sec

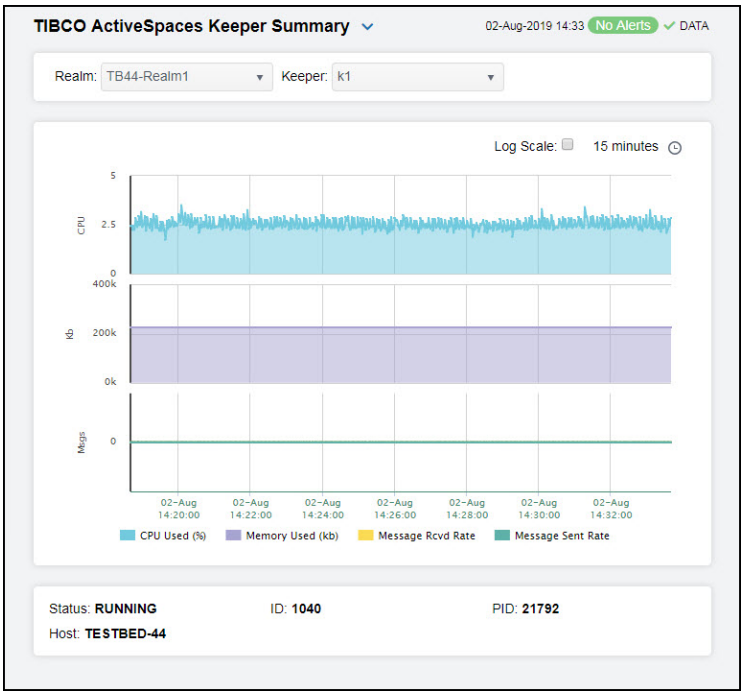
The rate of messages received.The color gradient bar , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgKeeperMsgsRcvdRateHigh**. The middle value in the gradient bar indicates the middle value of the range.

Msgs Sent/sec

The rate of messages received.The color gradient bar , populated by the current heatmap, shows the value/color mapping. The numerical values in the gradient bar range from 0 to the defined alert threshold of **TdgKeeperMsgsSentRateLow**. The middle value in the gradient bar indicates the middle value of the range.

TIBCO ActiveSpaces Keeper Summary - HTML

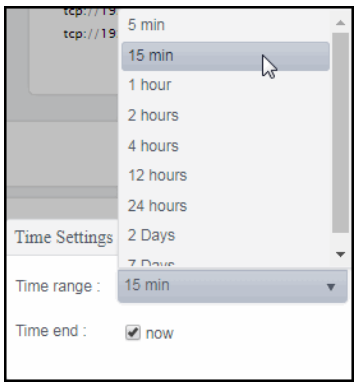
Clicking **Single Keeper Summary** in the left/navigation menu opens the **TIBCO ActiveSpaces Keeper Summary**, which provides a view of the current and historical metrics for a single keeper. The trend graph in the display traces the current and historical CPU usage percentage, process memory usage (in KB), rate of received messages, and the rate of sent messages for the keeper.



Filter By:

The display might include these filtering options:

Realm	Select the realm (containing the keeper) for which you want to show data in the display.
Keeper	Select the keeper for which you want to show data in the display.
Trends	Traces the following: **CPU Used (%)**-- traces the CPU usage percentage. **Memory Used (kb)** -- traces the memory usage, in kilobytes. **Message Rcvd Rate**-- traces the rate of messages received, per second. **Message Sent Rate**-- traces the rate of messages sent, per second.
Log Scale	Select to enable a logarithmic scale. Use **Log Scale** to see usage correlations for data with a wide range of values. For example, if a minority of your data is on a scale of tens, and a majority of your data is on a scale of thousands, the minority of your data is typically not visible in non-log scale graphs. **Log Scale** makes data on both scales visible by applying logarithmic values rather than actual values to the data.
Time Settings	Select a time range from the drop down menu varying from **5 Minutes** to **Last 7 Days**. By default, the time range end point is the current time.



To change the time range, deselect the **now** toggle, which displays some additional date fields. You can click the left and right arrow buttons to decrease the end time by one time period (the time selected in the **Time range** drop down) per click, or you can choose the date and time from the associated calendar and clock icons. You can also enter the date and time in the text field using the following format: **MMM dd, YYYY HH:MM:ss**. For example, Aug 21, 2018 12:24 PM. Click the **now** toggle to reset the time range end point to the current time.

Fields and Data:

Grid Name	The name of the grid.*
PID	The process ID of the StateKeeper.*
Status	The current status of the keeper.*
Host	The name of the host.
ID	The ID of the keeper.*

Drilldowns

The displays described in this section are only accessible from other displays. These displays are used for managing alerts at the component level.

This View includes the following displays:

- Alerts Table by Component - HTML: Track alerts associated with CIs shown in a display.
- Alert Detail for Component - HTML: Investigate an alert instance and its history.
- Alert Configuration for Component - HTML: Refine alert threshold settings.

Alerts Table by Component - HTML

As an alternative to the **Alerts Table**, use the **Alerts Table by Component** to track and manage all alerts that are specifically associated with the CIs shown in a display.

You access the **Alerts Table by Component** by clicking 17 Alerts (the alert status icon) in the title bar of other displays. The display in which you click 17 Alerts is the source display.

Package provides the technology label associated with the alerts shown. For example, **Jvm**, **Tomcat** and **Host** are the technology labels for Java Virtual Machines, Tomcat applications and servers (respectively). These labels are also correlated with the RTView solution package names (for example, the Solution Package for Host Agent). **Category** lists all alert categories related to the source display.

Use the **ACK** and **Cleared** drop-downs to filter the table by **All**, **True** or **False**.

See the **Alert Level** column icon, where:

The alert reached its ALARM LEVEL threshold in the table row.

The alert reached its WARNING LEVEL threshold in the table row.

To investigate, click:

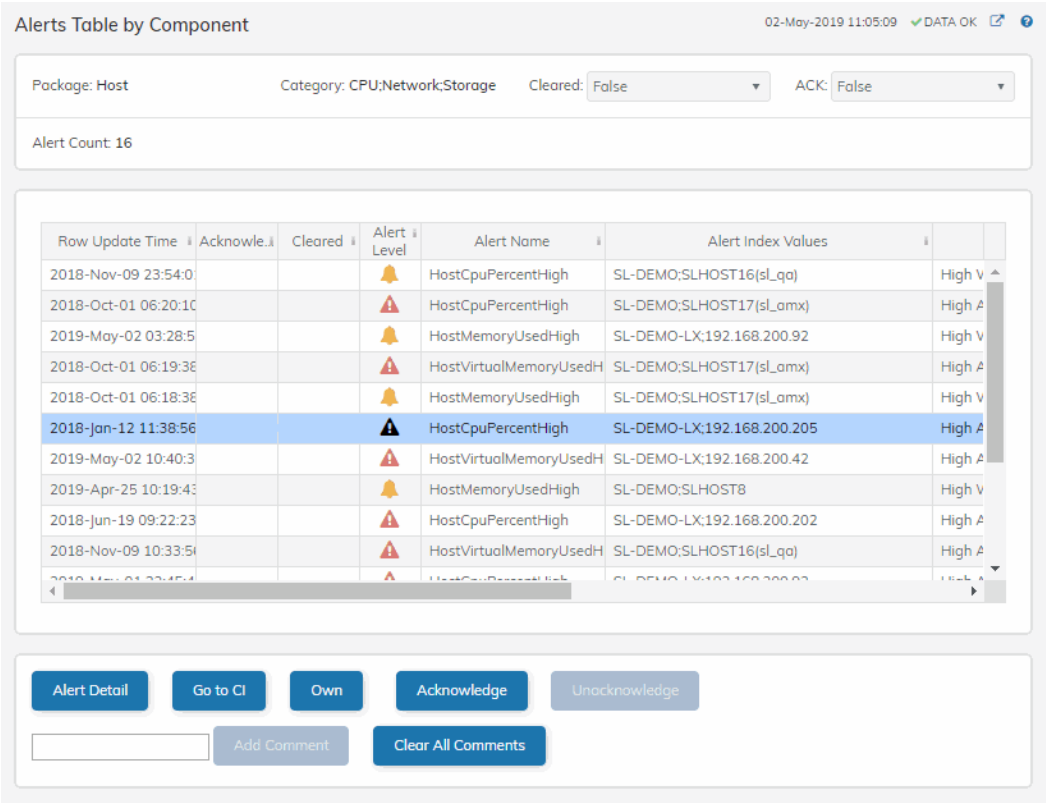
Alert Detail to open the **Alert Detail for Component** where you can see the current and historical conditions that precipitated the alert being executed.

Go to CI to open the summary display for the CI associated with the alert where you can investigate utilization metrics for the CI leading up to the alert being executed.

You can search, filter, sort and choose columns to include by clicking a column header icon (to the right of each column label) and selecting **Filter**, **Sort Ascending**, **Sort Descending** or **Columns**. Right-click on a table cell to **Export to Excel**. Use **Ctrl +** click or **Shift** + click to select multiple alerts.

With one or more alerts selected, click Own to set the alert(s) owner field, Acknowledge to acknowledge the alert(s), Unacknowledge to clear the acknowledgement on previously acknowledged alert(s), Add Comment to add a comment to the alert(s).

You must be logged in as rtvalertmgr or rtvadmin to perform the **Own**, **Ack**, **Unack**, or **Comment** actions. Otherwise, you get an error dialog.



Alert Detail for Component - HTML

Use the **Alert Detail for Component** display to investigate current and historical activity of a specific alert instance as it applies to the associated CI, and also compare against **Metric History** trends of the associated CI. A trend graph for the CI associated with the alert instance. You can hover over the trend graph to see the values at a particular time. You can specify the time range for the trend graph and view data based on a log scale, which enables visualization on a logarithmic scale and should be used when the range in your data is very broad.

Access the **Alert Detail for Component** display by clicking Details in the **Alerts Table** or Alert Detail in the **Alerts Table by Component** display.

The **Alert History** table at the bottom of the display contains a row of data for each time the alert instance was updated. See the alert **ID**, **Row Update Time**, **Cleared** status and **Reason**, **Owner** and the **Alert Level** column icon, where:


The alert reached its ALARM LEVEL threshold in the table row.

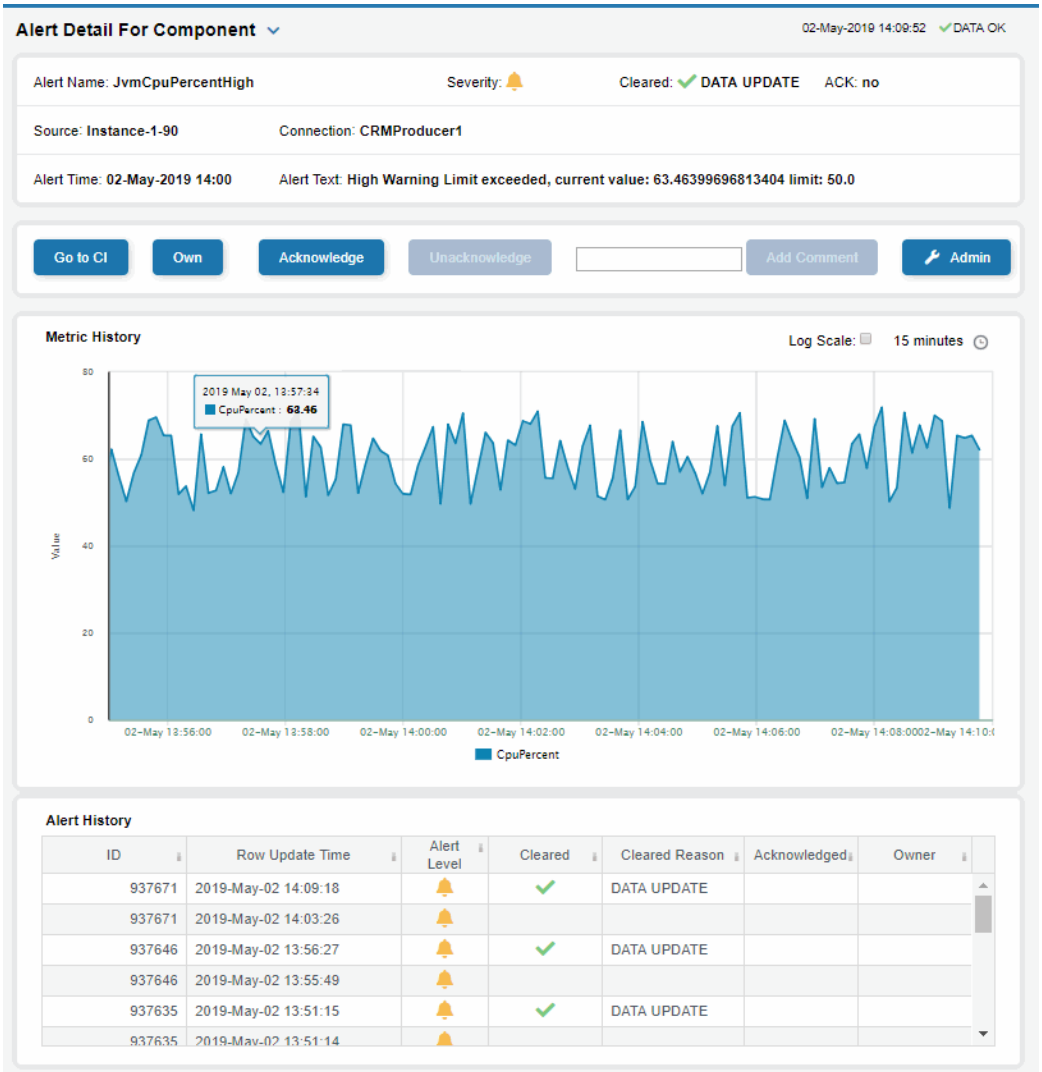

The alert reached its WARNING LEVEL threshold in the table row.

You can search, filter, sort and choose columns to include by clicking a column header icon (to the right of each column label) and selecting **Filter**, **Sort Ascending**, **Sort Descending** or **Columns**. Right-click on a table cell to **Export to Excel**. Use **Ctrl +** click or **Shift** + click to select multiple alerts.

To investigate, click:

Go to CI to see utilization conditions for the CI associated with the alert in a summary display.

Admin to open the **Alert Configuration for Component** display where you can see, modify and refine alert threshold settings for that particular alert. A trend graph traces the relevant alert metric for the CI so you can adjust thresholds in real-time.



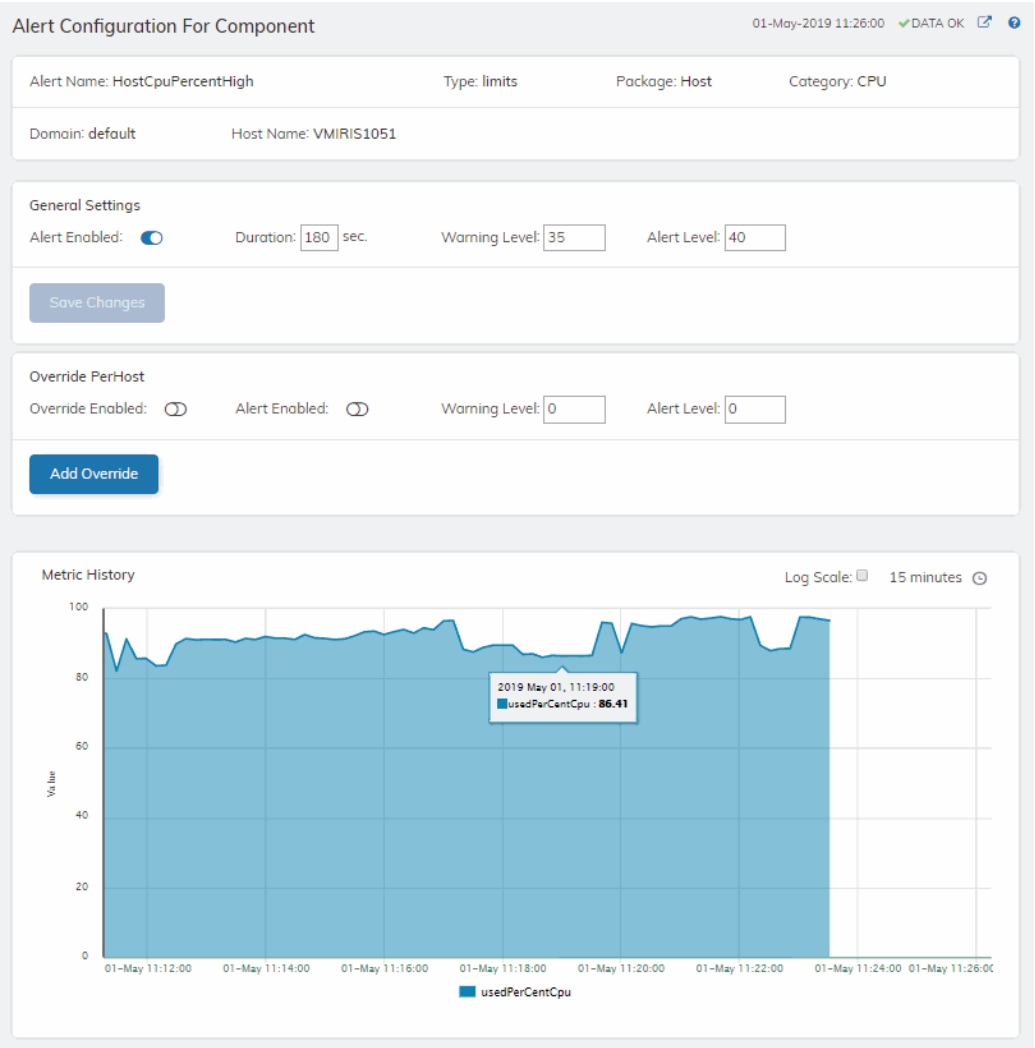
Alert Configuration for Component - HTML

Use the **Alert Configuration for Component** display to see, modify and refine alert threshold settings for a particular alert. A trend graph traces the history of the relevant metric for this alert so you can adjust thresholds in real-time. You can also modify alert thresholds, add an override alert and toggle ON or OFF both global and override alerts.

Access the **Alert Configuration for Component** display by clicking Admin in the **Alert Detail for Component** display.

The bottom half of the display provides a **Metric History** trend graph which traces the performance metric pertaining to the alert. You can hover over the trend graph to see the values at a particular time. You can specify the time range for the trend graph and view data based on a log scale, which enables visualization on a logarithmic scale and should be used when the range in your data is very broad.

You must be logged in as rtvalertmgr or rtvadmin to modify alerts.



Alerts

This section describes displays in the Alerts tab.

Alerts Table

Use this display to track and manage all alerts that have occurred in the system, where:

One or more alerts exceeded their ALARM LEVEL threshold in the table row

One or more alerts exceeded their WARNING LEVEL threshold in the table row

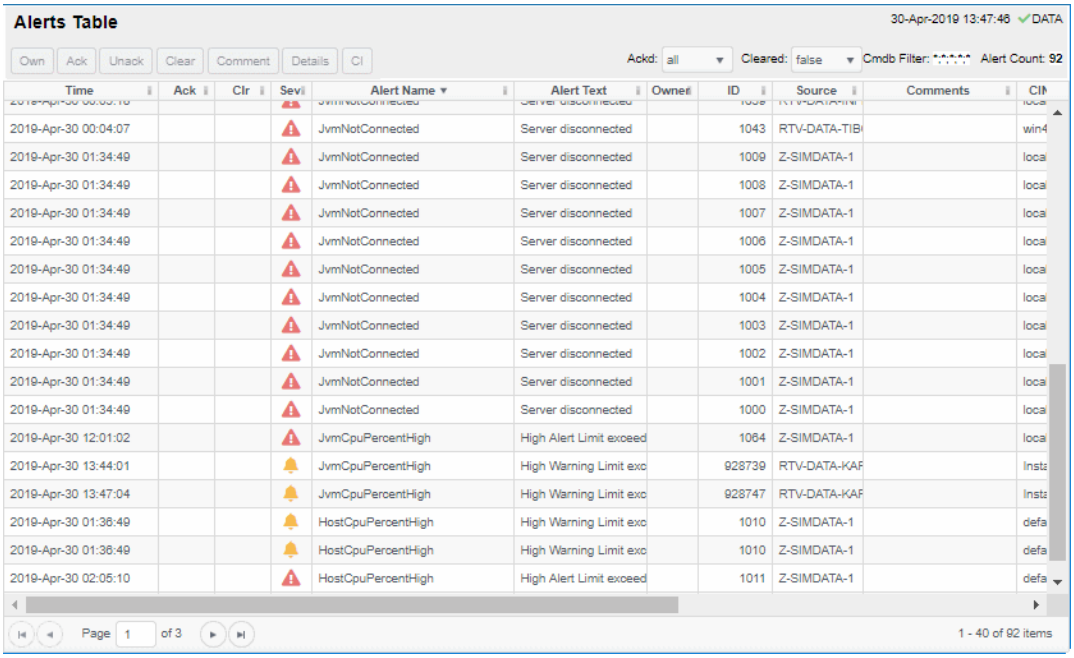
You can search, filter, sort and choose columns to include by clicking a column header icon (located to the right of each column label) and selecting **Filter**, **Sort Ascending**, **Sort Descending** or **Columns**. Use the **Ack'd** and **Cleared** drop-downs to filter the table by those columns. Right-click on a table cell to **Export to Excel** or **Copy Cell Value**. Use **Ctrl +** click or **Shift** + arrow to select multiple alerts. To investigate, select one alert and click:

Details to open the **Component Alert Detail** display to get details about that particular alert instance as it specifically applies to the associated CI.

CI to see utilization conditions for the CI associated with the alert during the seconds (minutes, hours or days) leading up to the alert being executed in a summary display.

With one or more alerts selected, you can click **Own** to set the alert(s) owner field, **Ack** to acknowledge the alert(s), **Unack** to clear the acknowledgement on previously acknowledged alert(s) and **Comment** to add a comment to the alert(s).

You must be logged in as rtvalertmgr or rtvadmin to perform the **Own**, **Ack**, **Unack**, or **Comment** actions. Otherwise, you get an error dialog.



Admin

This section describes displays in the Admin tab.

These displays enable you to set alert thresholds, observe how alerts are managed, and view internal data gathered and stored by RTView (used for troubleshooting with SL Technical Support). Displays in this View are:

- Alert Administration: Displays active alerts and provides interface to modify, enable and manage alerts.
- Admin: Set and modify alert overrides. Access this display from the Alert Administration display.
- Cache Table: View cached data that RTView is capturing and maintaining, and use this data use this for debugging with SL Technical Support.

Alert Administration

The **Alert Administration** display allows administrators to enable/disable alerts and manage alert thresholds. The table describes the global settings for all alerts on the system.

You can set the **Delay** time (the number of seconds that must pass before an alert is triggered, where **0** sets it to immediately execute).

You can set the **Warning Level** which executes a single warning alert when the number of seconds specified here is exceeded. To set the warning to occur sooner, reduce the **Warning Level** value. To set the warning to occur later, increase the **Warning Level** value.

You can set the **Alarm Level** which executes a single alarm alert when the number of seconds specified here is exceeded. To set the alarm to occur sooner, reduce the **Alarm Level** value. To set the alarm to occur later, increase the **Alarm Level** value.

Note: For low value-based alerts (an alert that executes based on a value going below a certain threshold), to set the alarm to occur sooner you increase the **Alarm Level** value. To set the alarm to occur later, reduce the **Alarm Level** value.

You can apply alert thresholds globally or as an *override*. Setting override alerts allows you to set thresholds for a subset of your resources, or for a single resource (for example, a single server). Override alerts are useful if the majority of your resources require the same threshold setting, but there are a few resources that require a different threshold setting. For example, you might not usually be concerned with execution time at a process level, but perhaps certain processes are critical. In this case, you can apply alert thresholds to each process individually. See below for instructions.

You can filter, sort and choose columns to include by clicking a column header icon (located to the right of each column label) and selecting Filter, Sort Ascending, Sort Descending or Columns. Use the Ack'd and Cleared drop-downs to filter the table by those columns. Right-click on a table cell to Export to Excel.

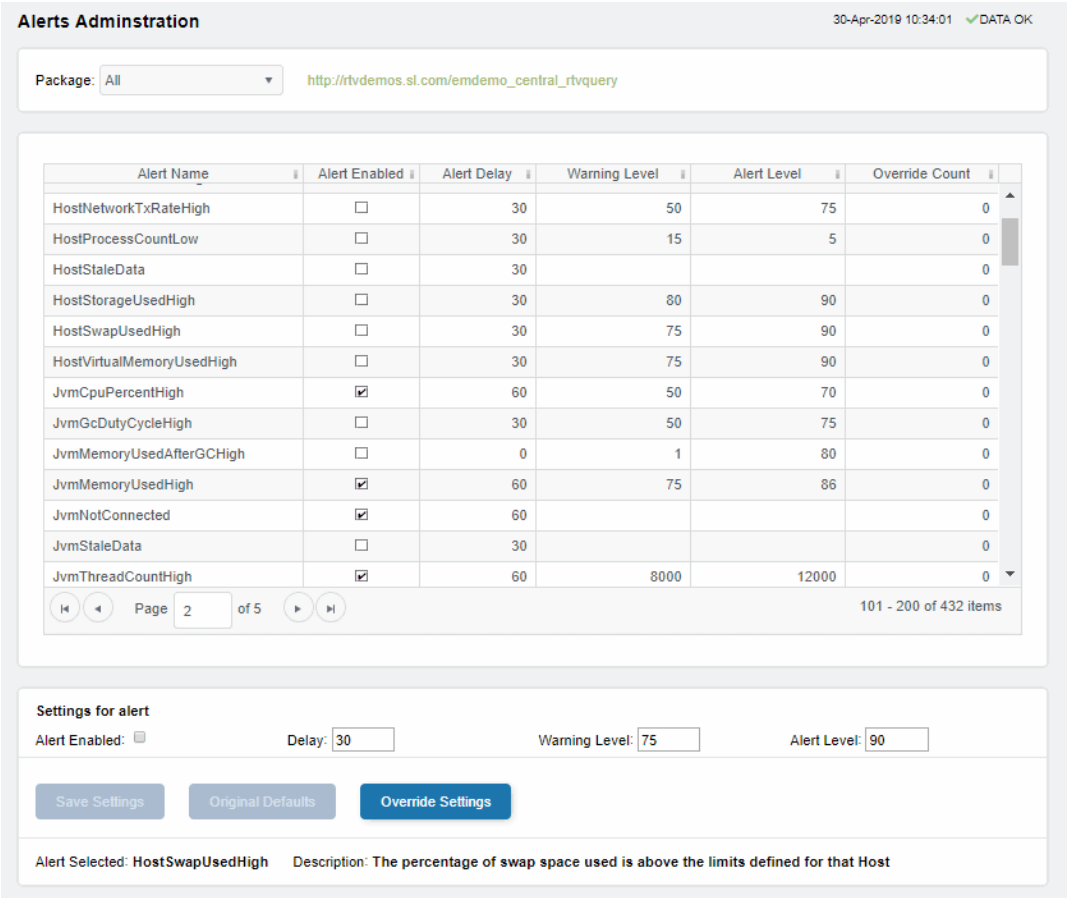
To set thresholds and enable a global alert:

Select an alert and, under **Settings for alert** (in the lower portion of the screen), modify settings for the alert **Delay**, **Warning Level** and/or **Alarm Level** and **Save Settings**. With that alert selected, check the **Alert Enabled** box under **Settings for alert** (in the lower portion of the screen) and **Save Settings**. The **Alert Enabled** box (next to the selected alert) is now checked.

You can also override the alert duration time per alert index instead of to all indexes. To override the duration for an alert index, select the alert in the **Alert Administration** display, click **Override** and edit the **Alert Delay**. For alert indexes that were overridden in a previous release (before duration override was supported) the override duration is set to **-1**, indicating that this is set to use the top level alert duration.

To set thresholds and enable an override alert:

To set an override alert, select an alert and click **Override Settings** to open the **Alert Overrides Admin** display.



For additional details, see Admin.

Alert Name	The name of the alert.
Alert Enabled	When checked, the alert is enabled globally.
Alert Delay	The amount of time (in seconds) that the value must be above the specified Warning Level or Alarm Level threshold before an alert is executed. **0** is for immediate execution.
Warning Level	The global warning threshold for the selected alert. When the specified value is exceeded a warning is executed.
Alert Level	The global alarm threshold for the selected alert. When the specified

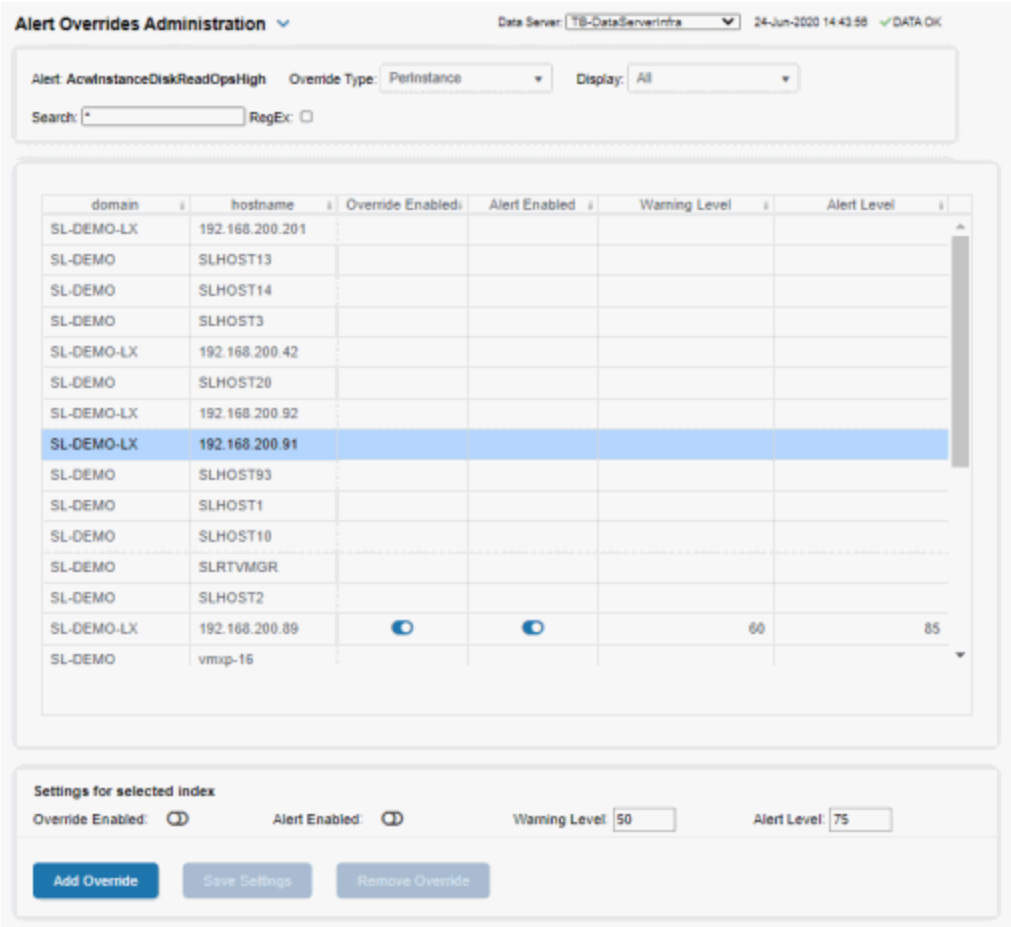
	value is exceeded an alarm is executed.
Override Count	The number of times thresholds for this alert have been defined individually in the **Tabular Alert Administration** display. A value of: **-0** indicates that no overrides are applied to the alert. **-1** indicates that the alert does not support overrides.

Settings for alert
Select an alert in the table to use the following options:

Alert Enabled	Check / uncheck this box to enable or disable the selected alert globally.
Delay	Enter the amount of time (in seconds) that the value must be above the specified Warning Level or Alarm Level threshold before the selected alert is executed. **0** is for immediate execution.
Warning Level	Enter the global warning threshold for the selected alert. When the specified value is exceeded a warning is executed. To set the warning to occur sooner, reduce the Warning Level value. To set the warning to occur later, increase the Warning Level value.
Alert Level	Enter the global alarm threshold for the selected alert. When the specified value is exceeded an alarm is executed. To set the alarm to occur sooner, reduce the Alarm Level value. To set the warning to occur later, increase the Alarm Level value. NOTE: For low value-based alerts (such as **EmsQueuesConsumerCountLow**), to set the alarm to occur sooner, increase the Alarm Level value. To set the alarm to occur later, reduce the Alarm Level value.
Save Settings	Click to apply alert settings for the selected alert.
Original Defaults	Click to revert to original alert settings for the selected alert.
Override Settings	Click to set an alert override in the **Alert Overrides Admin** display on the selected alert.

Alert Overrides Administration

Administrators use this display to override the alert settings defined in the **Alert Administration** display. To access this display, select an alert in the **Alert Administration** display and choose **Override Settings**.



The table lists all the possible overrides that can be defined for the alert you selected from the **Alert Administration** display. Each row in the table represents a different resource or group of resources that can be overridden. When the four last columns are blank, that means the resource has not been overridden, and the default settings for the alert apply. Otherwise, columns describe whether the alert is enabled, if the override itself is enabled, the overridden alert thresholds and the overridden duration for each row.

Use the **Override Type** drop-down menu to switch the list to a specific type of override (the options for this menu vary according to the alert type), and use the **Display** drop-down menu to list **All** resources, **Overridden** resources or **Free** resources.

You can also enter a pattern or regular expression in the **Search** string to limit the list.

The **RegEx** checkbox indicates whether the text you entered is treated as a search pattern or as a regular expression. Multiple rows can be selected to create/edit/remove many overrides simultaneously.

You can filter, sort and choose columns to include by clicking a column header icon (located to the right of each column label) and selecting **Filter**, **SortAscending**, **Sort Descending** or **Columns**. Use the **Display** drop-down to filter the table to show **All** resources, only resources with the **Overridden** alert applied or **Free** resources (to show only resources without the alert override applied). Right-click on a table cell to **Export to Excel** or **Copy Cell Value**.

To set overrides:

Select an **Override Type** from the drop-down menu (depending on the alert, there might be only one type) and then select one or more rows from the table. Under **Settings for selected index** (in the lower portion of the screen), modify settings for the **Override Enabled**, **Alert Enabled**, **Alert Delay**,

Warning Level and/or **Alarm Level**, then click **Add Override**. The table updates with your new settings.

To alter overrides:

To alter existing overrides with new settings, select them from the table, set all properties under **Settings for selected index** as desired, then click **Save Settings**. To clear existing overrides, select one or more rows, then click **Remove Override**.

> **Note:** You can override alert and warning levels without overriding duration by setting it to **-1**.

For alert indexes that were overridden in a previous release (before duration override was supported) the override duration is set to **-1**, indicating that this is set to use the top level alert duration.

Cache Table

View the raw data that RTView is capturing and maintaining to investigate utilization and capacity metrics, as well as connection details, for caches on a data server.

Select a **Data Server** from the drop-down menu. The upper table contains a row of data for each cache on the selected data server. You can see the current number of **Rows** and **Columns** in each table and the amount of **Memory** used. You can also find out the cache **Table** type of which there are five:

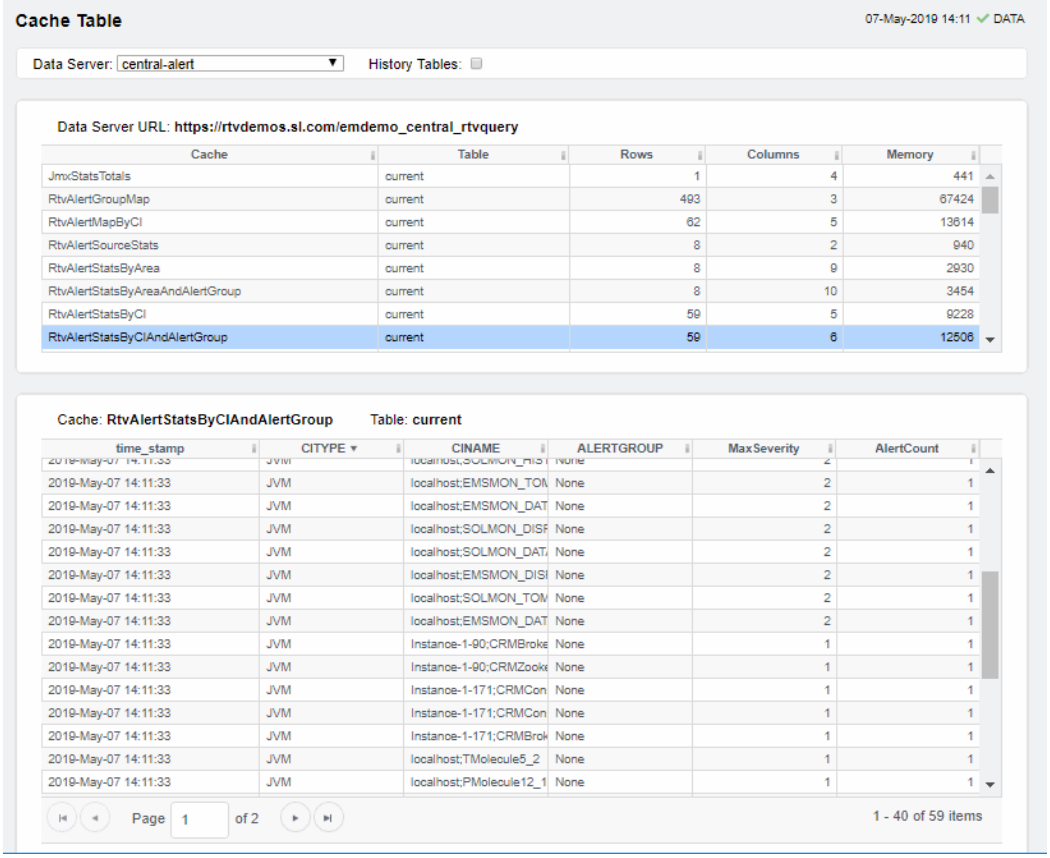
- **current** tables show the most recently received values for each index.
- **current_condensed** tables are current tables with primary compaction configured.
- **history** tables show the historical values for each index.
- **history_condensed** tables are history tables with primary compaction configured.
- **history_combo** tables are history tables with primary compaction configured, and which is also configured to store rows of recent raw data followed by rows of older condensed data.

Select a cache to see connection utilization details for that cache in the lower table. The lower table shows the contents of the selected cache table. Available columns vary by cache. For example, a JVM cache table might provide **BootClassPath** and **InputArgument** columns, and a Tomcat cache might provide **RateAccess** and **cacheMaxSize** columns.

You can search, filter, sort and choose columns to include by clicking a column header icon (to the right of each column label) and selecting **Filter**, **Sort Ascending**, **Sort Descending** or **Columns**. Or just click a column header to sort.

Right-click on a table cell to **Export to Excel** or **Copy Cell Value**. Use **Ctrl +** click or **Shift** + click to select multiple alerts. Use **History Tables** to include / exclude history tables in the table. Right-click on a table cell to **Export to Excel** or **Copy Cell Value**.

This low-level option can be useful to identify the source of the problem when the displays are not showing the expected data. Use this data for debugging and troubleshooting with Technical Support.



Alerts Administration

The **Alert Administration** display allows administrators to enable/disable alerts and manage alert thresholds. The table describes the global settings for all alerts on the system.

You can set the **Delay** time (the number of seconds that must pass before an alert is triggered, where **0** sets it to immediately execute).

You can set the **Warning Level** which executes a single warning alert when the number of seconds specified here is exceeded. To set the warning to occur sooner, reduce the **Warning Level** value. To set the warning to occur later, increase the **Warning Level** value.

You can set the **Alarm Level** which executes a single alarm alert when the number of seconds specified here is exceeded. To set the alarm to occur sooner, reduce the **Alarm Level** value. To set the alarm to occur later, increase the **Alarm Level** value.

> **Note:** For low value-based alerts (an alert that executes based on a value going below a certain threshold), to set the alarm to occur sooner you increase the **Alarm Level** value. To set the alarm to occur later, reduce the **Alarm Level** value.

You can apply alert thresholds globally or as an *override*. Setting override alerts allows you to set thresholds for a subset of your resources, or for a single resource (for example, a single

server). Override alerts are useful if the majority of your resources require the same threshold setting, but there are a few resources that require a different threshold setting. For example, you might not usually be concerned with execution time at a process level, but perhaps certain processes are critical. In this case, you can apply alert thresholds to each process individually. See below for instructions.

You can filter, sort and choose columns to include by clicking a column header icon (located to the right of each column label) and selecting **Filter**, **Sort Ascending**, **Sort Descending** or **Columns**. Use the **Ack'd** and **Cleared** drop-downs to filter the table by those columns. Right-click on a table cell to **Export to Excel**.

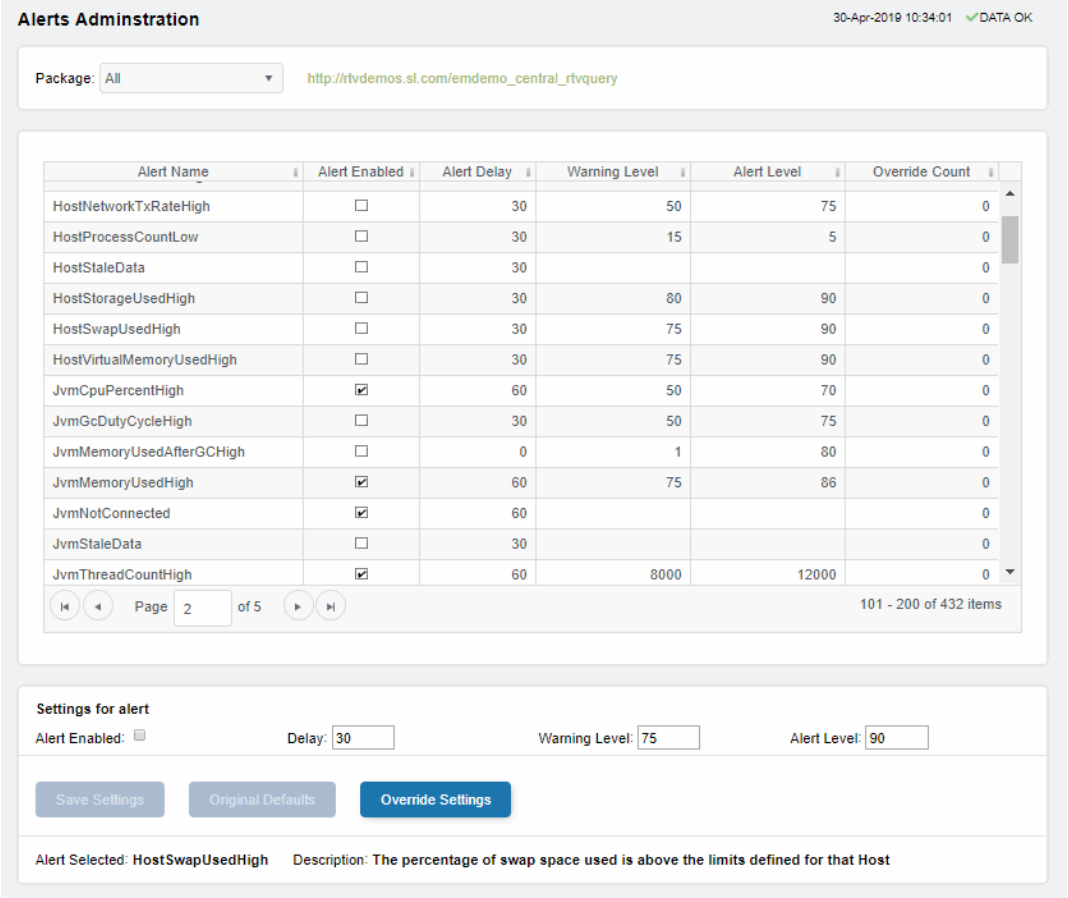
To set thresholds and enable a global alert:

Select an alert and, under **Settings for alert** (in the lower portion of the screen), modify settings for the alert **Delay**, **Warning Level** and/or **Alarm Level** and **Save Settings**. With that alert selected, check the **Alert Enabled** box under **Settings for alert** (in the lower portion of the screen) and **Save Settings**. The **Alert Enabled** box (next to the selected alert) is now checked.

You can also override the alert duration time per alert index instead of to all indexes. To override the duration for an alert index, select the alert in the **Alert Administration** display, click **Override** and edit the **Alert Delay**. For alert indexes that were overridden in a previous release (before duration override was supported) the override duration is set to **-1**, indicating that this is set to use the top level alert duration.

To set thresholds and enable an override alert:

To set an override alert, select an alert and click **Override Settings** to open the **Alert Overrides Admin** display.



Alert Name	The name of the alert.
Alert Enabled	When checked, the alert is enabled globally.
Alert Delay	The amount of time (in seconds) that the value must be above the specified Warning Level or Alarm Level threshold before an alert is executed. **0** is for immediate execution.
Warning Level	The global warning threshold for the selected alert. When the specified value is exceeded a warning is executed.
Alert Level	The global alarm threshold for the selected alert. When the specified value is exceeded an alarm is executed.
Override Count	The number of times thresholds for this alert have been defined individually in the **Tabular Alert Administration** display. A value of: **-0** indicates that no overrides are applied to the alert. **-1** indicates that the alert does not support overrides.

Settings for alert

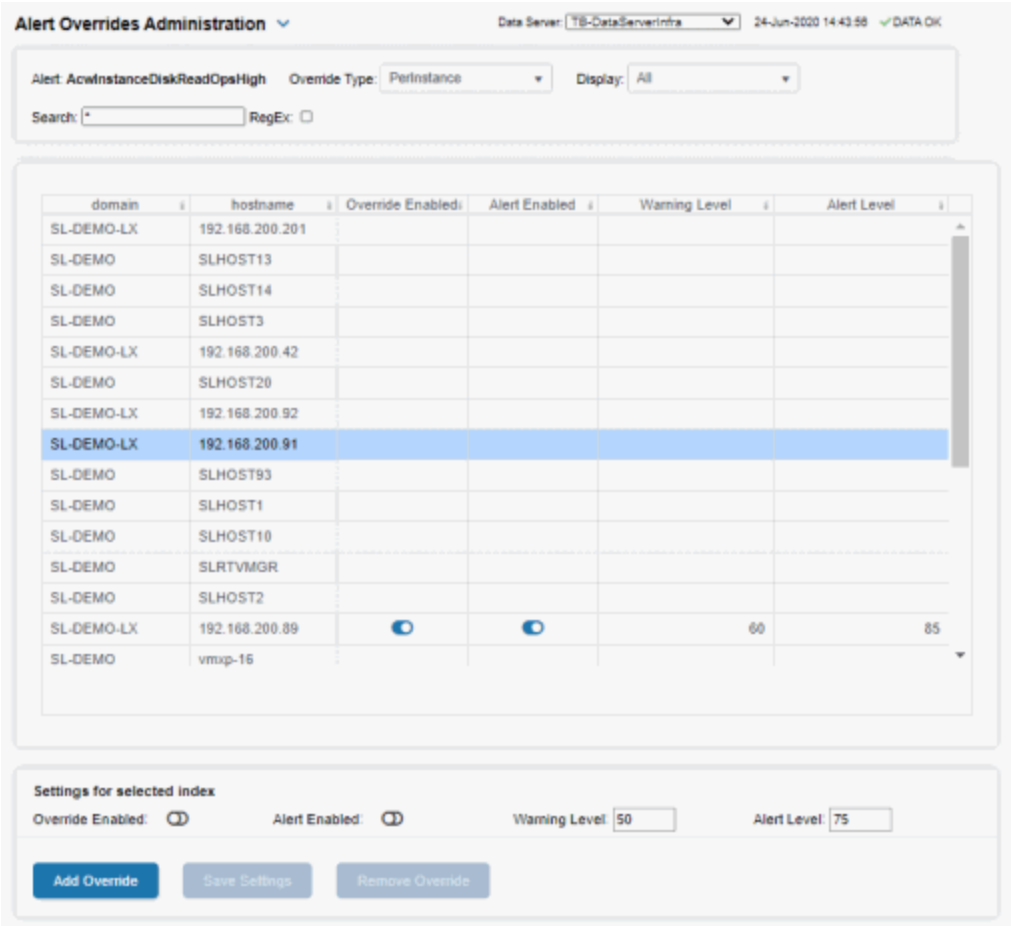
Select an alert in the table to use the following options:

Alert Enabled	Check / uncheck this box to enable or disable the selected alert globally.
Delay	Enter the amount of time (in seconds) that the value must be above the specified Warning Level or Alarm Level threshold before the selected alert is executed. **0** is for immediate execution.
Warning Level	Enter the global warning threshold for the selected alert. When the specified value is exceeded a warning is executed. To set the warning to occur sooner, reduce the Warning Level value. To set the warning to occur later, increase the Warning Level value.
Alert Level	Enter the global alarm threshold for the selected alert. When the specified value is exceeded an alarm is executed. To set the alarm to occur sooner, reduce the Alarm Level value. To set the warning to occur later, increase the Alarm Level value.

	NOTE: For low value-based alerts (such as **EmsQueuesConsumerCountLow**), to set the alarm to occur sooner, increase the Alarm Level value. To set the alarm to occur later, reduce the Alarm Level value.
Save Settings	Click to apply alert settings for the selected alert.
Original Defaults	Click to revert to original alert settings for the selected alert.
Override Settings	Click to set an alert override in the **Alert Overrides Admin** display on the selected alert.

Alert Overrides Administration

Administrators use this display to override the alert settings defined in the **Alert Administration** display. To access this display, select an alert in the **Alert Administration** display and choose **Override Settings**.



The table lists all the possible overrides that can be defined for the alert you selected from the **Alert Administration** display. Each row in the table represents a different resource or group of resources that can be overridden. When the four last columns are blank, that means the resource has not been overridden, and the default settings for the alert apply. Otherwise, columns describe whether the alert is enabled, if the override itself is enabled, the overridden alert thresholds and the overridden duration for each row.

Use the **Override Type** drop-down menu to switch the list to a specific type of override (the options for this menu vary according to the alert type), and use the **Display** drop-down menu to list **All** resources, **Overridden** resources or **Free** resources.

You can also enter a pattern or regular expression in the **Search** string to limit the list.

The **RegEx** checkbox indicates whether the text you entered is treated as a search pattern or as a regular expression. Multiple rows can be selected to create/edit/remove many overrides simultaneously.

You can filter, sort and choose columns to include by clicking a column header icon (located to the right of each column label) and selecting **Filter**, **SortAscending**, **Sort Descending** or **Columns**. Use the **Display** drop-down to filter the table to show **All** resources, only resources with the **Overridden** alert applied or **Free** resources (to show only resources without the alert override applied). Right-click on a table cell to **Export to Excel** or **Copy Cell Value**.

To set overrides:

Select an **Override Type** from the drop-down menu (depending on the alert, there might be only one type) and then select one or more rows from the table. Under **Settings for selected index** (in the lower portion of the screen), modify settings for the **Override Enabled**, **Alert Enabled**, **Alert Delay**, **Warning Level** and/or **Alarm Level**, then click **Add Override**. The table updates with your new settings.

To alter overrides:

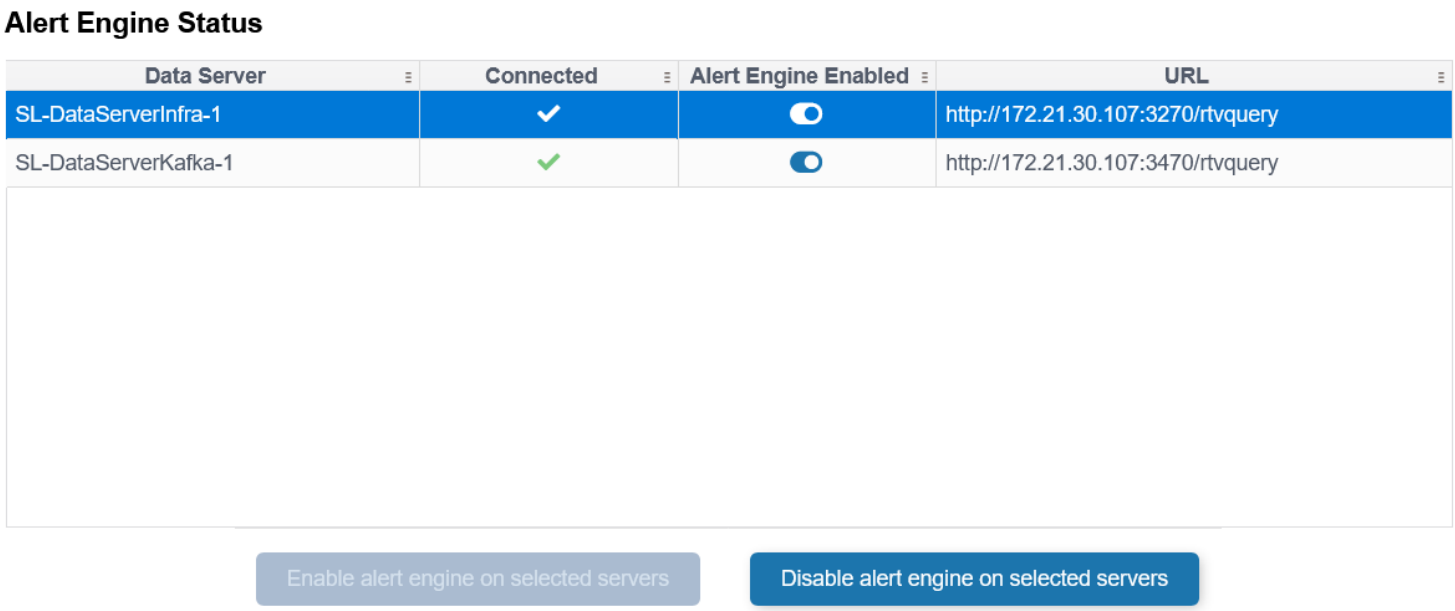
To alter existing overrides with new settings, select them from the table, set all properties under **Settings for selected index** as desired, then click **Save Settings**. To clear existing overrides, select one or more rows, then click **Remove Override**.

> **Note:** You can override alert and warning levels without overriding duration by setting it to **-1**.

For alert indexes that were overridden in a previous release (before duration override was supported) the override duration is set to **-1**, indicating that this is set to use the top level alert duration.

Alert Engine Admin

This display allows you to enable and disable the alert engine(s) of your Data Server(s) on a per-server basis. This display requires administrator privileges.



The **Alert Engine Status** table lists Data Servers that are connected to your deployment. The **Connected** column will display whether or not the Data Server is presently connected. If the Data Server is connected, the **Alert Engine Enabled** column will display whether the alert engine for that Data Server is enabled or not.

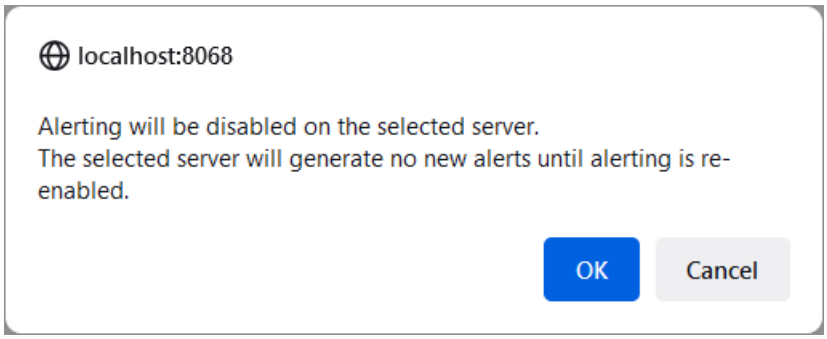
Disabling the alert engine on a Data Server clears all existing alerts on that server. In the Data Server's RtvAlertTable cache, the "Cleared Reason" column will show MANUAL for each alert that was cleared as a result of disabling the alert engine. No new alerts will be generated by that server until its alert engine is re-enabled..

By default the alert engine is enabled for a Data Server. When a Data Server is restarted, its alert engine is always re-enabled.

Disable Alert Engine

Select one or more Data Servers in the **Alert Engine Status** table, then click Disable alert engine on selected servers.

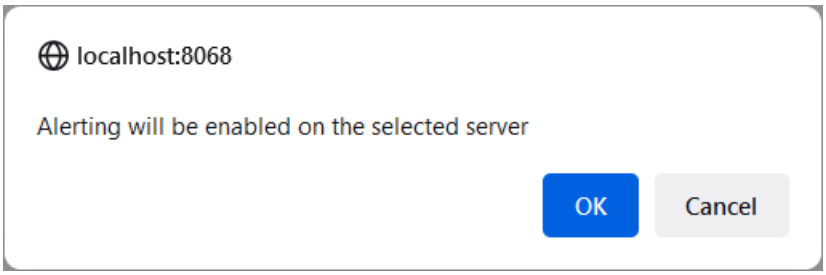
A confirmation dialog box will display. Click **OK** to continue and disable the selected alert engine(s).



Enable Alert Engine

Select one or more Data Servers in the **Alert Engine Status** table, then click Enable alert engine on selected servers .

A confirmation dialog box will display. Click **OK** to continue and enable the selected alert engine(s).



Warning! On Firefox, the enable/disable confirmation dialogs may display a checkbox with the text "Don't allow HOST:8068 to prompt you again". This is from the browser, not from RTView. Do not check that box, as it will prevent the display from working. If it is accidentally selected, clear the Firefox browser cache (**Options -> History -> Clear recent history ...**)

Cache Table

View the raw data that RTView is capturing and maintaining to investigate utilization and capacity metrics, as well as connection details, for caches on a data server.

Select a **Data Server** from the drop-down menu. The upper table contains a row of data for each cache on the selected data server. You can see the current number of **Rows** and **Columns** in each table and the amount of **Memory** used. You can also find out the cache **Table** type of which there are five:

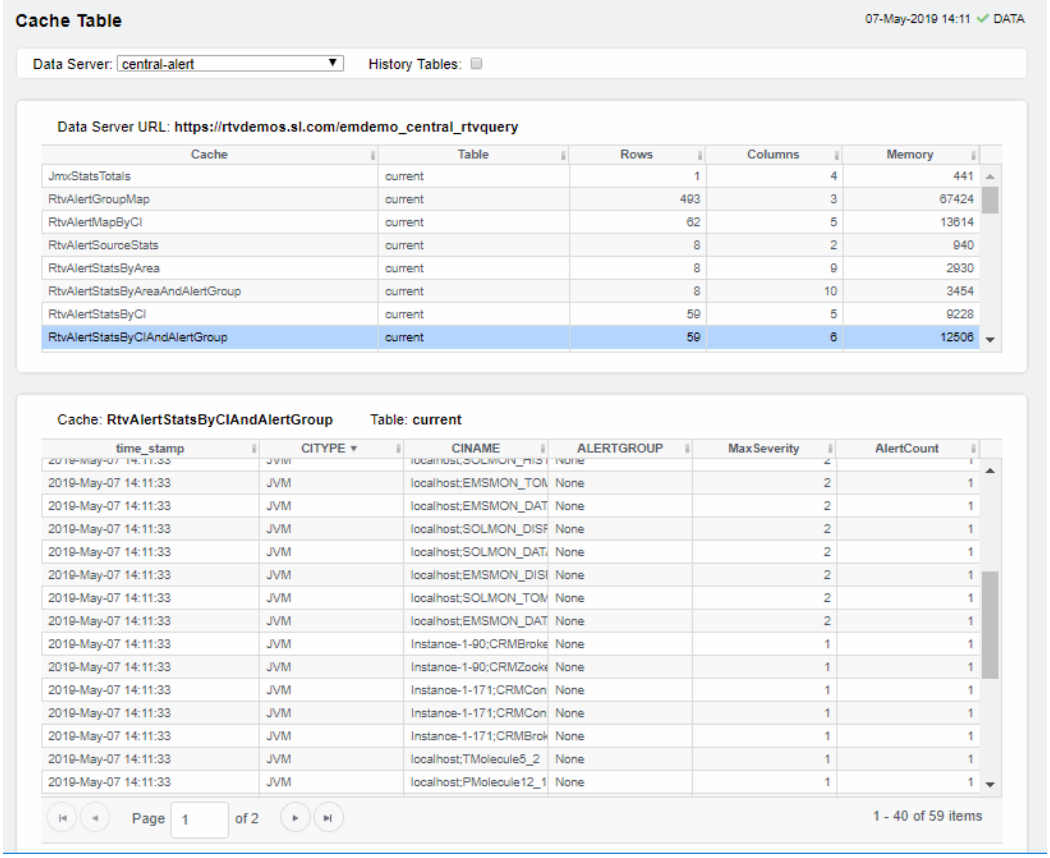
- **current** tables show the most recently received values for each index.
- **current_condensed** tables are current tables with primary compaction configured.
- **history** tables show the historical values for each index.
- **history_condensed** tables are history tables with primary compaction configured.
- **history_combo** tables are history tables with primary compaction configured, and which is also configured to store rows of recent raw data followed by rows of older condensed data.

Select a cache to see connection utilization details for that cache in the lower table. The lower table shows the contents of the selected cache table. Available columns vary by cache. For example, a JVM cache table might provide **BootClassPath** and **InputArgument** columns, and a Tomcat cache might provide **RateAccess** and **cacheMaxSize** columns.

You can search, filter, sort and choose columns to include by clicking a column header icon (to the right of each column label) and selecting **Filter**, **Sort Ascending**, **Sort Descending** or **Columns**. Or just click a column header to sort.

Right-click on a table cell to **Export to Excel** or **Copy Cell Value**. Use **Ctrl +** click or **Shift** + click to select multiple alerts. Use **History Tables** to include / exclude history tables in the table. Right-click on a table cell to **Export to Excel** or **Copy Cell Value**.

This low-level option can be useful to identify the source of the problem when the displays are not showing the expected data. Use this data for debugging and troubleshooting with Technical Support.


Cache Table
07-May-2019 14:11 DATA
Data Server: central-alert
History Tables:
Data Server URL: https://rtvdemos.sl.com/emdemo_central_rtvquery
Cache
Table
Rows
Columns
Memory
JmxStatsTotals current 1 4 441
RtvAlertGroupMap current 493 3 67424
RtvAlertMapByCI current 62 5 13614
RtvAlertSourceStats current 8 2 940
RtvAlertStatsByArea current 8 9 2930
RtvAlertStatsByAreaAndAlertGroup current 8 10 3454
RtvAlertStatsByCI current 59 5 9228
RtvAlertStatsByCIAndAlertGroup current 59 6 12506
Cache: RtvAlertStatsByCIAndAlertGroup
Table: current
time_stamp
CITYPE
CINAME
ALERTGROUP
MaxSeverity
AlertCount
2019-May-07 14:11:33 JVM localhost;EMSMON_TOM None 2 1
2019-May-07 14:11:33 JVM localhost;EMSMON_DAT None 2 1
2019-May-07 14:11:33 JVM localhost;SOLMON_DISF None 2 1
2019-May-07 14:11:33 JVM localhost;SOLMON_DATA None 2 1
2019-May-07 14:11:33 JVM localhost;EMSMON_DISI None 2 1
2019-May-07 14:11:33 JVM localhost;SOLMON_TOM None 2 1
2019-May-07 14:11:33 JVM localhost;EMSMON_DAT None 2 1
2019-May-07 14:11:33 JVM Instance-1-90;CRMBroke None 1 1
2019-May-07 14:11:33 JVM Instance-1-90;CRMZooke None 1 1
2019-May-07 14:11:33 JVM Instance-1-171;CRMCon None 1 1
2019-May-07 14:11:33 JVM Instance-1-171;CRMCon None 1 1
2019-May-07 14:11:33 JVM Instance-1-171;CRMBrok None 1 1
2019-May-07 14:11:33 JVM localhost;TMolecule5_2 None 1 1
2019-May-07 14:11:33 JVM localhost;PMolecule12_1 None 1 1
Page 1 of 2
1 - 40 of 59 items

APPENDIX A Monitor Scripts

This section describes Monitor scripts and the **rtvservers.dat** configuration file. This section includes:

- Scripts
- rtvservers.dat

Scripts

These instructions assume use of a BASH or a BASH-compliant shell. The following scripts are available when used from an initialized command window. The scripts can be executed from a Windows Command Prompt or UNIX terminal window. On Windows, you can type the commands as described in this section. On UNIX systems, you must add **.sh** to each command. For example, **rtvapm_init.sh**. Also on UNIX systems, it is a requirement that the installation directory path not contain spaces.

These instructions assume use of a BASH or a BASH-compliant shell.

Script Name	Description
my_alert_actions.bat/sh	Sample script to define actions for alerts. Location: The project directory. Format: **my_alert_actions** (Append **.sh** on UNIX)
rtvapm_init.bat/sh	Initializes a command window. Location: **rtvapm** This script must be executed in the directory in which it resides. Format: **rtvapm_init** (Append **.sh** on UNIX)
start_rtv.bat/sh	Starts processes in an RTView configuration as specified in the **rtvservers.dat** configuration file. Location: **rtvapm/common/bin** This script must be executed in the project directory (the directory containing the **rtvservers.dat** file). This script requires **rtvapm_init.bat/sh** be executed first. An RTView configuration might include a Data Server or Display Server, an Historian and a Central Server Database. **start_rtv** only attempts to start processes it detects are not running. The action can be applied to all RTView configurations, a single RTView configuration or a single process in an RTView configuration. Before starting an RTView server, this script detects port conflicts

	caused by another server. If the conflict is caused by another RTView server, it returns a message identifying that server by its **rtvapm**. For example:
	`...start_rtv.bat: another dataserver running with JMX port 3268 under C:\rtview\RTViewDataServer\rtvapm`
	
	If the port conflict is caused by a non-RTView process, it returns a message similar to this, for example:
	`...start_rtv.bat: JMX port 3268 in use by PID 1234`
	In both cases the script includes this advice:
	`Warning: server not started, port conflict`

	To avoid port conflicts, run your start script with the **-portprefix:** command line argument to change the first two (**2**) digits of all your server ports. To persist these port changes, change the port prefix in the RTView Configuration Application or use the **-saveportprefix** command line argument. Additional arguments can be included on the command line in which case they are passed to every server specified by the command. Additional arguments can also be included in the **rtvservers.dat** file, in which case they are only applied to the specific server in whose command they are included. **Note:** If you use the **-properties** or **-propfilter** argument with **start_rtv**, you should also use them with **status_rtv** and **stop_rtv**. Those commands use the JMX ports defined for the server, and if any of the properties specified by **-properties** or **-propfilter** arguments change those ports, subsequent commands will be unable to find the server unless also given those properties.

	–console (or **–c**) - Start the processes with a command window (which is useful for testing).
	When used without arguments, this script returns usage information and a list of available configurations. For example, **start_rtv** returns:
	`Usage: start_rtv config or 'all' [server or 'all'] [args...]`
	`Available configs:`
	`default`
	`dataserver`
	`historian`
	`displayserver`
	`database`
	`sender`
	`dataserver`

	all Starts all RTView configurations that are specified in the **rtvservers.dat** file. **all** applies the action to all RTView configurations specified in the **rtvservers.dat** file (and corresponding servers or clients specified in each configuration). **Note:** When multiple configurations are specified in the **rtvservers.dat** file and they have different project settings directory locations, the **all** argument processes all the configurations. However, if the configurations have the same project settings directory locations, the **all** argument processes only the first configuration as the others are considered alternative configurations. Example: **start_rtv all** **(Append .sh on UNIX)**
	[Configuration Name] Starts a single RTView configuration specified in the **rtvservers.dat** file: **start_rtv [Configuration Name]** **(Append .sh on UNIX)** Configuration Name is the RTView configuration name specified in the **rtvservers.dat** file. The action applies to all servers or clients specified in the configuration. Example: **start_rtv web_deployment** **(Append .sh on UNIX)**
	[Server Name] Starts a single process in an RTView configuration specified in the **rtvservers.dat** file: **start_rtv [Configuration Name] [Server Name]** **(Append .sh on UNIX)** Server Name is the name of a server or client member in the configuration. For example, **dataserver**, **displayserver**, **historian** and **database**. The action applies only to that server or client in the configuration. Example: **start_rtv web_deployment dataserver** **(Append .sh on UNIX)**
	Use With Secured JMX Ports This script works with RTView servers whose JMX ports are secured with either a username and password, or with SSL. You provide the scripts with the necessary credential information and the scripts manage authentication with the server. There are two ways that you can provide credential information to the scripts: via command-line arguments and via properties placed in any property file that is used by the server. **Securing with username and password** • To secure with a username and password via command-line, use the arguments as follows: **-jmxuser:...** **-jmxpass:...** • To secure with a username and password in a property file, use the properties as follows: **sl.rtview.jmxremote.username=...** **sl.rtview.jmxremote.password=....**

	Securing with SSL To secure with SSL, you provide the client KeyStore and TrustStore locations and their corresponding passwords. • To secure with SSL via command-line, use the arguments as follows: **-sslkeystore:...** **-sslkeystorepass:...** **-ssltruststore:...** **-ssltruststorepass:...** • To secure with SSL in a property file, use the properties as follows: **sl.rtview.ssl.client.keyStore=...** **sl.rtview.ssl.client.keyStorePassword=...** **sl.rtview.ssl.client.trustStore=...** **sl.rtview.ssl.client.trustStorePassword=....** **Password Encryption** To encrypt the passwords in your properties files, use the command-line tool "encode_string", for example: **encode_string encoder2 password** This will give you an encrypted value for "password" that you can use in your properties.
start_server.bat/sh	Starts the RTView DataServer. Location: **<installation directory>** This script must be executed in the directory in which it resides. You can also execute the script by double-clicking in an Explorer window. Format: **start_server** (Append **.sh** on UNIX)
status_rtv.bat/sh	Returns the status of all RTView configurations that are specified in the **rtvservers.dat** configuration file. Location: **rtvapm/common/bin** This script must be executed in the project directory (the directory containing the **rtvservers.dat** file). This script requires **rtvapm_init.bat/sh** be executed first. This action uses defined JMX ports. An RTView configuration might include a Data Server, a Display Server or Viewer, an Historian and a Central Server Database. **status_rtv** only attempts to start processes it detects are not running. The action can be applied to all RTView configurations, a single RTView configuration or a single process in an RTView configuration. Additional arguments can be included on the command line in which case they are passed to every server specified by the command. Additional arguments can also be included in the **rtvservers.dat** file, in which case they are only applied to the specific server in whose command they are included. Note that if you use **-properties** or **-propfilter** arguments with **start_rtv**, you should also use them with **status_rtv** and **stop_rtv**. Those commands use the JMX ports defined for the server, and if any of the properties specified by **-properties** or **-propfilter** arguments change those ports, subsequent

	commands will be unable to find the server unless also given those properties.
	all Returns the status of all RTView configurations specified in the **rtvservers.dat** file. **Note:** When multiple configurations are specified in the **rtvservers.dat** file and they have different project settings directory locations, the **all** argument processes all the configurations. However, if the configurations have the same project settings directory locations, the **all** argument processes only the first configuration as the others are considered alternative configurations. Example: **status_rtv all** (Append **.sh** on UNIX)
	[Configuration Name] Returns the status of a single RTView configuration specified in the **rtvservers.dat** file: **status_rtv [Configuration Name]** **(Append .sh on UNIX)** Configuration Name is the RTView configuration name specified in the **rtvservers.dat** file. The action applies to all servers or clients specified in the configuration. Example: **status_rtv web_deployment** (Append **.sh** on UNIX)
	[Server Name] Returns the status of a single process in an RTView configuration specified in the **rtvservers.dat** file: **status_rtv [Configuration Name] [Server Name]** **(Append .sh on UNIX)** Server Name is the name of a server or client member in the configuration. For example, **dataserver**, **displayserver**, **historian** and **database**. The action applies only to that server or client in the configuration. Example: **status_rtv web_deployment dataserver** **(Append .sh on UNIX)**
	Use With Secured JMX Ports This script works with RTView servers whose JMX ports are secured with either a username and password, or with SSL. You provide the scripts with the necessary credential information and the scripts manage authentication with the server. There are two ways that you can provide credential information to the scripts: via command-line arguments and via properties placed in any property file that is used by the server. **Securing with username and password** • To secure with a username and password via command-line, use the arguments as follows: **-jmxuser:...** **-jmxpass:...** • To secure with a username and password in a property file, use the properties as follows: **sl.rtview.jmxremote.username=...** **sl.rtview.jmxremote.password=....**

	Securing with SSL To secure with SSL, you provide the client KeyStore and TrustStore locations and their corresponding passwords. • To secure with SSL via command-line, use the arguments as follows: **-sslkeystore:...** **-sslkeystorepass:...** **-ssltruststore:...** **-ssltruststorepass:...** • To secure with SSL in a property file, use the properties as follows: **sl.rtview.ssl.client.keyStore=...** **sl.rtview.ssl.client.keyStorePassword=...** **sl.rtview.ssl.client.trustStore=...** **sl.rtview.ssl.client.trustStorePassword=....** **Password Encryption** To encrypt the passwords in your properties files, use the command-line tool "encode_string", for example: **encode_string encoder2 password** This will give you an encrypted value for "password" that you can use in your properties.
status_server.bat/sh	Returns the status of the RTView DataServer. Location: **<installation directory>** This script must be executed in the project directory (the directory containing the **rtvservers.dat** file). Format: **status_server** (Append **.sh** on UNIX)
stop_rtv.bat/sh	Stops processes in an RTView configuration as specified in the **rtvservers.dat** configuration file. Location: **rtvapm/common/bin** This script must be executed in the project directory (the directory containing the **rtvservers.dat** file). This script requires **rtvapm_init.bat**/**sh** be executed first. This action uses defined JMX ports. An RTView configuration might include a Data Server or a Display Server, an Historian and a Central Server Database. **stop_rtv** only attempts to start processes it detects are not running. The action can be applied to all RTView configurations, a single RTView configuration or a single process in an RTView configuration. Additional arguments can be included on the command line in which case they are passed to every server specified by the command. Additional arguments can also be included in the **rtvservers.dat** file, in which case they are only applied to the specific server in whose command they are included. Note that if you use **-properties** or **-propfilter** arguments with **start_rtv**, you should also use them with **status_rtv** and **stop_rtv**. Those commands use the JMX ports defined for the server, and if any of the properties specified by **-properties** or **-propfilter** arguments change those ports, subsequent commands will be unable to find the server unless also given those properties.

	Location: **project directory** **This script must be executed in the project directory (the directory containing the rtvservers.dat file).**
	all Stops all RTView configurations that are specified in the **rtvservers.dat** file. **all** applies the action to all RTView configurations specified in the **rtvservers.dat** file (and corresponding servers or clients specified in each configuration). **Note:** When multiple configurations are specified in the **rtvservers.dat** file and they have different project settings directory locations, the **all** argument processes all the configurations. However, if the configurations have the same project settings directory locations, the **all** argument processes only the first configuration as the others are considered alternative configurations. Example: **stop_rtv all** **(Append .sh on UNIX)**
	[Configuration Name] Stops a single RTView configuration specified in the **rtvservers.dat** file: **stop_rtv [Configuration Name]** **(Append .sh on UNIX)** Configuration Name is the RTView configuration name specified in the **rtvservers.dat** file. The action applies to all servers or clients specified in the configuration. Example: **stop_rtv web_deployment** **(Append .sh on UNIX)**
	[Server Name] Stops a single process in an RTView configuration specified in the **rtvservers.dat** file: **stop_rtv [Configuration Name] [Server Name]** **(Append .sh on UNIX)** **Server Name** is the name of a server or client member in the configuration. For example, **dataserver**, **displayserver**, **historian** and **database**. The action applies only to that server or client in the configuration. Example: **stop_rtv web_deployment dataserver** **(Append .sh on UNIX)**
	Use With Secured JMX Ports This script works with RTView servers whose JMX ports are secured with either a username and password, or with SSL. You provide the scripts with the necessary credential information and the scripts manage authentication with the server. There are two ways that you can provide credential information to the scripts: via command-line arguments and via properties placed in any property file that is used by the server. **Securing with username and password** • To secure with a username and password via command-line, use the arguments as follows: **-jmxuser:...** **-jmxpass:...**

	• To secure with a username and password in a property file, use the properties as follows: **sl.rtview.jmxremote.username=...** **sl.rtview.jmxremote.password=....** **Securing with SSL** To secure with SSL, you provide the client KeyStore and TrustStore locations and their corresponding passwords. • To secure with SSL via command-line, use the arguments as follows: **-sslkeystore:...** **-sslkeystorepass:...** **-ssltruststore:...** **-ssltruststorepass:...** • To secure with SSL in a property file, use the properties as follows: **sl.rtview.ssl.client.keyStore=...** **sl.rtview.ssl.client.keyStorePassword=...** **sl.rtview.ssl.client.trustStore=...** **sl.rtview.ssl.client.trustStorePassword=....** **Password Encryption** To encrypt the passwords in your properties files, use the command-line tool "encode_string", for example: **encode_string encoder2 password** This will give you an encrypted value for "password" that you can use in your properties.
stop_server.bat/sh	Stops the RTView DataServer. Location: **<installation directory>** This script must be executed in the directory in which it resides. Format: **stop_server** (Append **.sh** on UNIX)
update_wars.bat/sh	Creates/updates the primary Monitor servlets. Location: **<installation directory>/projects/rtview-server** This script must be executed in the directory in which it resides. This script requires **rtvapm_init.bat**/**sh** be executed first. Format: **update_wars.sh [appname [host [portprefix]]]** For example: **update_wars.sh my-appname my-hostname 99** The name, host, and portprefix are declared in variables at the top of the script for easy editing, and can be passed into the scripts on the command-line.

	You can use **?** or **help** to get a usage message. For example: **update_wars.sh help** You can edit other variables at the top of the scripts to set properties for high-availability (HA). **Set HA_HOST** to the hostname of the backup data server. **Set HA_DISPLAYHOST** to the hostname of the backup display server. **Set HA_FAILBACK** to true to automatically reconnect to the primary display server.

rtvservers.dat

This section describes the **rtvservers.dat** configuration file which is used to manage your TIBCO RTView for TIBCO ActiveSpaces deployment and RTView processes. This section includes:

The **rtvservers.dat** text file contains one or more RTView configurations. An RTView configuration is a group of servers that should be started together. For example, the configuration might include any of the following: a Data Server, Historian, HSQLDB database, and a Display Server (for a Web Deployment). The **rtvservers.dat** file is used when the following scripts are executed:

- start_rtv Starts RTView processes specified in the **rtvservers.dat** file.
- stop_rtv Stops the RTView processes specified in the **rtvservers.dat** file.
- status_rtv Returns status information for RTView processes specified in the **rtvservers.dat** file.

The following **rtvservers.dat** file, located in your project directory, contains a single RTView configuration, named **default**.

```
default . dataserver rundata
default . historian runhist -ds
default . database rundb
```

> **Note:** The last line in the **rtvservers.dat** file must end with a new line, or be followed by a blank line.

In this example, to start the **default** configuration type: **start_rtv default** or **start_rtv all**. To start a single server in the configuration, type **start_rtv <Configuration Name> <Server Name>**. For example: **start_rtv default displayserver**.

Each line has the following format consisting of four fields:

<Configuration Name> <Project Settings Directory Location> <Property Filter Identifying the Server> <Command>

<Configuration Name>	The name of the RTView configuration (**default** in this example).
<Project Settings Directory Location>	The TIBCO RTView for TIBCO ActiveSpaces project settings directory location, relative to the location of the **rtvservers.dat** file (**.**, the current directory, in this example).
<Property Filter Identifying the Server>	The property filter that identifies the server, which is the property filter under which the server's JMX port is defined. By default, this is the server name, such as **dataserver** and **historian**.

<Command>	The script used to start the process. Valid values are: • rundata: Starts the Data Server. • runhist: Starts the Historian. • rundb: Starts the HSQLDB Database.

APPENDIX B Alert Definitions

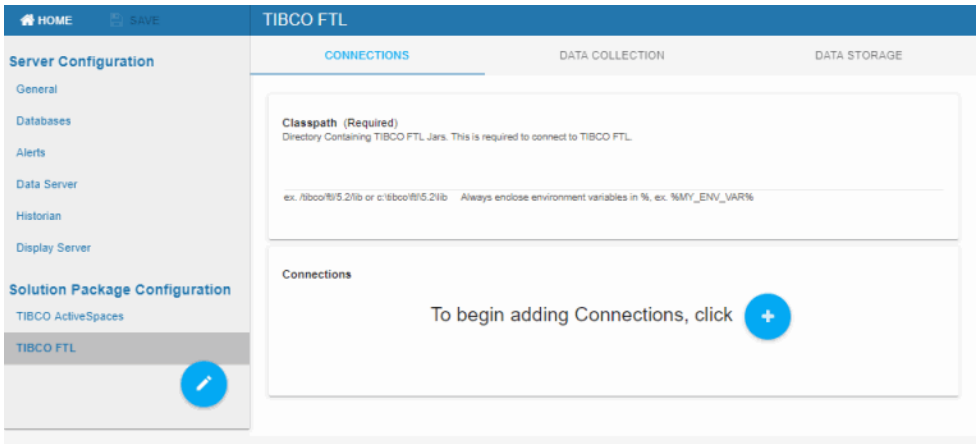
This section describes alerts for TIBCO ActiveSpaces Monitor and their default settings.

Alert Name	WARN. LEVEL	ALARMLEVEL	DURATION	ENABLED
TdgKeeperCpuUsageHigh The keeper CPU usage rate (msec/sec) is above the defined thresholds. **Index Type(s)**: PerTdgKeeper	60	80	30	FALSE
TdgKeeperExpired RTView is not receiving metrics updates from this Keeper. The Expired flag of the Keeper was set to true. **Index Type(s)**: PerTdgKeeper	NaN	NaN	30	FALSE
TdgKeeperMemoryUseHigh The keeper's usage of memory, in KB, is above the threshold. **Index Type(s)**: PerTdgKeeper	1600000	2000000	30	FALSE
TdgKeeperMsgsRcvdRateHigh The incoming message rate, in messages per second, is higher than expected for this keeper. **Index Type(s)**: PerTdgKeeper	160000	200000	30	FALSE
TdgKeeperMsgsSentRateLow The keeper's rate of messages sent is below the threshold. **Index Type(s)**: PerTdgKeeper	15	5	30	FALSE
TdgKeeperNotRunning The current status for this keeper is not "RUNNING." **Index Type(s)**: PerTdgKeeper	NaN	NaN	30	FALSE
TdgNodeCpuUsageHigh The node CPU Usage rate (msec/sec) is above threshold. **Index Type(s)**: PerTdgNode	60	80	30	FALSE
TdgNodeExpired RTView is not receiving metrics updates from this Node. The Expired flag of the Node was set to true. **Index Type(s)**: PerTdgNode	NaN	NaN	30	FALSE
TdgNodeLiveDataSizeHigh The node's live data size is above the threshold. **Index Type(s)**: PerTdgNode	1600000	2000000	30	FALSE

TdgNodeMemoryUseHigh The node's usage of memory, in KB, is above the threshold. **Index Type(s)**: PerTdgNode	1600000	2000000	30	FALSE
TdgNodeMsgsRcvdRateHigh The incoming message rate, in messages per second, is higher than expected for this node. **Index Type(s)**: PerTdgNode	160000	200000	30	FALSE
TdgNodeMsgsSentRateLow The outgoing message rate, in messages per second, is lower than expected for this node. **Index Type(s)**: PerTdgNode	15	5	30	FALSE
TdgNodeNotRunning The current status for this node is not "RUNNING". **Index Type(s)**: PerTdgNode	NaN	NaN	30	FALSE
TdgNodeOpsCompletedRateLow The rate of completed operations on the node is below the threshold. **Index Type(s)**: PerTdgNode	15	5	30	FALSE
TdgNodeOpsFailedRateHigh The rate of failed operations on the node is above the threshold. **Index Type(s)**: PerTdgNode	10	20	30	FALSE
TdgNodeTxnRollbackRateHigh The node's rate of transactions rolled back is above the threshold. **Index Type(s)**: PerTdgNode	50	100	30	FALSE
TdgProxyCpuUsageHigh The proxy CPU Usage rate (msec/sec) is above the defined threshold. **Index Type(s)**: PerTdgProxy	60	80	30	FALSE
TdgProxyExpired RTView is not receiving metrics updates from this Proxy. The Expired flag of the Proxy was set to true. **Index Type(s)**: PerTdgProxy	NaN	NaN	30	FALSE
TdgProxyMemoryUseHigh The proxy's usage of memory, in kilobytes, is above the threshold. **Index Type(s)**: PerTdgProxy	1600000	2000000	30	FALSE
TdgProxyMsgsRcvdRateHigh The incoming message rate, in messages per second, is higher than expected for this proxy. **Index Type(s)**: PerTdgProxy	160000	200000	30	FALSE
TdgProxyMsgsSentRateLow The outgoing message rate, in messages	15	5	30	FALSE

per second, is lower than expected for this proxy. **Index Type(s)**: PerTdgProxy				
TdgProxyNotRunning The current status for this proxy is not "RUNNING." **Index Type(s)**: PerTdgProxy	NaN	NaN	30	FALSE
TdgProxyTxnRollbackRateHigh The proxy's rate of transactions rolled back is above the threshold. **Index Type(s)**: PerTdgProxy	50	100	30	FALSE
TdgRealmOpsCompletedRateLow The rate of completed operations on the realm is below the threshold. **Index Type(s)**: PerTdgRealm	15	5	30	FALSE
TdgRealmOpsFailedRateHigh The rate of failed operations on the realm is above the threshold. **Index Type(s)**: PerTdgRealm	10	20	30	FALSE
TdgRealmServerCpuUsageHigh The CPU utilization of the Realm Server, as a percentage, is above the threshold. **Index Type(s)**: PerTdgRealm	60	80	30	FALSE
TdgRealmServerExpired RTView is not receiving metrics updates from this Realm Server. The Expired flag was set to true. **Index Type(s)**: PerTdgRealm	NaN	NaN	30	FALSE
TdgRealmServerMemoryUseHigh The Realm Server memory usage (RSS) is above threshold. Units are kilobytes. **Index Type(s)**: PerTdgRealm	160	200	30	FALSE
TdgRealmTxnRollbackRateHigh The node's rate of transactions rolled back is above the threshold. **Index Type(s)**: PerTdgRealm	50	100	30	FALSE

APPENDIX C RTView Configuration Application



The RTView Configuration Application is a tool that you can use to help configure the Monitor by defining various properties and connections via an easy-to-use interface. The RTView Configuration Application consists of three different sections: the top-level **Projects** page, the **Server Configuration** view, and the **Solution Package Configuration** view. This section will provide high-level definitions for each option within each view. More detailed descriptions on how this tool can be used to set up the Monitor can be found in the Configuration chapter, as well as in the Quick Start chapter.

This section contains the following:

- Accessing the RTView Configuration Application
- Projects Page
- Server Configuration View
- Solution Package Configuration View

Accessing the RTView Configuration Application

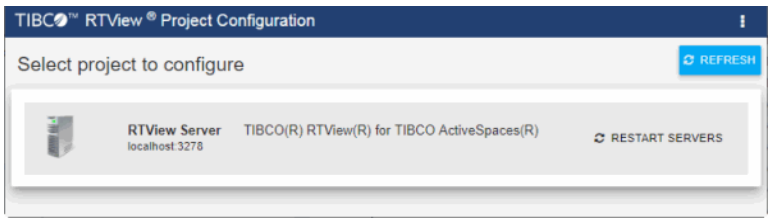
You can access the RTView Configuration Application via URL by performing the following steps:

1. Download and extract the TIBCO ActiveSpaces compressed .zip file.
2. Set the **JAVA_HOME** environment variable.
3. Run **start_server** from your project directory to start all servers.

4. Open a browser and enter **http://localhost:3270/rtview-tdgmon-rtvadmin**. Use username/password rtvadmin/rtvadmin.

See Quick Start for additional details.

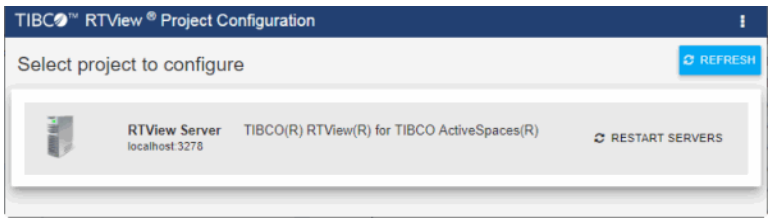
Note: Once you have finished making changes in the RTView Configuration Application, you must restart your data server for your changes to take place in the Monitor.



See Quick Start for additional details.

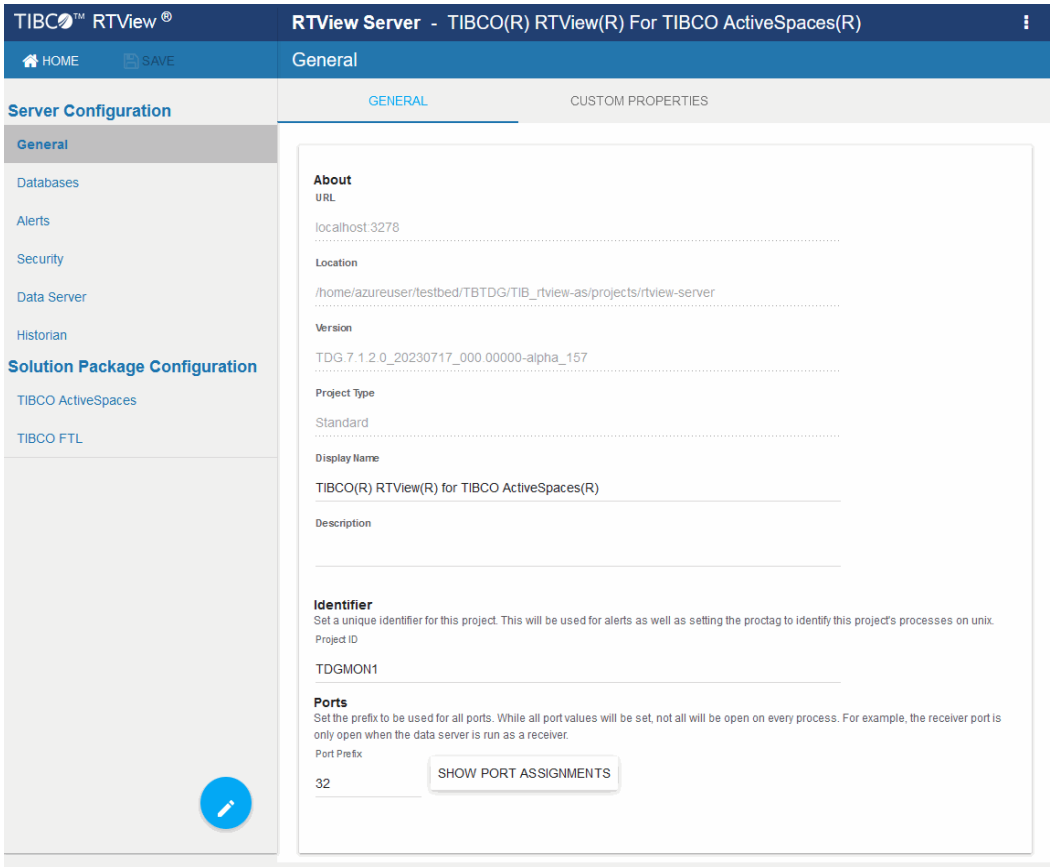
Note: Once you have finished making changes in the RTView Configuration Application, you must restart your data server for your changes to take place in the Monitor.

Projects Page



The Projects Page lists the project(s) in your project directory. Click the project to access the Configuration Views.

Server Configuration View



The **Server Configuration** View provides options that allow you to modify the default settings for the project including the project name and default port, define the alert threshold database connection and alert notification settings, define custom properties, define data server properties, define display server properties, and define the historian database connection and other historian properties. This section contains the following:

- General
- Databases
- Alerts
- Data Server
- Historian
- Display Server

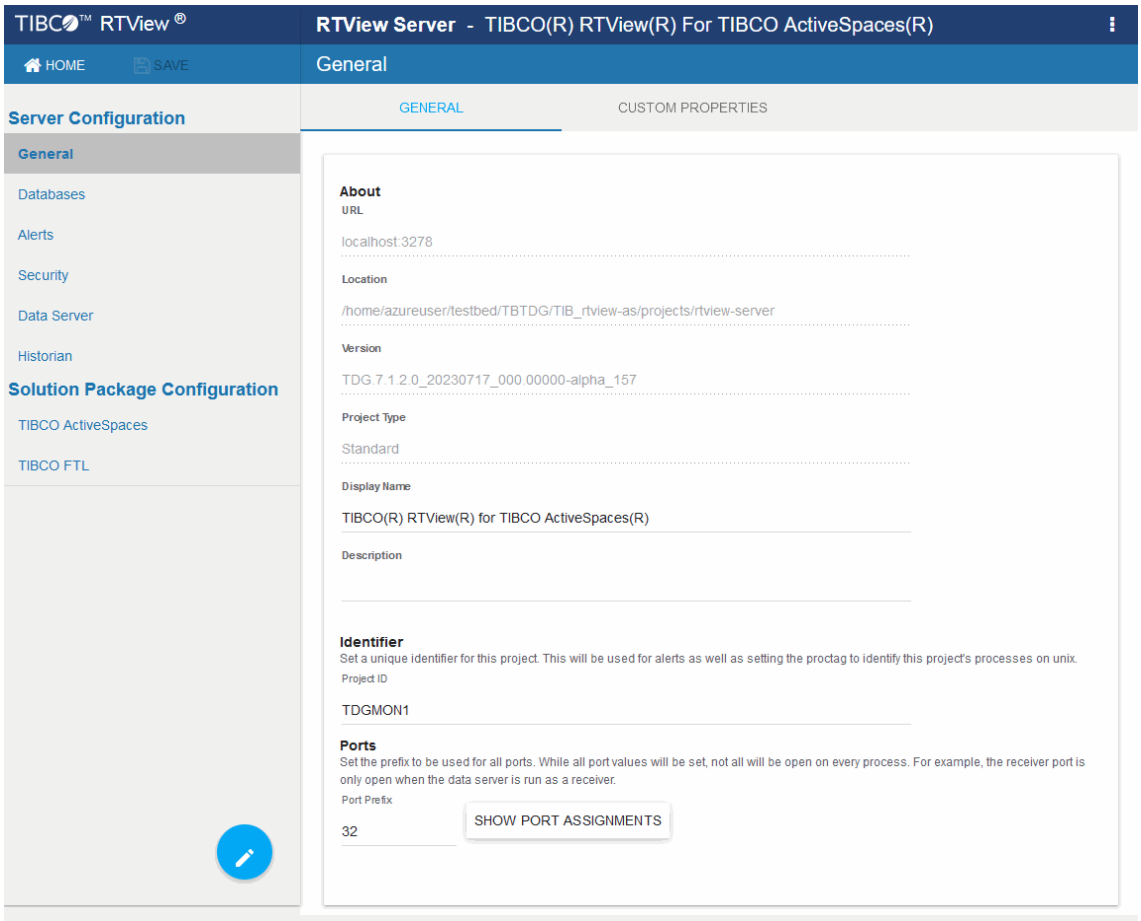
Tip: Gray text shows the default setting for the field which you can edit. To return to the default setting, delete the text you entered.

General

The **General** option consists of two different tabs that allow you to define the values for the project, specify the port, and define any custom properties you might need to create. The available tabs are:

- General Tab
- Custom Properties Tab

General Tab



The **General/GENERAL** tab contains the following regions:

About

URL: Displays the URL used to the connect to the server. This field cannot be edited.

Location: Displays the project directory location (path). This field cannot be edited.

Version: Displays the current version of TIBCO ActiveSpaces installed. This field cannot be edited.

Project Type: Displays the type of project (Standard, Sender, or ConfigClient). This field cannot be edited.

Display Name: Displays the default name for the project and displays on the Home/**RTView Project Configuration** (top level) page. This field can be edited.

Description: Optionally specify a description that will display on the Home/**RTView Project Configuration** (top level) page.

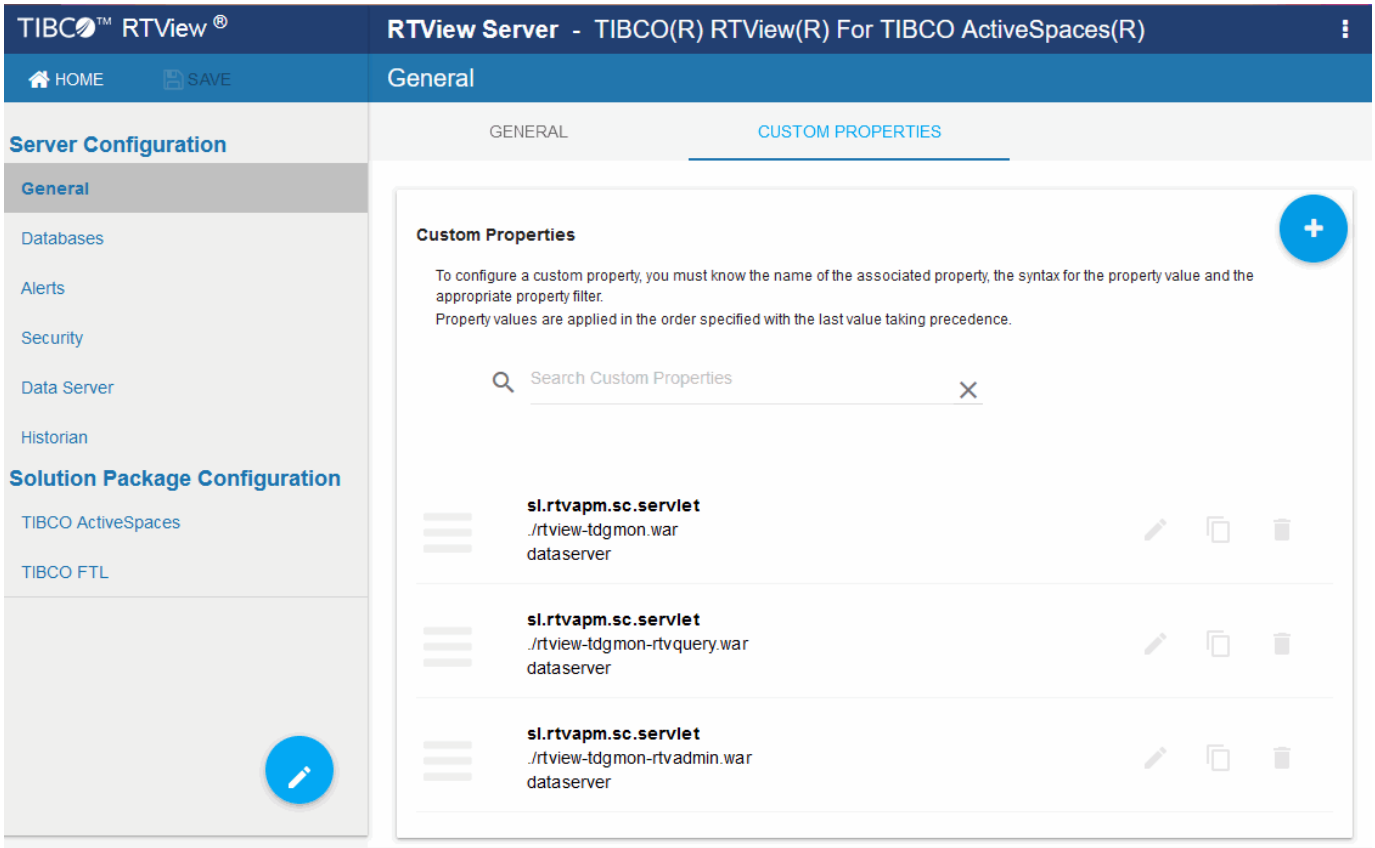
Identifier

Project ID: Displays a default unique identifier for the project, which you can modify.

Ports

Port Prefix: Displays the default port prefix (first two numbers used for the port) that will be used for all ports, which you can modify. The latter two numbers in the port are predefined and cannot be modified. Click **Show Port Assignments** to view the Port Assignments.

Custom Properties Tab



The **General/CUSTOM PROPERTIES** tab allows you to create custom properties. Property values are applied in the order specified with the last value taking precedence. To create properties you need the name of the associated property, the syntax for the property value,

and the appropriate property filter. Click the icon to open the **Add Property** dialog, which has the following fields:

Name: (Required) The name of the associated property.

Value: (Optional) The value for the associated property (using the correct syntax).

Filter: (Optional) The filter for the associated property.

Comment: (Optional) Enter useful details about the property and its behavior for yourself and other users.

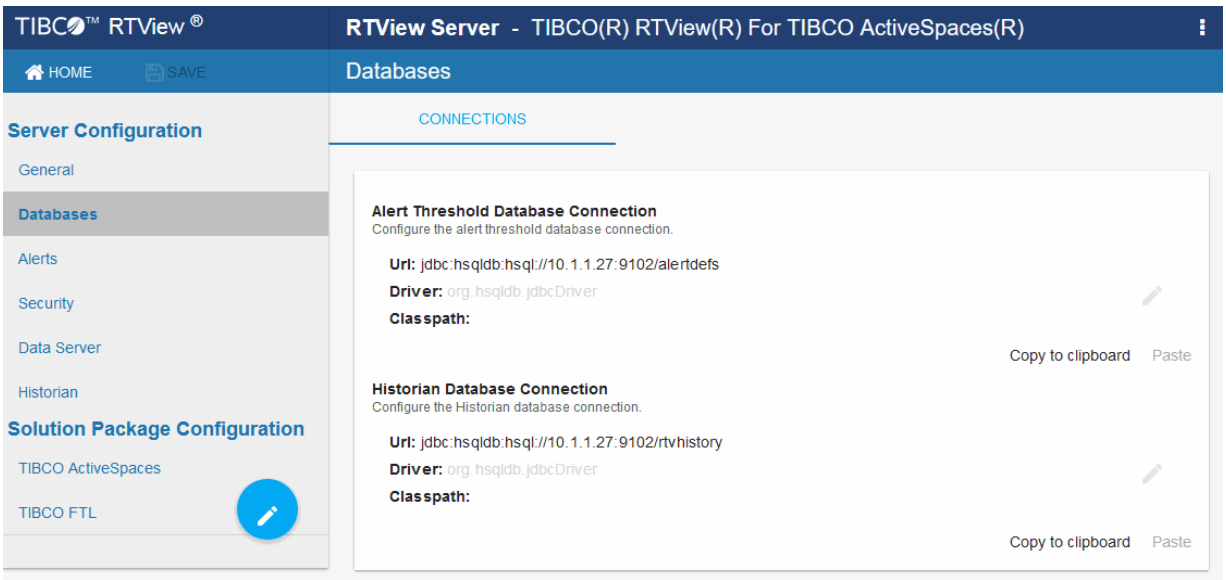
See Configure Alert Notification for an example of when you can use **Custom Properties**.

Databases

The **Databases** option consists of the **Connections** tab that allows you to define Alert Threshold Database and Historian Database connections.

- Connections Tab

Connections Tab



This tab contains the following regions:

Alert Threshold Database Connection

If you want to use local alert threshold database connection, add the connection information where:

URL: The complete URL for the database connection.

Driver: The full name for the driver.

Classpath: The complete classpath for the jar location.

Username: The username is used when creating the connection. This field is optional.

Password: This password is used when creating the connection. This field is optional. By default, the password entered is hidden. Click the icon to view the password text.

Run Queries Concurrently: When selected, database queries are run concurrently.

Historian Database Connection

URL: The complete URL for the database connection.

Driver: The full name for the driver.

Classpath: The complete classpath for the jar location.

Username: The username is used when creating the connection. This field is optional.

Password: This password is used when creating the connection. This field is optional. By default, the password entered is hidden. Click the icon to view the password text.

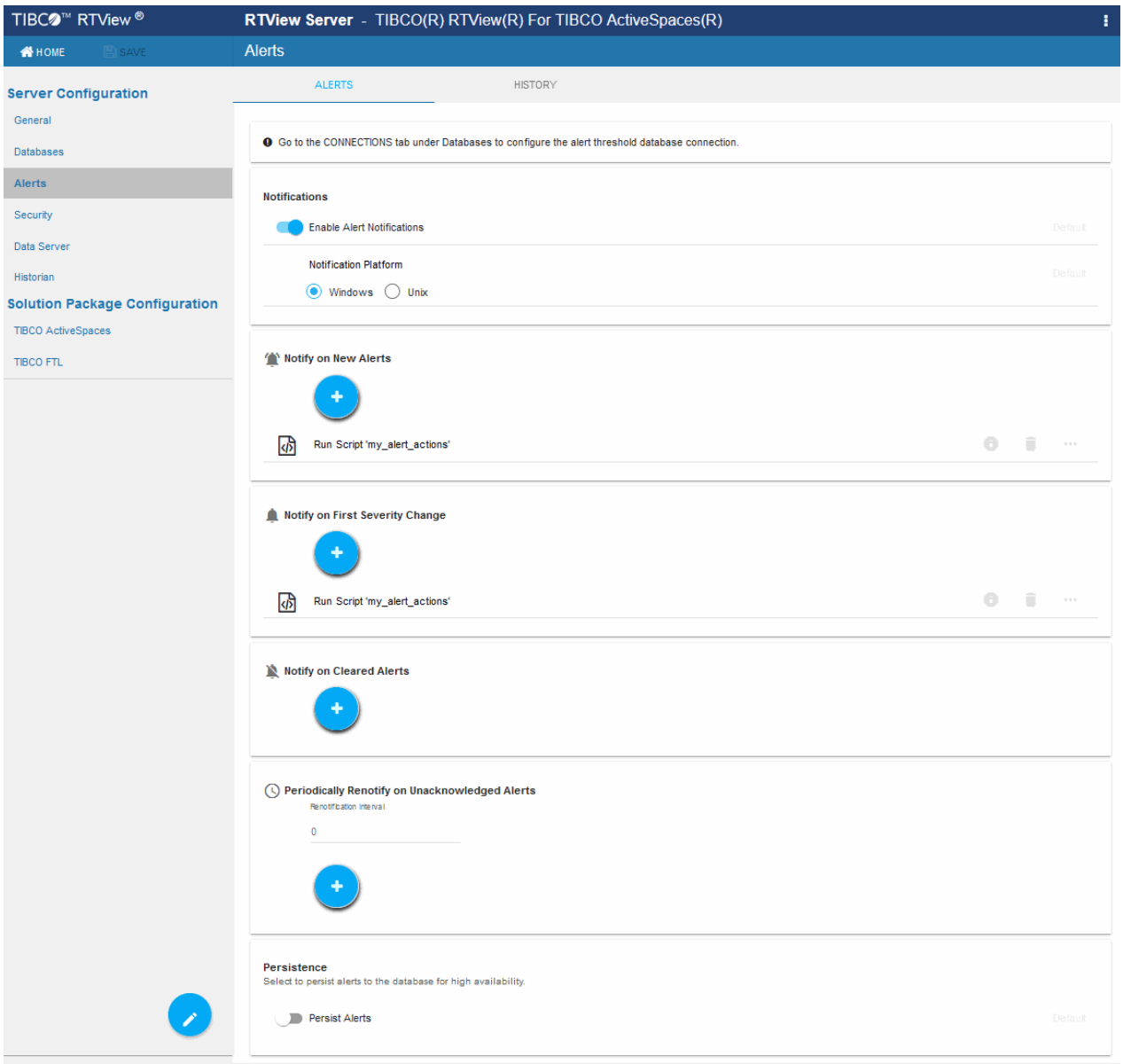
Run Queries Concurrently: When selected, database queries are run concurrently.

Alerts

The Alerts option consists of the Alerts tab and the History tab, which allow you to define the alert and history properties. Alert and Historian database connections are set up using the Databases option. The following tabs are available:

- Alerts Tab
- History Tab

Alerts Tab



This tab contains the following regions:

Notifications

- **Enable Alert Notifications**: Selecting this toggle enables alert notifications to be sent.
- **Notification Platform**: Select the platform type (**UNIX** or **Windows**).

Alert Event Options

- **Notify on New Alerts**: A notification is executed every time a new alert is created.
- **Nofity on First Severity Change**: A notification is executed the first time the **Severity** changes for each alert.
- **Nofity on Cleared Alerts**: A notification is executed every time an alert is cleared.
- **Periodically Renotify on Unacknowledged Alerts**: Enter the **Renotification Interval** (number of seconds). A notification is executed for each unacknowledged alert per the interval you specify here. If the Renotification Interval is greater than **0** and no actions are defined, the **New Alerts** action will be used for renotifications.

Selecting the  button next to each of the Alert Event Options displays the following options:

 Run a Script	This alert notification action executes the following script in the **TIB_rtview-ems/projects/rtview-server** directory: • **my_alert_actions.bat/sh** – New and First Severity Change • **my_alert_actions.cleared.bat/sh** – Cleared • **my_alert_actions.renotify.bat/sh** – Periodically Renotify This action can only be added once per notification type. In addition to selecting this action in the Configuration Application, you must also modify the appropriate script to execute the actions for your notification. This script has access to the following fields from the alert: **Alert Name**, **Alert Index**, **ID**, **Alert Text** and **Severity**.
 Execute Java Code	This alert notification action allows you to implement your alert notification actions using Java code. It executes the **my_alert_notification.$domainName.$alertNotifyType.$alertNotifyCol** command in your Custom Command Handler and passes the row from the alert table that corresponds to the alert. This action can only be added once per notification type. In addition to selecting this action the Configuration Application you must also modify the custom command handler to execute the actions for your notification. A sample custom command handler is included under **projects/custom**. It prints the alert notification to the console. You will modify this command handler to implement your own notification actions. Make the following entries: • **Custom Command Handler Class Name**: Enter the fully qualified name of the Custom Command Handler class. This defaults to the sample Custom Command Handler in the

TIB_rtview-ems/projects/custom directory.

- **Custom Command Handler Jar**: Enter the path and name of the jar containing the Custom Command Handler class. The path may be absolute or relative to the location of data server. This defaults to the sample Custom Command Handler in the **TIB_rtview-ems/projects/custom** directory.

Note that if you can only have one custom command handler per Data Server, so changing these settings for one notification event will change them for the rest of the notification events.

Send Email

This alert notification action sends an email. This action can be added multiple times per notification type. No additional setup is required beyond filling in the **Send Email** dialog in the Configuration Application.

Make the following entries:

- **SMTP Host**: The SMTP host address. This is required. Consult your administrator.
- **SMTP Port**: The SMTP port number. This is required. Consult your administrator.
- **From**: The email address to which to send the email. This is required.
- **To**: The email address to which to send the email. This is required and may contain multiple entries.
- **Subject**: The subject for the email. This is required. You can include the value from any column in the alert table in your subject. Click the **Show More** link at the bottom of the dialog to see the alert column values you can use in the **Subject**.
- **Body**: The body of the email. This is optional. Click the **Show More** link at the bottom of the dialog to see the alert column values you can use in the **Subject**.
- **User**: The user name for the account from which you are sending the email. This is optional.
- **Password**: The password for the account from which you are sending the email. This is optional.

Send SNMP Trap

This alert notification action sends an SNMP Trap as described in **rtvapm/common/lib/SL-RTVIEW-EM-MIB.txt**. This action can be added multiple times per notification type. No additional setup is required beyond filling in the **Send Email** dialog in the Configuration Application

Make the following entries:

- **Trap Type**: Select the SNMP version of the trap. This is required.
- **Destination Address**: The system name or IP address of the receiving system. This is required.
- **Destination Port**: The UDP port on the receiving system. This is required.

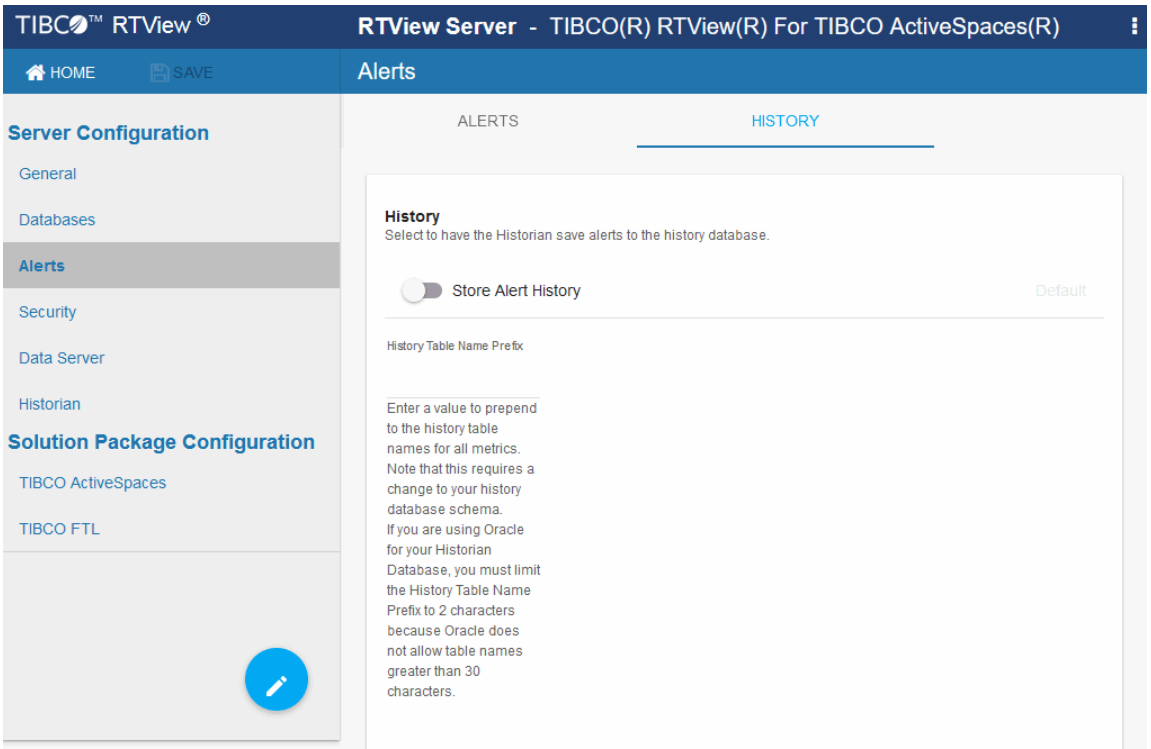
- **Community Name**: (This field is visible when **Trap Type v2/v3** is selected.) The SNMP v2 Community Name string. This is required.

	This alert notification action executes a specified command. This action can be added multiple times per notification type. Make the following entry:
Run Command String	**Command String**: Enter the command string for any command supported by RTView Classic. To enter a command string, you must know the correct syntax for the command. Contact Technical Support for assistance on syntax. You can include the value from any column in the alert table using the syntax in the Show More link at the bottom of the dialog.
	This alert notification action alert allows you to execute different actions for different alerts based on information in the alert. For example, you can configure EMS alerts to send emails to your EMS team and Solace alerts to send emails to your Solace team. This action can be added multiple times per notification type.
	To create a condition, make the following entries:
Conditional Filter	• **Alert Field**: Select an alert field: **Alert Name**, **Alert Index**, **Category, CI Name, Owner, Package, Primary Service** or **Severity**. This is required. Note that **CI Name** and **Primary Service** fields are for RTViewCentral only. • **Operator**: Select one - **EQUALS**, **DOES NOT EQUAL**, **STARTS WITH**, **ENDS WITH** or **CONTAINS**. This is required. • **Value**: Enter the value to which to compare the Alert Field. Cannot contain wildcard characters. This is required. • **Action(s)**: Select one or more actions to execute when this condition is met.

Persistence

Persist Alerts: When enabled, saves alerts to the database for high availability purposes.

History Tab



This tab contains the following region:

History

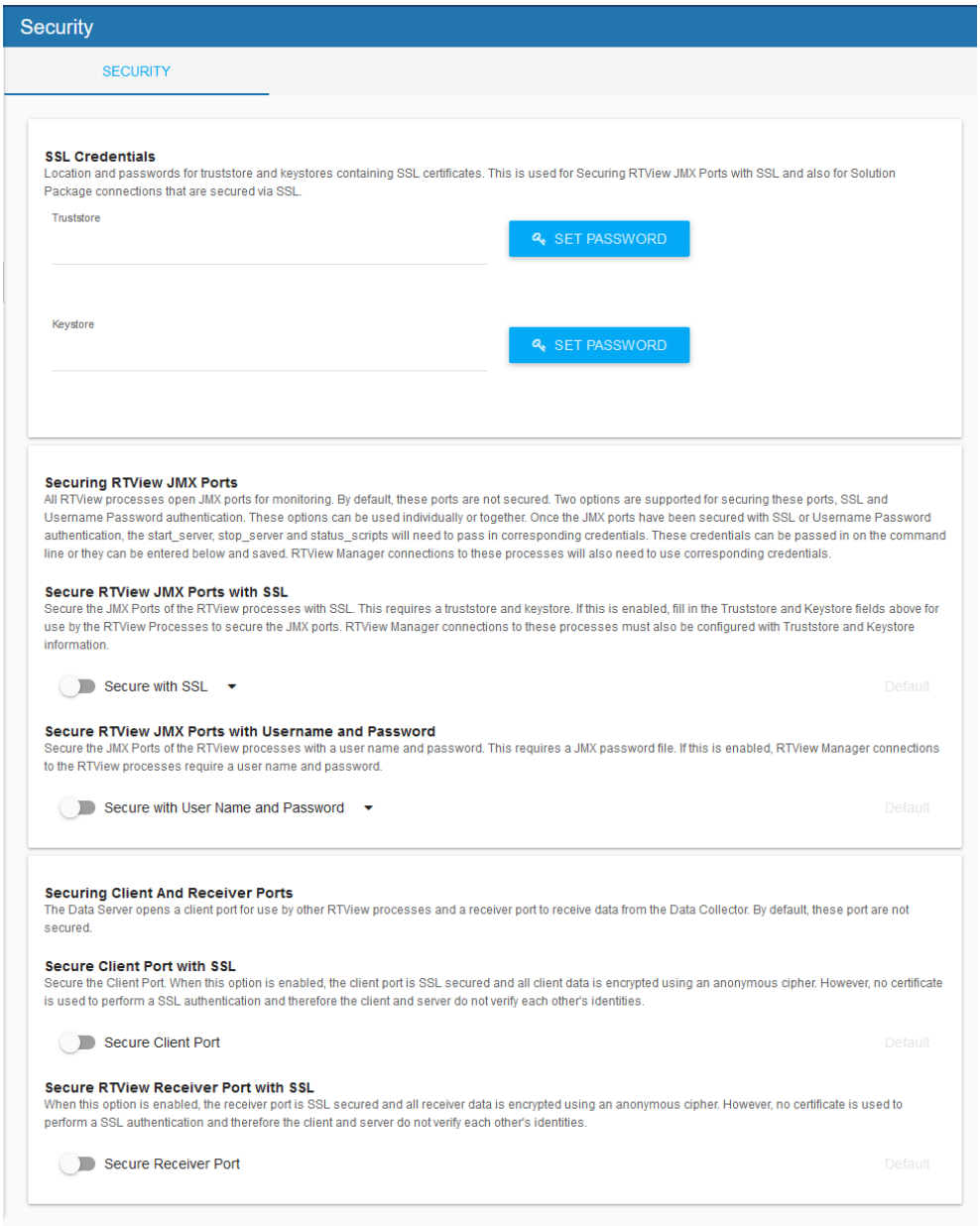
Store Alert History: Toggle to enable/disable **Store Alert History** to store alerts in the history database. This value is used in the **Alerts Table** (which makes it easier to find the alerts).

History Table Name Prefix: This field allows you to define a prefix that will be added to the database table names so that the Monitor can differentiate history data between data servers when you have multiple data servers with corresponding Historians using the same solution package(s) and database. In this case, each Historian needs to save to a different table, otherwise the corresponding data server will load metrics from both Historians on startup. Once you have defined the **History Table Name Prefix**, you will need to create the corresponding tables in your database as follows:

- Locate the .sql template for your database under **TIB_rtview-as/rtvapm/tdgmon/dbconfig** and make a copy of it
- Add the value you entered for the **History Table Name Prefix** to the beginning of all table names in the copied .sql template
- Use the copied .sql template to create the tables in your database

Security

All RTView processes (Data Server, Historian, Display Server) open JMX ports for monitoring which, by default, are not secured. The **Security** tab allows you secure these ports as well as specify credentials needed to connect to SSL secured servers from RTView's Solution Packages.



SSL Credentials

This region allows you to specify the path to the **Truststore** and **Keystore** files (and their associated passwords) that contain the SSL credentials needed to secure the RTView JMX Ports and/or access any SSL secured servers associated with RTView's Solution Packages. This is required if the **Secure with SSL** option is enabled (see below for details).

Optional: To obscure the credentials of the truststore and keystore in the output of the **ps** and **jps** commands, add the following custom property to each Data Server on which SSL Credentials have been configured:

```
Name: sl.rtview.jvm
Value: -Drtv.hidesslprops=true
Comment: hide ssl properties in ps/jps output
```

Securing RTView JMX Ports

This region provides a couple of options for securing the JMX ports that are opened by the RTView processes: **Secure with SSL** and/or **Secure with Username and Password**.

Secure with SSL

When toggled on, this option secures the JMX ports for the RTView processes with SSL. When the JMX ports are SSL secured, an SSL handshake is performed between the client and the server when the client attempts to connect. For this handshake, the client must provide a certificate the server trusts, and the server must provide a certificate the client trusts. A Keystore file contains the application's certificate and private key and a Truststore file contains the application's trusted certificates. These files are created by running the Java keytool on the command line. When this option is enabled, you must specify the path to the server's Truststore and Keystore files (and their associated passwords) in the **SSL Credentials** region (see above).

The **start_server**, **stop_server**, and **status_server** scripts are all connected to the JMX Ports of the RTView processes to execute operations and get status. If the JMX ports have been secured with SSL, these scripts need the path and passwords for the truststore and keystore files containing the client credentials in order to connect. You can either pass these in on the command line each time you run (**-sslkeystore:clientkeystore.jks-sslkeystorepass:clientkeystorepass-ssltruststore:clienttruststore.jks-ssltruststorepass:clienttruststorepass**) or you can fill in the fields under **SSL Credentials for RTView Scripts**.

The RTView Manager application also connects to the JMX Ports of the RTView processes in order to monitor them. If you are using the RTView Manager and the JMX ports have been secured with SSL, you must fill in the **SSL Credentials** on the **Security** tab of the RTView Manager Configuration Application to specify the path the truststore and keystore files containing the client credentials.

Secure with Username and Password

This region allows you to secure the JMX ports for RTView processes with a username/password. This can be used in addition to Securing with SSL (see above). If this option is enabled, you must specify the path to a JMX password file containing all valid user names and passwords.

Important! A JMX password file must be read-only to the owner. See Java documentation for details on the creation of a JMX remote password file.

The **start_server**, **stop_server**, and **status_server** scripts are all connected to the JMX Ports of the RTView processes to execute operations and get status. If the JMX ports have been secured with a username and password, the scripts need a valid user name and password in order to connect. You can either pass these into the command line each time you run (**-jmxuser:userName-jmxpass:myPassword**) or you can fill in the **Username and Password Credentials** and enable the **Use for Scripts** toggle.

The RTView Manager application also connects to the JMX Ports of the RTView processes in order to monitor them. If you are using the RTView Manager in RTViewCentral and the JMX ports have been secured with username and password, you must fill in the **Username and Password Credentials** that the RTView Manager should use to connect. If you are using the RTView Manager in a deliverable other than RTViewCentral, you will need to fill in the user name and password in the connection to this RTViewDataServer in the RTView Manager Configuration Application.

Securing RTView JMX Ports

All RTView processes open JMX ports for monitoring. By default, these ports are not secured. Two options are supported for securing these ports, SSL and Username Password authentication. These options can be used individually or together. Once the JMX ports have been secured with SSL or Username Password authentication, the start_server, stop_server and status_scripts will need to pass in corresponding credentials. These credentials can be passed in on the command line or they can be entered below and saved. RTView Manager connections to these processes will also need to use corresponding credentials.

Secure RTView JMX Ports with SSL

Secure the JMX Ports of the RTView processes with SSL. This requires a truststore and keystore. If this is enabled, fill in the Truststore and Keystore fields above for use by the RTView Processes to secure the JMX ports. RTView Manager connections to these processes must also be configured with Truststore and Keystore information.

Secure with SSL

SSL Credentials for RTView Scripts

The start_server, stop_server and status_server scripts connect to the RTView processes using JMX. You can either save the client Truststore and Keystore properties below for use by the scripts or you can pass them in on the command line each time you execute those scripts. For example, start_server.sh -sslkeystore:clientkeystore.jks -sslkeystorepass:clientkeystorepass -ssltruststore:clienttruststore.jks -ssltruststorepass:clienttruststorepass.

Client Truststore

SET PASSWORD

Client Keystore

SET PASSWORD

Secure RTView JMX Ports with Username and Password

Secure the JXM Ports of the RTView processes with a user name and password. This requires a JMX password file. If this is enabled, RTView Manager connections to the RTView processes require a user name and password.

Secure with User Name and Password

Password File

Username and Password Credentials

A user name and password are required in order for the RTView Manager in RTViewCentral to monitor these RTView processes.

Username

SET PASSWORD

The start_server, stop_server and status_server scripts also connect to RTView processes using JMX. You can optionally allow the scripts use the user name and password entered above or you can enter them on the command line each time you run the start_server, stop_server and status_server scripts. For example, start_server.sh -jmxuser:userName -jmxpass:myPassword.

Use for Scripts

Default

Secure Client and Receiver Ports with SSL

The Data Server opens a client port for use by other RTView processes and a receiver port to receive data from the Data Collector. By default, these port are not secured.

When **Secure Client Port with SSL** is enabled, the client port is SSL secured and all client data is encrypted using an anonymous cipher. However, no certificate is used to perform a SSL authentication and therefore the client and server do not verify each other's identities.

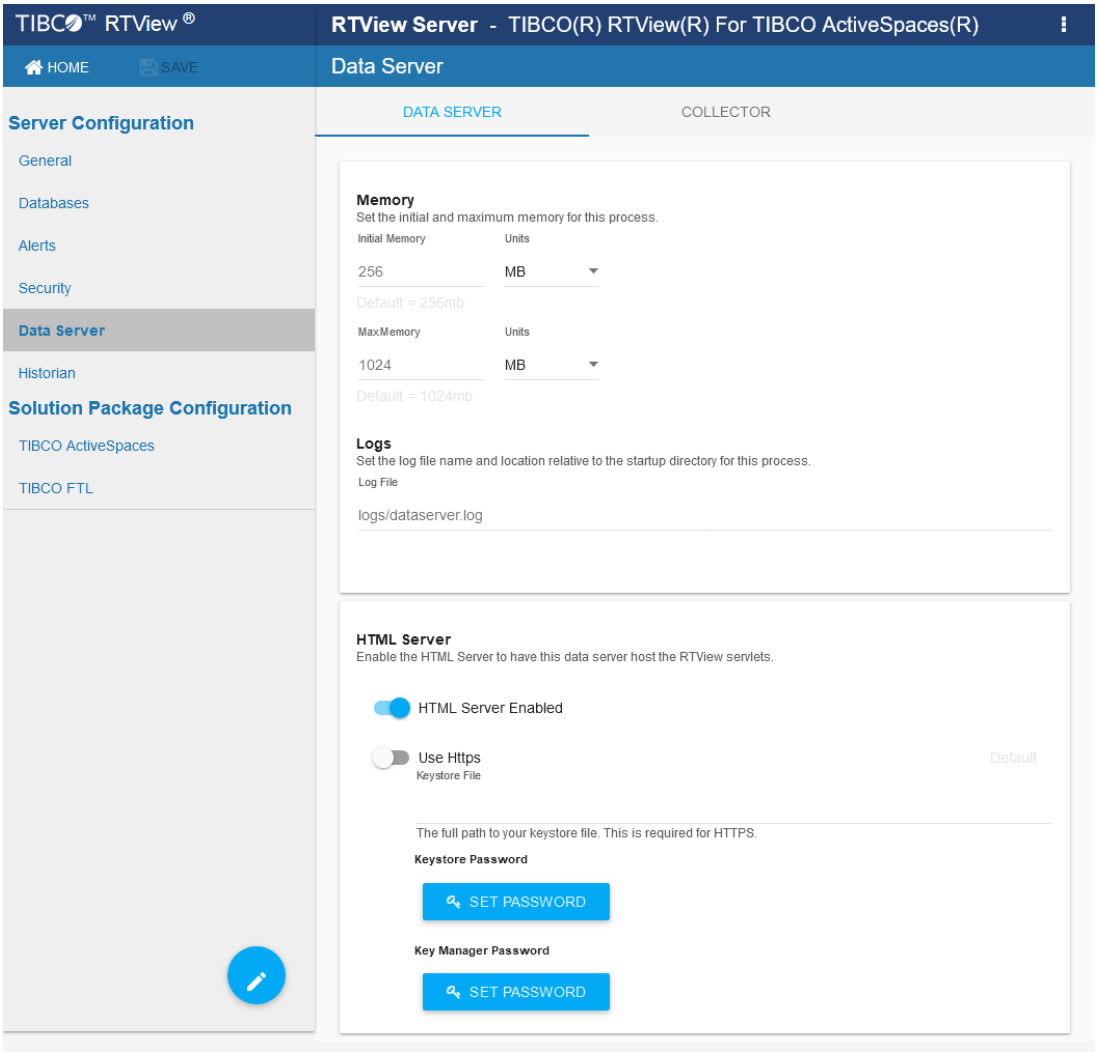
When **Secure RTView Receiver Port** is enabled, the receiver port is SSL secured and all receiver data is encrypted using an anonymous cipher. However, no certificate is used to perform a SSL authentication and therefore the client and server do not verify each other's identities.

Data Server

This section describes the Data Server Configuration settings. There are two tabs available:

- Data Server Tab
- Collector Tab

Data Server Tab



The **Data Server/DATA SERVER** tab contains the following:

Memory: Set the initial memory and maximum memory for the Data Server process. Specify a number followed by a unit. Units are k (kilobyte), m (megabyte), g (gigabyte). If no unit is used, the number is assumed to be bytes. **Note**: Use caution when you change the memory

allocation. If the memory allocation is too small the server might crash during startup and if too large the server might eventually exceed the available CPU/memory and fail.

Initial Memory: The initial amount of memory to allocate for this process.

Max Memory: The maximum amount of memory to allocate for this process.

Logs

Log File: The log file name and location relative to the startup directory for this process. In the **Log File** field, use the following format: **<directory name>/<log file name>**.

For example, logs/dataserver.log.

HTML Server

HTML Server Enabled: Enable the Eclipse Jetty HTML Server in the Data Server. If enabled, Eclipse Jetty will host the RTView Servlets at **http://localhost:XX70**, where **XX** is the port prefix specified on the **Server Configuration** > **General** > **GENERAL** tab. **Note**: You cannot disable this option if the RTView Configuration Application is being hosted by Eclipse Jetty in the Data Server. All RTView Servlets hosted by Eclipse Jetty are automatically configured with the correct Data Server port at runtime. The following RTView Servlets are hosted in Eclipse Jetty:

rtview-tdgmon-classic

rtview-tdgmon-rtvadmin

rtvadmin

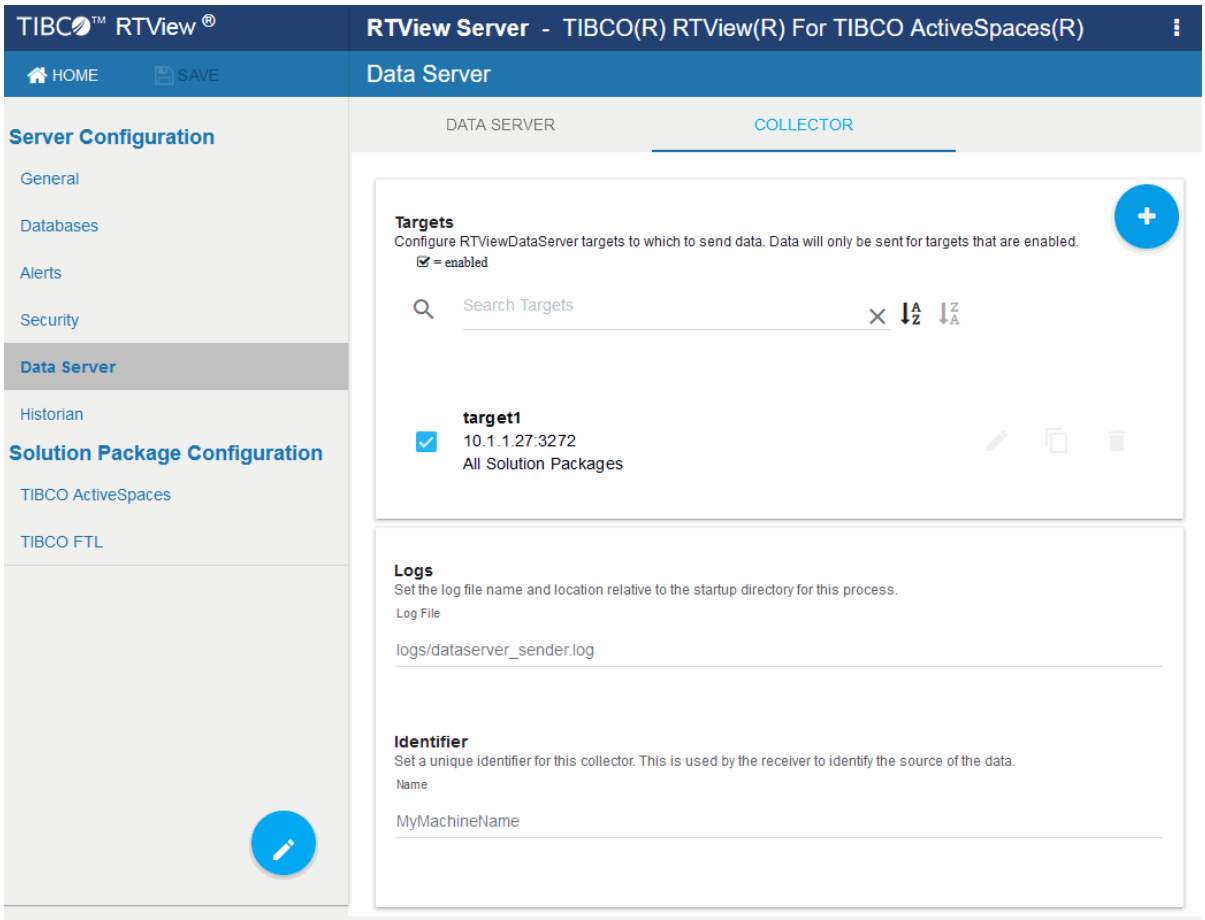
rtvdata

rtvquery

rtvagent

rtvpost

Collector Tab



The **Data Server/Collector** tab is only available when the data server is configured to be a sender. See Sender/Receiver: Distributing the Load of Data Collection for more information. This tab contains the following:

Targets: You can specify multiple targets by adding them one at a time. All fields on the **Add Target** dialog are required. Click the icon to open the **Add Target** dialog, which has the following fields:

ID: A unique name for the target.

URL: Specify the URL for the receiver. The url can be **host:port** (for example, somehost:3372) or an **http url** for the rtvagent servlet on the receiver. For example, if you are using Tomcat, you would use **http://somehost:8068/tdgmon-rtvagent**. If you are using Jetty, you would use **http://somehost:3270/rtvagent**.

Targets: Select the **All solution packages** option.

Enabled: Select this check box to enable the target.

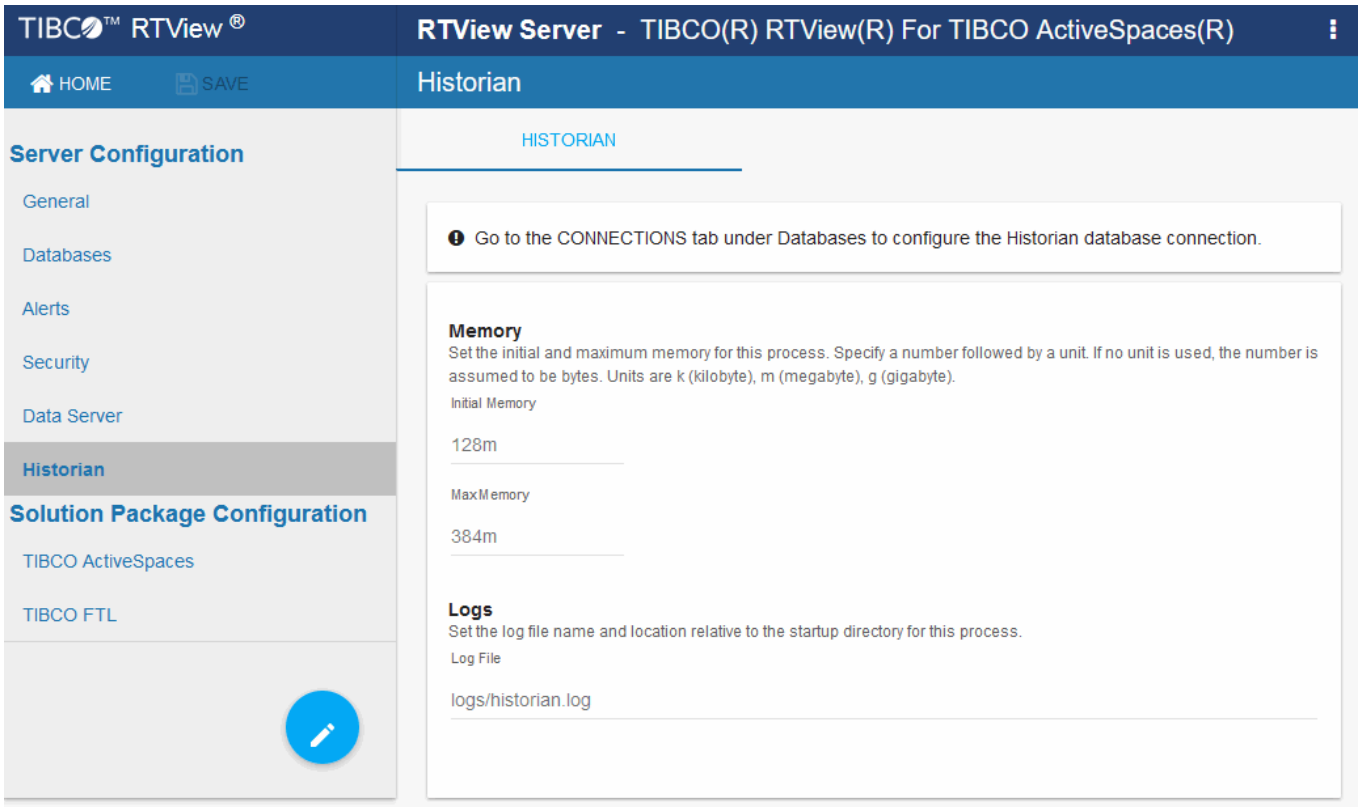
Logs

Log File: The log file name and full path.

Identifier

Name: A unique name for the data server, which is typically your machine's name.

Historian



The **Historian** option consists of the **Historian** tab, which allows you to define the history properties. Historian database connections are set up using the Databases option. This option contains the following regions:

Memory: Set the initial memory and maximum memory for the Historian process. Specify a number followed by a unit. Units are k (kilobyte), m (megabyte), g (gigabyte). If no unit is used, the number is assumed to be bytes.

> **Note:** Use caution when you change the memory allocation. If the memory allocation is too small the server might crash during startup and if too large the server might eventually exceed the available CPU/memory and fail.

Initial Memory: The initial amount of memory to allocate for this process.

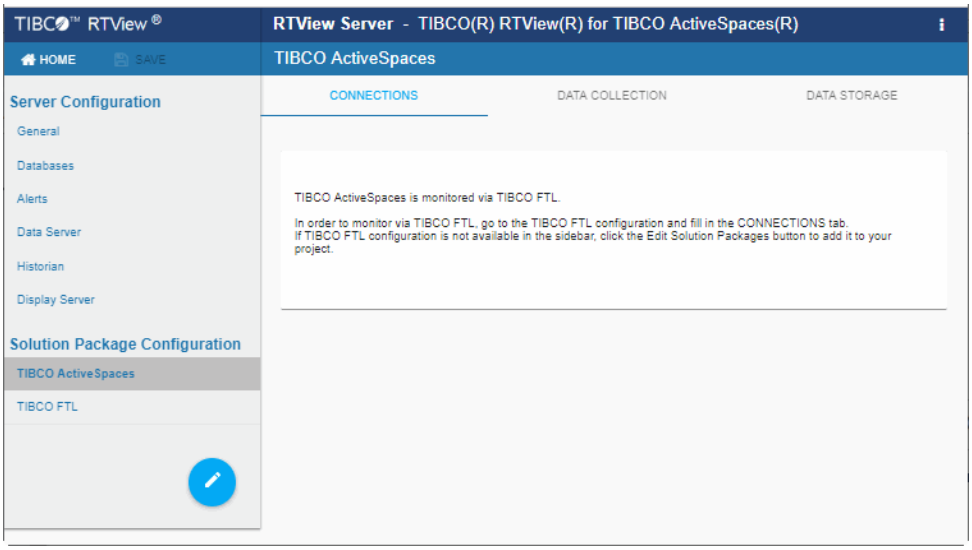
Max Memory: The maximum amount of memory to allocate for this process.

Logs

Log File: The log file name and location relative to the startup directory for this process. In the **Log File** field, use the following format: **<directory name>/<log file name>**.

For example, logs/historian.log.

Solution Package Configuration View



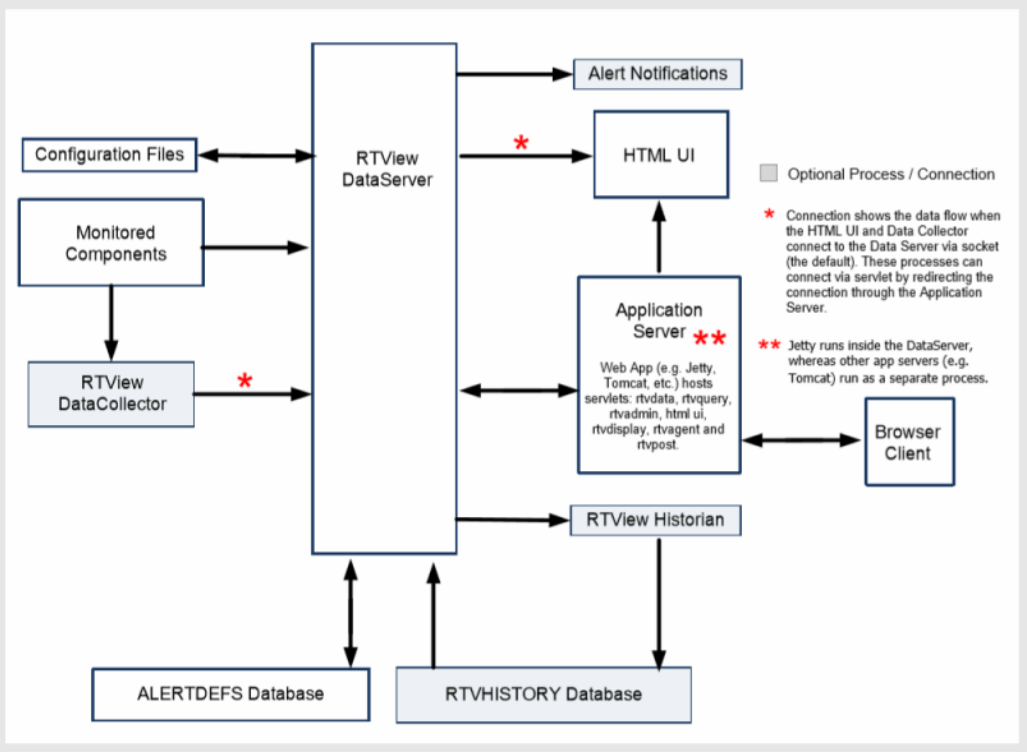
The **Solution Package Configuration** View provides options that allow you to modify the default settings for the project, define the classpaths and connections for the Monitor, and define the data collection and data storage properties for the Monitor. Descriptions for all of the properties for these options, as they pertain to the Monitor, are explained in detail in the Configuration chapter. You can also view the basic steps to get the Monitor up and running in the Quick Start chapter.

APPENDIX D Security Configuration

This section provides details for securing a direct connection RTView deployment. This section contains:

Introduction

The following diagram shows how data flows through the RTView deployment. The Data Server connects to the Monitored Components to collect metric data which it stores in local caches. The Data Server uses the collected data to generate alerts based on enabled threshold settings in the ALERDEFS database. If the user has (optionally) defined alert notifications, the Data Server also executes them.



In cases where the data collection needs to be distributed, one or more Data Collectors can be deployed to connect to the Monitored Components and forward the collected data to the Data Server.

The HTML UI is a browser-based user interfaces that show metric and alert data from the Data Server and also allow the user to enable, disable and set thresholds on alerts.

The Historian is an optional process that stores historical metric and alert data to the RTVHISTORY database. When the Historian is enabled, the Data Server queries historical data from the RTVHISTORY on startup to populate in-memory history and also any time the HTML UI request history data that is older than the data in the in-memory history.

The Configuration Application is a browser based application for configuring the RTView processes. It connects to the Data Server to read and write Configuration Files.

The next sections provide a more detailed description of each process.

Data Server

The Data Server gathers and caches the data from the applications being monitored and also hosts the alerts for that data. Because the Data Server can exist behind firewalls, it simplifies and strengthens the secured delivery of information to clients beyond the firewall. The Data Server serves metric and alert data to the Historian via socket on port **3278** and receives data via socket from the optional Data Collector on port **3272**. It also serves metrics and alert data to the HTML UI via the rtvquery servlet which connects via socket on port **3278**.

The Historian runs in the same directory as the Data Server, while the optional Data Collector (s) typically run in a different directory or a different system. By default, socket connections to the Data Server are unsecured. The Data Server supports secure socket connections (SSL) with or without certificates. It also supports client whitelist and blacklist. Secure socket and

client whitelist/blacklist configuration are described in the *RTView Core User's Guide* under Deployment/Data Server/Security.

The HTML UI connects to the Data Server via the rtvquery servlet. See HTML UI in this document for information on how to enable authentication in the HTML IU and rtvquery servlets. The rtvquery servlet will connect via secure socket if the Data Server is configured for SSL sockets.

The Data Collector can optionally be configured to send data to the Data Server via the rtvagent servlet instead of the socket. In this case, the rtvagent servlet connects to the Data Server via socket on port **3272**. While the rtvagent servlet cannot be configured for authentication, Tomcat access filters can be used to restrict access. The rtvagent servlet will connect via secure socket if the Data Server is configured for SSL sockets.

The Configuration Application connects to the Data Server via the rtvadmin servlet to read and write properties files. The rtvadmin servlet connects to the Data Server via socket on port **3278**. See Configuration Application in this document for information about servlet authentication. The rtvadmin servlet will connect via secure socket if the Data Server is configured for SSL sockets.

If the Historian is enabled, the Data Server connects to the RTVHISTORY database on startup to read initial cache history data and if the thin client or HTML UI request history data older than the in memory cache history. It also connects to the ALERTDEFS database to query and set alert thresholds. See Database in this document for more information.

The Data Server optionally executes alert notifications based on user settings. Since the notification actions are user defined, security must be determined by the user.

The Data Server opens a JMX port on **3268** to enable monitoring. By default, the JMX port is not secured. See Monitored Components for information on securing this connection.

By default, the Data Server runs a Jetty process which hosts all of the RTView servlets and accepts HTTP client requests on port **3270**. You can optionally configure Jetty to use HTTPS instead of HTTP.

Also see "Port Settings".

HTML UI

The new user interface is implemented in HTML and is deployed as a servlet, **rtview-tdgmon**, which is configured by default to use BASIC HTTP authentication. Browser clients connect via HTTP or HTTPS depending on the Application Server configuration. It should be used with HTTPS since BASIC authentication does not encrypt user credentials. The HTML UI connects to the Data Server via the rtvquery servlet. See Data Server for information on securing the connection between the rtvquery servlet and the Data Server. By default, the rtvquery servlet is not configured for authentication, but can be modified to require BASIC HTTP authentication as follows (this should be used with HTTPS since BASIC authentication does not encrypt user credentials):

1. Extract the web.xml file from the rtvquery servlet as follows:

   jar -xf rtview-tdgmon-rtvquery.war WEB-INF/web.xml

2. Open **WEB-INF/web.xml** in a text editor and uncomment the security section.
3. Pack the modified **web.xml** file back into the rtvquery servlet as follows:

   jar -uf rtview-tdgmon-rtvquery.war WEB-INF/web.xml

After you enable BASIC HTTP authentication in the rtvquery servlet, you will also need to modify the HTML UI to pass in credentials:

1. Extract the **setup.js** file from **rtview-tdgmon.war** as follows:

   jar -xf rtview-tdgmon.war setup.js

2. Open **setup.js** in a text editor and remove the **//** from the beginning of the **authValueC** line: **//authValueC: 'Basic ' + btoa('rtvuser:rtvuser')**
3. Pack the modified **setup.js** file back into the HTML UI servlet as follows:

   jar -uf rtview-tdgmon.war setup.js

Data Collectors

This process is optional and is used to distribute connections to Monitored Components Data Collectors intead of having the Data Server connect to each component to be monitored directly. This process collects data from Monitored Components and forwards it to the Data Server via socket or the rtvagent servlet. See Data Server for information about securing the connection between the Data Collector and Data Server. This process does not keep history or process alerts - those are handed in the Data Server. While the Data Collector typically does not have data clients, it accepts data requests via socket on port **3276** which can be secured as described in the Data Server section. It runs Jetty on port **3270** and also opens JMX on port **3266** for monitoring. By default, the JMX port is not secured. See Monitored Components for information on securing this connection.

Also see "Port Settings".

Configuration Application

The Configuration Application connects to the Data Server via the rtvadmin servlet which is configured with BASIC HTTP authentication. It should be run on HTTPS since Basic Authentication does not encrypt user credentials. Passwords saved by the configuration application are scrambled except in the case where they are added in the **CUSTOM PROPERTIES** section. See Data Server for information about securing the connection between the Configuration Application and Data Server.

Configuration Files

Configuration (**.properties** and **.properties.json**) files are stored on the file system and read by all RTView processes to control configuration. Additionally, the Configuration Application writes these files, scrambling all connection and database passwords. Passwords entered in the **CUSTOM PROPERTIES** tab are not scrambled.

Historian

The Historian connects to the Data Server via socket and saves cache history to a database via JDBC. This process is optional and the user can configure which data will be saved. See Data Server for information about securing the connection between the Historian and Data Server. See Database for information about the connection between the Historian and the database. This process opens JMX port **3267** for monitoring. By default, the JMX port is not secured. See Monitored Components for information on securing this connection.

Also see "Port Settings".

Database

The ALERTDEFS database stores alert threshold information and optionally alert persistence information. The Data Server connects to the ALERTDEFS database to query thresholds and also to set thresholds when the user interacts with the **Alert Administration** page in the user interface. The RTVHISTORY database stores cache data (if the Historian is enabled). The Historian connects to the RTVHISTORY database to insert cache history data and to perform data compaction. The Data Server connects to the RTVHISTORY database on startup to load initial history into the caches and also when the user interface asks for history data older than what is contained in the in-memory history caches.

By default, the Data Server and Historian connect to the HSQLDB database that is included with RTView using an unsecured JDBC connection. See the HSQLDB documentation for information on configuring it for secure JDBC connections. Alternately, you can use your own database and secure the JDBC connection according to the documentation for that database.

Application Servers

By default, the Data Server runs a Jetty process which hosts all of the RTView servlets and accepts HTTP client requests on port **3270**. You can optionally configure Jetty to use HTTPS instead of HTTP. This will require you to provide a certificate for your domain.

Also see "Port Settings".

When you have a certificate, do the following in the Configuration Application in the **Data Server** tab:

1. Turn on the **Use HTTPS** toggle.
2. Set the **Keystore File** to the keystore file name (including the path) that contains the certificate for your domain.
3. Optionally enter the **Keystore Password** and **Key Manager Password** if they are required for your keystore.
4. **Save** your configuration and restart the data server.

The Configuration Application and HTML UI use BASIC HTTP authentication and require the following roles which are preconfigured. You can modify the user names and passwords (but not the roles) in **RTVAPM_HOME/common/lib/ext/jetty/rtvadmin-users.xml**:

- rtvadmin
- rtvuser
- rtvalertmgr

Jetty does not limit the number of failed login attempts which leaves it open to brute force attacks. If this is a concern, you should deploy with Tomcat or another Application Server.

You can optionally use Tomcat or another Application Server in addtion to or instead of the Jetty process that comes with RTView. To deploy your servlets to your application server, go into the **RTVAPM__HOME/tdgmon/projects/sample** directory and run **update_wars.bat** or **update_wars.sh**. Copy all of the generated war files to the **webapps** directory in your application server.

Tomcat and most other Application Servers can be configured for HTTPS. This will require you to provide a certificate for your domain. Follow the application server instructions to enable HTTPS.

Additionally, Tomcat access filters can be configured to restrict access according to the remote client host or address. Tomcat also has a feature named LockOut Realm to protect against

brute force login attacks. After 5 successive login attempts for a given username with invalid password, then all logins for that username are rejected for the next 5 minutes. The LockOut Realm parameters are configurable. See Apache Tomcat documentation for more information.

You will need to add the following roles to your Application Server for use with the Configuration Application and HTML UI authentication. For Tomcat, users and roles are defined in **conf\tomcat-users.xml**:

- rtvadmin
- rtvuser
- rtvalertmgr

You can optionally disable Jetty in the Data Server when using Tomcat or another Application Server. To disable Jetty, you must access the Configuration Application from Tomcat or another Application Server. In the Configuration Application, go to the **Data Server** tab and do the following:

- Turn off the **HTML Server Enabled** toggle.
- **Save** your configuration and restart.

Monitored Components

Monitored Components are the processes that the Data Server and Data Collector connect to in order to request metric data. Some examples of Monitored Components are EMS Servers, Oracle Databases and RTView Processes. Connections to Monitored Components are made through application-specific APIs, so the options for securing these connections differ based on the Monitored Component.

This section contains:

- TIBCO FTL
- TIBCO ActiveSpaces

TIBCO FTL

Support for secure connections to TIBCO FTL was added in version 5.2. Previous versions of RTView do not support secure connections to TIBCO FTL. The Data Server connects to TIBCO FTL using the TIBCO FTL API. The TIBCO FTL Server can be configured to run with transport encryption and additionally with username/password authentication. If transport encryption is enabled, follow the instructions in the *TIBCO FTL Administration Guide* to create a trust file (certificate), which by default is named **ftl-trust.pem**. Copy this file into your **projects/rtview-server** directory, and also import it into your JVM keystore with a command such as:

keytool -alias ftl -file ftl-trust.pem -import -keystore $JAVA_HOME/jre/lib/security/cacerts -storepass changeit

In the RTView Configuration Application TIBCO FTL **Connection** dialog, use an HTTPS URL to connect to TIBCO FTL Servers with transport encryption enabled. If the TIBCO FTL server is configured with authentication, fill in the **Username** and **Password** fields.

TIBCO ActiveSpaces

The Data Server connects to the ActiveSpaces Data Grid using TIBCO FTL. See TIBCO FTL for instructions on securing those connections.

Security Summary

Security options per RTView process are included in the section for each component above. This section provides a summary of security options for the entire deployment organized by priority.

This section contains:

- Secure Installation Location - High Priority
- Login and Servlet Authentication - High Priority
- Application Server Security - High Priority
- Secure Connections between RTView Processes - Medium-to-Low Priority*
- Secure Connections to Monitored Components - Medium-to-Low Priority*
- Secure Connections to Monitored Components - Medium-to-Low Priority*

Secure Installation Location - High Priority

The RTView installation and Application Server should be run in a secure location to ensure displays and configuration files are secure and access-restricted.

Login and Servlet Authentication - High Priority

- **HTML UI** - By default, the HTML UI is configured with BASIC HTTP authentication which should use HTTPS since BASIC authentication does not encrypt user credentials. The HTML UI connects to the Data Server via the rtvquery servlet. The rtvquery servlet does not have authentication enabled by default. See the HTML UI section in this document for information on enabling authentication in the rtvquery servlet.
- **Configuration Application** - By default, the Configuration Application is configured with BASIC HTTP authentication which should use HTTPS since BASIC authentication does not encrypt user credentials.

Application Server Security - High Priority

It is highly recommended that you configure your Application Server to use HTTPS as described in the Application Servers section of this document. The RTView servlets that support HTTP authentication all use BASIC authentication which does not encrypt user credentials.

It is highly recommended that you change the user credentials in your Application Server for the rtvadmin, rtvuser and rtvalertmgr roles since the default credentials are documented and publicly available.

Secure Connections between RTView Processes - Medium-to-Low Priority*

The Historian, Data Server, Data Collector, rtvquery servlet, rtvdata servlet, rtvadmin servlet and rtvagent servlet all connect to the Data Server via socket which is unsecured by default. The Data Server supports secure socket connections (SSL) with or without certificates. It also supports client whitelist and blacklist. Secure socket and client whitelist/blacklist configuration are described in the RTView Core User's Guide under **Deployment/Data Server/Security**.

Secure Connections to Monitored Components - Medium-to-Low Priority*

The Data Server uses component specific API's to connect to Monitored Components. See the Monitored Components section in this document for information on how to secure these connections.

Secure Connections to Databases - Medium-to-Low Priority*

The Data Server and Historian both create database connections using JDBC. See the Database section in this document for information on securing JDBC connections to your database.

*If Secured Installation Location has been met, these are lower priority.