

TIBCO Administrator™

Server Configuration Guide

Software Release 5.10
August 2015

Two-Second Advantage®



Important Information

SOME TIBCO SOFTWARE EMBEDS OR BUNDLES OTHER TIBCO SOFTWARE. USE OF SUCH EMBEDDED OR BUNDLED TIBCO SOFTWARE IS SOLELY TO ENABLE THE FUNCTIONALITY (OR PROVIDE LIMITED ADD-ON FUNCTIONALITY) OF THE LICENSED TIBCO SOFTWARE. THE EMBEDDED OR BUNDLED SOFTWARE IS NOT LICENSED TO BE USED OR ACCESSED BY ANY OTHER TIBCO SOFTWARE OR FOR ANY OTHER PURPOSE.

USE OF TIBCO SOFTWARE AND THIS DOCUMENT IS SUBJECT TO THE TERMS AND CONDITIONS OF A LICENSE AGREEMENT FOUND IN EITHER A SEPARATELY EXECUTED SOFTWARE LICENSE AGREEMENT, OR, IF THERE IS NO SUCH SEPARATE AGREEMENT, THE CLICKWRAP END USER LICENSE AGREEMENT WHICH IS DISPLAYED DURING DOWNLOAD OR INSTALLATION OF THE SOFTWARE (AND WHICH IS DUPLICATED IN THE LICENSE FILE) OR IF THERE IS NO SUCH SOFTWARE LICENSE AGREEMENT OR CLICKWRAP END USER LICENSE AGREEMENT, THE LICENSE(S) LOCATED IN THE "LICENSE" FILE(S) OF THE SOFTWARE. USE OF THIS DOCUMENT IS SUBJECT TO THOSE TERMS AND CONDITIONS, AND YOUR USE HEREOF SHALL CONSTITUTE ACCEPTANCE OF AND AN AGREEMENT TO BE BOUND BY THE SAME.

This document contains confidential information that is subject to U.S. and international copyright laws and treaties. No part of this document may be reproduced in any form without the written authorization of TIBCO Software Inc.

TIBCO, Two-Second Advantage, TIBCO Hawk, TIBCO Rendezvous, TIBCO Runtime Agent, TIBCO ActiveMatrix BusinessWorks, TIBCO Administrator, TIBCO Designer, TIBCO ActiveMatrix Service Gateway, TIBCO BusinessEvents, TIBCO BusinessConnect, and TIBCO BusinessConnect Trading Community Management are either registered trademarks or trademarks of TIBCO Software Inc. in the United States and/or other countries.

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

All other product and company names and marks mentioned in this document are the property of their respective owners and are mentioned for identification purposes only.

THIS SOFTWARE MAY BE AVAILABLE ON MULTIPLE OPERATING SYSTEMS. HOWEVER, NOT ALL OPERATING SYSTEM PLATFORMS FOR A SPECIFIC SOFTWARE VERSION ARE RELEASED AT THE SAME TIME. SEE THE README FILE FOR THE AVAILABILITY OF THIS SOFTWARE VERSION ON A SPECIFIC OPERATING SYSTEM PLATFORM.

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

THIS DOCUMENT COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN; THESE CHANGES WILL BE INCORPORATED IN NEW EDITIONS OF THIS DOCUMENT. TIBCO SOFTWARE INC. MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED IN THIS DOCUMENT AT ANY TIME.

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

Copyright © 1999-2015 TIBCO Software Inc. ALL RIGHTS RESERVED.

TIBCO Software Inc. Confidential Information

Contents

Figures	ix
Tables	xi
Preface	xiii
Changes from the previous Release of this Guide	xiv
Related Documentation	xv
TIBCO Administrator Documentation	xv
Other TIBCO Product Documentation	xv
.....	xvi
Typographical Conventions	xvii
Connecting with TIBCO Resources	xx
How to Join TIBCOCommunity	xx
How to Access TIBCO Documentation	xx
How to Contact TIBCO Support	xx
Chapter 1 Introduction	1
Architecture Overview	2
Appendix A An <i>administration domain</i> is a collection of users, machines, and services. A domain is managed by an administration server, which is assisted by a TIBCO Hawk agent running on each machine in the domain.	2
Administration Server	3
TIBCO Hawk Agent	4
LDAP Directory Server	4
Client Applications	4
Application Domains	5
Database and File-based Storage	6
Client Application Data Storage	7
Deployment Choices	9
Using Local Application Data	9
Using Server-based Application Data	10
Security Considerations	12
Access to Domain Data	12
Domain Transport Security	13

Access to the TIBCO Administrator GUI	14
Deployed Applications	14
Domain Password Policy	15
Administration Server Shutdown String	15
Password Policy	16
Changing the Password Policy for Existing Domains	16
Distributing Changed Passwords	17
Domains Integrated with an LDAP Directory	17
Password Policy Choices	17
Chapter 2 Fault Tolerance and Load Balancing in a Rendezvous Administration Domain . . .	21
Overview	22
Multiple Secondary Administration Servers	23
Load-Balanced Servers	25
One Server in One Subnet	27
Options	27
Multiple Load-Balanced Servers in a Subnet	28
Options	29
Multiple Administration Domains in a Subnet	30
Multiple Administration Domains	30
Multiple Servers Using Different Daemon and Service Properties	31
Administration Domain Spanning Subnets	32
Introduction	32
Setup Tasks	33
Troubleshooting	41
Multiple Load-Balanced Servers Spanning Subnets	42
Chapter 3 Fault Tolerance and Load Balancing in a EMS-Based Administration Domain . . .	45
Overview	46
Multiple Secondary Administration Servers	46
Creating a Secondary Administration Server	48
To Add a Secondary Server Using the GUI	48
To Add a Secondary Server Using the Command Line Utility	49
Secondary Server Fields	49
Removing a Secondary Administration Server	54
To Remove a Secondary Server Using the GUI	54
To Remove a Secondary Server Using the Command Line Utility	54
Specifying Fault Tolerance EMS Servers	55
Chapter 4 Administration Server Properties File	57
Introduction	58

General Properties.	59
File Storage Properties	63
Database Storage Properties	64
TIBCO Rendezvous Properties	66
Logging Properties.	67
Security Audit Properties	70
Load Balancing Properties.	71
Server Security Properties.	73
Persist Commit History Properties.	76
TIBCO Hawk Agent Properties	77
Chapter 5 Advanced Topics	79
Server-Based Repository Locator String	80
TIBCO Rendezvous	80
HTTP and HTTPS	82
Local Repository Locator String.	86
Using a Properties File to Encapsulate Locator String Components.	88
Configuring Connection Pool Size for the Database Server	89
Enabling an SSL Connection to an LDAP Directory Server.	90
Maintaining Connections to an LDAP Directory Server	91
Internationalization.	92
Introduction.	92
Design-Time Encoding vs. Runtime Encoding	92
Setting Encoding for an Administration Domain	93
Users and Passwords with Double-Byte Characters	95
Configuring EHCACHE	96
Writing Cache Statistics to a Log File.	97
Authenticating Users	98
Authentication Using the JAAS Authentication Web Service	98
Authentication Code Example	99
Logging Information to a File	100
Repository Locking	102
How to Break the Lock	102
Global Variables.	103
Defining Variables in TIBCO Designer	103
Defining Service-Settable Variables in TIBCO Designer	103
Setting Variable Values.	104

Chapter 6 Command Line Utilities 105

Command List 106

CorpRoleSynchronizer 108

DeleteInvalidUsers 109

CorpUserSynchronizer 110

ExportDomainSecurity 111

ImportDomainSecurity 116

MoveMachine 119

RedeployAllAppsForUser 123

RepoConvert 126

RepoPing 130

RepoCreateInstance 132

RepoDelete 133

RepoDeleteInstance 134

RepoDiff 135

RepoExport 137

RepoImport 139

RepoListInstances 142

RepoRename 143

AppStatusCheck 146

Appendix B Log Files Generated for an Administration Domain 149

Introduction 150

Installer Logs 151

Tomcat Logs 152

Administration Server Log 153

TIBCO Hawk Agent Logs 154

Application Logs 155

Audit Logs 156

TIBCO Domain Utility Log 157

Repository Logs 158

Standard TIBCO Trace Message Format 159

Appendix C Using TIBCO Hawk Methods 161

Introduction 162

Starting TIBCO Hawk Software 163

Starting TIBCO Hawk Software on Unix	163
Starting TIBCO Hawk Software on Microsoft Windows	163
The Auto-Discovery Process	164
Invoking Microagent Methods	165
Available Microagent Methods	167
onCopyInstance()	169
onCreateInstance()	170
onDeleteInstance()	171
onModifyInstance()	172
onServerShutdown()	173
onStartInstance()	174
onStopInstance()	175
onInstanceChange()	176
getConfigProperties()	177
getRvConfig()	179
getConfig()	180
getStatus()	181
getQueueStats()	183
getLogConfig()	184
getTracelds()	186
onContentChange()	187
setLogConfig()	188
setTracelds()	190
doShutdownServer()	191
getInstanceInfos()	192
getInstanceStatus()	194
getSizeOfInstance()	195
getListOfLockedNodes()	196
getConnectedClients()	197
doStopInstance()	198
doStartInstance()	199
doRefreshCache()	200
doRefreshSecurity()	201
doTerminateClient()	202
Appendix D Schema Files used when Exporting and Importing	203
XML Formats Used by TIBCO Projects	204
Schema Files Included With TIBCO Runtime Agent	205
Using RepoExport and RepoImport	206
Creating a New Adapter Configuration Schema	208
Including Custom Schemas During Import	209

Appendix E System Messages 211

AEREPO Messages 212

POF Messages 246

POFUTIL Messages 274

POOL Messages 280

PLUG-IN Messages 283

Index 287

Figures

Figure 1	Components in Administration Domain	2
Figure 2	One Server in a Single Subnet.	27
Figure 3	Multiple Load-Balanced Servers in One Administration Domain	28
Figure 4	Multiple Domains in a Single Subnet	31
Figure 5	Setup of rvrd for Subject "com.tibco.repo.>"	34
Figure 6	Enter Name of the Router	36
Figure 7	Configure Local and Neighbor Interfaces.	37
Figure 8	Add Local Network Interface	38
Figure 9	Specify the Subject to be Used	39
Figure 10	Router Configuration	40
Figure 12	Circular Role Hierarchy	111
Figure 13	Discovered Agents	164
Figure 14	The Microagents, Methods and Arguments Dialog Displays	165
Figure 15	Select Microagent Name	166

Tables

Table 1	General Typographical Conventions	xvii
Table 2	Syntax Typographical Conventions	xviii
Table 3	Domain Setup	12
Table 4	Domain Transport Security	13
Table 5	Application Deployment Options	14
Table 6	Service and Daemons for Machine	33
Table 7	Secondary Server Fields	49
Table 8	Optional properties for server-based locator string (TIBCO Rendezvous)	80
Table 9	Optional properties for server-based locator string (HTTP)	82
Table 10	Properties Required for HTTPS urlFiles	84
Table 11	Optional properties for local repository locator string	86
Table 12	Command List	106
Table 13	ExportDomainSecurity Properties	112
Table 14	ImportDomainSecurity Properties	116
Table 15	MoveMachine Property	120
Table 16	RedeployAllAppsForUser Property	124
Table 17	RepoConvert Property	126
Table 18	RepoPing Property	130
Table 19	RepoCreateInstance Parameter	132
Table 20	RepoDelete Parameter	133
Table 21	RepoDeleteInstance Parameters	134
Table 22	RepoDiff Parameter	135
Table 23	RepoExport Parameter	137
Table 24	RepoImport Parameter	139
Table 25	RepoListInstances Parameters	142
Table 26	RepoRename Parameters	143
Table 27	AppStatusCheck Parameters	146
Table 28	Trace Message Data	159

Preface

This document provides server configuration information for administrator users of TIBCO Administrator™. *TIBCO Administrator User's Guide* explains how to get started with TIBCO Administrator.

Topics

- [Changes from the previous Release of this Guide, page xiv](#)
- [Related Documentation, page xv](#)
- [Typographical Conventions, page xvii](#)
- [Connecting with TIBCO Resources, page xx](#)

Changes from the previous Release of this Guide

There are no changes from the previous release of this guide.

Related Documentation

This section lists documentation resources you may find useful.

TIBCO Administrator Documentation

The following documents form the *TIBCO Administrator*[™] documentation set:

- *TIBCO Administrator*[™] *Installation* Read this manual for instructions on site preparation and installation.
- *TIBCO Administrator*[™] *User's Guide* Read this manual for instructions on using the product to manage users, resources, and applications inside an administration domain.
- *TIBCO Administrator*[™] *Server Configuration Guide* Read this manual for instructions on using the administration server to manage projects and repositories, using command-line tools, performing conversions, and so on. The manual is written primarily for system administrators.
- *TIBCO Administrator*[™] *Release Notes* Read the release notes for a list of new and changed features. This document also contains lists of known issues and closed issues for this release.

Other TIBCO Product Documentation

You may find it useful to read the documentation for the following TIBCO products:

- *TIBCO Runtime Agent*[™] : TIBCO Runtime Agent is a bundle of TIBCO software and third-party software that is needed to run many TIBCO applications such as TIBCO ActiveMatrix BusinessWorks and TIBCO Adapters.
- *TIBCO Designer*[™]: This graphical user interface is used for designing and creating integration project configurations and building an Enterprise Archive (EAR) for the project. The EAR can then be used by TIBCO Administrator for deploying and running the application.
- *TIBCO Hawk*[®]: This is a tool for monitoring and managing distributed applications and operating systems.
- *TIBCO Rendezvous*[®]: Rendezvous enables programs running on many different kinds of computers on a network to communicate seamlessly. It includes two main components: the Rendezvous application programming interface (API) in several languages, and the Rendezvous daemon.

- TIBCO Enterprise Message Service™: This software lets application programs send and receive messages using the Java Message Service (JMS) protocol. It also integrates with TIBCO Rendezvous and TIBCO SmartSockets® messaging products.
- TIBCO ActiveMatrix BusinessWorks™: ActiveMatrix BusinessWorks is a scalable, extensible, and easy to use integration platform that allows you to develop integration projects. ActiveMatrix BusinessWorks includes a GUI for defining business processes and an engine that executes the process.
- TIBCO® Adapter software: TIBCO Runtime Agent is a prerequisite for TIBCO Adapter products. You will therefore find TIBCO Adapter product documentation useful.

Typographical Conventions

The following typographical conventions are used in this manual.

Table 1 General Typographical Conventions

Convention	Use
<i>ENV_NAME</i> <i>TIBCO_HOME</i> <i>TRA_HOME</i>	<p>TIBCO products are installed into an installation environment. A product installed into an installation environment does not access components in other installation environments. Incompatible products and multiple instances of the same product must be installed into different installation environments.</p> <p>An installation environment consists of the following properties:</p> <ul style="list-style-type: none"> • Name Identifies the installation environment. This name is referenced in documentation as <i>ENV_NAME</i>. On Microsoft Windows, the name is appended to the name of Windows services created by the installer and is a component of the path to the product shortcut in the Windows Start > All Programs menu. • Path The folder into which the product is installed. This folder is referenced in documentation as <i>TIBCO_HOME</i>. <p><i>TIBCO Administrator</i> installs into a directory within a <i>TIBCO_HOME</i>. This directory is referenced in documentation as <ProductAcronym>_HOME. The default value of <ProductAcronym>_HOME depends on the operating system. For example on Windows systems, the default value is C:\tibco\<ProductAcronym>\<ReleaseNumber>.</p>
code font	<p>Code font identifies commands, code examples, filenames, pathnames, and output displayed in a command window. For example:</p> <p>Use MyCommand to start the foo process.</p>
bold code font	<p>Bold code font is used in the following ways:</p> <ul style="list-style-type: none"> • In procedures, to indicate what a user types. For example: Type admin. • In large code samples, to indicate the parts of the sample that are of particular interest. • In command syntax, to indicate the default parameter for a command. For example, if no parameter is specified, MyCommand is enabled: MyCommand [enable disable]

Table 1 General Typographical Conventions (Cont'd)




Convention	Use
<i>italic font</i>	<p>Italic font is used in the following ways:</p> <ul style="list-style-type: none">• To indicate a document title. For example: See <i>TIBCO ActiveMatrix BusinessWorks Concepts</i>.• To introduce new terms For example: A portal page may contain several portlets. <i>Portlets</i> are mini-applications that run in a portal.• To indicate a variable in a command or code syntax that you must replace. For example: <code>MyCommand <i>PathName</i></code>
Key combinations	<p>Key name separated by a plus sign indicate keys pressed simultaneously. For example: <code>Ctrl+C</code>.</p> <p>Key names separated by a comma and space indicate keys pressed one after the other. For example: <code>Esc, Ctrl+Q</code>.</p>
	<p>The note icon indicates information that is of special interest or importance, for example, an additional action required only in certain circumstances.</p>
	<p>The tip icon indicates an idea that could be useful, for example, a way to apply the information provided in the current section to achieve a specific result.</p>
	<p>The warning icon indicates the potential for a damaging situation, for example, data loss or corruption if certain steps are taken or not taken.</p>

Table 2 Syntax Typographical Conventions

Convention	Use
[]	<p>An optional item in a command or code syntax.</p> <p>For example:</p> <p><code>MyCommand [optional_parameter] required_parameter</code></p>
	<p>A logical OR that separates multiple items of which only one may be chosen.</p> <p>For example, you can select only one of the following parameters:</p> <p><code>MyCommand param1 param2 param3</code></p>

Table 2 *Syntax Typographical Conventions (Cont'd)*

Convention	Use
{ }	<p>A logical group of items in a command. Other syntax notations may appear within each logical group.</p> <p>For example, the following command requires two parameters, which can be either the pair param1 and param2, or the pair param3 and param4.</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command requires two parameters. The first parameter can be either param1 or param2 and the second can be either param3 or param4:</p> <pre>MyCommand {param1 param2} {param3 param4}</pre> <p>In the next example, the command can accept either two or three parameters. The first parameter must be param1. You can optionally include param2 as the second parameter. And the last parameter is either param3 or param4.</p> <pre>MyCommand param1 [param2] {param3 param4}</pre>

Connecting with TIBCO Resources

How to Join TIBCOCommunity

TIBCOCommunity is an online destination for TIBCO customers, partners, and resident experts. It is a place to share and access the collective experience of the TIBCO community. TIBCOCommunity offers forums, blogs, and access to a variety of resources. To register, go to <http://www.tibcommunity.com>.

How to Access TIBCO Documentation

You can access TIBCO documentation here:

<http://docs.tibco.com>

How to Contact TIBCO Support

For comments or problems with this manual or the software it addresses, contact TIBCO Support as follows:

- For an overview of TIBCO Support, and information about getting started with TIBCO Support, visit this site:

<http://www.tibco.com/services/support>

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

<https://support.tibco.com>

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

Chapter 1 **Introduction**

This chapter introduces the administration server that is used to manage applications placed in an administration domain.

Topics

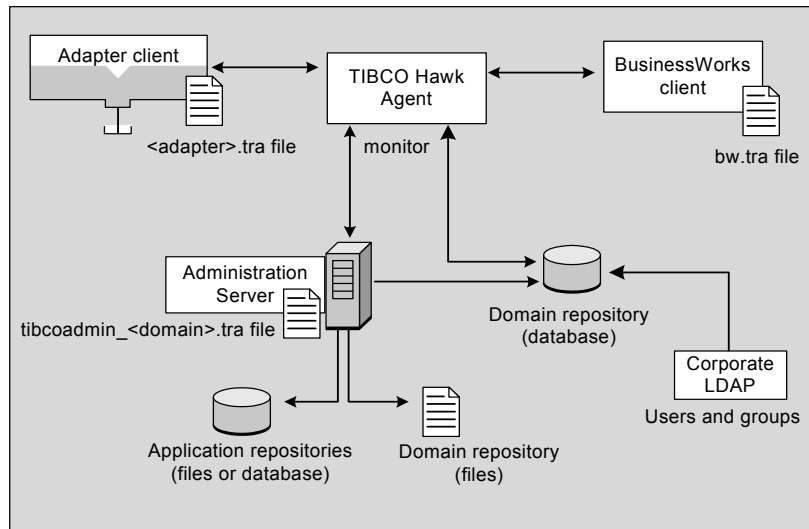
- [Architecture Overview, page 2](#)
- [Database and File-based Storage, page 6](#)
- [Deployment Choices, page 9](#)
- [Security Considerations, page 12](#)
- [Password Policy, page 16](#)

Architecture Overview

An *administration domain* is a collection of users, machines, and services. A domain is managed by an administration server, which is assisted by a TIBCO Hawk agent running on each machine in the domain.

The next diagram shows the basic components in an administration domain. TIBCO Hawk agent uses microagents and rulebases to monitor activity among the components in the administration domain. Depending on the transport used for inter-domain communication, domain information can be stored in a database or in files.

Figure 1 Components in Administration Domain



Administration Server

An administration server manages resources in an administration domain. A machine can have multiple domains, with each domain assigned a unique server. Each time a domain is created, a new server is created for the domain and named with the domain name.

When configuring an administration server, you specify the transport to use for managing communications in the administration domain. Either TIBCO Rendezvous or TIBCO Enterprise Message Service can be used as the transport.

If TIBCO Rendezvous is used as the transport in an administration domain, you can create secondary servers to load balance client-server activity in the domain. In this case, the initial administration server is known as a primary server and other servers are known as secondary servers. Multiple secondary servers are allowed in the same administration domain, but each must be on a different machine. The machine that hosts the primary server cannot also host a secondary server.

If TIBCO Enterprise Message Service is used as the transport, there is no need for secondary servers. (You can create secondary servers, but they are just like primary servers.) The only deployment option is Local, which means that client applications run independent of the administration server (unless an application performs certain operations that require connecting to the administration server, such as HTTP and SOAP authentication). See [Deployment Choices on page 9](#) for more information.

The administration server runs under the Tomcat web server as a servlet. Tomcat ports are defined when using TIBCO Domain Utility to configure the administration domain. The ports can be changed later using Domain Utility.

Data about the machines, registered software, users, roles, access control lists, application configurations and deployment history is maintained for an administration domain. If you have configured a domain to use the TIBCO Enterprise Message Service as the transport, domain storage must be in a database. Domains configured to use TIBCO Rendezvous can store data in a file repository or in database. See [Database and File-based Storage on page 6](#) for details.

Each administration domain contains one or more machines. You add a machine to a domain using TIBCO Domain Utility. You can start Domain Utility at any time to manage domain machines, domain configurations, administration server settings, upgrade domains and register a TIBCO Enterprise Message Service server.

TIBCO Hawk Agent

TIBCO Hawk agent is part of the TIBCO Runtime Agent package and is installed on each machine in an administration domain. TIBCO Hawk agent monitors local resources and conditions. An Agent uses collections of locally loaded rules organized into rulebases to apply monitoring logic.

The Hawk agent also builds local client .tra and .exe files when a client application is deployed, and on Windows platforms, creates NT services for applications.

The administration server retrieves the monitoring information displayed in the TIBCO Administrator GUI from a Hawk Agent, which runs as a separate process on each machine in the domain.

The TIBCO Administrator GUI provides a dialog to configure rulebases, set monitoring options, and display status. See the *TIBCO Administrator User's Guide* for more information.

LDAP Directory Server

You can integrate an administration domain with an LDAP directory server to use the users and groups defined in the LDAP directory. The groups then become visible in TIBCO Administrator GUI. As part of domain creation, the TIBCO Domain Utility prompts for the information to retrieve, the synchronization interval, and other options.



The administration server only retrieves information from an LDAP directory server, it never sends or stores information there.

Client Applications

Applications can communicate directly with the administration server when configuration data for the application is stored in the administration domain repository. The administration server can also send application data directly to the machine on which that application is running, and the application will then work independently of the administration server. If you are using TIBCO Enterprise Message Service as the domain transport, application data is always sent to the local machine where the application is running.

Associated with each client application is a `.tra` file, which determines certain aspects of how the application is run. The `.tra` file is created during deployment and available on the deployment machine in the `TIBCO_TRA_DOMAIN_HOME/domain-name/application/application-name` directory.



You can manually edit the client application's `.tra` file, however, each time you deploy (or redeploy) the corresponding client application, TIBCO Administrator overwrites the `.tra` file. Using the TIBCO Administrator GUI to manage `.tra` file content and not editing the file is therefore highly recommended.

Programmatically, applications such as adapters use a project locator string to specify location and protocol information. See [Server-Based Repository Locator String on page 80](#) for details.

Application Domains

If your TIBCO application supports this feature, you can create multiple application domains and assign client applications to use them. An application domain allows you to specify a repository to hold application data only. This is useful, for example, when your applications need to use a local database rather than that used by the administration domain.

Database and File-based Storage

An administration domain stores domain information in a database repository or file repository, depending on the transport type. If TIBCO Rendezvous is set as the transport, you can store domain data in a database repository or file repository. If TIBCO Enterprise Message Service is set as the transport, you can only store domain data in a database repository.

In most cases, it is recommended to use a database for the domain repository and files for client application repositories.

- In the case of a file-based domain, domain data is stored in the `SYS_domain.dat` and `AUTH_domain.dat` files. Data stored in the `SYS_domain.dat` file is referred to as the administration domain while data stored in the `AUTH_domain.dat` file is referred to as the authorization domain.

The authorization domain contains the users, roles and data access ACLs. Everything else is stored in the administration domain: installed software, machines, applications, plug-ins, TIBCO Administrator ACLs, and so on. As such, the administration domain file is usually much larger than the authorization domain file.



Editing these files is *not* recommended. It can potentially lead to loss of all domain data and to unpredictable behavior of TIBCO Administrator.

In a file-based domain, the Hawk agent accesses the domain repository through the administration server.

- If a database is used, the administration domain is stored in tables beginning with `Ad`, while authorization domain data is stored in tables beginning with `Au`.

In a database-based domain, the Hawk agent accesses the domain repository directly and does not go through the administration server.

Performance can improve if domain data is saved into a database. During deployment, there are a number of steps during which domain data is committed. If domain data is stored in files, the files are loaded at startup and each time a change to the domain is saved, the contents of the corresponding file must be regenerated and the file saved. Each commit for a file-based domain requires writing the entire file, which is expensive both because of I/O and calculating the large number of individual strings.

When using a file-based repository with a primary and secondary administration server, performance will be slower than when using a database to store administration domain data.

When using a database, only the changed data is written. In addition to faster performance when saving domain data, there also is significant memory savings. Domain data will not be loaded and cached by TIBCO Administrator in the way that application data is. Note that moving application data to a database has no such savings and will slightly increase the server startup and deployment times.

To migrate from a file-based domain to a database-based domain, simply create a new domain with database settings, then use the AppManage utility to export your applications from the old domain and import them into the new domain.

Restrictions on Database Storage

Database storage has the following restrictions:

- If two differently named administration servers use database storage and connect to the same database instance, you must create separate user accounts for each server. This prevents the servers from accessing the same set of database tables simultaneously.
- If applications that store data in databases are used inside an administration domain, the domain data must also be stored in a database. Note that the reverse is not true. Domain data can be stored in a database when application repositories are stored in files.



Storing application data in databases has been deprecated with this release. It is recommended that you use the standard options for application deployment.

- In a database-based domain, the TIBCO Hawk agents on all client machines *must* have database connectivity because they access the domain repository directly instead of accessing it through the administration server.

Client Application Data Storage

TIBCO Administrator creates application repositories when you deploy an application. For information about the deployment process, see the *TIBCO Administrator User's Guide*.

- Application configurations are created using TIBCO Designer and are included in the enterprise archive file that is imported into TIBCO Administrator. The enterprise archive file contains information about the services and client application to run, such as TIBCO BusinessWorks processes and TIBCO Adapter services.
- The deployment process creates client application data in the administration server or sends the data to the application's target machine(s). If TIBCO

Enterprise Message Service is used as the transport, application data is always sent to the application's target machine.

- When application data is sent to the target machines, the application runs independently of the administration server (unless it performs operations that require connecting to the administration server in a *file-based* domain, such as HTTP or SOAP authentication).
- Security for access to application repositories is set using the TIBCO Administrator GUI.

Deployment Choices

When configuring an administration domain, you can set how the administration server creates and stores application data. This section explains the choices available and discusses their implications. You have the following choices when configuring an administration domain that uses TIBCO Rendezvous. If TIBCO Enterprise Message Service is used, only the local application data choice is available. Server-based application data is not available.

- [Using Local Application Data](#)
- [Using Server-based Application Data](#)

For both choices, a directory is created for each deployed application on the target machine under `TIBCO_TRA_DOMAIN_HOME\domain-name\application`. The executables, property files and deployment files required for the application are stored under this directory.

The `TIBCO_TRA_DOMAIN_HOME\domain-name\datafiles` directory is also created as needed and only for local deployment. This directory contains the project files used by an application's non adapter components such as TIBCO BusinessWorks. Adapter repository files (.dat) are stored in the `TIBCO_TRA_DOMAIN_HOME\domain-name\data` directory.

Using Local Application Data

When using TIBCO Domain Utility to create an administration domain, you can select the Local Application Data option. When the option is chosen, deployment files for an application are sent to target machines. This allows the application to run independently of the administration server (unless an application performs operations that require connecting to the administration server in a *file-based* domain, such as HTTP or SOAP authentication). Selecting the option makes it the default in the domain. For domains that use TIBCO Rendezvous, you can change the option per application to use Rendezvous, or HTTP/HTTPS when deploying the application in the TIBCO Administrator GUI or the appManage utility.

Choosing the Local Application Data option results in the following benefits:

- Because the local service instances use no memory, thread or CPU resources on the administration server, the solution is infinitely scalable.
- At runtime, because there is no communication between the deployed application and the administration server, a server failure has no effect on the runtime application (unless an application performs certain operations that require connecting to the administration server in a *file-based* domain, such as HTTP or SOAP authentication).

- Application data for TIBCO BusinessWorks is stored in its native format as a multifile project, while adapters use their native format . (dat files). The required storage format is determined based on the application's components installed on the target machine. For example, if an application uses a TIBCO Adapter, a .dat file is written to the target machine. If the application uses TIBCO BusinessWorks, a project file is written to the target machine.
- For server-based domains, TIBCO BusinessWorks applications must do an in-memory transformation at startup.

Issues with this Choice

The main issue with this choice is synchronization. If deployment is done cleanly, all client machines will remain synchronized as appropriate processes are stopped and each will receive updated deployment files. However, if a subset of client machines are unavailable during deployment and later started, their service instances, metadata and configurations can be out of sync. This could result in severely aberrant behavior.



When using this option, you must ensure that each application deploys successfully. If an application does not deploy successfully, a warning message will appear in the TIBCO Administrator GUI Deployed Configuration pane.

The person responsible for the application should ensure that the components on the failed machine are not started until they are successfully updated.

The other significant concern is security. Repository instances can contain sensitive information such as database or application user ids and passwords. While the passwords are encrypted, the encryption algorithm used is 3DES. To protect this data, you must provide strong file system security on each machine to which applications are deployed.

Using Server-based Application Data

For domains using TIBCO Rendezvous as the transport, the server-based application data option is available. When deploying an application with this option set, a repository instance for the application is created on the administration server and the repository instance is referenced in each application's instance's property file (.tra file).

- Two transport protocols are available for server-based communication, TIBCO Rendezvous, and HTTP/HTTPS. The default transport option is configured using TIBCO Domain Utility when creating the domain. The utility can be run later to change the default transport option for a domain. The transport can be changed when deploying an application, either in the TIBCO Administrator

GUI or with the appManage utility. This allows one application in the domain to use Rendezvous, another application to use HTTP and another to use the local deployment option.



If you configure an administration server for HTTPS, that server's GUI will not be accessible via HTTP. You can, however, access the same server programmatically using both HTTP and HTTPS.

You must use TIBCO Rendezvous or the local deployment option to connect to a version 4.x TIBCO Adapter. HTTP or HTTPS cannot be used.

Issues with this Choice

Using this choice, there are no significant issues related to synchronization.

The main concern with this approach is that each repository instance on the administration server consumes memory and threads. The memory requirement is 3-5 times the size of the `.dat` file, which can range in size up to tens of megabytes. With many applications starting simultaneously, the server could become overwhelmed causing time-outs and application failures. Other issues include increased network traffic, and maintaining network and administration security.

Another issue is related to fault tolerance. Runtime applications can not start, and in some cases can not function without being able to communicate with the administration server. This makes the administration server a single point of failure at runtime. The server does have some fault tolerance with the secondary servers, but deployment can be significantly slowed when secondary servers are used.

Security Considerations

This section outlines the security issues to consider when mapping out an administration domain. Read this section carefully and take all necessary security precautions if the environment in which you are setting up an administration domain is not secure. Examples of an insecure environment include:

- The subnet in which the domain operates is accessed by persons of varying security clearance.
- Elements of the domain (such as deployed applications) exist outside of the intranet firewall.
- The domain spans subnets that are configured with different levels of security.
- Communication between domain elements overlaps insecure network environments, such as the internet.

Access to Domain Data

Data storage locations and access methods may differ depending on your domain setup. Note that administration servers, Hawk agents, and deployed applications act as clients to the domain data and access it *directly*.

Consult the table below to make sure that all connections in the course of data access are secure for your setup.

See [Domain Transport Security on page 13](#) for recommendations on transport security.

Table 3 Domain Setup

Domain Setup	Data Storage and Access	Security Considerations
File-based without LDAP	Data kept on the administration server host and accessed using Rendezvous or Enterprise Message Service.	Consider changing to LDAP integration with SSL for better security of authentication tasks.
File-based with LDAP	Data kept on the administration server host and accessed using Rendezvous or Enterprise Message Service. Authentication delegated to the LDAP server, which is accessed directly.	LDAP connection can be secured with SSL. Note: Access to users and roles that are created in the TIBCO Administrator GUI <i>cannot</i> be secured by SSL.

Table 3 Domain Setup

Domain Setup	Data Storage and Access	Security Considerations
Database-based without LDAP	Data kept on the database server and accessed directly.	Database connection can be secured with SSL. Alternatively, consider changing to LDAP integration with SSL for better security of authentication tasks.
Database-based with LDAP	Data kept on the database server and accessed directly. Authentication delegated to the LDAP server, which is accessed directly.	Database connection can be secured with SSL. LDAP connection can be secured with SSL. Note: Access to users and roles that are created in the TIBCO Administrator GUI <i>can</i> be secured by securing the database connection.

Domain Transport Security

Two options are available for domain transport: Rendezvous or Enterprise Message Service.

Table 4 Domain Transport Security

	Rendezvous	Enterprise Message Service
Underlying Protocol	TCP between client and daemon (rvd) UDP between daemons within subnet (rvd) TCP between daemons spanning subnets (rvrd)	TCP
Security Options	No end-to-end security available: <ul style="list-style-type: none"> TCP traffic can be secured using Rendezvous secure daemon (rvsd). Traffic between subnets can be secured using Rendezvous secure routing daemon (rvrsd). UDP broadcast within subnet is not secure. 	End-to-end security available by enabling SSL.

Table 4 Domain Transport Security

	Rendezvous	Enterprise Message Service
Recommendation	Safe to use in a benign subnet or across subnets that are benign. Note that password-type data is always encrypted.	Safe to use in all environments.

Access to the TIBCO Administrator GUI


The TIBCO Administrator GUI enables you to access and manipulate sensitive domain data and activity, including user and deployment management. When you access the administration server using the TIBCO Administrator GUI, all information is transmitted over the HTTP protocol by default. This is not secure when you access the TIBCO Administrator GUI from an insecure subnet or the Internet.

You can secure your access to the TIBCO Administrator GUI by enabling HTTPS in TIBCO Domain Utility. See *TIBCO Runtime Agent Domain Utility User's Guide* for instructions.

Deployed Applications

The following table makes recommendations on different application deployment options. When an application is deployed, the deployment operation uses the domain transport. When the application is running, it accesses application data using the transport option you select at the time of deployment. See the table below for recommendations on the transport options for application data access.

You can select HTTPS as a transport option for application data if you enable HTTPS in TIBCO Domain Utility. See *TIBCO Runtime Agent Domain Utility User's Guide* for instructions.



An application may perform basic authentication tasks (for example, authenticating an application user) and may need to access domain data. See [Access to Domain Data on page 12](#) for security recommendations.

Table 5 Application Deployment Options

Deployment Option	Transport Option	Security Considerations
Local Application Data	Local	The most secure option, since no network traffic is necessary to access application data.

Table 5 Application Deployment Options

Deployment Option	Transport Option	Security Considerations
Server-based Application Data	<ul style="list-style-type: none"> • HTTP • HTTPS • Rendezvous <p>Note: Enterprise Message Service requires that applications are deployed as Local Application Data.</p>	Use HTTPS to access application data securely.

Domain Password Policy

See [Password Policy on page 16](#) for more information.

Administration Server Shutdown String

While you can shutdown the administration server remotely via the shutdown port, you should define a shutdown string to protect the shutdown port. This shutdown string is not considered to be a password and is not governed by the password policy. However, it is best to specify a string that can pass the restrictive policy (see [Stopping the Administration Server on page 19](#) in *TIBCO Administrator User's Guide* for more information).

Password Policy

When using TIBCO Domain Utility to configure an administration domain, you can specify whether to enforce a password policy in the domain. Doing so allows the applications running in the domain to meet the fundamental security requirements defined in the Sarbanes-Oxley Act. For example, to meet Sarbanes-Oxley security compliance, a password must meet the following policy:

- Minimum Password Length — forcing a password to be at least a certain number of characters in length, for example, eight characters.
- Password Complexity — forcing a password to include letters, numbers, and special characters such as punctuation.
- Password Aging — forcing a password to be changed after a certain time period has passed, for example, three months.
- Forced Initial Password Change — forcing a password to be changed on a user's first login.
- Account Lockout — disabling a user account after a certain number of failed login attempts, for example, after five attempts.
- Password History — forcing a certain number of recently used passwords to be remembered so that they cannot be used again, for example, not allowing the past four passwords to be used.

The password policy applies to user account passwords that allow access to TIBCO Administrator GUI modules and application passwords that are used to login to applications that have been deployed using the TIBCO Administrator GUI.

When a user is deleted, the user's password history is also deleted.



The password policy that you configure in TIBCO Domain Utility applies to all users and groups in the administration domain. You should use an LDAP directory server if you wish to customize password policies for different users and groups.

Changing the Password Policy for Existing Domains

A user with write permission to the User console in the TIBCO Administrator GUI can change the password policy that was set when the administration domain was created. See the *TIBCO Administrator User's Guide* for details about changing the password policy.

Distributing Changed Passwords

After a user changes his or her password successfully, the newly changed password must be distributed to all deployed applications on target machines, so that each application can use the newly changed password to login in an unattended mode. The easiest way to distribute changed passwords is to use the `RedeployAllAppsForUser` utility. See [RedeployAllAppsForUser on page 123](#) for details.

Domains Integrated with an LDAP Directory

If your administration domain is integrated with an LDAP directory, you can have both LDAP users and local users (users defined directly in TIBCO Administrator GUI). The password policy applies only to local users and does not apply to LDAP users.

Password Policy Choices

This section lists the password policies that can be applied to an administration domain using TIBCO Domain Utility. The password policy is set when a domain is created and can be changed later in the TIBCO Administrator GUI Users console.

No Policy

This choice allows an administration domain to be created with no policy enforced for passwords. This allows user accounts to be created in the TIBCO Administrator GUI without assigning passwords. If passwords are assigned, they will not expire. A user can attempt any number of logons without having the account locked out.

Default Policy

If selected, the following password policy is enforced. A password:

- Will be saved in encryption mode, and the algorithm used is 3DES-CBC with a 192-bit key. That is, `SaveHashMode` is set to `false`.
- Must contain at least three characters.

Normal Policy

This is not a selectable policy in TIBCO Domain Utility, but is provided as a password policy template file. See [Custom Policy on page 18](#) for information on finding this template file.

If you use this template file, the following password policy is enforced. A password:

- Is saved in encryption mode, and the algorithm used is 3DES-CBC with a 192-bit key. That is, SaveHashMode is set to false.
- Must contain at least six characters.
- Must contain at least three of the following:
 - One or more characters in lower case (a-z,)
 - One or more characters in upper case (A-Z)
 - One or more numeric characters (0-9)
 - One or more punctuation characters (.,!@#\$\$%^&*()_+ | ~-=\`{}[]:;'<>?,./)

Restrictive Policy

If selected, the following password policy is enforced. A password:

- Is saved in hash mode, and the algorithm used is SHA1. That is, SaveHashMode is set to true.
- Must contain at least eight characters.
- Cannot contain the current password.
- Cannot contain the user's name.
- Cannot contain the space character.
- Must contain at least three of the following:
 - One or more characters in lower case (a-z,)
 - One or more characters in upper case (A-Z)
 - One or more numeric characters (0-9)
 - One or more punctuation characters (.,!@#\$\$%^&*()_+ | ~-=\`{}[]:;'<>?,./)
- A password must be changed after 90 days.
- A password must be changed on a user's first login or when the password is reset.
- A user account is disabled after five failed login attempts.
- The last five passwords used cannot be reused.

Custom Policy

You can provide a custom policy that is based on the password policy templates

and schema file provided in the *TIBCO_HOME/tra/version/config/security* directory. After copying a template to another location and modifying it, click the ... icon and load the custom policy file. The file contents are written to the administration domain.

Chapter 2

Fault Tolerance and Load Balancing in a Rendezvous Administration Domain

This chapter discusses fault tolerance, which also provides load balancing for the TIBCO Administrator server when using a TIBCO Rendezvous administration domain.

The needs of your integration project determine the complexity of your administration server configuration. This chapter also presents different configuration scenarios.

Topics

- [Overview, page 22](#)
- [Load-Balanced Servers, page 25](#)
- [One Server in One Subnet, page 27](#)
- [Multiple Load-Balanced Servers in a Subnet, page 28](#)
- [Multiple Administration Domains in a Subnet, page 30](#)
- [Administration Domain Spanning Subnets, page 32](#)
- [Multiple Load-Balanced Servers Spanning Subnets, page 42](#)

Overview

TIBCO Administrator provides fault tolerance and load balancing capabilities for client applications. To allow client applications to access the same data from multiple administration servers, a primary server and one or more secondary servers can be installed. Each server should be installed on a separate machine and each machine must be part of the same administration domain.



Note that this chapter applies only to administration domains that use TIBCO Rendezvous as the message transport for communication within the domain. The chapter does not apply to your domain if it has been configured to use TIBCO Enterprise Message Service.

Fault tolerance and load balancing is implemented using the TIBCO Rendezvous distributed queue protocol (RVDQ). Each logical server has a distributed queue based on the server name. The servers in a distributed queue group share the same server name. The TIBCO Administrator client just sees one logical server.

Primary and secondary administration servers can be configured in one of these ways:

- During installation when using TIBCO Domain Utility to configure the initial administration domain.
- After installation, when using TIBCO Domain Utility to create or modify an administration domain.
- After installation, when editing the administration server's property file.

When a client application writes to the application or domain repository (for file-based domains), the primary administration server propagates the changes to the secondary administration servers. If the primary server goes down, clients continue to receive data from the secondary server.

While all update requests are handled by the primary administration server, read requests are shared among the primary and secondary servers by use of RVDQ. Updates will automatically propagate from the primary to the secondary servers. You can have any number of secondary servers.

Note that this is not full fault tolerance.

However, if you are using file storage, there is a small time gap between when the primary receives an update and when it is propagated to the secondary servers, so converting a secondary to a primary for short downtimes is not recommended, and definitely not to deal with network partitions. The most reliable configuration is to have your data stored in a fault tolerant, distributed database that is shared by all servers.

Multiple Secondary Administration Servers

If you install one or more secondary administration servers to run in a fault-tolerant mode, each client request for project information in the administration domain is load-balanced across all administration servers, even if the primary administration server is down.

If multiple secondary administration servers are used:

- Each secondary administration server must run on a different machine and have its own `tibcoadmin_domain-name.tra` file.
- If you make updates to the configuration using either the TIBCO Domain Utility (recommended) or by editing the `tibcoadmin_domain-name.tra` files, you must restart the administration server on each machine so the changes take effect.
- If TIBCO Rendezvous is used as the transport between a client application and the administration server, the appropriate administration server responds automatically and there is load balancing across administration servers.

For example, if a TIBCO BusinessWorks process uses Rendezvous to make a request to a primary administration server and that server fails, the process request will automatically go to the secondary administration server.

- If HTTP or HTTPS is used as the transport between a client and the administration server, you must either use an IP redirector or explicitly point to the secondary administration server to be used if the primary server fails.

For example, if a TIBCO BusinessWorks process uses HTTP or HTTPS to communicate with TIBCO Administrator, when the primary administration server is unavailable, the process will be unable to communicate with the secondary server as it has different IP or hostname. In this case, the `tibco.repourl` property in the BusinessWorks process `.tra` file must be changed to point to the secondary server machine's IP address or hostname.

Alternatively an external IP redirector can be used to redirect the primary machine's IP address or hostname to the secondary machine. See your IP Redirector or HTTP server documentation for information on how to do this.

- If the primary administration server fails, you can use the secondary server to start or stop processes, but other management tasks such as deploying, undeploying, or deleting application are not allowed.
- By default, all machines within an administration domain are expected to be in the same network subnet. You can, however, set up your system to use the TIBCO Rendezvous `rvr` daemon so the server can communicate with machines on different subnets. See [Administration Domain Spanning Subnets on page 32](#).

- If the secondary servers are distributed across a WAN, regions can be used to have clients use the local server. See [Load-Balanced Servers on page 25](#) for details.

Secondary Administration Server Startup Fails

When setting up a primary and a secondary administration server, if startup of the secondary server fails because it cannot get its initial data, a stack trace message starting with the following appears in the secondary server's console and log file:

```
java.lang.Exception: Load security program error:null
```

To resolve this, verify that the time zone settings on the machines that host the primary and secondary servers are in sync.

Load-Balanced Servers

If you wish to use multiple load-balanced servers, you must install TIBCO Administrator once for each server. You must designate one server as the primary, and all other servers as secondary servers.

For the simplest case, you can just install multiple servers and designate them as a primary or secondary server using the TIBCO Domain Utility, which is launched at the end of installation and can also be invoked separately afterwards. In some cases you also edit properties in the `tibcoadmin_domain.tra` file for each administration server in the domain and restart each server.

Optionally, if you set up multiple servers to talk to each other across a WAN, you can use a subject prefix to specify a preference for local servers for clients communicating using TIBCO Rendezvous. The subject prefix is specified in the server's property file in the `repo.prefixRegionalSubject` property as explained later in this section.

The following properties affect load balancing. The properties are set in the administration server's configuration file, `tibcoadmin_domain-name.tra`.

- The `repo.master` property distinguishes the primary server from the secondary servers. If the value is set to a hostname or default IP address, it is a primary server. Any other value indicates a secondary server. For example, the bogus value `255.255.255.255` indicates a secondary server.

As a side effect of enabling fault tolerance, a small `_RVFT` message is sent once a second. If you do not intend to run in fault tolerant mode, that is, you wish to use only a primary server and no secondary servers, you can disable fault tolerance by commenting out the `repo.master` property in the configuration (`.tra`) file. In that case, the `_RVFT` message is no longer sent.



If you comment out the `repo.master` property, you cannot add secondary servers until you have manually removed the comment.

- If the `repo.isMaster` property is set true, it overrides the `repo.master` property to allow the primary server to be specified in a cluster environment.
- The `repo.state` property defaults to `READ_WRITE` if the `repo.master` property is set. You can set this property to `READ_ONLY` explicitly to have a server come up in that state upon startup. The primary server and all secondary servers should have the same value set. This value indicates the global state for the set of servers.

By default, the global state is `READ_WRITE`. This allows the primary server to handle both read and write requests, while the secondary servers only handle

read requests. Some sites wish to limit updates to specific windows of time. The following states are supported:

- `READ_ONLY`—If the primary has been set to `READ_ONLY`, and the client application tries to perform an operation that involves locking or writing to the application or domain repository, a `SERVER_STATE_READ_ONLY` exception results.
- `READ_WRITE`—The primary server's state must be `READ_WRITE` if you wish to make changes to projects. Only the primary server can handle write request, but all secondary servers will continue serving read requests.
- The `repo.prefixRegionalSubject` property specifies a prefix for the TIBCO Rendezvous subject. In load balancing mode, servers with that prefix are considered preferred servers for read operations.

This string is used for giving preference to a group of load balancing servers over the rest of load balancing servers. Each repository client and server which participates in load-balancing may have optional regional information. If this optional property is set, servers in load-balancing mode listen on the extra read subject that incorporates this region. If an optional regional subject is set for a client, the client directs read operations to that subject first. If no server is available, then the client retries using the regular (non-prefixed) read subject.

You must set this property explicitly in the `tibcoadmin_domain.tra` file and for each repository locator string used in client `.tra` files.

- The `repo.updateSecondaryServerPersistentStore` property only applies to secondary servers. It determines whether or not secondary servers working in load balancing mode write data into their persistent backend. If the backend is shared by the primary server and this secondary server, this property should be set to `false`. If this property is set to `true`, the secondary server will update not only its cache but also its backend whenever it synchronizes a change with the primary server. Currently this property cannot be set to `true` for secondary servers that use a database backend. Default value for this property is `true`.

You must set this property explicitly in the `tibcoadmin_domain.tra` file.

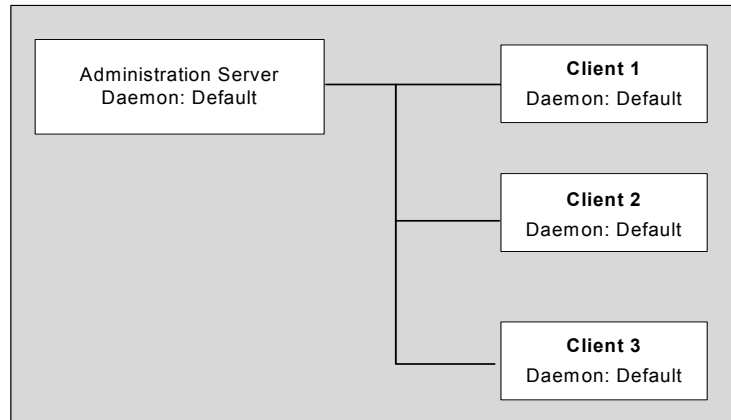
One Server in One Subnet

Using defaults, an administration server can communicate with multiple client applications. For example, a TIBCO Adapter instance could be requesting adapter configurations or a TIBCO BusinessWorks engine could be accessing information saved to an application repository.

The next diagram shows a simple configuration in a single subnet. No setup is required because defaults are used. The default daemon is the default TIBCO Rendezvous daemon, but HTTP or HTTPS could also be used in this scenario.

Note that you cannot use HTTP or HTTPS to connect to a 4.x adapter. You must configure HTTPS before using it.

Figure 2 One Server in a Single Subnet



Options

You can use TIBCO Domain Utility to set the following options:

- Specify a database backend as the persistent store for domain data.
- Add secondary servers.
- Specify non-default TIBCO Rendezvous properties or configure HTTPS.
- Synchronize with an LDAP directory server to manage users and groups.
- Set Local Application Data as the default deployment option
- Specify a Password Policy
- Specify ports for a web server

Multiple Load-Balanced Servers in a Subnet

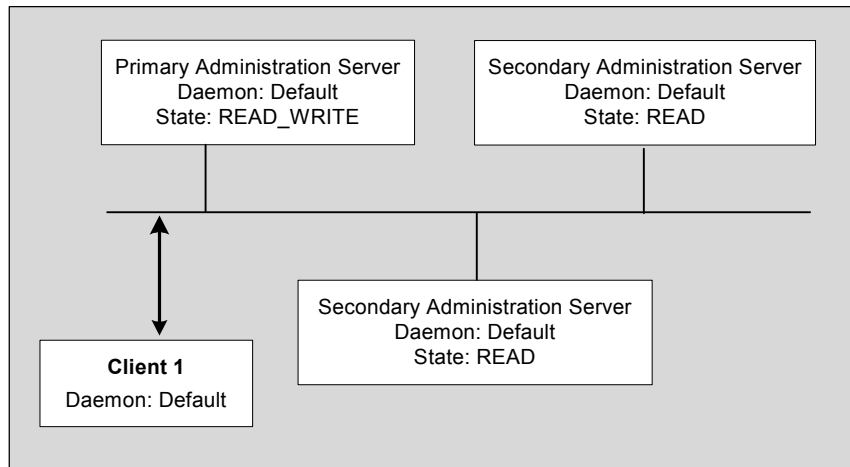
You can install multiple load-balanced servers on different machines in a subnet.

When you install TIBCO Administrator, you are prompted to install a primary or a secondary server.

- The first time you install TIBCO Administrator, you must install a primary server and create an administration domain.
- All subsequent installations must either create a differently named primary server or set up secondary servers, as follows:

All servers in a load balanced group must use the same TIBCO Rendezvous service and port properties. In [Figure 3](#), all servers use the default TIBCO Rendezvous daemon. When a client sends a request for information, the next available server responds (this speeds up responses in a high-volume situation). When a client writes information, the primary server picks up the information and enters it into the appropriate project. It then automatically sends update notices to all secondary servers in the domain.

Figure 3 Multiple Load-Balanced Servers in One Administration Domain



Using TIBCO Rendezvous, load balancing is automatic. To get the load balancing and fault tolerance benefit with HTTP, you must either use an IP redirector or explicitly point to a backup server to be used when a server fails.

See your IP Redirector or HTTP Server documentation for information on how to do this.

Options

You may decide to take advantage of the available options listed in [Options on page 27](#).

Updates are propagated automatically to secondary servers, which persistently store the updated data. However, this is a push process with version checks to handle synchronization issues rather than a distributed two-phase commit. To ensure that persistent storage for the server is always synchronized, a database or shared file system must be used. Otherwise, there is a small window of time during which an update may not have been propagated. If the primary server goes down at that time, the secondary server is not able to receive the update until both the secondary and the primary server are running again.

Multiple Administration Domains in a Subnet

You can have multiple administration domains in a single subnet (or even on a single machine). The primary servers may be used in conjunction with multiple administration domains, or standalone without domains.

- [Multiple Administration Domains](#) discusses multiple TIBCO administration domains, each with its own primary server.
- [Multiple Servers Using Different Daemon and Service Properties](#) discusses the same scenario for situations where the domain management and security component is not installed.

Multiple Administration Domains

You can have multiple TIBCO administration domains in the same subnet or on the same machine. Each administration domain must have its own administration server.

Each domain on a specific machine must have its own name.



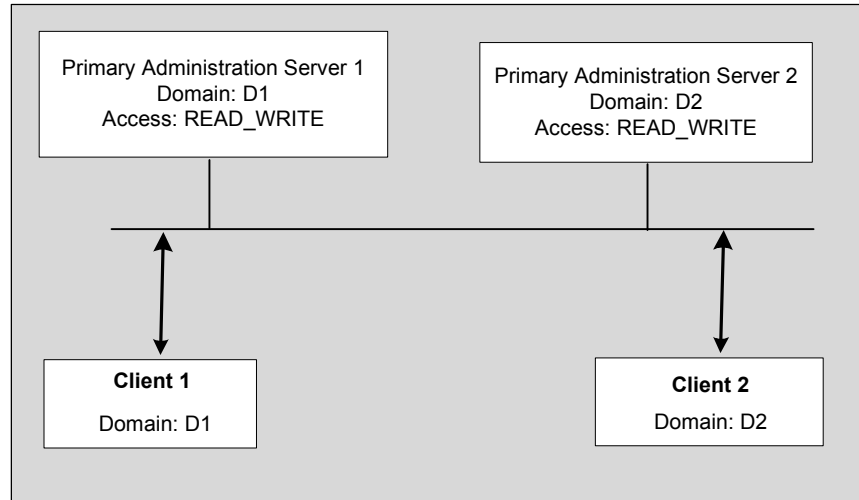
On each machine, each domain must be distinguished by having a different name.

Within a subnet, either a different name or different connection properties will distinguish a domain. However, if connection properties are the same, then only one domain can have a project or application repository of any given name. Deployment will create an application repository name with the domain name prefix. However, for maximum security, it is recommended that you use a different name and different connection properties.

In the next illustration, one server is using a domain D1 and another server is using a domain D2. Each server must be a primary server.

You can choose to use HTTP for communication between the server and associated project or application repository. However, when you deploy applications or view machine inventory or alert information, the server retrieves this information using TIBCO Hawk agent and it can only communicate with the server using TIBCO Rendezvous.

Figure 4 Multiple Domains in a Single Subnet



Multiple Servers Using Different Daemon and Service Properties

You can use the daemon port and service properties to set up multiple servers in a single subnet. The service port number defines the TIBCO Rendezvous service to be used for the subnet. The different service numbers assigned to different administration servers and their client applications can be used to partition subnets.

You can assign different service and daemon port numbers using the TIBCO Domain Utility.



Daemon and service port numbers are not required to be the same. However, the recommended practice is to use the same daemon and service port number to avoid confusion.

Each client application must set the service and daemon to the port number of the administration server on the same subnet. The properties are set when a machine joins an administration domain and when an application is deployed.

On computers with more than one network interface, the network parameter instructs the Rendezvous daemon to use a particular network for all communications involving this transport. To communicate over more than one network, a program must create a separate transport object for each network. See the *TIBCO Rendezvous Concepts* guide for details.

Administration Domain Spanning Subnets

You can extend an administration domain to span subnets by using the TIBCO Rendezvous routing daemon (`rvrd`).

TIBCO Administrator does not provide built-in support for installation or configuration of `rvrd`; this section explains how to do it using the `rvrd` setup utilities.



It is recommended that you simplify network topology as much as possible in the analysis and planning phase. If at all possible, you should architect the solution to use a single subnet.

Also, when you plan your network's firewalls, keep in mind that TIBCO Hawk agents in different subnets than the database server (in a database-based domain) must be able to connect directly to the domain repository on the database server.

Introduction

This section explains the basic setup that is required for having your administration domain span subnets. Setup needs to be performed:

- on two machines
- for the local and neighbor network
- for each daemon/service pair and for all appropriate associated TIBCO Rendezvous subjects (as discussed in [Setup Tasks on page 33](#)):
 - Subject `com.tibco.repo.>`, using daemon and service (7500 by default). To filter out unneeded messages, you can also choose from this list. Replace *repo_server_name* with the name of your server and *repo_instance* with your server instance name.

```
com.tibco.repo.repo_server_name.repo_instance.>
com.tibco.repo.instance_discovery.request
com.tibco.repo.server_discovery.request
com.tibco.repo.serverHeartbeat.>
com.tibco.repo.instance_mgmt.repo_server_name.request
```

If you use secondary servers in your domain, you must also perform setup for these subjects:

```
_RVFT.>
_RVCM.>
```

`_RVCMQ.>`
— Subject `_HAWK.>` (7474 by default) and subject `com.tibco.pof`.



It is also possible to set up for the TIBCO Rendezvous subject `>` (all subjects) if there is relatively little TIBCO Rendezvous traffic on the subnet.

This setup can be used at development time for all platforms. At runtime, you have two options:

- On UNIX systems, you can prepare scripts that can start `rvrd` for both subjects automatically.
- On Windows systems, only one Windows Service for `rvrd` is allowed. Because you would need one service for each subject, using a Windows system for runtime monitoring may therefore be challenging and is not recommended.

Setup Tasks

If you need to set up your domain to span subnets, the following steps are involved:

Install Administration Server

Install an administration server on one machine (you may later install secondary servers on additional machines for load balancing). Specify the domain name, user name, and password and note them for later use.

Decide the Services and Daemons the two Machines will use

For each machine, specify the services and daemons used for the two subjects on which the administration server sends and receives messages, for example (by default):

Table 6 Service and Daemons for Machine

Subject	service	daemon
<code>_HAWK.></code>	7474	tcp:7474
<code>com.tibco.repo.></code> <code>com.tibco.pof.></code>	7500	tcp:7500



It is necessary to listen to each subject you are interested in. See [page 32](#).

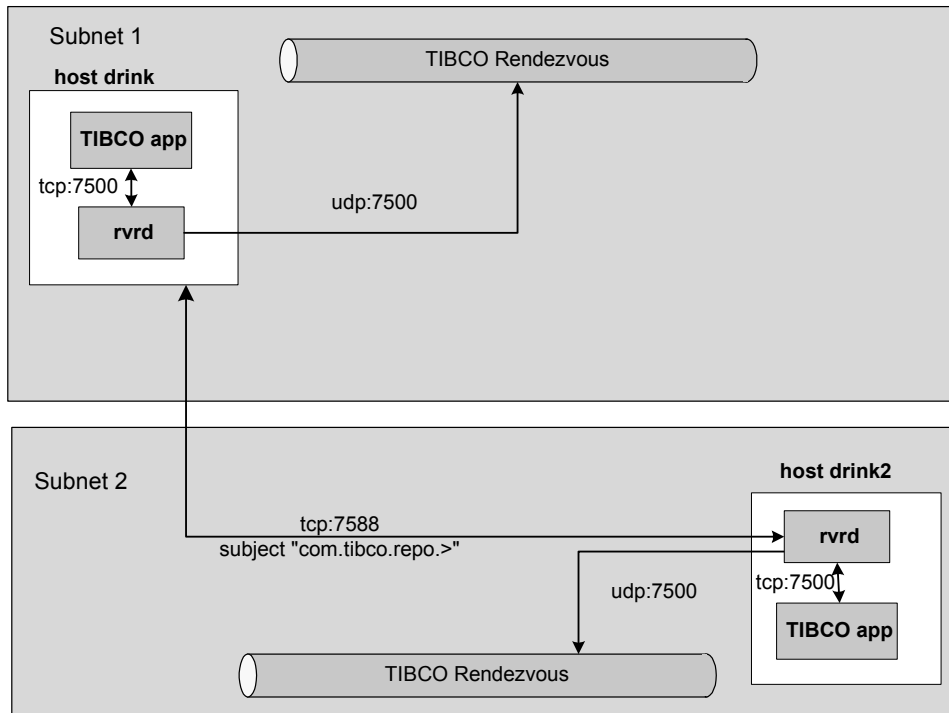
Figure 5 illustrates how rvrdr communicates within and across the subnet: On host drink, rvrdr will send out messages via rvrdr to other applications in the subnet using udp 7500.

- Locally, rvrdr communicates with those applications that use it directly (for example, TIBCO Designer or TIBCO Administrator) using TCP 7500.
- rvrdr communicates using TCP 7588 to connect to the rvrdr on host priya-dt, which is also using tcp 7588. This rvrdr is set up in this example to use the same sending and listening port on both machines. It's also possible to use a different port for sending and listening for added security.
- On host drink2, rvrdr is sending out messages via TIBCO Rendezvous using udp 7500.



This description and diagram only illustrates the messages on the subject `com.tibco.repo.`. The setup needs to be duplicated with different tcp and udp connections for the other subjects, for example, `_HAWK.>` or `com.tibco.pof.>`

Figure 5 Setup of rvrdr for Subject "com.tibco.repo.>"



Set up and start rverd on the Administration Server Machine

On the administration server machine, TIBCO Rendezvous has been installed as part of the TIBCO Runtime Agent installation and you can set up rverd without additional installation.

Setting up rverd communication on the first machine consists of these steps:

1. Shut down any TIBCO applications (for example, adapters, TIBCO BusinessWorks).
2. On Windows, use the Services Control Panel and shut down:
 - TIBCO Hawk agent service for the administration domain
 - Administration server service for the administration domain
 - Any other TIBCO services that require a TIBCO Rendezvous daemon to be running.

On UNIX systems, kill the above processes.

3. Using the task manager, stop any rvd processes. Make sure the process is not restarted.



You cannot configure rverd if rvd is running.

4. Start two local rverd services, which write the appropriate storage and log files. As you start rverd, it is also useful to specify the http port from which each daemon can be configured.

You can start the services from the command-line, or using a batch file or script. Here's an example of a Windows batch file:

```
@echo on
start c:\tibco\tibrv\bin\rvrd -store c:\tibco\drink7500.rverd
      -listen 7500 -logfile c:\tibco\drink7500.out -http 7599
start c:\tibco\tibrv\bin\rvrd -store c:\tibco\drink7474.rverd
      -listen 7474 -logfile c:\tibco\drink7474.out -http 7499
```

Configure the two rverd daemons

To configure the two rverd daemons, follow these steps:

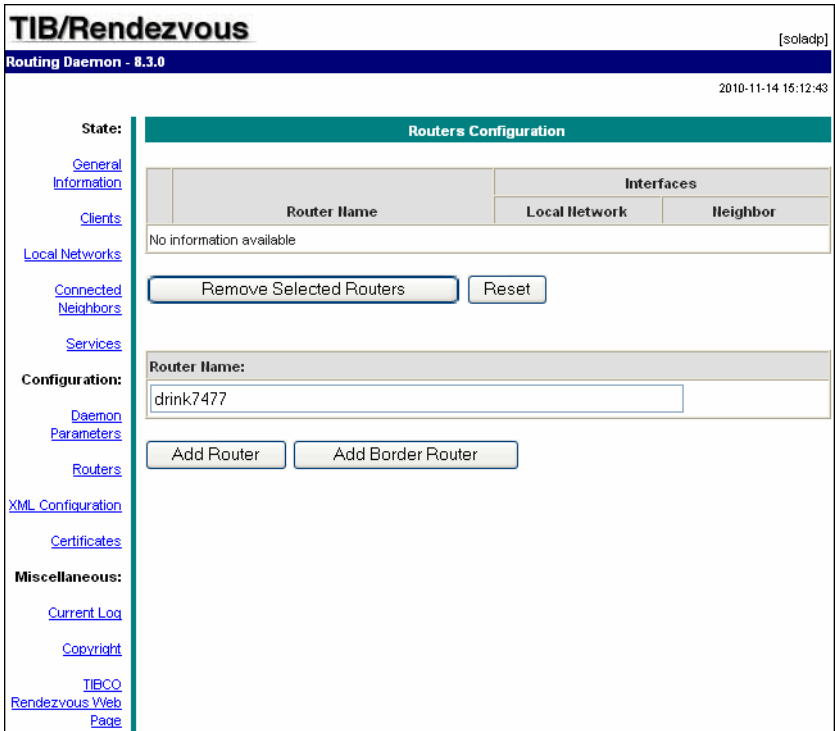
1. Open a web browser and go to the rverd setup screen.

To access the TIBCO Rendezvous browser administration interface, point your browser to `http://host_name:port`. Host name can be a machine name or IP address, but sometimes a fully qualified name is required. Default port

numbers are listed in *TIBCO Rendezvous Administration*. For example, **http://ip-address:7580/**.

- 2. In the Router Name field, give a name for the router (for example, drink7474), then click **Add Router**.

Figure 6 Enter Name of the Router



The router is listed, with a choice to configure local and neighbor interfaces.

Figure 7 Configure Local and Neighbor Interfaces

TIB/Rendezvous

Routing Daemon - 8.3.0

2010-11-14 15:13:21

[solidp]

State:

General Information

Clients

Local Networks

Connected Neighbors

Services

Configuration:

Daemon Parameters

Routers

XML Configuration

Certificates

Miscellaneous:

Current Log

Copyright

TIBCO Rendezvous Web Page

Routers Configuration

	Router Name	Interfaces	
		Local Network	Neighbor
<input type="checkbox"/>	drink7477	_0_	_0_

Remove Selected Routers

Reset

Router Name:

Add Router

Add Border Router

3. Click the number under **Local Network**. In the screen that is displayed, type the local network name and the service, then click **Add Local Network Interface**.

Figure 8 Add Local Network Interface

TIB/Rendezvous

[soladp]

Routing Daemon - 8.3.0

2010-11-14 15:13:46

State:

Local Network Interfaces Configuration [drink7477]

General Information

Clients

Local Networks

Connected Neighbors

Services

Configuration:

Daemon Parameters

Routers

XML Configuration

Certificates

Miscellaneous:

Current Log

Copyright

TIBCO Rendezvous Web Page

Local Network Name	Service	Network Specification	Cost
No information available			

Remove Selected Local Network Interface(s)

Reset

Local Network Name	Service	Network Specification	Cost
drink7477	7477		1

Add Local Network Interface

Reset



If you do not include the service, it defaults to 7500. This may result in an error if 7500 is already in use.

4. In the screen that appears, specify the subject to be used, either `com.tibco.repo.>` or `_HAWK.>`, then click **Import and Export**.



For `_Hawk.>` and the other subjects use different daemons and services. Be sure you are using the appropriate daemon and service for the subject.

Figure 9 Specify the Subject to be Used

TIB/Rendezvous

Routing Daemon - 8.3.0

2010-11-14 15:14:22

State:

General Information

Clients

Local Networks

Connected Neighbors

Services

Configuration:

Daemon Parameters

Routers

XML Configuration

Certificates

Miscellaneous:

Current Log

Copyright

TIBCO Rendezvous Web Page

Subject Configuration [drink7477]

Import Subjects	Import Weight
No information available	

Export Subjects
No information available

Remove Selected Subjects

Reset

Subject	Import Weight
<input type="text" value="_HAWK.>"/>	<input type="text" value="10"/>

Import

Export

Import and Export

5. Click **Routers** again in the left column and choose the number under **Neighbor**.

Figure 10 Router Configuration

TIB/Rendezvous

Routing Daemon - 8.3.0

2010-11-14 15:16:16

State:

General Information

Clients

Local Networks

Connected Neighbors

Services

Configuration:

Daemon Parameters

Routers

XML Configuration

Certificates

Miscellaneous:

Current Log

Copyright

TIBCO Rendezvous Web Page

Routers Configuration

	Router Name	Interfaces	
		Local Network	Neighbor
<input type="checkbox"/>	drink7477	1	0

Remove Selected Routers

Reset

Router Name:

Add Router

Add Border Router

6. In the next screen, specify the host, port, and router name for the local and remote port.
It is convenient to use the same port locally and remotely, but it is not necessary.
7. Click **Add Neighbor Interface**. The information, including the IP address, is displayed.
8. Now repeat the steps for the neighbor machine. You have to specify:
 - An interface ID.
 - A router name.
 - The local network name and service.
 - A neighbor interface configuration.

9. Repeat the process for the second daemon and the associated subjects. You can find the HTTP address at which you can configure that daemon by displaying the `.out` file to the console.

Restart Administration Server and TIBCO Hawk Agent on First Machine

One easy way to do this on Windows platforms is to reboot the machine.

Set up Second Machine

The second machine is on a different subnet than the first. Setup of the second machine is similar to that of the first. Follow these steps:

1. Install TIBCO Rendezvous, either standalone or as part of TIBCO Runtime Agent.



You can perform a custom installation of TIBCO Runtime Agent and only install the TIBCO Rendezvous product.

2. Start two `rverd` daemons. See [Set up and start rverd on the Administration Server Machine on page 35](#).
3. Configure the two `rverd` daemons. Follow the steps under [Configure the two rverd daemons on page 35](#).



As you step through daemon configuration, you use the same ports and subjects but a different local and neighbor daemon.

Second Machine Joins Domain

The final step is to invoke the TIBCO Domain Utility on the second machine and have that machine join the administration domain.

Troubleshooting

If your setup is not successful, that is, the components from the second machine do not show up on the domain inventory, consider these points:

- The TIBCO Runtime Agent version on the two machines must be exactly the same.
- Consider restarting the administration server on the machine where it is located.
- Consider using the wildcard subject (`>`) if there isn't too much TIBCO Rendezvous traffic on your network.

Multiple Load-Balanced Servers Spanning Subnets

You can use the mechanism discussed in [Administration Domain Spanning Subnets on page 32](#) to set up an administration domain with multiple load-balanced servers.

To successfully work in this scenario, follow these steps:

1. Install the desired number of servers in the different subnets and configure them, as discussed in [Administration Domain Spanning Subnets on page 32](#).

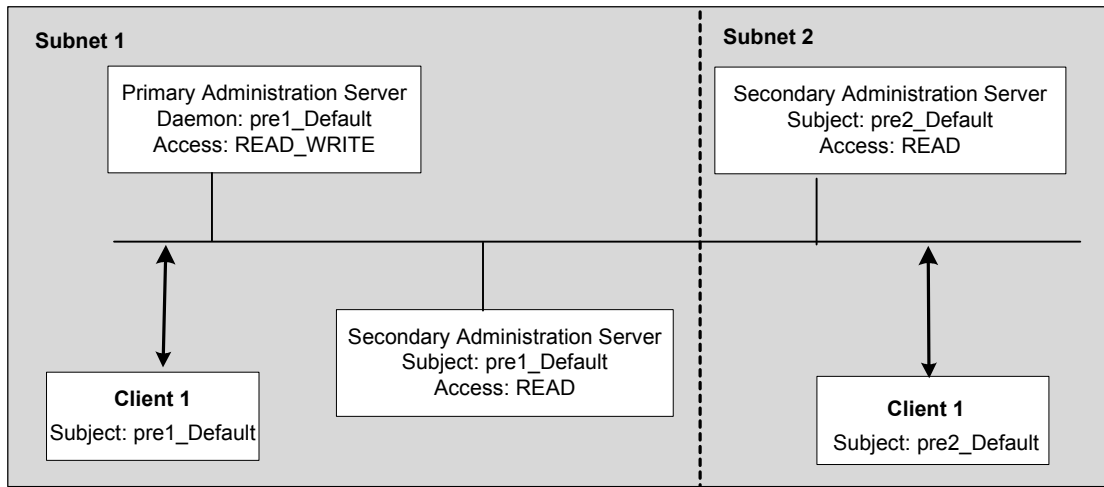


Install only one primary server. All other servers should be secondary (read-only) servers. You must carefully consider which server should be the primary (read-write access) server. The primary server does all project management such as adding and deleting projects, and handles all updates. Consider reliability and network traffic when selecting the primary server.

2. Set up the `tibcoadmin_domain.tra` file for each server to have a `repo.prefixRegionalSubject` value. This is optional.

The `repo.prefixRegionalSubject` property is used to identify and give preference to local servers.

If this optional property is set, servers in load-balancing mode listen on the extra read subject that incorporates this region. If an optional region subject is set for a client, the client directs read operations to that subject first. If no server is available, then the client retries using the regular (non-prefixed) read subject. However, deployment does not use this attribute so the `repoURL` values in each client's property (`.tra`) file must be updated each time a deployment occurs to take advantage of this.

Figure 11 Multiple Load-Balanced Servers Across Subnets

Chapter 3

Fault Tolerance and Load Balancing in a EMS-Based Administration Domain

This chapter discusses fault tolerance, which also provides load balancing for the TIBCO Administrator server when using a TIBCO EMS-based administration domain.

Topics

- [Overview, page 46](#)
- [Creating a Secondary Administration Server, page 48](#)
- [Removing a Secondary Administration Server, page 54](#)
- [Specifying Fault Tolerance EMS Servers, page 55](#)

Overview

TIBCO Administrator provides fault tolerance and load balancing capabilities for client applications. To allow client applications to access the same data from multiple administration servers, a primary server and one or more secondary servers can be installed. Each server should be installed on a separate machine and each machine must be part of the same administration domain.



This chapter applies only to administration domains that use TIBCO Enterprise Message Service as the message transport for communication within the domain. The chapter does not apply to your domain if it has been configured to use Rendezvous Service.

When you use TIBCO Enterprise Message Service for your domain transport, both the primary and secondary servers use the same database and same EMS server. Secondary servers function exactly like the primary server and can deploy, undeploy, and delete applications. Note that Fault Tolerance Administration servers only applies to management tasks and clients must use local data.



If the administration domain is created using TIBCO Enterprise Message Service as the transport and the version of TIBCO Hawk installed is 4.9.0, an extra topic (`_LOCAL.HAWK.DUPCHECK`) is required to start the domain hawkagent. Add the topic entry in the `topics.conf` file.

Multiple Secondary Administration Servers

If multiple secondary administration servers are used:

- Each secondary administration server must run on a different machine and have its own `tibcoadmin_domain-name.tra` file.
- If you make updates to the configuration using either the TIBCO Domain Utility (recommended) or by editing the `tibcoadmin_domain-name.tra` files, you must restart the administration server on each machine so the changes take effect.

Secondary Administration Server Startup Fails

When setting up a primary and a secondary administration server, if startup of the secondary server fails because it cannot get its initial data, a stack trace message starting with the following appears in the secondary server's console and log file:

```
java.lang.Exception: Load security program error:null
```

To resolve this, verify that the time zone settings on the machines that host the primary and secondary servers are in sync.

Creating a Secondary Administration Server

A secondary server can be added to an administration domain. The secondary server is equivalent to a primary server in that it has read and write permissions enabled. You can include or create additional secondary servers if a secondary is up and running. Secondary servers can be used for load balancing and fault tolerance.

Use this procedure to add a secondary server to an administration domain that uses TIBCO EMS as the transport.

To Add a Secondary Server Using the GUI

1. Start Domain Utility and click the **Next** button on the main screen.
2. Under **Category**, click **Domain Configuration**, then click **Add a secondary server**.
3. Click **Next** and in the screen that appears:

Select **Show Advanced** and click TIBCO EMS as the transport. The EMS parameters must be defined for the secondary server.

Click the **Discover** button and select a domain. If a domain does not appear, select **Show Advanced** and increase the **Discover Timeout** value. The value sets the amount of time Domain Utility has to connect to the master server. If no connection is made in the specified time, the discovery operation will time out. Increase this number on slow systems and click the **Discover** button again.

Provide domain details. See [Secondary Server Fields on page 49](#) for field descriptions.

4. Click the **Next** button. The screen that appears displays the ports used by the Web Server.
5. Click the **Next** button. Provide the administration credentials that were defined when the domain was created.
6. Click the **Next** button. The screen that appears displays a summary of the values you have provided.
7. Click the **Next** button to add the secondary server to the domain. After joining the domain, the services that support the secondary server are listed. You must start each service to enable the secondary server.
8. Click **Finish** to end the session.

To Add a Secondary Server Using the Command Line Utility

1. Create a working directory that will hold the XML file that defines configuration options.
2. Copy the following file to your working directory:
`TRA_HOME\template\domainutility\cmdline\AddSecondaryServer.xml`
3. Open `AddSecondaryServer.xml` in a text editor.
After changing the parameters, save the file and exit the text editor.
4. Execute the following command to apply your changes to the domain:
`domainutilitycmd -cmdFile
working-dir-path\AddSecondaryServer.xml`

Secondary Server Fields

Table 7 Secondary Server Fields

Domain Details	
Administration Domain	Click Discover Domains to choose the domain for which you wish to install a secondary server.
Project Directory	Location in which the server looks for the domain data store, for project files saved as single-file projects, and for deployment configuration files.
Machine	The name of the machine is provided in the <code>Machine</code> field and must not be changed.
Hawk Cluster	Machines are grouped in the Administrator GUI under the value provided in the <code>Hawk Cluster</code> field. If you change the default value, this machine displays in the Administrator GUI under the value you provide. The cluster name must be enclosed within quotes, if the name contains spaces.
Advanced Options	
Show Advanced	If you specified custom TIBCO Rendezvous parameters for the master server, click this check box, to specify the same custom parameters for the secondary server.

Table 7 Secondary Server Fields

Discover Timeout	Time Domain Utility allows for connecting to the master server. If no connection is made in the specified time, the master server may be down or slow to respond and Domain Utility times out. Increase this number on slow systems.
TIBCO Rendezvous TIBCO EMS	Select the transport to use for administration domain communication. When adding a machine, you must select the transport already set for the primary domain. The fields change, depending on the transport selection.
TIBCO EMS parameters for TIBCO Administrator	
Server URL	The URL of the TIBCO Enterprise Message Service server in the following format: <code>tcp://hostname:port</code> . If you have configured multiple fault tolerant servers, specify all of them here, separating them by commas. For example: <code>tcp://host1:7222,tcp://host2:7222</code>

Table 7 Secondary Server Fields

Username	<p>Specify the user account name authorized to administer the TIBCO Enterprise Message Service server. Specify a user that is a member of the \$admin group (for example, the predefined admin user), or a user who has the following permissions:</p> <ul style="list-style-type: none"> publish, subscribe, and create permissions to the following topics: com.tibco.repo.> com.tibco.pof> or com.tibco.pof.HawkConfig.> com.tibco.pof.MonitoringManagement.> com.tibco.repo.server_discovery.> com.tibco.pof.domain-name.> (for each domain)> com.tibco.pof.AUTH_domain-name.> (for each domain) com.tibco.repo.instance_mgmt.*.trustworthy_HAWK_domain-name (for each domain) _LOCAL.HAWK.DUPCHECK public, subscribe, and create permissions to the com.tibco.repo.> queues <p>Note: You must add the following topics to the <i>TIBCO_HOME</i>/ems/bin/topics.conf file: com.tibco.pof.domain-name.> com.tibco.repo.server_discovery.> com.tibco.pof.AUTH_domain-name.> com.tibco.repo.instance_mgmt.*.trustworthy_HAWK.domain-name (one line for each domain) _LOCAL.HAWK.DUPCHECK</p> <p>You must also add the following queue to the <i>TIBCO_HOME</i>/ems/bin/queues.conf file: com.tibco.repo.></p> <p>Note that if <i>domain-name</i>, contains the characters '.', '>' and '*', the characters must be replaced by the following strings:</p> <p> "." replaced by "2E" ">" replaced by "3E" "*" replaced by "%2A"</p>
Password	<p>Specify the password for the user account given in the Username field.</p>

Table 7 Secondary Server Fields

Enable SSL	Select to enable Secure Sockets Layer (SSL) for use with TIBCO Enterprise Message Service.
Domain Home Paths Configuration	
TRA Domain Home	Click the ... button and navigate to a drive shared by all nodes in the cluster. Specify the location of the TIBCO Runtime Agent domain home. For example, C:\tibco\tra\domain. See the <i>TIBCO Runtime Agent Installation</i> guide for information about installing TIBCO software in a cluster environment.
Administration Domain Home	Click the ... button and navigate to a drive shared by all nodes in the cluster. Specify the location of the administration server domain home. For example, C:\tibco\administrator\domain.
TIBCO Rendezvous parameters for TIBCO Hawk HMA	
Use default values	Select the Use default values check box unless you are an experienced user.
Hawk HMA Daemon	Hawk HMA Daemon instructs the transport creation function about how and where to find the Rendezvous daemon and establish communication. The default is tcp:7474
Hawk HMA Network	Hawk HMA Network specified, uses the specified network for all the communications.
Hawk HMA Service	Hawk HMA Service is used for client-server communication. Each transport communicates on a single service. A transport can communicate only on the same service with other transports. The default is 7475.
Cluster Group Configuration	
Machine is Logical	Select to specify that the machine is a node in a cluster.
Virtual IP Address	Enter the cluster virtual IP.

Table 7 Secondary Server Fields

Symmetric Key	
Dynamic Symmetric Key	<p>Select to use dynamically generated keys to encrypt sensitive data by default when deploying applications locally. A static key is used if this checkbox is cleared.</p> <p>Note: This option does <i>not</i> affect sensitive data in the deployment configuration files exported using the AppManage utility. See <i>TIBCO Runtime Agent Scripting Deployment Guide</i> for information on how to protect sensitive data in the deployment of configuration files using an encryption password.</p>

Removing a Secondary Administration Server

Use this procedure to remove a secondary server from an administration domain that uses TIBCO EMS as the transport. The secondary administration server must be stopped before deleting it.

To Remove a Secondary Server Using the GUI

1. Start Domain Utility and click the **Next** button on the main screen.
2. Under **Category**, click **Domain Configuration**, then click **Delete an Administration Domain**.
3. Click **Next** and in the screen that appears, select an administration domain.
4. Click **Next** and in the screen that appears, provide the credentials for the domain in which the secondary server is running.
5. Click **Next** and in the screen that appears, verify that the values you have provided are correct.
6. Click **Next** to delete the secondary server.
7. Click **Finish** to end the session.

To Remove a Secondary Server Using the Command Line Utility

1. Create a working directory that will hold the XML file that defines configuration options.
2. Copy the following file to your working directory:
`TRA_HOME\template\domainutility\cmdline\DeleteDomain.xml`
3. Open `DeleteDomain.xml` in a text editor.
After changing the parameters, save the file and exit the text editor.
4. Execute the following command on the machine on which the secondary server has been installed:

```
domainutilitycmd -cmdFile  
                  working-dir-path\DeleteDomain.xml
```

Specifying Fault Tolerance EMS Servers

Primary and secondary administration servers can be configured when they are created and they can also be modified afterwards.

To create an EMS administration domain, refer to *TIBCO Domain Utility User's Guide*, Creating a Domain that Uses a Database.

To create a secondary administration domain, refer to [Creating a Secondary Administration Server on page 48](#).

To modify an existing administration domain, refer to *TIBCO Domain Utility User's Guide*.

Chapter 4 **Administration Server Properties File**

This chapter explains how to perform advanced TIBCO Administrator configuration by editing properties in the administration server properties file.

Topics

- [Introduction, page 58](#)
- [General Properties, page 59](#)
- [Database Storage Properties, page 64](#)
- [TIBCO Rendezvous Properties, page 66](#)
- [Logging Properties, page 67](#)
- [Load Balancing Properties, page 71](#)
- [Server Security Properties, page 73](#)
- [Persist Commit History Properties, page 76](#)
- [TIBCO Hawk Agent Properties, page 77](#)

Introduction

Administration server properties are stored in the `tibcoadmin_domain.tra` file. The file is located in `TIBCO_ADMIN_DOMAIN_HOME\domain-name\bin`.



Most properties in the `tibcoadmin_domain.tra` file that start with `tibcoadmin` deal with administration server internal communications and should *not* be changed.



Many TIBCO Administrator users need not view or edit properties in the administration server property file directly. Instead, they set up the administration domain and interact with the administration server using TIBCO Domain Utility or the TIBCO Administrator GUI, which writes changes to the property file.

In some situations, however, it may be necessary to make explicit changes to the administration server property file. This chapter therefore explains the properties in each property file, what they mean, and possible values.

Changes made to the property file take effect the next time the administration server or TIBCO Hawk agent is started.

General Properties

The following properties are available to all server-based projects:

- `file.encoding`—Optional. Default value is `ISO8859-1`.

The encoding format for the `tibcoadmin_domain.tra` property file. By default, the property file is encoded as `ISO8859-1`. The encoding can be changed to `UTF-8` to support multibyte characters.

Typically, this property is not required if the JVM encoding format and the property file encoding format are the same. However, if your JVM and the property file use different encoding formats, this property must be set to correspond to the property file encoding format.

- `repo.forceInstanceLocks`—Optional. Default value is `false`.

If this property is set to `true`, the server takes over any existing instance locks. This automatically allows its projects to be modified by client requests.



Warning. This property should only be changed if instance locks have been left hanging by another server and need to be broken. Careless use of this property causes the former lock holder to immediately become read-only, and any pending changes by any of its clients to be lost.

- `repo.instanceNames`—Optional. Use of this option is not recommended.

A comma-separated list of projects to which the server connects. If the list is empty, the server connects to all projects matching its search criteria.

A server-based project exists for a server under these circumstances:

- Storage is a file, there is a project with a `.dat` extension in the server's search directory, and the file's contents is a valid TIBCO project.
- Storage is a database and there is project entry in the database tables of the database to which the server connects with the specified JDBC properties and user ID.

If the list contains entries, the server connects to the existing server-based projects and throws warnings for non-existing server-based projects.

- `repo.lockTimeout`—Optional. Default value is 120 seconds.

The number of seconds the server is willing to wait without hearing from a client that has locked various resources, before automatically freeing those resources.

This option does not apply to project instance locks.

- `repo.minimizeHandshakeInterval`—Optional. Default value is `false`.

Bypass delay while checking for duplicate servers during startup if `true`. *This is for use only while testing.*



Warning. This is an **extremely dangerous option**. If there are multiple servers in the environment, this could allow two servers with the same name to be operational simultaneously for a short period of time. During this time clients might get duplicate or contradictory messages leading to invalid states, exceptions, and possibly data corruption.

Be **absolutely** sure to set this to `false` in production environments.

- `repo.qThreadCnt`—Optional. The default is 1. The value is set to 2 by the TIBCO Domain Utility.

The queue thread count controls how many threads the server uses to process client read requests for each project. If this value is 2 or more, read requests are processed during commits, but the required memory and number of threads increases. The number can be increased if client applications are waiting and the CPU is running at less than 100 percent.

- `repo.serverHeartbeatInterval`—Optional. Default value is 30000 (30 seconds).

Specifies the server heartbeat interval in milliseconds. Server heartbeats allow clients to detect when a server goes down. Consider increasing the interval if network delays are causing false time-outs after waits of less than the value given.

Any number less than 1000 is considered to be in seconds. For example 900 is not 0.9 seconds, but rather 15 minutes.

- `repo.serverName`—Required.

Name of the administration domain.



Do not change this property. The server name is set during installation in multiple locations based on user input. Changing the name only here results in an inconsistent environment.

- `repo.serverPassword`—Optional.

Use this password to avoid accidental changes to projects (for example, deleting projects) when security is turned off.



This password is not used if you are running TIBCO Administrator and included the security and domain management component during installation.

- `repo.secureStatementUseOldLocale`—This property specifies the locale to be used by the statement generation logic when generating date and time strings. This is false by default and the default locale is US English. However, for compatibility with old clients, administrator may set this to true and it will generate date/time strings that are compatible with both older and new clients.

Security authentication does not work if the administration server is 5.0.1 or later and the client is pre-5.0.1. Setting the `repo.secureStatementUseOldLocale` property to true allows this combination to work.

- `java.property.tra_deploy_timeout`—When deploying large applications the TIBCO Hawk agent can time-out before deployment completes, causing it to fail. Depending on your TIBCO Runtime Agent installation version, this can be addressed as follows:
 - If TIBCO Runtime Agent 5.2.1 or greater is installed on the deployment target machines and on the machine that hosts the administration server, you can increase the value for the `java.property.tra_deploy_timeout` property in the `TIBCO_ADMIN_DOMAIN_HOME\domain-name\bin\tibcoadmin_domain-name.tra` file.
 - If you are using the `appManage` utility (version 5.2.1 or greater) to deploy applications, you can increase the value for the `java.property.tra_deploy_timeout` property in the `TRA_HOME\bin\AppManage.tra` file.
 - If TIBCO Runtime Agent 5.2.0 or greater is installed on the deployment target machines and TIBCO Runtime Agent 5.2.0 or greater is installed on the machine that hosts the administration server, you can increase the value for the `tra_deploy_timeout` property in the `TIBCO_TRA_DOMAIN_HOME\domain-name\hawkagent.cfg` file on each target machine where deployment is failing due to the Hawk agent timing out.
- `tibcoadmin.client.encoding`—Specifies the character encoding that is used to encode the HTML sent to the browser. In most cases UTF-8 is the best choice as it can handle all character sets. In some cases, a different encoding that is tailored to the specific character set(s) being used may offer slightly better performance.
- `tibcoadmin.autoRefreshInterval`—Specifies the interval, in milliseconds, that is used for the auto-refresh feature on the monitor screens. Default is 20000.
- `tibcoadmin.monitor.traceLogMaxLines`—Specifies the maximum number of lines to retrieve during a trace log search. Default is 1000.

- `DisableAdministratorClearLocks`—Objects stored in the administration, authorization and application domains can acquire locks. By default, these locks are cleared when the administration server starts. You can change this behavior by setting `DisableAdministratorClearLocks` to true so that locks are not cleared when the administration server starts for the administration domain where the property is set. The property can be set in the `AdministrationDomain.properties` and `AuthorizationDomain.properties` files, which are located in the `TIBCO_HOME\tra\domain\domain-name` folder.

File Storage Properties

The following properties affect administration domains that use a file repository for storage.

- `repo.directory`—Location in which the server will look for file-based projects.

Set by TIBCO Domain Utility to

`TIBCO_ADMIN_DOMAIN_HOME/domain_name/data`. If this property is removed or commented out, the location is set to `administrator/bin`.

You can also specify the path. Note that you must use *forward* slashes, for example, `c:/tibco/administrator/data`.

- `repo.fileType`—Optional. The file type is XML.

The format to use for storing projects on the server.

The values are in XML.

XML. XML has the advantage of being human readable, making it easy to use for development and debugging. However, XML format files are easily corrupted.

TIBCO Designer can export all projects, regardless of storage format, to an XML file. Similarly, TIBCO Designer can also import the contents of a TIBCO Repository XML file to any project, however, the export file cannot be used as a project.

Database Storage Properties

The following properties are required for administration server-based projects that use a database for storage. If values are not specified for any of these properties, the server uses a file for storage.

These properties must be defined for a database backend for project and application repositories.

- `repo.jdbcDriver`—Required if database storage is used.

There is no default value.

Name of the JDBC driver this server uses to connect server-based projects to their database. An example would be `oracle.jdbc.driver.OracleDriver`.

Must be commented for file-based storage.

- `repo.jdbcURL`—Required if database storage is used. There is no default value.

JDBC URL this server uses to connect its server-based projects to their database.

Must be commented for file-based storage.

- `repo.username`—Required if database storage is used. No default value.

JDBC database account user name this server uses to connect its server-based projects to their database.

Must be commented for file-based storage.

- `repo.password`—Optional. No default value.

JDBC database account password this server uses to connect its server-based projects to their database. This property is completely unrelated to TIBCO Administrator security or the password for the super user of the administration domain.

Must be commented for file-based storage.

- `repo.dbtype`—Required if database storage is used. No default value.

Database type. Values can be `Oracle`, `Db2`, `Sybase` or `Sql` (`Sql` is Microsoft SQL Server). Case is significant.

Must be commented for file-based storage.

- `repo.delayForDBStartup`—Optional. Default value is 0.

Changing this property is useful if a database is not ready immediately. It allows for a delay between when the server is started and when it first tries to communicate with the database. The delay is in seconds.

This property has no effect for file-based projects.

- `database.default.minConnections`—Number of database connections to cache per domain instance. No connections are allocated by default.
- `database.default.reconnectInterval`—Interval in milliseconds to ping to determine if a server has come back online. Default is 5 seconds.
- `database.default.pingInterval`—Interval in milliseconds to ping the server for keepalive through firewalls. Default is 5 minutes.

TIBCO Rendezvous Properties

The following properties are optional. In most cases, the default settings for these properties are appropriate.



Use TIBCO Domain Utility to change these properties. If you change these properties in the administration server properties file, your administration domain may become inconsistent.

- `repo.encoding`—Optional. The default value is ISO8859-1.

Specifies the character encoding for communication between the administration server and the components it manages.

Valid values are ISO8859-1 or UTF-8. To use a character set other than Latin-1, use UTF-8. See [Internationalization on page 92](#).



If you do not set this value to UTF-8, data loss may result if your data are multi-byte characters.

- `repo.rvDaemon`—Optional. The default value is `tcp:7500`.

This property specifies where to find the TIBCO Rendezvous daemon for this session. This allows a client to access a server on a different subnet without using `rvrd`.

- `repo.rvNetwork`—Optional. No default value.

Network to use. If not given, the primary network interface for the host computer is used. Set this property on computers that have more than one network interface to specify the interface to use.

- `repo.rvService`—Optional. Default is 7500.

Service group for this session. Set this property to partition servers on different networks.

Logging Properties

Information about administration server management operations can be logged to the console window, a file in a specified directory, or both.

In addition, log records can be sent to the Windows event log if you are running on a Windows operating system. File names are of the form *repo.logFileNamePrefix.number*, where *repo.logFileNamePrefix* is based on the property listed below and *number* is a unique ID. Each time the administration server is started, a new log file is created. By default twenty log files are kept. Log files normally grow only to a certain maximum size (default is 100KB). After reaching this limit, the log file is closed and a new log file is opened.

The following properties define how logging occurs.



The properties listed in this section only apply to project (data store) logging operations. TIBCO Administrator has a separate log for administration server operations. To look at that log, see

TIBCO_TRA_DOMAIN_HOME/domain-name/logs/Administrator.log

- `tibcoadmin.monitor.traceLogMaxLines`—Default value is 1000.

Specifies the max number of lines to retrieve in trace log search. If the TIBCO Administrator GUI is not able to display the log because it is too big, change this number or add filters to the log search.

- `repo.logOperations`—Optional. Default value is `false`.

If set to `true`, information about project management operations is logged. Information is logged to a file sink, as well as to the console.

To set up a file sink, use the `repo.logDirectory` property. To specify the level of the severity of events to log, use the `repo.logLevel` property.

- `repo.logDirectory`—Optional. Default value is `TRA_HOME/tra/domain/domain/administrator/version/tomcat`.

Path to directory. If set, logging is printed to the console and to a file in the specified directory.

If the property is not set, then the `log` directory under the current working directory is used.

File names are of the form *repo.logFileNamePrefix.number*, where *number* is a unique id, and the *repo.logFileNamePrefix* value is given as another property of this file. Each time the server is started, a new file is created.

- `repo.logFileNamePrefix`—Optional. Default value is `RepositoryServer.repo.serverName.log` where the `repo.serverName` value is given as another property of this file.

This property holds a prefix for the log file name. This property is effective only when `repo.logOperations` is set to `true`.

- `repo.logFileMaxNumber`—Optional. Default value is 20.

Maximum number of most recent log files that are maintained by the server. Only the most recent log files are kept.

- `repo.logFileMaxSize`—Optional. Default value is 100KB.

Log file size limit. Log files normally grow only to a certain maximum size. After reaching this limit, the file is closed and a new file is opened. The limit is in bytes.

- `repo.logLevel`—Optional. Default value is `Info`.

Specify the minimum severity level of events to log. The server supports the following logging levels.

— `Error`: only error messages are logged.

— `Warn`: warning messages are logged in addition to all the messages at `Error` level.

— `Info`: informative messages are logged in addition to all the messages of `Warning` level.

— `Verbose`: verbose messages are logged in addition to all the messages of `Info` level. This logs debugging messages.

- `repo.windowsEventLogLogLevel`—Optional. Default value is `None`.

Specifies the level of logging on Microsoft Windows Event Log. When the server is running on a Microsoft Windows system, it can write log messages on the Windows event log service as well as console or file. This property is effective only when the operating system is Microsoft Windows.

The server supports the following levels of logging.

- None: no messages are logged to the Microsoft Windows event log.
- Error: only error messages are logged to the Microsoft Windows event log.
- Warning: warning messages are logged in addition to all the messages of Error level.
- Info: informative messages are logged in addition to all the messages of Warning level.
- Debug: debug messages are logged in addition to all the messages of Info level.

Security Audit Properties

The following properties affect the security audit file:

- `repo.auditFileSize`
Specifies the default audit file size. Default value is 200,000.
- `repo.auditFileNum`
Specifies the default audit file number. Default value is 20.
- `repo.auditFileName`
Specifies the default audit file name. Default value is `audit.log`.

Load Balancing Properties

The following properties are available for load balancing of multiple administration servers.

- `repo.master`—Optional. Default is no value, in that case, the server does not perform load balancing.

The `repo.master` property must be set to either the host name or the IP address of the administration server you want to use as the primary server. The property is set during installation and can be changed by editing this file.



During domain creation, `repo.master` is set to the hostname of the server. When secondary servers are added, it is given the value "255.255.255.255".

As a side effect of enabling fault tolerance, a small `_RVFT` message is sent out once a second. If you started running in fault tolerant mode, but then wish to use only one server, you can disable fault tolerance by commenting out the `repo.master` property in the configuration (`.tra`) file. In that case, the `_RVFT` message is no longer sent.



If you comment out `repo.master`, you cannot add secondary servers until you have manually removed the comment.

- If the `repo.isMaster` property is set true, it overrides the `repo.master` property to allow the primary server to be specified in a cluster environment.
- `repo.state`—Optional. Defaults to `READ_WRITE`.

Specifies the global state. If other servers in the load-balanced group are up, this server will switch to their state before processing client requests.

You can set this property explicitly to `READ_ONLY` to have a primary server come up in that state upon startup.

`repo.state` will not effect Local deployment on a Database based domain, but will block RV or HTTP deployment on a Database based domain. Secondary servers should have the values set to the same as the primary server. This property should not be `READ_ONLY` unless the primary can lead to synchronization problems.



`repo.state` should not be used with file based domains and EMS based domains.

- `repo.prefixRegionalSubject`—Optional. Default is no value.

The `repo.prefixRegionalSubject` property specifies a prefix for the TIBCO Rendezvous subject. In load balancing mode, this prefix is used to consider servers with that prefix preferred servers for read operations.

This string is used for giving preference to a group of load balancing servers over the rest of the load balancing servers. Each client and server participating in load-balancing may have optional regional information. If this optional property is set, servers in load-balancing mode listen on the extra read subject that incorporates this region. If an optional region subject is set for a client, the client directs read operations to that subject first. If no server is available, then the client retries using the regular (non-prefixed) read subject.

- `repo.updateSecondaryServerPersistentStore`—Optional. Defaults to `true`.

The `repo.updateSecondaryServerPersistentStore` property only applies to secondary servers. It determines whether or not secondary servers working in load balancing mode write data into their persistent backend. If the backend is shared by the primary server and this secondary server, this property should be set to `false`. If this property is set to `true`, the secondary server will update not only its cache but also its backend whenever it synchronizes a change with the primary server. Currently this property is ignored by secondary servers that use a database backend. Default value for this property is `false`.

Server Security Properties

The following properties govern how the administration server handles security.

- `repo.isSecurityEnabled`—Optional. Default value is `true`.



This is the master switch for security. If its value is `false`, other security related properties have no meaning.

Specify whether the server is security-enabled.

If this property value is a text string set to `true`, the server is security-enabled. The server is always either secure or not. There is no way to programmatically turn security on or off.

- `repo.checkSecurityOnlyOnPolicyManagement`—Optional. Default value is `false`.

Specify whether the server checks security only on the operations that update security policies.

If this property value is a text string set to `true`, the server checks security only on the policy management operations that update security policies. Other operations do not get any security checked by the server.

This property is relevant only if the `repo.isSecurityEnabled` property is set to `true` for this server.

This property is for production time when replacing an old version of a legacy server with a new, secured server. Users can configure security policies first while other operations are still being served without security checked. Using the default value is strongly recommended.

- `repo.secureGuestPassword`—Optional. Default value is an empty string.

This property must match the password for the user specified by `repo.secureGuestUsername`. Use this option to allow 3.x applications to access an instance of a repository 4.x or later on a secure server.

This property requires that security be enabled via the `isSecurityEnabled` property.

- `repo.secureGuestUsername`—Optional. Default value is `guest`.

When security is enabled, this property specifies the user to use for 3.x applications. Use this option to allow 3.x applications to access an instance of a repository 4.x or later on a secure server. A client can continue accessing a project after its password has been changed until its security statement times out.

Any message received from a client that doesn't contain an authentication statement is treated with the authority of the `repo.secureGuestUsername` user. To create an authentication statement, the server uses `repo.secureGuestUsername` user with its password specified in another property of this file, `repo.secureGuestPassword`.

The `repo.secureGuestUsername` user must be explicitly defined in the security policy backend with its password specified by `repo.secureGuestPassword`.

It is perfectly valid for the `secureGuestUsername` user not to exist. Such a situation completely blocks access to unauthenticated users for all instances maintained by that server. Similarly, the `secureGuestUserId` user must be explicitly granted access rights in the same way any other user would be granted rights.

There are no automatic rights, nor are there any limits to the rights granted. All unauthenticated users share the same privileges.

This property requires that security be enabled via the `isSecurityEnabled` property.

- `repo.secureUsername`—Optional. Name of the user with full Administrator privileges.

This is the user specified as the domain administrator for the administration domain when it was created. The domain administrator can use the TIBCO Administrator user management module to specify additional users with full Administrator privileges.

This property is meaningful only when `repo.isSecurityEnabled` property is set to true.

- `repo.securePassword`—Optional. Password for the username specified in the `repo.secureUsername` property.

This is the password specified for the domain administrator for the administration domain. This property is meaningful only when `repo.isSecurityEnabled` property is set to true.

- `repo.secureStatementDuration`—Optional. Default value is 3600.

The number of seconds a security statement generated by a server is valid for a given client. Clients automatically renew their security statements when they expire.

- `repo.isRepoNavigatorEnabled`—Optional. Enables the use of the Repository Navigator.

When set (`repo.isRepoNavigatorEnabled=true`) the repository instances on the server can be viewed using the URL: `http://host:port/administrator/repo`.



When this property is enabled, any user can view global variables, which may be a security risk. If security is a concern, this property should not be enabled.

Persist Commit History Properties

The following properties are used to configure history files for the administration server.



Commit history files are different than revision histories, which are created by the deployment process and stored in the administration domain data. It is always safe to delete commit history files. They are used as an optimization for synchronization and HTTP client applications.

- `repo.commitHistoryDirectory`—Directory for commit history files. Default is a history directory inside the directory where Administrator was installed, for example, `c:\tibco\administrator\<version>\bin\history`.

If this property is set, commit files are generated in the specified directory. If the property is not set, then a history directory under the current working directory will be created and used.

File names are of the form "*project.number.commit*", where *project* is a project name, *number* is a commit number. By default there is no limit on the number of files. This can be changed by the `repo.commitHistoryFileMaxNumber` property.

- `repo.commitHistoryFileMaxNumber`—Optional. Default value is -1.

This property specifies the maximum number of commit files that are maintained for a project. If the number of files exceeds this limit, the file that has the oldest commit number will be overwritten.

If the value is set to 0, then no external file will be generated. If the value is set to -1, then an unlimited number of external files is generated.

Unless you use HTTP to access the application data, this property is not needed. It is recommended to set this value to 0.

- `repo.deploymentDirectory`—Deployment variable directory

This property specifies the directory where deployment variables are stored persistently. The name of the file will be *instanceName.substvar*. If this property is not set, the deployment variables will be stored in the directory holding file-based projects.

TIBCO Hawk Agent Properties



Use TIBCO Domain Utility to change these properties. If you change these properties directly in the administration server properties file, your administration domain may become inconsistent. These values effect the administration servers' own microagent.

- `repo.hawkImplant`—Optional. Default value is `true`.

This property specifies whether the server must create a TIBCO Hawk microagent (implant) when it starts.

If the value is set to `true`, the server creates a microagent. If the value is set to `false`, the server does not create a microagent. If the value is set to `true`, but TIBCO Hawk is not running, the server does not create a microagent.

- `repo.hawkDaemon`—Optional. The default value is `tcp:7474`

The TIBCO Rendezvous daemon property that specifies the TIBCO Hawk Rendezvous session used by this server.

- `repo.hawkNetwork`—Optional. No default value.

The rvd network for the server's Hawk RV session used by this Administrator process.

- `repo.hawkService`—Optional. Default value is `7474`.

The rvd service for the server's Hawk RV session used by this Administrator process.

- `repo.hawkMaxThread`—Optional. Default is `2`.

This value specifies how many threads should process TIBCO Hawk AMI requests for this server.

This value is limited to 9. If a number larger than 9 is used, the server defaults to 9 and ignores the given value.

- `repo.hawkMicroAgentName`—Optional. Default value is `COM.TIBCO.REPOSITORYSERVER`.

Specify the name of the TIBCO Hawk microagent for this server.

This name uniquely identifies this server as a TIBCO Hawk microagent in the Hawk console.

Java Properties

- `java.heap.size.initial`—defines the initial heap size for the Java VM that will be invoked.
- `java.heap.size.max`—specifies the maximum heap size. The JVM reserves this much memory at start-up and memory used for object allocation cannot exceed this amount.
- `java.thread.stack.size`—specifies the stack size.
- `java.extended.properties`—specifies any java property. For example, to set the java PermGen size to 128 megabytes:

`-XX:MaxPermSize=128m`

Insufficient PermGen space is indicated by the exception:

`java.lang.OutOfMemoryError: PermGen space`

Chapter 5 **Advanced Topics**

This chapter discusses advanced topics.

Topics

- [Server-Based Repository Locator String, page 80](#)
- [Local Repository Locator String, page 86](#)
- [Using a Properties File to Encapsulate Locator String Components, page 88](#)
- [Configuring Connection Pool Size for the Database Server, page 89](#)
- [Enabling an SSL Connection to an LDAP Directory Server, page 90](#)
- [Maintaining Connections to an LDAP Directory Server, page 91](#)
- [Internationalization, page 92](#)
- [Configuring EHCACHE, page 96](#)
- [Authenticating Users, page 98](#)
- [Repository Locking, page 102](#)
- [Global Variables, page 103](#)

Server-Based Repository Locator String

Applications built using the TIBCO Adapter SDK use a locator string to specify their application data’s repository location and other properties. The locator string is also used by repository management tools (see [Chapter 6, Command Line Utilities](#)), TIBCO BusinessWorks engines, and other TIBCO products.

Depending on how a client accesses the domain repository (4.x legacy projects) or application repository, the locator string differs, as discussed in the following sections:

- [TIBCO Rendezvous](#)
- [HTTP and HTTPS](#)

TIBCO Rendezvous

In case of a TIBCO Rendezvous transport, the locator string begins with `tibcr://` or `tibcr@`, followed by the instance name. In addition, the optional properties listed in [Table 8](#) are supported. Each property is separated by a colon.

Table 8 Optional properties for server-based locator string (TIBCO Rendezvous)

Property	Description
daemon	TIBCO Rendezvous rvd daemon value.
service	TIBCO Rendezvous rvd service value.
network	TIBCO Rendezvous rvd network value.
subject	Instance discovery subject.
discoveryTime	Timeout value in seconds for instance discovery.
timeout	Timeout value in seconds for server requests.
operationRetry	Number of retries when timeout occurs.
userName	Any identifier (null or empty implies read only with guest privileges).
password	User password for security.

Table 8 Optional properties for server-based locator string (TIBCO Rendezvous)

Property	Description
regionalSubject	TIBCO Rendezvous subject prefix used for regional read-operation in the load balancing mode. For additional information see Load Balancing Properties on page 71 .
typeAccess	<p>Type of client connection. Valid values are:</p> <ul style="list-style-type: none"> CLIENT_USAGE_DONT_CARE—Client reads until update, then switches to write. This is the default. CLIENT_USAGE_READ_ONLY—Client is not allowed to do updates. CLIENT_USAGE_READ_WRITE—Client can do both reads and updates.
urlFile	<p>Property file. The property file identifier can either be a fully qualified path or a relative path. The legal properties in this file are the same as optional properties specified above. The properties in the file are appended to the repository locator string. If the same property appears in both locator string and property file, the properties in the locator string take precedence. Property values starting with #! are considered marked for encryption.</p> <p>Encryption can be performed explicitly by running the obfuscate tool in <i>TIBCO_HOME\tra\version\bin</i>. The tool obfuscates all values starting with #!.</p> <p>For example, if a file contains this value:</p> <pre>repo.serverPassword=#!RepoTestServerPass</pre> <p>Running the obfuscate utility makes this change in the file:</p> <pre>repo.serverPassword=#!yecjXiS6JikkPrfNre8i/Hqb09afoyl6ACkGtzRPFFhaupSC1v0ApRLyyS+LOXtG</pre> <p>See the <i>TIBCO Runtime Agent Installation</i> guide for additional information.</p>

Examples

```
tibcr://myInst:service=5456:userName=ann:timeout=4000
tibcr@myInst:service=5456:urlFile=/tibco/props/fredsProps.txt
tibcr://myInst:urlFile=/tibco/props/fredsProps.txt
```

HTTP and HTTPS

In case of HTTP transports, the locator string begins with `http://`.

In case of HTTPS transports (HTTP over Secure Socket Layer) the locator string begins with `https://`.

Host name and port number are next (`http://host:port`).

- For Java clients, the port number is optional. If it is not specified, the default value used is 8080 for HTTP and 8443 for HTTPS.
- For C++ clients the port number is required.

The host name and port number are followed by the instance name, which is preceded by a question mark (?), for example, `http://host:8080/?inst1`

Optionally, `administrator/repo` may be included as part of the instance name, for example, `http://host:8080/administrator/repo/?inst1`



HTTPS-specific properties should be placed in a property file and that file should be specified using `urlFile=`. If you do not wish to specify HTTPS validation properties, `urlFile` is not a required property for HTTPS.

In addition, remote HTTP or HTTPS clients support the following optional properties separated by `&`. Note that when `&` is used as the separator of properties, and the URL is specified on the command line, the URL should be enclosed in quotes so that the shell does not interpret it.

Table 9 Optional properties for server-based locator string (HTTP)

Property	Description
timeout	Timeout value in seconds for server requests.
operationRetry	Number of retries if a timeout occurs.
userName	Any identifier (null or empty implies read only with guest privileges).
password	User password for security.

Table 9 Optional properties for server-based locator string (HTTP)

Property	Description
typeAccess	<p>Specifies whether access is read only or read-write. Valid values are:</p> <ul style="list-style-type: none">CLIENT_USAGE_DONT_CARE—Client reads until update, then switches to write. This is the default.CLIENT_USAGE_READ_ONLY—Client is not allowed to do updates.CLIENT_USAGE_READ_WRITE—Client can do both reads and updates.
urlFile	<p>Property file. The property file identifier can either be a fully qualified path or a relative path. The legal properties in this file are the same as optional properties specified above. The properties in the file are appended to the repository locator string. If the same property appears in both locator string and property file, the properties in the locator string take precedence. Property values starting with #! are considered marked for encryption.</p> <p>Encryption can be performed explicitly by running the obfuscate tool in <i>TIBCO_HOME\tra\version\bin</i>. The tool obfuscates all values starting with #!.</p> <p>For example, if a file contains this value: repo.serverPassword=#!RepoTestServerPass</p> <p>Running the obfuscate utility makes this change in the file: repo.serverPassword=#!yecjXiS6JikkPrfNre8i/Hqb09afoyl6ACkGtzRPFFhaupSC1v0ApRLyyS+LOXtG</p> <p>See the <i>TIBCO Runtime Agent Installation</i> guide for additional information.</p>

Examples

```
"http://host:8080?myInst&userName=ann&timeout=4000"
"http://host:8080/administrator/repo?myInst&userName=ann&timeout=4000"
"https://host:8443?myInst&urlFile=httpsProps.ini"
"https://host:8443/administrator/repo?myInst&server=s1&userName=bob&password=bswrd"
"https://host:8443/administrator/repo?myInst&urlFile=httpsProps.ini"
```

Creating a urlFile for Use With HTTPS

To use HTTPS, the following information must be included in the properties file:

Table 10 Properties Required for HTTPS urlFiles

Property	Description
<code>trustedCertFormat</code>	Format of the SSL certificate. Can be one of P12, P7, PEM, DER, EPF, or keystore. Only PEM is supported for C++ repository clients.
<code>httpsVendor</code>	Name of the SSL provider. For Java repository clients, this can be either <code>j2se</code> or <code>entrust61</code> . This string is case sensitive. For C++ repository clients, <code>openssl</code> is the only vendor supported so this property is ignored.
<code>keyFile</code>	Key file. Keys can either be embedded (P12 and keystore) or non-embedded (PEM, P7, DER). Key file is relevant only for non-embedded key files, that is, PEM, DER, and P7.
<code>keyPassword</code>	Password associated with the key file.
<code>identityFile</code>	Location of the identity file.
<code>identityType</code>	Format of the identity file. Can be one of P12, P7, PEM, DER, keystore, EPF.
<code>trustedCertPassword</code>	Password for the certificate specified by <code>trustedCerts</code> . This property is not relevant for C++ repository clients.
<code>trustedCerts</code>	Location of the trusted certificate or certificate chain.
<code>egdSocket</code>	Not relevant for Java applications. For C++ applications, using HTTPS transport on UNIX platforms other than Linux, this property specifies the location of the socket from which to get the random number.

Example for Java Applications

Following is an example for a url file containing HTTPS-specific properties for Java applications:

```
httpsVendor=j2se
trustedCerts=H:/downloads/certs/clientcerts/trustedcerts/RSA/PEM/
                                     RSA1024ca1.cert.PEM
trustedCertFormat=PEM
trustedCertPassword=RSA1024ca1
identityFile=H:/downloads/certs/clientcerts/identity/RSA/P12/
                                     RSA1024ca2.cert.P12
identityType=P12
keyPassword=RSA1024ca2
```

Example for C++ Applications

Following is an example for a url File containing HTTPS specific properties for a C++ application:

```
httpsVendor=j2se
trustedCerts=H:/downloads/certs/clientcerts/trustedcerts/RSA/PEM/
                                     RSA1024ca1.cert.PEM
trustedCertFormat=PEM
trustedCertPassword=RSA1024ca1
identityFile=H:/downloads/certs/clientcerts/identity/RSA/P12/
                                     RSA1024ca2.cert.P12
identityType=P12
keyPassword=RSA1024ca2
egdSocket=/etc/egd-pool
```

Local Repository Locator String

Applications built using the TIBCO Adapter SDK use a locator string to specify the repository location and other properties. For local repositories, the string starts with the instance name, which can optionally be preceded by `localrepo@`. The instance name can either be a fully qualified path or a relative path. The `.dat` extension is optional.

In addition, clients support the following optional properties separated by colons:

Table 11 Optional properties for local repository locator string.

Property	Description
userName	Any identifier (if not present or empty makes a read-only client).
typeAccess	<div>Type of client connection. Valid values are:<ul style="list-style-type: none">CLIENT_USAGE_DONT_CARE—Client reads until update, then switches to write. This is the default.CLIENT_USAGE_READ_ONLY—Client is not allowed to do updates.CLIENT_USAGE_READ_WRITE—Client can do both reads and updates.CLIENT_USAGE_REACQUIRE_INSTANCE_LOCK—Client is allowed to overwrite this local repository even if a lock file exists, as long as it’s the same user.CLIENT_USAGE_FORCE_INSTANCE_LOCK—Client is allowed to overwrite this local repository even if a lock file exists. WARNING: Using this option may result in other users overwriting your project.</div> <div>See Repository Locking on page 102 for additional information.</div>

Table 11 Optional properties for local repository locator string.

Property	Description
urlFile	<p>Property file. The property file identifier can either be a fully qualified path or a relative path. The legal properties in this file are the same as optional properties specified above. The properties in the file are appended to the repository locator string. If the same property appears in both locator string and property file, the properties in the locator string take precedence. Property values starting with #! are considered marked for encryption.</p> <p>Encryption can be performed explicitly by running the obfuscate tool in <code>TIBCO_HOME\tra\version\bin</code>. The tool obfuscates all values starting with #!.</p> <p>For example, if a file contains this value:</p> <pre>repo.serverPassword=#!RepoTestServerPass</pre> <p>Running the obfuscate utility makes this change in the file:</p> <pre>repo.serverPassword=#!yecjXiS6JikkPrfNre8i/Hqb09afoyl6AckGtzRPFFhaupSC1v0ApRLyyS+LOXtG</pre> <p>See the <i>TIBCO Runtime Agent Installation</i> guide for additional information.</p>

Examples

```
./instances/myInst.dat:userName=deborah
c:/tibco/administrator/myinstances/myInst.dat:urlFile=c:/tibco/
administrator/props/deborah
myProj.dat
myProj
myProj/myrepo.dat
```

Using a Properties File to Encapsulate Locator String Components

Locator strings are included in an applications properties file in the `tibco.configurl` and `tibco.repourl` fields. If, however, you do not want to expose the complete locator string, for example, to protect your passwords, you can instead point to a `urlFile` as part of the locator string:

```
tibcr@default:userName=myName:urlFile=abc
```

Any `:name=value` url option to a utility tool can be used as a property *name/value* pair in the property file. The *name/value* pairs in the property file are appended to the url string. Note that this option is designed for command line utilities and is not available through deployment using the TIBCO Administrator GUI.

Defining a URL

Suppose you want to define the following url.

```
tibcr@default:userName=myName:password=myPassword
```

You can define the url using the `myPropertiesFile` as follows.

```
tibcr@default:userName=myName:urlFile=myPropertiesFile
```

Then provide the password in `myPropertiesFile`. Obfuscate the file using the `obfuscate` tool in `TIBCO_HOME/tra/version/bin`.

Property Precedence

If the same *name/value* pair appears in both url string and property file, the *name/value* pair in the url string takes precedence. You can overwrite the *name/value* pair in property file with the value directly given in url string.

For example, if you have the following:

```
tibcr@default:userName=myName:urlFile=myPropertiesFile
```

and `password=myName` is defined in `myPropertiesFile`. The final url string is

```
tibcr@default:userName=myName:password=myPassword
```

Configuring Connection Pool Size for the Database Server

When using a database-based domain, it is recommended that you fine tune the database server's JDBC connection pool to avoid running out of connections.

The following components in a domain keep their own JDBC connection pool for database access:

- each primary administration server
- each secondary administration server
- each application domain on an administration server
- each Hawk agent
- AppManage utility

The maximum JDBC connections available to each component described above is defined by the `tibcoadmin.database.maxConnections` property in the following two files for your domain:

- `TIBCO_HOME/domain/domain-name/AdministrationDomain.properties`
- `TIBCO_HOME/domain/domain-name/AuthenticationDomain.properties`

By default, both `.properties` files set the `tibcoadmin.database.maxConnections` property to 10. This means each of the components described above uses a maximum of 20 connections (the sum of the administration domain and the authentication domain). You must multiply this number by the total number of components running in your domain.

Enabling an SSL Connection to an LDAP Directory Server

SSL (Secure Sockets Layer) is a network protocol that allows authentication and encryption of data. SSL provides a secure connection between a client and a server.

You can use SSL to secure the user and group data transmitted to your TIBCO servers and applications from the LDAP directory server. Doing so ensures privacy, integrity, and authenticity of data from the LDAP directory server.

TIBCO Domain Utility specifies SSL usage for the LDAP integration of an administration domain. Once SSL is specified for a domain's LDAP integration, the administration servers and applications depend on the security features of the JVM they run on to establish SSL connections with the LDAP server (do not actively participate in establishing the SSL connections).

To configure an administration domain to connect to the LDAP directory server over SSL, do the following:

1. Enable SSL authentication on the LDAP directory server. You may need to contact the IT department in your organization that manages your LDAP servers. This requires installing a valid server certificate and CA trust certificate from a certificate authority on the LDAP directory server.
2. Import the CA trust certificate for your LDAP server certificate into the keystores of all JREs associated with the administration domain on each machine. This includes JREs for all primary and secondary servers, as well as for BusinessWorks processes that perform basic authentication. Note that the TIBCO JRE keystores already contains certificates from well-known certificate authorities such as Verisign, Thawte and Entrust, so you can skip this step if your LDAP server certificate is issued by one of these certificate authorities.
3. When integrating your administration domain with the LDAP directory server using TIBCO Domain Utility (see *Changing a Domain's Integration With an LDAP Directory Server* in *TIBCO Runtime Agent Domain Utility User's Guide*), select **SSL** in the LDAP Authentication drop-down list.

See *Configuring LDAP Integration With SSL Connections* in *TIBCO Runtime Agent Domain Utility User's Guide* for detailed instructions.

Maintaining Connections to an LDAP Directory Server

TIBCO applications maintain a pool of connections to an LDAP directory server for authorization purposes. These connections are automatically reconnected when these are found to be broken.

The connection pool properties are defined with default values when the installer sets up TIBCO Administrator. The values for some of the properties can be overridden in the `tibcoadmin_domain-name.tra` file, which is written in the `TIBCO_ADMIN_DOMAIN_HOME\domain-name\bin` folder after the domain is created.

These properties apply to both LDAP directory server connections and database connections. The properties are listed next with their default values:

- `java.property.tibcoadmin.database.maxConnections=10`
- `java.property.tibcoadmin.database.expiryInterval=1800000`
- `java.property.tibcoadmin.database.waitTimeoutInterval=30000`

The `expiryInterval` and `waitTimeoutInterval` property values are specified in milliseconds.

At runtime if a value is specified for one of the above properties in the `tibcoadmin_domain.tra` file, it is used instead of the value set at install time.

The connections are maintained for subscribing to group information changes. They are automatically reconnected when found to be broken. These connections must be kept alive when a TIBCO application is across a firewall, since a firewall can break these connections after certain period of inactivity.

When these connections are reconnected, it is necessary to perform complete synchronization of group information again. Similarly when real-time subscriber connections used by the TIBCO Administrator service are reconnected, the group-synchronized roles must be synchronized again. See the *TIBCO Administrator User's Guide* for details.

Internationalization

This section discusses internationalization. You learn about the following topics:

- [Introduction](#)
- [Design-Time Encoding vs. Runtime Encoding](#)
- [Setting Encoding for an Administration Domain](#)

Introduction

TIBCO products can store, process, display and transmit data in multiple character sets including Latin-1, Korean, Japanese, Chinese, and so on. To support this, encoding is set as follows:

- To communicate with vendor applications (such as SAP, Oracle, Siebel), TIBCO products use native encodings (like ISO-8859 series, Microsoft Windows series, Shift_JIS, Big5, GB2312, etc.). The encoding for communicating with the vendor application depends on the nature of the data exposed by the application and must be set during configuration of the TIBCO product.
- For internal representation of text data, TIBCO products use UCS-2. The TIBCO products take care of the encoding conversion between UCS-2 and any vendor application's native encoding, according to the encoding specified during configuration of the TIBCO product.
- For internal communication, TIBCO products use ISO8859-1 or UTF-8 as the wireformat encoding
 - ISO8859-1 is the preferred encoding for exchanging ASCII or Latin-1 (ISO8859-1) data between TIBCO products or product components
 - UTF-8 must be used for data in character sets other than ASCII and Latin-1.

This wireformat encoding is called *TIBCO Messaging Encoding* and is determined by the encoding property of the administration domain in which the TIBCO application is running. Having one encoding per administration domain ensures that all TIBCO products in the administration domain use the same encoding to communicate.

Design-Time Encoding vs. Runtime Encoding

At design-time, you specify a TIBCO messaging encoding for a project. This encoding is used when the project is running in debug mode, or if you run it as a legacy project using a local repository (that is, exported as .dat file).

After the project has become a deployed application in TIBCO Administrator, the administration domain encoding supersedes the project-level encoding. This assures all TIBCO products working in the domain use the same wireformat encoding to communicate.

When an enterprise archive file is loaded, the design-time encoding is verified to be compatible with the domain encoding.



If a .ear file created using UTF-8 is deployed in a non-EMS domain, it gives a validation error and prompts the user to change the parameters values `repo.encoding` and `tibcoadmin.client.encoding` of the .tra file for the TIBCO Administrator. There is no such issue in the EMS domain that uses ISO 8859 - 1.

Setting Encoding for an Administration Domain

The encoding for an administration domain is set during domain creation. You can create multiple administration domains on the same machine, each using a different encoding.



Secondary servers automatically use the same encoding as the primary server.

For the encoding:

- ISO8859-1 should be used as the preferred encoding for projects in which the text data to be exchanged is only ASCII or Western European data.
- UTF-8 must be used for projects that handle data other than ASCII and Western European character set. Note that all the world's major language characters can be represented using UTF-8.

Default Encoding

When you install an administration server and create an administration domain, the default encoding used for the domain is ISO8859-1 (Latin-1). The value is used for different properties of which three are in the `tibcoadmin_domain.tra` file:

- `tibcoadmin.client.encoding`—encoding that is used to encode the HTML that displays the TIBCO Administrator GUI.
- `file.encoding`—encoding of the `tibcoadmin_domain.tra` file contents.
- `repo.encoding`—TIBCO Messaging encoding (discussed in the previous section). This encoding used by TIBCO Administrator and other TIBCO products in the domain to communicate.

- `-character_encoding`?used by the TIBCO Hawk agent. The property is defined in the `hawkagent.cfg` file.

Changing Encoding for a Domain

In some situations, it may be necessary to choose UTF-8 as the encoding for all these properties. TIBCO Administrator stores encoding information per domain in two files. One is the domain's `.tra` configuration file used to launch TIBCO Administrator, and the other is the configuration file used by the TIBCO Hawk agent.

For example, to change the encoding for a domain called `myDomain`:

1. Change directory to `TIBCO_ADMIN_DOMAIN_HOME\domain-name\bin`.
2. Using a text editor, open the `tibcoadmin_myDomain.tra` file.
3. Change the `tibcoadmin.client.encoding` entry to UTF-8.
4. Change the `repo.encoding` entry to UTF-8.
5. Save the file and restart the administration server.
6. Change directory to `TIBCO_TRA_DOMAIN_HOME\domain-name`.
7. Using a text editor, open the `hawkagent.cfg` file.
8. Change the `-character_encoding` entry to UTF-8.
9. Save the file and restart the TIBCO Hawk agent

The `client.encoding` value can be any valid encoding value. It is used to enforce the encoding of the contents pushed to the browser. The `repo.encoding` value is used as TIBCO Rendezvous encoding and only UTF-8 and ISO8859-1 are supported.

If this domain has a secondary server or if other machines join this domain, you must apply the same changes on each secondary server and machine.

Changing Selected Properties

In some situations, you may wish to change the value of one, but not all properties. For example, if you are going to use `aeXm1` as the message format, you must change `repo.encoding` to UTF-8 even if you are actually using ASCII characters. If you are using a `.tra` file that uses non-Latin1 characters, you must change `file.encoding` to UTF-8.

You can edit individual properties in the `tibcoadmin_domain-name.tra` file.



The application's configuration data in persistent storage is always UTF-8. This includes TIBCO Designer projects in different formats (including legacy server-based projects with a database backend) and TIBCO Administrator application repositories. The encoding for these files is independent of the TIBCO Messaging encoding chosen for the administration domain.

The following table shows some example for which project content should be used with what TIBCO Messaging encoding.

Project Content	TIBCO Message Encoding
Latin-1, ASCII 7-Bit	ISO8859-1 (preferred) or UTF-8
Latin 2 (Eastern European) Korean characters, Chinese characters, Japanese characters	UTF-8

Users and Passwords with Double-Byte Characters

You can create users and assign passwords with double-byte characters if you configure the administration domain to use UTF-8.

- If Local Application Data is used, run-time encoding can be set at deployment time.
- If TIBCO Enterprise Message Service is used for administration domain communication, design-time encoding is used by default at runtime.

Configuring EHCACHE

TIBCO Administrator includes the EHCACHE product that provides a simple, fast and thread-safe cache facility. A default cache configuration is configured in the `ehcache-failsafe.xml` file, which is included in `ehcache-1.7.2.jar`. It is used unless you create and configure a custom cache file. The default configuration stores 5000 objects of each object type in memory without expiration.

EHCACHE allows you to configure caching per object type. You can specify that only certain objects are cached after the first iteration. More information about configuring EHCACHE is available from the following location:

<http://ehcache.sourceforge.net/documentation/>

To create and configure a custom cache file, you must:

1. Use winzip or another similar utility to unpack the EHCACHE jar file in:
`TIBCO_HOME\tpcl\version\lib\ehcache-1.7.2.jar`
2. Copy the `ehcache-failsafe.xml` file to a working directory and use an XML editor to configure the file for your environment.

To configure the default cache to use for all object types in the store, edit the `defaultCache` configuration in the `ehcache-failsafe.xml` file. To configure a cache for a specific object type, create a cache configuration with the full Java class name of the object type in the `ehcache-failsafe.xml` file. In such a configuration, typically only the `maxElementsInMemory` value is changed. For example:

```
<cache name="class name" maxElementsInMemory="number" eternal="true"
      overflowToDisk="false"/>
```

3. After modifying the `ehcache-failsafe.xml` file, copy it to the `TIBCO_TRA_DOMAIN_HOME/domain-name` directory (the same directory that contains the `AdministrationDomain.properties` file). Note that if a cache is not specified for an object type, the default cache's configuration is used to create a new cache for the object type.
4. Copy the file to `domainName_ehcache.xml` and `AUTH_domainName_ehcache.xml` for each domain, or put the `ehcache-failsafe.xml` file back into the jar file for use with all domains.

Writing Cache Statistics to a Log File

Periodically, cache statistics can be dumped to a log file. This provides information needed to alter the cache configurations. Each store can configure how often it wants all of its cache statistics dumped to the INFO role of the Administrator.log file. The following properties can be added to and configured for each store (in `AdministrationDomain.properties` and `AuthorizationDomain.properties`):

- `CacheDumpInterval`: Set to the number of milliseconds to dump the cache statistics to the INFO role. If less than 0 or not set, the statistics are never dumped. By default, this property is not set.
- `CacheDumpSizeEnabled`: Set to "true" if the amount of memory used by the cache should be dumped as well. This is a separate option because determining the size of the cache is an expensive operation — around one second per MB of cache size. All use of the cache *stops* for this period and the application will seem to stop as well. If this property is not set, it defaults to "false".



If your dump interval is shorter than the length of time it takes to generate the cache size, the application will effectively freeze until this is not the case, which may require termination.

The INFO log message has the following form for the default cache or a store wide cache:

```
[ name = acmedb:admin:
com.tibco.pof.admindomain.AdministrationDomain status = active
eternal = true overflowToDisk = false maxElementsInMemory = 10000
timeToLiveSeconds=120 timeToIdleSeconds = 120 diskPersistent =
false diskExpiryThreadIntervalSeconds = 600 hitCount = 50
memoryStoreHitCount = 50 diskStoreHitCount = 0 missCountNotFound =
203 inMemorySize = 76454 ]
```

See the EHCACHE documentation for details on each of these attributes (<http://ehcache.sourceforge.net/documentation/>). The `inMemorySize` element is the number of bytes of the objects in the cache. This number is the size of the Java serialization of the objects and does not account for any transient variables (which should not significantly add to the size).

The name of the cache is a concatenation of the following. The `ClassLoaderHashCode` element is a number (an internal identifier).

storeName:userName:ClassLoaderHashCode/entityTypeClassName

Similarly, the name of a cache that is specific to an object type is a concatenation of the following. The `storeImplementationClassName` element can be removed if each application domain is uniquely named.

storeName:userName:storeImplementationClassName/objectTypeClassName

Authenticating Users

TIBCO Administrator implements the Java Authentication and Authorization Service (JAAS) framework. The framework allows you to perform web authentication and authorization in applications that use the administration server.

You must create an administration domain that uses HTTPS. The JAAS Authentication web service is then deployed on the tomcat server for the administration domain.

Authentication Using the JAAS Authentication Web Service

The Internet provides many tutorials for using the JAAS framework. See <http://java.sun.com/developer/JDCTechTips/2001/tt0727.html> for a tutorial that shows the JAAS Authentication web service checking the identity of a caller. Other tutorials may be available on the Internet that show the same functionality.

The above URL displays a tutorial that provides an introduction to JAAS and explains how to configure a JAAS LoginModule to validate passwords. The only difference between what is presented in the tutorial and how to use the Authentication web service is in the tutorial's Configuration File section where the following lines

```
SimpleLogin {
    SimpleLoginModule required;
};
```

instead would be:

```
AuthenticationService {
com.tibco.atlas.auth.jaas.AuthenticationServiceLoginModule
    soap_username="admin"
    soap_password="#!P3A46TfV7Mf4bOGq0cCUnad3hQ1lcn2W"
    authority="putYourAdministratorHostNameHere:8443"
    scheme="https"
};
```

The password is encrypted using the obfuscate utility. See the *TIBCO Runtime Agent Installation* guide for details about using the utility.

Authentication Code Example

This example provides example code that shows how to validate a password for a user. The administration domain must have SSL enabled for the example to work. The following arguments specify the username and password to test and are used to configure the connection to the service provider.

- `username` — name of the user to validate.
- `obfuscatedPassword` — the obfuscated password of the user.
- `superuserName` — the name of a superuser defined in TIBCO Administrator.
- `obfuscatedSuperuserPassword` — the obfuscated password of the superuser.
- `authority` — the *host:port* specification of the server where TIBCO Administrator is running. The default is **localhost:8443**

A `MalformedURLException` is thrown if the service's URL is not formed correctly. This occurs if an unknown authority is set.

```
package sample.com.tibco.atlas.authentication;

import java.net.MalformedURLException;
import com.tibco.atlas.auth.authentication.AuthenticationService;
import com.tibco.atlas.auth.authentication.AuthenticationServiceFactory;

public class AuthenticateUser {

    public static void main(String[] args) throws MalformedURLException {
        String username = (args.length>0) ? args[0] : "user";
        String obfuscatedPassword = (args.length>1) ? args[1] : "";
        String superuserName = (args.length>2) ? args[2] : "superuserName";
        String obfuscatedSuperuserPassword = (args.length>3) ? args[3] : "";
        String authority = (args.length>4) ? args[4] : "localhost:8443";

        try{
            boolean retVal = isUserValid(username, obfuscatedPassword, superuserName,
                                         obfuscatedSuperuserPassword, authority);

            System.out.println("Password for user " + username + " is " + ((retVal) ?
                                                                    "valid" : "not valid"));
        }
        catch(Exception ex)
        {
            {
                ex.printStackTrace()
            }
        }
    }

    public static boolean isUserValid(String username, String obfuscatedPassword,
        String superUserName, String obfuscatedSuperuserPassword, String authority)
        throws Exception {
        AuthenticationServiceFactory factory = new AuthenticationServiceFactory();
```

```

        factory.setAuthority(authority);
        factory.setSoapUsername(superUserName);
        factory.setObfuscatedSoapPassword(obfuscatedSuperuserPassword);
        AuthenticationService service = factory.createAuthenticationService();
        return service.isPasswordValid(username, obfuscatedPassword);
    }
}

```

Logging Information to a File

By default, the Authentication web service prints all runtime trace information to the TIBCO Administrator console. This section explains how to configure the Authentication web service to print trace information to a file.

The following system property needs to be set to the path of a logging configuration file. The property is then specified at startup in the client application command line.

`-Djava.util.logging.config.file=`*path to logging properties file*

Your logging properties file should be similar to the following:

```

# Logging Properties File

# "handlers" specifies a comma separated list of log Handler
# classes. These handlers will be installed during VM startup.
# Note that these classes must be on the system classpath.
# To log to the console use java.util.logging.ConsoleHandler below:
handlers= java.util.logging.FileHandler

# Default global logging level.
# This specifies which kinds of events are logged across
# all loggers. For any given facility this global level
# can be overridden by a facility specific level
# Note that the ConsoleHandler also has a separate level
# setting to limit messages printed to the console.
package.level= FINEST

# The default file output is in user's home directory.
java.util.logging.FileHandler.pattern = path to log file
java.util.logging.FileHandler.limit = 50000
java.util.logging.FileHandler.count = 1
java.util.logging.FileHandler.level = INFO

# XML can be output using java.util.logging.XMLFormatter:
java.util.logging.FileHandler.formatter = java.util.logging.XMLFormatter

```

- `handlers` specifies the set of handlers to be loaded on startup.

- `package.level` sets the default global logging level for each package. The values are ALL, OFF, FINEST, FINER, FINE, CONFIG, INFO, WARNING, SEVERE.
- The global logging level can be overridden for a specific handle.
 - `pattern` provides the path to the log file for the given handle. If the path is not specified, the file is placed in the user's home directory.
 - `limit` is the maximum size of the output file in bytes.
 - `count` is the number of output files to use. An integer is appended to the base file name.
 - `level` is the logging level. It uses the same values as the global logging level.
 - `formatter` describes the output style, simple or XML. The value can be `java.util.logging.SimpleFormatter` or `java.util.logging.XMLFormatter`

Repository Locking

When an administration server accesses a repository for update, the repository is marked as locked so that other servers cannot update the same repository.

- For server-based repositories, the actual lock occurs the first time a client tries to connect to the administration server in read/write mode, not at server initialization.
- For a local file repository, the locking occurs if the repository is accessed, the username is non-empty and the access type is not read-only.

Once a repository is locked, it remains so until the server is cleanly shut down. For local file repositories, the lock is released when the client is destroyed.

- Files. In the case of a file-based repository (for both local and server-based repositories), a lock file is created which specifies the locking user name or server name and lock time.

The file exists in the same directory as the project file, but has a `.lck` extension.

- Database. In the case of a database backend, the Instance table has the `locker` column set to the server name for that particular instance.

How to Break the Lock

In some situations, a repository may remain locked although it is actually no longer in use.

For server-based repository, you can break the lock by setting the `repo.forceInstanceLocks` property in the `tibcoadmin_domain-name.tra` file to `true` and restarting the administration server. See also [Administration Server Properties File on page 57](#) for additional details.

For local file repositories, a lock file exists in the same directory as the project file and has a `.lck` extension rather than a `.dat` extension. You can force the removal of a lock on a repository by deleting its `.lck` file.

Global Variables

You can define variables for your project, either during configuration or at deployment. Global variables are defined in TIBCO Designer during configuration.

A TIBCO project contains a number of predefined global variables. For some of the variables, default values are assigned. You can use TIBCO Designer to specify additional variables and, optionally, define their values.

At runtime, a client application can choose to overwrite the variable values or use the predefined value for the global variable.

Examples for global variables include:

- Use a variable for your TIBCO Rendezvous service. Then supply the value 7001 during testing and 7002 for production runs.
- Allow your clients to set a TIBCO Rendezvous subject. For example, the same adapter can be run in different applications using two different subjects.



Global variables are stored as XML data, so only valid XML characters are allowed for variable values.

Defining Variables in TIBCO Designer

To define and use variables within TIBCO Designer:

1. In the TIBCO Designer project window, choose the Global Variables tab.
2. Click the Pen icon.
3. Click the abc icon and specify the variable name and (optionally) the value as a string.
4. To use a variable, copy and paste the variable into any location in your project where you need it.
5. When you save the project, the variables and their values are included.

If your application doesn't override the variable value, it is used. The program code can, however, override it. You can also override it upon startup using a command-line argument.

Defining Service-Settable Variables in TIBCO Designer

If you wish to be able to change the value of a variable on a per-service basis, you can do so following these steps:

1. In the TIBCO Designer project window, choose the Global Variables tab.
2. Click the pen icon to bring up the Global Variables Editor
3. Specify the variable, an optional value, and whether it should be settable on a per-deployment, per-service basis, or both.



For more information about defining variables, see the *TIBCO Designer User's Guide*.

Setting Variable Values

You can preset variable values when you define the variables. You can then override any existing values as part of your application code or in the properties file for the application. For some applications, you can also override variables using the command-line.

Variables set on the command line override variables set in the properties file and variables set in the properties file override variables set in the application.

Change a variable value using the TIBCO Administrator GUI by selecting:

- An application's Advanced tab to set application-specific variables.
- A service's Advanced tab to set service-specific variables.
- A service instance's Advanced tab to set service instance-specific variables.

A variable value can be changed in a properties file or on the command line:

- Locally define the value for a variable in the properties file for a specific domain repository using the `tibco.clientVar.varname` property. The local value takes precedence over any global value.
- Specify a value for a variable through a command-line argument. This overrides all other specification.

```
-system:clientVar varName=value
```

Multiple `-system:clientVar` options can be specified on the command line. If the same variable already exists in the command line, the later one replaces the first one. No space is allowed for either `varName` or the value, when using `-system:clientVar`.



You can also set variable values using the AppManage utility. This can be useful if only a few out of a large number of global variables change between domains.

Chapter 6 Command Line Utilities

This chapter provides the command-line syntax and a brief discussion of each command line utility used for project management.

The AppManage command line utility is used to deploy applications. This utility is described in the *TIBCO Runtime Agent Scripting Deployment User's Guide*.

Topics

- [Command List, page 106](#)
- [CorpRoleSynchronizer, page 108](#)
- [CorpUserSynchronizer, page 110](#)
- [ExportDomainSecurity, page 111](#)
- [ImportDomainSecurity, page 116](#)
- [MoveMachine, page 119](#)
- [RedeployAllAppsForUser, page 123](#)
- [RepoConvert, page 126](#)
- [RepoPing, page 130](#)
- [AppStatusCheck, page 146](#)

Command List

The user account used to run most command line utilities must have `Write` permissions set in the TIBCO Administrator GUI for the application, application repository, and domain repository that is being changed. Some command utilities require the user to be a member of the Super Users role.



Many of the commands use a repository URL to point to a project. The URL syntax is discussed in [Server-Based Repository Locator String on page 80](#) and [Local Repository Locator String on page 86](#)

Table 12 Command List

Command	Description
CorpRoleSynchronizer	Syncs an administration domain with its associated LDAP directory.
CorpUserSynchronizer	Pre loads user objects into an application.
DeleteInvalidUsers	Removes the users from an LDAP domain if the users do not exist in the associated LDAP directory.
ExportDomainSecurity	Exports security data from a domain. This includes users and roles.
ImportDomainSecurity	Imports security data into a domain.
MoveMachine	Moves a domain from one machine to another.
RedeployAllAppsForUser	Updates password changes to all deployed applications in a given domain.
RepoConvert	Converts back and forth between single-file (.dat) and multi-file projects.
RepoCreateInstance	Creates a new project.
RepoDelete	Deletes nodes from a domain repository.
RepoDeleteInstance	Deletes a repository instance.
RepoDiff	Logically compares two projects and outputs the differences.

Table 12 Command List

Command	Description
RepoExport	Exports all parts of a project to an XML file.
RepoImport	Loads data from a file created with RepoExport into a project.
RepoListInstances	Lists currently available projects found at the location specified by a URL. The URL can either be local or remote.
RepoPing	Checks whether a file based administration server is running and communicating.
RepoRename	Renames a node in a local single-file or server-based project. Do not use this command if you have saved the project in multi-file format.
AppStatusCheck	This utility is used to query status of all deployed applications in a domain.

CorpRoleSynchronizer

Command

```
CorpRoleSynchronizer -domain domain-name [-h|-?]
```

The command must be started with an option or an exception will result.

Purpose

The CorpRoleSynchronizer command line utility syncs an administration domain with its associated LDAP directory. The sync occurs based on the search criteria for LDAP groups that was defined when the administration domain was created.

Your administration domain may be out of sync because the auto sync settings for the domain were not been enabled when using TIBCO Domain Utility to configure the domain, or significant changes have been made to the LDAP directory since the last automatic sync and you do want to wait for the next auto sync cycle to occur, or do not want to do an manual sync from the TIBCO Administrator GUI.

A summary of results is provided in the console where you launched the command and in the TIBCO Administrator log file.

DeleteInvalidUsers

Command

```
DeleteInvalidUsers -domain <domain> [-h|-?]
```

This command only applies to domains which sync with an LDAP server.

Purpose

It Removes the users from an LDAP domain if the users do not exist in the associated LDAP directory. You need to specify the name of the LDAP domain when using this command. The domain name is case sensitive.

Location

TIBCO_HOME\tra\version\bin

CorpUserSynchronizer

Command

```
CorpUserSynchronizer -domain domain-name [-clean] [-h|-?]
```

The command must be started with an option or an exception will result.

Purpose

When an application is running, a new user profile is created when a user first accesses the application. At that time, the default objects for the user’s profile are created in the administration domain’s LDAP repository.

Because of this activity, the administration server can become overloaded if many new users access the application in a short period of time. For example, this is especially applicable to TIBCO PortalBuilder applications directly after they go into production. In these situations, performance will be improved if you run the CorpUserSynchronizer command.

Pre loading users is generally done before an application goes live, but can be done at any time when a mass import of users would be useful.

Property	Description
domain-name	Name of an administration domain.
clean	Removes users in the administration domain who are no longer found in the LDAP directory.

If you have a large number of users to import, the import operation may fail after processing only part of the LDAP directory’s contents. This happens because of low default values in certain iPlanet and Sun ONE Directory Server settings.



It is assumed that your corporate LDAP (primary as well as any referral LDAPs) is appropriately configured so that LDAP search queries return all matching users in that LDAP.

- For iPlanet and Sun ONE Directory Servers, the `timelimit`, `sizelimit` and `look-throughlimit` are appropriately set.
- For Microsoft Active Directory, the `NTDSUTIL` utility is used to configure the appropriate settings. See the Active Directory documentation for instructions.

A summary of results is provided in the console where you launched the utility and in the TIBCO Administrator log file.

ExportDomainSecurity

Command

```
ExportDomainSecurity -domain domain [-allUsers | -minimalUsers]
[-userIncrement number] [-roleMembership] [-roles [role1:role2: ... :
roleN] ] [-super] [-acls] [-date date] [-GUID] [-writeDates]
[-encryptKey key] ((-user user -pw password) | -cred cred) -file file
```

Purpose

This command exports administration domain information to an XML file. The file can be loaded back into the domain or into another domain using the ImportDomainSecurity command.

The ExportDomainSecurity and ImportDomainSecurity commands are designed for data loading (for example, batch loading users into a domain), synchronizing user and role data between domains, and backing up domain data.

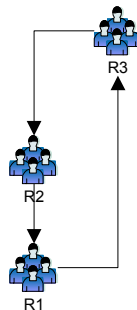
You can use ExportDomainSecurity to create example templates.

- Either -allUsers or -minimalUsers must be specified to export users.
- Use the -GUID flag to keep two domains synchronized.

Data exported from an administration domain integrated with an LDAP directory server can only be imported into another LDAP based domain. Similarly, data exported from a non-LDAP domain can only be imported into a non-LDAP domain.

Circular role hierarchies can be defined in an LDAP directory. For example, as shown next, R1 is a child of R2 which is a child of R3 which is a child of R1.

Figure 12 Circular Role Hierarchy



Circular role hierarchies are not supported in TIBCO Administrator. If you import a role hierarchy that contains circular role hierarchies into another LDAP based domain, the LDAP synchronization will fail. To resolve this, you must remove the roles that are part of the circular hierarchy from the exported file and import the file into your target LDAP based domain. When TIBCO Administrator syncs with the LDAP directory server, the roles you removed will be restored.

Table 13 *ExportDomainSecurity Properties*

Property	Description
<code>-domain domain</code>	Name of a domain to export from (case sensitive).
<code>-allUsers</code>	Exports all users.
<code>-minimalUsers</code>	Exports minimal set of users only (only relevant if <code>roleMembership</code> is specified).
<code>-userIncrement number</code>	If specified, a new file will be created after this many users has been exported. The file will have a dash and an incrementing number appended to the value of <code>-file</code> .
<code>-roleMembership</code>	Exports the list of users belonging directly to the role for standard roles.
<code>-roles role1:role2: ...: roleN</code>	List the roles to export, each separated by a colon. If specified without any values, all roles are exported. If not specified, no roles are exported.
<code>-super</code>	If specified, members or the super users role are exported.
<code>-acls</code>	If specified, all access control lists are exported.
<code>-date date</code>	Only entities updated since the given date are exported. The syntax for date must conform to that required of the Java method, <code>Date.parse()</code> . For example, "Sat 13 Nov 2004 13:30:00 GMT".
<code>-GUID</code>	Include GUIDs for improved merge on import. Note that if you do not specify this property, there is no way to determine whether a user or role has been renamed. Without this property, all user and role names are considered new names.
<code>-writeDates</code>	Outputs last modification dates (automatic if GUID is specified).

Table 13 *ExportDomainSecurity Properties*

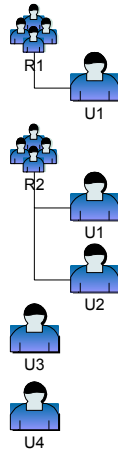
Property	Description
-encryptKey <i>key</i>	User passwords will be encrypted using the specified key (and a DES3 encryption algorithm). If a key is used to encrypt, that key must be supplied to ImportDomainSecurity when importing the file. If you lose the key, the password data is lost.
-user <i>user</i>	Name of an authorized user.
-pw <i>password</i>	Password of an authorized user. The password is case sensitive and should not be encrypted.
-cred <i>cred</i>	<p>Name and path to a property file that contains the user name and encrypted password. Use the obfuscate utility to encrypt the password. The utility is documented in the <i>TIBCO Runtime Agent Installation</i> guide. For example, the property file could contain this entry after running the utility:</p> <pre>user=admin pw=#!YZIAIiZ5DnkhFMqP+RUQkiSYHqP6jIGU</pre> <p>If this property is used, -user and -pw should not be specified.</p>
-file <i>file</i>	Name of the file in which the data is stored.

Examples

The following command line exports only users who have been created or changed after the given date.

```
exportdomainsecurity -domain tp0513 -allUsers -date "Tue 09 Nov
2004 12:30:00 GMT" -user admin -pw admin -file \tibco\export.xml
```

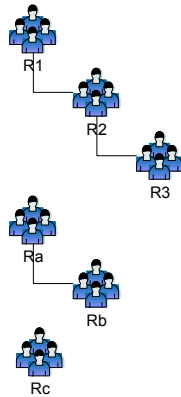
Given the following users and roles:



The following command exports only users U1 and U2, and roles, R1 and R2.

```
exportdomainsecurity -domain tp0513 -roles -minimalUsers -user
admin -pw admin -file \tibco\export.xml
```

Given the following roles:



The following command exports roles R1, R2, R3, Ra, and Rb.

```
exportdomainsecurity -domain tp0513 -roles R1 Ra -user admin -pw admin -file
\tibco\export.xml
```

The following command exports roles R3, and Rb.

```
exportdomainsecurity -domain tp0513 -roles R3 Rb -user admin -pw admin -file
\tibco\export.xml
```

The following command exports roles R2, R3 and Rc.

```
exportdomainsecurity -domain tp0513 -roles R2 Rc -user admin -pw admin -file  
\tibco\export.xml
```

Location

TIBCO_HOME\tra\version\bin

ImportDomainSecurity

Command

```
ImportDomainSecurity -domain domain [-noUsers | -noRoles] [-noACLs]
[-overwrite] [-encryptKey key] ((-user user -pw password) | -cred cred)
-file file
```

Purpose

This command imports domain security information from an XML file. The file is created using the ExportDomainSecurity command.

The ExportDomainSecurity and ImportDomainSecurity commands are designed for data loading (for example, batch loading users into a domain), synchronizing user and role data between domains, and backing up domain data.

You can use this command to load user, role and access control information from another source. The XML file you create must conform to the DomainSecurity.xsd file. The DomainSecurity.xsd file is located in the TRA_HOME\template\domainutility\cmdline directory.

Only valid XML characters can be used in user names, role names and properties.

Data exported from a domain integrated with an LDAP directory server can only be imported into another LDAP based domain. Similarly, data exported from a non-LDAP domain can only be imported into a non-LDAP domain.



After exporting LDAP group synchronized roles from an LDAP based domain and importing the roles into another LDAP based domain, all the exported roles will display in the imported LDAP based domain. The roles that are not part of the imported LDAP based domain will disappear when the administration server synchronizes with its LDAP directory.

An export file that contains circular role hierarchies is not supported. See the [ExportDomainSecurity](#) command for more information.

Table 14 ImportDomainSecurity Properties

Property	Description
-domain <i>domain</i>	Name of the domain to import (case sensitive).
-noUsers	Do not import users.
-noRoles	Do not import roles.
-noACLs	Do not import ACLs except those protecting an imported role.

Table 14 ImportDomainSecurity Properties

Property	Description
-overwrite	<p>If specified, existing role values are replaced. If not specified, existing role values will be appended. If a role in the file has no membership, even if this property is specified, the role members will be left unchanged.</p> <p>This property removes any parents to a role not found in the import file and removes any members from standard roles not found in the imports membership list, unless the membership list is empty.</p> <p>This property has no effect on users, non-standard role membership configurations, or role properties.</p>
-encryptKey <i>key</i>	Provide the key that was used to used to encrypt the user passwords stored in the file. If you lose the key, the password data is lost.
-user <i>user</i>	Name of a user in the domain that has Super User permissions assigned.
-pw <i>password</i>	Password of the authorized user (case sensitive). The password can either encrypted or be in clear text.
-cred <i>cred</i>	<p>Name and path to a property file that contains the user name and encrypted password. Use the obfuscate utility to encrypt the password. The utility is documented in the <i>TIBCO Runtime Agent Installation</i> guide. For example, the property file could contain this entry after running the utility:</p> <pre>user=admin pw=#!YZIAIiZ5DnkhFMqP+RUQkiSYHqP6jIGU</pre> <p>If this property is used, -user and -pw should not be specified.</p>
-file <i>file</i>	Name of the file in which the data is stored.

Example

```
importDomainSecurity -domain tp0513 -user admin -pw admin -file
\tibco\export.xml
```

Location

TIBCO_TRA_DO_HOME\bin

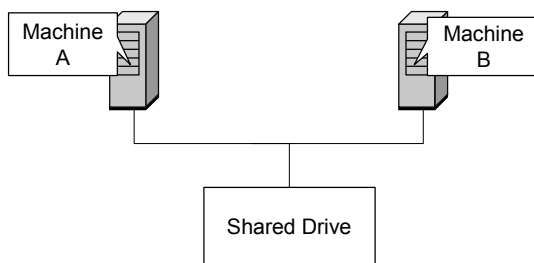
MoveMachine

Command

```
MoveMachine -domain domain-name [-oldMachine old machine name]
[-newMachine new machine name] [-doAdminUI] [-help]
```

Purpose

This command moves a domain from one machine to another machine. For example, the next diagram shows two machines, A and B that used the same shared drive. The moveMachine command can move the domain from machine A to machine B. Note that machine B must not be part of the administration domain.



The command changes all references from the old machine to the new machine in the specified domain.


Note that:

- In all cases, the TIBCO Hawk agent must be stopped on the old machine for the specified domain and started on the new machine.
- All service instances and process engines running on the old machine must be stopped before running the MoveMachine command.
- If you moving a 5.1 domain and have not upgraded the domain to 5.2 or greater, and have registered an EMS server in the domain, you should use TIBCO Domain Utility on the new machine to update the EMS server after using the MoveMachine command.
- All TIBCO software installed on the old machine must be installed on the new machine.
- Both machines must be running the same operating system.



The command does not update HTTP repository URIs for deployed components.

Table 15 MoveMachine Property

Property	Description
-domain <i>domain-name</i>	Name of the administration domain to move.
-oldMachine <i>old machine name</i>	The existing machine name in the domain files that is to be replaced. The name is also used for validation. If not specified, all domain files are updated, but without validation.
-newMachine <i>new machine name</i>	<p>The new machine name to put in the domain files. If not provided, the hostname of the machine on which the command is run is used.</p> <p>The new machine must not already be part of the administration domain specified in -domain.</p> <p>Two machines that share the same drive on <i>TIBCO_HOME</i> cannot be in the same administration domain.</p>
-doAdminUI 	<p>If specified, updates the TIBCO Administrator web page that lists all available domains.</p> <p>This changes entries for all domains from the old machine to new machine. The property is applied only if the oldMachine property was specified.</p>
-help	Prints this help information.

Moving an Administration Domain on a Server Machine

- To move an administration domain on a machine that hosts the administration server to another machine:
1. Ensure that all TIBCO software installed on the server machine is installed on the new machine.
 2. Stop the TIBCO Hawk agent on the server machine.
 3. Stop all service instances and process engines that are running in the administration domain on the server machine.
 4. Either use a shared drive, or copy the administration domain from the server machine to the new machine. For example, if the files are under `c:\tibco` on the server machine, you must copy the entire directory (`c:\tibco`) to the new machine. The directory path on the new machine must be identical to that on the server machine.

5. Execute the following command on the new machine. In the example command line, `machineA` is the server machine that hosts the administration server and `machineB` is the machine to which the administration domain is moved.

```
MoveMachine -domain tp034 -oldMachine machineA -newMachine machineB
```

6. Stop the administration server on the server machine.
7. Start the administration server on the new machine.
8. Start the TIBCO Hawk agent on the new machine.
9. Start the service instances and business processes on the new machine.

Moving an Administration Domain on a Client Machine

To move an administration domain on a client machine to another machine:

1. Ensure that all TIBCO software installed on the client machine is installed on the new machine.
2. Stop all service instances and process engines that are running on the client machine in the administration domain.
3. Stop the TIBCO Hawk agent on the old machine.
4. Either use a shared drive, or copy the administration domain from the client machine to the new machine. For example, if the files are under `c:\tibco` on the client machine, you must copy the entire directory (`c:\tibco`) to the new machine. The directory path on the new machine must be identical to that on the client machine.
5. Execute the following command on the new machine. In the example command line, `machineC` is the client machine that hosts the administration domain and `machineD` is the machine to which the administration domain is moved.

```
MoveMachine -domain tp987 -oldMachine machineC -newMachine machineD
```

6. Start the TIBCO Hawk agent on the new machine.
7. Start the service instances and business processes on the new machine.

Moving an Administration Domain on a Secondary Server Machine

To move an administration domain on a machine that hosts a secondary server to another machine:

1. Ensure that all TIBCO software installed on the secondary server machine is installed on the new machine.
2. Stop the TIBCO Hawk agent and secondary server on the old machine.

3. Stop all service instances and process engines that are running in the administration domain.
4. Copy the administration domain from the secondary server machine to the new machine. For example, if the files are under `c:\tibco` on the secondary server machine, you must copy the entire directory (`c:\tibco`) to the new machine. The directory path on the new machine must be identical to that on the secondary server machine.
5. Execute the following command on the new machine. In the example command line, `machineE` is the secondary server machine that hosts the administration domain and `machineF` is the machine to which the administration domain is moved.

```
MoveMachine -domain tp987 -oldMachine machineE -newMachine machineF
```

6. Start the TIBCO Hawk agent on the new machine.
7. Start the secondary server on the new machine.

Renaming a Machine

To rename a machine:

1. Stop all service instances and process engines that are running in the administration domains on the machine.
2. Stop the TIBCO Hawk agent on the machine.
3. Execute the following command on the new machine. In the example command line, `machineG` is the old machine name and `machineH` is the new machine name.

```
MoveMachine -domain tp987 -oldMachine machineG -newMachine machineH
```

4. Stop the administration server if one is running on the machine.
5. Change the machine name.
6. Start the TIBCO Hawk agent on the machine.
7. Start the administration server on the machine.
8. Start the service instances and business processes on the machine.

Location

```
TIBCO_HOME\tra\version\bin\
```

RedeployAllAppsForUser

Command

```
RedeployAllAppsForUser  
-domain domain  
[-desc description]  
(-user user -pw password | -cred cred  
(-targetUser targetUser -newPW new pass | -file file)
```

Purpose

Changes the password in all deployed applications’ properties files (.tra files) that are used to retrieve configuration data from the administration server. The command determines which applications in the given domain use the specified userid, and then updates their configurations and redeploys them. This is a convenience compared to doing these steps manually either in the TIBCO Administrator GUI or through command line deployment.

Table 16 *RedeployAllAppsForUser Property*

Property	Description
domain	Name of the administration domain. The name is case sensitive.
user	Name of a user that is authorized to change the deployed application.
password	Password for the above user.
cred	<p>Name of a file containing the user and encrypted password. Use the obfuscate utility to encrypt the values in the file. Each entry in the file must use the following syntax:</p> <p><i>user=password</i></p> <p>Use either the -cred property, or the -user and -password properties, not both.</p>
targetUser	Name of a user that is used by the deployed application to read its administration server based configuration data.
new pass	Updated password for the targetUser.
file	<p>Absolute path to a file that contains a list of target users and passwords. Each entry in the file must use the following syntax:</p> <p><i>targetUser=password</i></p> <p>Use either the -file property, or the -targetUser and -new pass properties, not both.</p>



If the `-file` property is specified or if the `-user` property value doesn't match the `-targetUser` property value, then the user running this command must be a super user in the domain.

Be very careful that you have no undeployed changes to the relevant applications as these will be propagated to the deployed applications when this command is run.

RepoConvert

Command

```
RepoConvert
  -in repoURL1
  [-out repoURL2]
  [-r project root]
  [-n]
  [-y]
  [-v log file]
  [-o]
  [-t trace property file]
  [-d directive file]
  [pattern]
  [pattern] . . .
```

Purpose

Converts projects from single-file (.dat) to multi-file format or back. By default (no -n or -o option):

- If the target project does not exist, create a new one.
- If the target project already exists, the content of the target project is completely deleted first, then new data is written in the target project.



For TIBCO BusinessWorks 2.x projects, use bw20migrate rather than RepoConvert or TIBCO Designer. The bw20migrate command is in the TIBCO BusinessWorks bin folder.

Table 17 RepoConvert Property

Property	Description
-in repoURL1	Source project.

Table 17 RepoConvert Property

Property	Description
<code>-out repoURL2</code>	<p>Target project.</p> <ul style="list-style-type: none"> • If the source project is in single-file (.dat) format, the target project must be a multi-file project • If the source project is a multi-file project, the target project must be in single-file (.dat) format • If the target project does not exist, a new project is created. If the input is a multi-file project, and the output project does not exist, the output project created in that case is in XML. • If the source project is in .dat format, and the target project does not exist, <code>-r <project root></code> must be specified. <p>Either this property or <code>-r</code> (for multi-file format) must be specified.</p> <p>repoURL2 cannot be the same as repoURL1.</p>
<code>-o</code>	<p>If you specify this flag, RepoConvert appends new information to the target project and overwrites existing nodes in the target project if target project exists.</p> <ul style="list-style-type: none"> • For .dat to multi-file project migration, append/overwrite applies to files in the multi-file project. • For multi-file to .dat migration, append/overwrite applies to project nodes. <p>When both <code>-n</code> and <code>-o</code> options are specified, <code>-o</code> option takes precedence. RepoConvert behavior in that case is the same as if <code>-o</code> only had been specified.</p>
<code>-v log file</code>	<p>Validates the multi-file format during multi-file to .dat migration. If not specified, no validation is performed. If a log file is specified, validation errors and warnings are written to it.</p>
<code>-t trace property file</code>	<p>A property file specifying trace options. Defaults to <code>repoConvert.ini</code></p>
<code>-d directive file</code>	<p>An directive file with a list of patterns. One pattern per line. You can specify <code>-d</code> or <code><pattern></code> but not both.</p>

Table 17 RepoConvert Property

Property	Description
pattern	Pattern to be matched, such as /tibco/private/adapter/* You can specify either -d or <pattern>, but not both.
-r project root	A file URI specifying the root directory of the multi-file project. If no target project is specified, and the source project is in .dat format, the project is migrated to the specified root directory. Either this property or -out must be specified.
-n	If you specify this flag, RepoConvert appends to the target project but does not overwrite information in the target project if the target project exists. <ul style="list-style-type: none">For .dat to multi-file project migration, new files are appended and existing files are not overridden in the multi-file project directory.For multi-file to .dat migration, new nodes are appended and existing nodes files are not overwritten in the .dat file. When both -n and -o options are specified, the -o option takes precedence. RepoConvert behavior in that case is the same as if -o only had been specified.
-y	Enable rename of 4.x project components (no rename is the default). In TIBCO Designer 4.x, moving resources had no effect on the project file. In TIBCO Designer 5.x, the file layout corresponds to the GUI layout. If you convert a 4.x project to TIBCO Designer 5.x multi-file format, and you wish to use TIBCO Designer 4.x mapping information to rename the files that are generated, you can use the -y option.

Single-file (.dat) to Multi-file Project Examples

Convert...	Using...
Local file	RepoConvert -in sdk_example.dat -out c:/repos/se/vcrepo.dat

Convert...	Using...
Server-based project using TIBCO Rendezvous	<code>RepoConvert -in tibcr://sdk_example -out c:/repos/se/vcrepo.dat</code>
Server-based project using HTTP	<code>RepoConvert -in http://host:8080?sdk_example -out c:/repos/se/vcrepo.dat</code>
Local file to directory	<code>RepoConvert -in sdk_example.dat -r c:/repos/se</code>
Only classes	<code>RepoConvert -in sdk_example.dat -out ./se/vcrepo.dat /tibco/public/class/*</code>

Multi-file to Single-file (.dat) Project Examples

Convert...	Using...
Local file	<code>RepoConvert -in c:/repos/se/vcrepo.dat -out sdk_example.dat</code>
Remote RV	<code>RepoConvert -in c:/repos/se/vcrepo.dat -out tibcr://sdk_example</code>
Local file with validation	<code>RepoConvert -in c:/repos/se/vcrepo.dat -out sdk_example.dat -v convert.log</code>
Only AE schemas (no overwrite)	<code>RepoConvert -in c:/repos/se/vcrepo.dat -out sdk_example.dat -n /AESchemas/*</code>

Location

`TIBCO_HOME\tra\version\bin\`

RepoPing

Command

```
RepoPing -url url [-retries retries sleep_interval]
```

Purpose

Checks whether a project can be accessed. This could be server based application data or domain data for file based domains. Checking if the project holding domain data is accessible is an easy way to determine that the Administration server is running for file based domains.



RepoPing utility works only with File based domains.

Table 18 RepoPing Property

Property	Description
-url url	<p>Standard repository locator string for the project. The URL must be specified and must be the first property.</p> <p>Use RepoPing -help url for information on the URL format.</p> <p>The URL must be specified.</p>
-retries retries	<p>Number of times to try connecting to the project. If you don't specify this number, the default number of retries is 10.</p> <p>This property is optional.</p>
sleep_interval	<p>Number of milliseconds to sleep between retries. If you don't specify this number, the default value is 600.</p> <p>This property is optional.</p>

Returns

Return values are:

- 0 Successful connection
- 1 Instance unavailable
- 2 Exception raised while connecting to instance - indicates a problem with the environment or a bad URL string

Examples

```
RepoPing -url
tibcr://SYS_filerv57:filerv57:daemon=tcp:7500:service=7500:network
=10.105.68.32:username=fred:password=pw
```

The above command attempts to ping an Administrator server having domain name as filerv57. The Rendezvous session parameters are specified using daemon, service and network options. The expected output is:

```
C:\tibco\tra\<version>\bin>repoping -url
tibcr://SYS_filerv57:filerv57:daemon=tcp:7500:service=7500:network
=10.105.68.32:username=vkhatri:password=vkhatri
Trying to connect to Repo Instance:
tibcr://SYS_filerv57:filerv57:daemon=tcp:7500:service=7500:network
=10.105.68.32:username=vkhatri:password=vkhatri
Nov 10, 2010 1:04:51 PM
com.tibco.security.providers.SecurityVendor_entrust61 <init>
INFO: Initializing Entrust crypto provider in NON FIPS 140-2 mode;
insert provider as normal
Connected to repository:
tibcr://SYS_filerv57:filerv57:daemon=tcp:7500:service=7500:network
=10.105.68.32:username=fred:password=pw
```

Location

TIBCO_HOME\tra\version\bin

RepoCreateInstance

Command

```
RepoCreateInstance -url url [-configFile configFile]
```

Purpose

Create a new project (repository instance). This command is primarily used with 4.x instances.

Table 19 RepoCreateInstance Parameter

Parameter	Description
-url url	Location of the project. Type RepoCreateInstance -help url for information on the url format. (-url must be specified). To create a project managed by a remote server, the server name must be provided by the server parameter inside url. See Server-Based Repository Locator String on page 80 .
-configFile configFile	Name of the instance property file. Its contents is used to initialize the InstanceInfo properties. If this argument is not given, RepoCreateInstance uses default property files. The default property files are localInstanceInfo.ini and remoteInstanceInfo.ini. If the URL specifies a remote instance, the default property file is remoteInstanceInfo.ini. If the url specifies a local instance, the default property file is localInstanceInfo.ini.

Examples

The following command (entered on one line) creates a server-based project.

```
RepoCreateInstance -url
tibcr://myIns:server=serv1:service=9000:username=sky:password=blue

RepoCreateInstance -url localrepo@/temp/myInst.dat
```

Location

```
TIBCO_HOME\tra\version\bin\
```


RepoDelete

Command

```
RepoDelete -url url RootNode RootNode ...
```

Purpose

This is only useful with very old versions of the product. Delete the node at the specified location in the project. This command is primarily used with 4.x instances.

RepoDelete can be combined with RepoExport to completely maintain a project through scripts.

Notes

If url points to a project saved in Multi-Project format, the following applies:

- If all nodes mapped to the file are deleted, then the file will be deleted.
- If multiple nodes (as in the case of an adapter) map to the particular file, and only some of them are deleted, the file will merely be updated.

Table 20 RepoDelete Parameter

Parameter	Description
-url url	Deletes from the specified URL. Type RepoDelete -help url for information on the url format. url must be specified.
RootNode	Node to delete. If the specified node is a directory node, all child nodes of the directory node are also deleted. At least one node must be specified.

Examples

```
RepoDelete -url test.dat /tibco/public/ae/class/class1  
/tibco/public/ae/union/union1
```

```
RepoDelete -url tibcr://test:service=7500  
/tibco/public/ae/class/class1
```

Location

```
TIBCO_HOME\tra\version\bin
```

RepoDeleteInstance

Command

```
RepoDeleteInstances -url url [-help url]
```

Purpose

Used to delete a repository instance from the specified URL. Type **RepoDeleteInstances -help url** for information on the url format. This command is primarily used with 4.x instances.

Table 21 RepoDeleteInstance Parameters

Parameter	Description
-url url	Deletes from the specified URL. Type RepoDeleteInstance -help url for information on the url format. url must be specified.

Examples

```
RepoDeleteInstance -url test.dat  
RepoDeleteInstance -url tibcr://test:service=7500
```

Location

```
TIBCO_HOME\tra\version\bin
```

RepoDiff

Command

```
repoDiff [-help | -?] [-i] [-s url] [-x url] [-v] repoUrl1 repoUrl2
```

Purpose

Logically compares two projects and outputs the differences. This command is primarily used with 4.x instances.

The output report contains the following information:

- Differences in global variables and in instance variables.
- Differences in project tree content.
- Summary of all differences.

Table 22 RepoDiff Parameter

Parameter	Description
-help -?	When you use this option, a help message is displayed. Type RepoDiff -help url for information on the url format.
-i	Compare the two projects ignoring case.
-s url	Start node for comparison. Only nodes descending from url are included in the comparison.
-x url	When an exclude url is specified, RepoDiff excludes the subtree that starts at this url.
-v	Verbose reporting. When specified, the complete content change is reported; otherwise, only the existence of a change is supported.
repoURL1 repoURL2	URL for the two instances to compare.

Examples

```
RepoDiff .\instance1.dat .\instance2.dat
```

```
RepoDiff -i tibcr://instance1 tibcr://instance2
```

```
RepoDiff -x /tibco/public http://localhost:8080?instance1  
http://localhost:8080?instance2
```

Location

TIBCO_HOME\tra\version\bin

RepoExport

Command

```
RepoExport -url url [options] file[RootNode] [RootNode]
```

Purpose

Use this command to export data from a project to an XML file. This command is primarily used with 4.x instances.

Table 23 *RepoExport Parameter*

Parameter	Description
-url <i>url</i>	Exports from the specified URL. Type RepoExport -help url for information on the url format. A valid <i>url</i> must be specified.
options	You can supply one or more options. See Options below.
file	The file that will be written.
RootNode (Optional)	The nodes from which to start exporting. If not specified, exports the entire project. Multiple root nodes are supported.

Option	Description
-bypassInvalidRoot	Export process skips invalid roots, prints a warning, and continues exporting from other valid roots.
-format <i>format</i>	Output file format. Valid file formats are generic and AEXML . If not specified, the default is generic . If you use generic format, XSD validation cannot be performed when you re-import the file.
-nochase	Contents of referenced nodes are not exported. By default referenced nodes are exported as if they had been specified as root nodes.
-exclude <obj>	Do not export anything at or below the specified directory or object. Use multiple -exclude arguments to exclude multiple directories/objects.

Option	Description
-substitute [<i>name=value...</i>]	Substitute global variables in exported strings. If this option is not specified, the strings are exported in their raw format. There can be an arbitrary number of name=value pairs which override any matching global variables.
-fullname	Global names are not shortened in AEXML format. By default, they are shortened.

Examples

```
RepoExport -url /repos/MyLocalRepo.dat -format AEXML MyRepo.exp  
  
RepoExport -url tibcr://MyRemoteRepo -format AEXML EntireRepo.exp  
  
RepoExport -url tibcr://MyRemoteRepo -exclude /tibco/public/class  
-exclude /tibco/public/sequence -format AEXML JustPrivate.exp  
/tibco/private
```

See Also

[Appendix D, Schema Files used when Exporting and Importing](#)

Location

```
TIBCO_HOME\tra\version\bin
```

RepoImport

Command

```
RepoImport -url url [options] import file [schema files]
```

Purpose

Import the specified file into a project using the specified options. This command is primarily used with 4.x instances. For more information see [Appendix D, Schema Files used when Exporting and Importing](#).

Table 24 RepoImport Parameter

Parameter	Description
-url url	Import from the specified URL. Type RepoImport -help url for information on the url format. -url must be specified.
options	You can set one or more options. See Options for AEXML or Generic format below.
import file	It is the instance file. The instance file may have any extension except .xsd. It must be in one of the formats used by RepoExport.
schema files	You can use one or more XML schema files. The schema file must have extension .xsd. This argument does not apply to instance files in generic format.

Options for AEXML or Generic format

Option	Description
-overwrite	If this option is used, imported objects replace existing objects if same name object exists. Otherwise, existing objects are left unchanged. Objects with names that don't currently exist are imported.

Option	Description
-overwriteVar	<p>Imported objects replace existing global variable if variables of the same name exist. Otherwise, existing global variables are left unchanged.</p> <p>Objects with names that do not currently exist are imported.</p>
-nullRef	<p>If this option is <code>true</code>, references that point to non-existing nodes are changed to null references. Otherwise, an empty object is created.</p>
-exclude <i>object</i>	<p>Do not overwrite the contents of the given directory or object. Use multiple <code>-exclude</code> arguments to specify multiple objects and directories.</p> <p>New objects are still added to this directory.</p>

Options for AEXML only

Option	Description
-caseSensitive	Case-sensitive XML validation in enumeration facet matching. This option does not apply if the instance file is in generic format.
-continue	Continue the import after encountering a validation error in the instance file. By default, <code>RepoImport</code> terminates if a validation error is encountered.
-lev 1 2 3	<p>The error/warning level generated by validator.</p> <ul style="list-style-type: none"> • 1 means error and warning, • 2 means error only, • 3 means warning only. <p>Default is 1.</p>
-novalidate	Do not validate instance files in AEXML format.
-log <i>file</i>	Log validation error in the specified log file. If no log file is specified, validation errors can be found in <code>default.log</code> in the current directory.

Examples

```
RepoImport -url /repos/MyLocalRepo.dat -continue MyLocalFile.exp  
Repository.xsd myAdapter.xsd AESDK.xsd AEService.xsd AESchema.xsd
```

```
RepoImport -url tibcr://MyRemoteRepo -novalidate MyLocalFile.exp  
Repository.xsd
```

```
RepoImport -url tibcr://MyRemoteRepo -exclude /tibco/public/class  
-exclude /tibco/public/sequence -continue JustPrivate.exp  
Repository.xsd
```

Location

TIBCO_HOME\tra\version\bin

RepoListInstances

Command

```
RepoListInstances -url url
```

Purpose

Use RepoListInstances to list instances at the location specified by a URL. The URL can either be local or remote. This command is primarily used with 4.x instances.

- If local, the directory should be specified rather than a specific instance.
- If remote, the server may be specified to get instances from a particular server, otherwise all instances will be lists.

The list goes to standard output and contains each instance name on a separate line.

Table 25 RepoListInstances Parameters

Parameter	Description
-url url	URL specifying the project for which you want to list the instances. Type RepoDelete -help url for information on the url format. The URL must be specified.

Examples

```
RepoListInstances -url c:/temp  
RepoListInstances -url tibcr://:server=TEST  
RepoListInstances -url tibcr://
```

Location

```
TIBCO_HOME\tra\version\bin
```

RepoRename

Command

```
RepoRename -url url [-type type] [-ignoreErr] -file file | oldName
newName
```

Purpose

Renames a node in a project. This command is primarily used with 4.x instances.



Do not rename the `SYS_domain-name` and `AUTH_domain-name` repository instances that store domain information. If you do, the administration server can no longer operate properly.

Table 26 RepoRename Parameters

Parameter	Description
-url <i>url</i>	Renaming starts at this URL. The URL must be specified. Type RepoRename -help url for information on the url format.
-type <i>type</i>	<p>Specifies the type of node to rename. This provides a prefix for convenience. If you do not specify the type, you must fully qualify the node names. Even if you specify the type, you may use an absolute path for one or both names. See Types below for a list of legal types</p> <p>Names beginning with a slash are absolute paths and will not have the type prefix applied.</p> <p>This parameter is optional.</p>
-ignoreErr	<p>If this option is specified, errors are ignored and processing continues if multiple items were specified for rename. If this option is not specified, processing stops when the first error is encountered.</p> <p>This parameter is optional.</p>
-file <i>file</i>	<p>Specify a file with a list of name pairs for rename. There can be multiple nodes to rename, but only one per line. Format of the file is a set of lines each consisting of oldname = newname</p> <p>Either -file or <oldname newname> must be specified, but not both</p>

Table 26 RepoRename Parameters

Parameter	Description
<i>oldname</i> <i>newname</i>	Node name for the node existing in the project that is to be renamed, followed by the target name. Either <code>-file</code> or <code>oldname newname</code> must be specified, but not both

Types

The following types can be used in conjunction with the `-type` option.

gn	fully qualified global name
association	/tibco/public/association/ae
class	/tibco/public/class
aeclass	/tibco/public/class/ae
scalar	/tibco/public/scalar
aescalar	/tibco/public/scalar/ae
sequence	/tibco/public/sequence
aesequence	/tibco/public/sequence/ae
union	/tibco/public/union/ae
channel	/tibco/public/channel
endpoint	/tibco/public/endpoint
session	/tibco/public/session
adapter	/tibco/private/adapter
mb	/tibco/private/mb

Examples

```
RepoRename -url tibcr@inst1 -type aeclass foo bar
RepoRename -url tibcr@inst1 -type scalar ae/i4 ae/integer
RepoRename -url tibcr@inst1 -type scalar ae/i4 com/i4
RepoRename -url tibcr@inst1 -type scalar ae/myI4
/tibco/private/adapter/myAdapter/myScalars/I4
RepoRename -url tibcr@inst1 /tibco/private/foo/T1
/tibco/private/ba/typeT1
RepoRename -url c:/tibco/repos/foo.dat -file c:/temp/renList.txt
```

```
RepoRename -url c:/tibco/repos/foo.dat -ignoreErr -file  
c:/temp/ren.txt
```

Location

TIBCO_HOME\tra\version\bin

AppStatusCheck

Command

```
AppStatusCheck [options] [args...]
```

Purpose

This utility is used to query status of all deployed applications in a domain. You can use it within scripts. Results are returned to the standard output.

The AppStatusCheck utility must be run on a machine that is part of an administration domain. The user account used to run the AppStatusCheck utility must have read permissions set in the TIBCO Administrator GUI for the application, domain repository, and application repository.

To start the utility, change directory to *TIBCO_HOME/tra/version/bin* and type AppStatusCheck with the appropriate command line arguments as described in the following table.

Table 27 AppStatusCheck Parameters

Parameter	Description
-domain	Domain name of the domain for the applications to query for status. If the -app parameter is not used, the status of all applications in a domain is provided.
-user	Name of an authorized user who has read permission for the applications and to access data from <i>SYS_domain</i> .
-pw	Password of an authorized user (case sensitive), provided in plain text.
-cred	Name of a property file containing an authorized <i>userid</i> and encrypted password. Use the <i>obfuscate</i> utility to encrypt the passwords for the property file. If the -user and -pw options are used along with -cred option, the command line values of user and password takes precedence over the property file values.
-app	Application name. To check the status of a specific application, use this option.
-service	Service, within an application. The status of the specified service is provided. This argument must be used along with the -app argument.
-binding	Service instance, within a service. The status of the specified service instance is provided. This argument must be used along with the -app and -service arguments.
-outfile	The file name with absolute path to generate the status output in XML format

Log File

The utility writes information to a log file. The default log file location and name is *TIBCO_HOME/tra/domain/domain_name/logs/AppStatusCheck.log* file.

Location of AppStatusCheck Utility

The location of AppStatusCheck Utility is *TIBCO_HOME\tra\version\bin*.

Appendix A **Log Files Generated for an Administration Domain**

This appendix briefly discusses the different log files created for an administration domain and the trace message format.

Topics

- [Introduction, page 150](#)
- [Installer Logs, page 151](#)
- [Tomcat Logs, page 152](#)
- [Administration Server Log, page 153](#)
- [TIBCO Hawk Agent Logs, page 154](#)
- [Application Logs, page 155](#)
- [Audit Logs, page 156](#)
- [TIBCO Domain Utility Log, page 157](#)
- [Repository Logs, page 158](#)
- [Standard TIBCO Trace Message Format, page 159](#)

Introduction

This appendix briefly lists the different log files generated inside an administration server. If you are facing an issue you need to resolve, you may need to look at one or two of the logs before you can determine what the problem is. In many cases, it is also recommended that you try starting the server from the command prompt to see the resulting errors.

[Appendix E, System Messages](#) lists system messages that TIBCO Administrator may display in addition to information in the logs.

Installer Logs

For each TIBCO product you install, the installer creates a log file in the *USER_HOME\TIBCO* folder. The log includes information about all actions the installer performs.

Tomcat Logs

Tomcat is used by TIBCO Administrator, and keeps a separate set of logs in the *TIBCO_ADMIN_HOME\tomcat\logs\domain-name* folder. A new set of files is created each day you use the software.

Because of the way that TIBCO Administrator forces Tomcat to pre-compile JSPs, a number of error messages are stored in the Tomcat log file. These messages do not indicate a problem (though they are confusing).

This log is generated by Tomcat and can help troubleshoot Tomcat issues.

Administration Server Log

Most administration server traces are stored in the file

TIBCO_TRA_DOMAIN_HOME\domain-name\logs\Administrator.log. The file uses standard TIBCO trace message format. See [Standard TIBCO Trace Message Format on page 159](#) for details.

If the file exceeds 5000 KB, a new file is generated while the old file is maintained with a suffix. Five files are generated before the oldest file is overwritten. If that is too much space for your system or you want to maintain more log information, edit the values in the *AdministrationDomain.Properties* file. The file is located in *TIBCO_TRA_DOMAIN_HOME\domain-name*.

```
LogGenerations= //number of files
LogGenerationSize= //in kilobytes
```

TIBCO Hawk Agent Logs

Two different TIBCO Hawk agent logs are included in *TIBCO_TRA_DOMAIN_HOME\domain-name\logs*:

- *tsm.log* includes the high-level Hawk Agent messages.
- *HawkAgent.log* includes the lower-level Hawk Agent messages.

The TIBCO Hawk product creates a number of log files in *TIBCO_TRA_HOME\logs\hma_port*.

Application Logs

Application logs are stored under the domain in which the application is running. For example, if you created and deployed a TIBCO BusinessWorks application, the log for that application is stored in:

`TIBCO_TRA_DOMAIN_HOME\application\logs`

The information in these logs depend on the monitoring setup for that application. The setup is performed in the TIBCO Administrator GUI, and the information in the application logs is available in Administrator. See *TIBCO Administrator User's Guide*.

Audit Logs

Audit logs track security events. Any TIBCO application that accesses the security module in a domain writes information to the audit log. The log is stored on the machine that hosts the administration server in *TIBCO_ADMIN_DOMAIN_HOME\domain-name\logs*.

TIBCO Domain Utility Log

TIBCO Domain Utility creates a log in a file named `domainutility.log` in the `TRA_HOME\logs` folder. The log contains information about user selections in TIBCO Domain Utility.

Repository Logs

By default, no repository logs are written. You can change this behavior by setting the appropriate properties in the `tibcoadmin_domain-name.tra` file. See [Logging Properties on page 67](#) for more information.

Standard TIBCO Trace Message Format

The implementation of the administration server tracing facility conforms to the tracing specification of TIBCO ActiveEnterprise. The ActiveEnterprise tracing specification provides a common framework for error messages and their handling across all TIBCO product lines (ActiveEnterprise, ActiveExchange, ActivePortal).

Each trace message includes the following information:

Table 28 Trace Message Data

Field Name	Description	Example
Timestamp	Timestamp of occurrence.	See the example in Installer Logs on page 151 .
ApplicationID	Name of the server	Repository.server name
Role	Role of the trace message, which is one of ERROR, WARN, INFO, and VERBOSE (DEBUG) roles.	Error
Category	Category of the message. One of the following: Configuration CreateInstanceTool Database HawkImplant Security Server Servlet System TibRvComm	Configuration
MessageCode	Unique code for the message. The code consists of one alphanumeric key followed by a dash "-" and a six-digit numeric code.	AEREPO-100015
Message	Text message.	"Could not read server process property file abc"

Appendix B Using TIBCO Hawk Methods

TIBCO Hawk monitors and manages distributed applications and systems. A TIBCO Hawk microagent contains methods you can use to change and display application properties at run-time.

The TIBCO Administrator microagent provides a number of methods that allow you to monitor and administer and control an administration server and its clients.



TIBCO Hawk Methods are only available for domains using RV for messaging, not for those using EMS.



Much of the information available via these methods is available from the TIBCO Administrator GUI. However, if you have purchased the TIBCO Hawk product, you can use these microagents for more sophisticated automatic monitoring management.

Topics

- [Introduction, page 162](#)
- [Starting TIBCO Hawk Software, page 163](#)
- [The Auto-Discovery Process, page 164](#)
- [Invoking Microagent Methods, page 165](#)
- [Available Microagent Methods, page 167](#)

Introduction

In a TIBCO Hawk environment, agents on each local computer perform the monitoring work. A TIBCO Hawk agent is an autonomous process that uses microagents to monitor local applications and systems activity. Microagents represent managed objects such as operating system subsystems, agent components, log files, event logs, and applications. Each microagent exposes a set of methods to the agent for collecting information and taking actions. This design allows for separation between management data and management rules or policies.

TIBCO Hawk Display is an application for viewing and managing TIBCO Hawk objects on your network. The Display application is not a centralized console, but a local window into network activity. In TIBCO Hawk Display, an administrator can discover TIBCO Hawk agents that are running, and communicate with microagents on each agent machine. Using application menus and dialogs, administrators can also build and distribute rulebases, which control the monitoring behavior of TIBCO Hawk agents.

For detailed information about TIBCO Hawk product components, see the *TIBCO Hawk Administrator's Guide*.

Starting TIBCO Hawk Software

When running in a domain, the domain configuration for a machine will include a configured TIBCO Hawk agent. The following sections describe how to start TIBCO Hawk product components on Microsoft Windows NT and UNIX platforms on a non-domain machine.

Starting TIBCO Hawk Software on Unix

To start TIBCO Hawk product components on Unix:

1. In a command window, start the TIBCO Hawk agent by typing:
startagent
2. In a new command window, set the DISPLAY environment variable for your workstation by typing:
setenv DISPLAY hostname:0.0
3. Start TIBCO Hawk Display by typing:
startdisplay &

After TIBCO Hawk Display is started, the operation is the same on the Unix and Windows platforms.

Starting TIBCO Hawk Software on Microsoft Windows

Once installed, the TIBCO Hawk agent runs as a Windows service. The service is configured to start automatically. You must start the TIBCO Hawk Display application on your workstation to access the user interface.

To start an instance of TIBCO Hawk Display on Microsoft Windows:

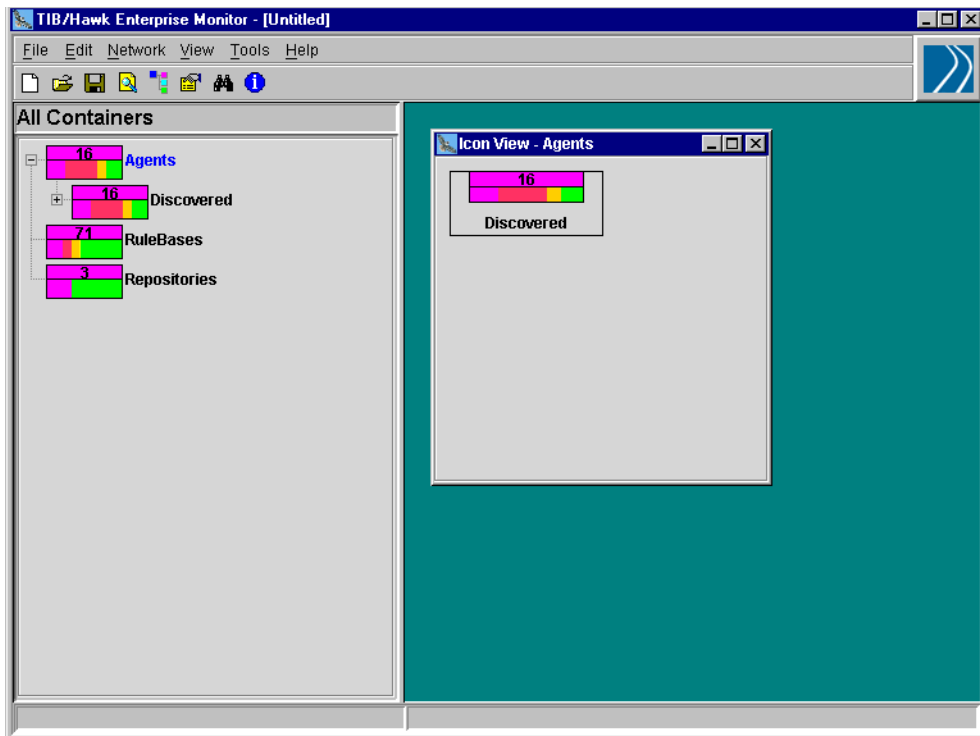
- Select TIBCO Hawk **Software>Hawk Display** from the Start menu, or double-click **Hawk Display** in the TIBCO Hawk program group.

The Auto-Discovery Process

After you start an instance of TIBCO Hawk Display, it continually discovers machines running TIBCO Hawk agents on your network. Container icons are created for each agent, and arranged hierarchically. By default, agent icons are clustered according to subnets.

At first, the Agents container is empty. Its counter displays a value of zero. On the right, the Discovered counter is also at zero. Both icons are initially green to show that no alerts or warning messages are in effect. As agents are discovered, the counters increment to reflect the current number of discovered agents:

Figure 13 *Discovered Agents*



Monitored network nodes are arranged in a hierarchical tree of containers. Clicking a container in the left panel displays nested items on the right.

Icon colors change to reflect the highest level of alert found on discovered agents. For explanations of icon elements and characteristics, see the *TIBCO Hawk Administrator's Guide*.

Invoking Microagent Methods

A set of default microagents is loaded when a TIBCO Hawk agent is started. If requested during installation, TIBCO Repository microagents are dynamically added to the local agent.

The following steps describe how to invoke a microagent method by specifying a microagent, method name, and optional method arguments.

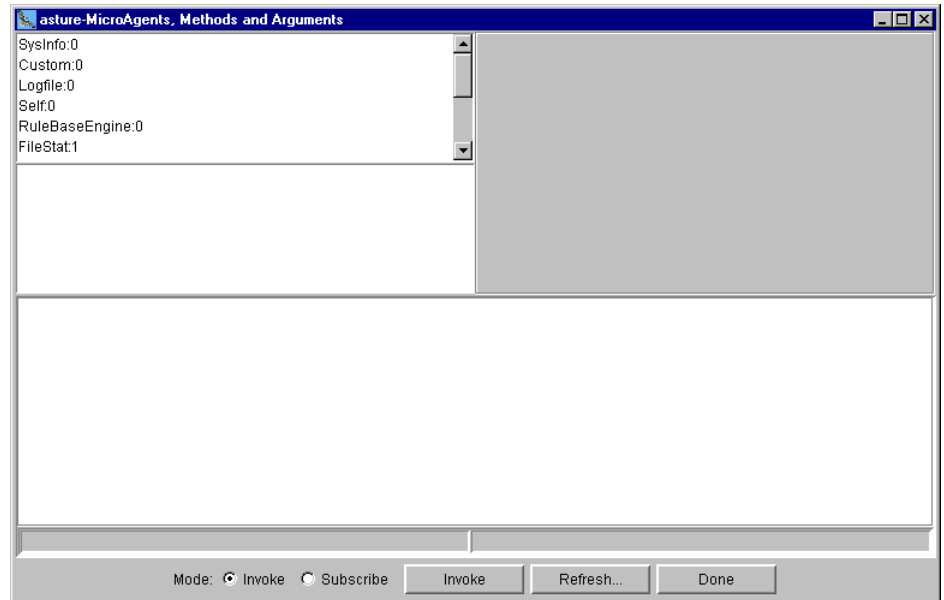
To invoke a microagent method on a TIBCO Hawk agent:

1. In TIBCO Hawk Display, right-click the agent icon and select **Get Microagents**.

If TIBCO Hawk security is implemented on your system and you do not have access to microagents on this agent, an error dialog displays. Select another agent, or contact your system administrator to obtain access.

2. The Microagents, Methods and Arguments dialog displays. The panel on the upper left lists microagents you can access on the current agent.

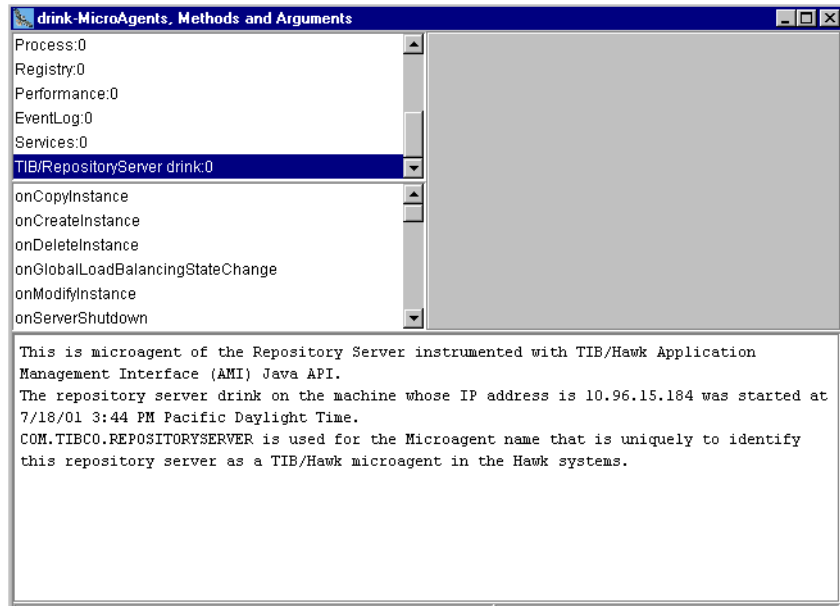
Figure 14 The Microagents, Methods and Arguments Dialog Displays



This dialog has two modes, Invoke and Subscribe. Invoking a method immediately returns a single set of current results. Subscribing provides updates of current results at regular intervals. Radio buttons at the bottom of the dialog control these modes.

- Click the microagent name, such as **TIBCO RepositoryServer** microagent to display a list of associated methods and text descriptions in the panels below.
- Click the name of the method to invoke, such as **GetComponentInfo**.

Figure 15 Select Microagent Name



If the method accepts arguments, fields for each argument display in the upper right panel. Detailed help text displays in the lower panel.

- Specify any arguments for the method invocation.
- Verify that the **Invoke** radio button is selected.
- Click the **Invoke** button to invoke the selected method.

The Invocation Results dialog displays the results returned by the method.

- Click **Done** to close the dialog.

These steps describe how to interactively invoke a microagent method and receive a single set of results in TIBCO Hawk Display. You can also use a microagent method as the data source of a TIBCO Hawk rule. Rules automatically receive method results, apply tests to evaluate them, then take action if necessary. For more information on building TIBCO Hawk rules and rulebases, see the *TIBCO Hawk Administrator's Guide*.

Available Microagent Methods

One microagent, named `COM.TIBCO.REPOSITORY.SERVER` includes all methods for monitoring TIBCO Repository. TIBCO Repository does not run in EMS based domains.

The following table lists, in alphabetical order, each method available for the repository microagent and the page on which the method is explained.

Method	Description	Page
doRefreshCache()	Refreshes the server's cache.	200
doRefreshSecurity()	Refreshes the security objects of the server.	201
doShutdownServer()	Shuts down the server process.	191
doStartInstance()	Starts the specified instance.	199
doStopInstance()	Stops the specified instance.	198
doTerminateClient()	Disconnects the specified client from the repository instance.	202
getConfig()	Returns information about the server process.	180
getConfigProperties()	This method returns properties used for initializing the server.	177
getConnectedClients()	Returns the names of all clients connected to a specified instance.	197
getInstanceInfos()	Returns properties of a specified instance.	192
getInstanceStatus()	Returns information about the current state of the specified instance.	194
getListOfLockedNodes()	Returns the names of all locked nodes.	196
getLogConfig()	Returns information about the log configuration of this server.	184

Method (Cont'd)	Description (Cont'd)	Page
getQueueStats()	Returns information about the TIBCO Rendezvous queues of the server.	183
getRvConfig()	Return information related to the TIBCO Rendezvous session the server uses.	179
getSizeOfInstance()	Returns the current size of an instance.	195
getStatus()	Returns information about the current status of the server process.	181
getTraceIds()	Returns the currently enabled trace IDs of the server.	184
onContentChange()	Notify local TIBCO Hawk agents when content is changed in any of the instances.	186
onCopyInstance()	Notify local TIBCO Hawk agents when a copy of a repository instance is made.	169
onCreateInstance()	Notify local TIBCO Hawk agents that a repository instance has been created.	170
onDeleteInstance()	Notify local TIBCO Hawk agents that a repository instance has been deleted.	171
onInstanceChange()	Notify local TIBCO Hawk agents that a repository instance has been changed.	177
onModifyInstance()	Notify local TIBCO Hawk agents that a repository instance has been modified.	172
onServerShutdown()	Notify local TIBCO Hawk agents that a server is being shut down.	173
onStartInstance()	Notify local TIBCO Hawk agents that a repository instance is started.	174
onStopInstance()	Notify local TIBCO Hawk agents that a repository instance has been stopped.	175
setLogConfig()	Controls logging properties.	188
setTraceIds()	Sets current trace IDs.	190

onCopyInstance()

Purpose Notify local TIBCO Hawk agents that a copy of a repository instance was made. The return value includes the names of the source and destination instance.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
Source instance name	Name of the instance that is used as the source.
Destination instance name	Name of the instance that is used as the destination.

onCreateInstance()

Purpose Notify local TIBCO Hawk agents when a repository instance is created.
The return value is the name of the instance.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns

NameDescription	
Instance name	Name of the instance that is created.

onDeleteInstance()

Purpose Notify local TIBCO Hawk agents when a repository instance is deleted.
The name of the instance is returned.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
Instance name	Name of the instance that is deleted.

onModifyInstance()

Purpose Notify local TIBCO Hawk agents that a repository instance has been modified.
The name of the instance is returned.
You modify instances in TIBCO Designer, which allows you to change instance properties such as the display name and description.
To get notification when the content of the instance changes use [onContentChange\(\)](#) on page 187.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns	NameDescription	
	Instance name	Name of the instance that is modified.

onServerShutdown()

Purpose Notify local TIBCO Hawk agents that a server is being shut down.
The name of the server is returned.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
Server name	Name of the server that is being shut down.

onStartInstance()

Purpose Notify local TIBCO Hawk agents when a repository instance has started.
The name of the instance is returned.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns	NameDescription	
	Instance name	Name of the instance that has started.

onStopInstance()

Purpose Notify local TIBCO Hawk agents when a repository instance has stopped.
You can stop instances using the TIBCO Hawk method [doStopInstance\(\)](#), [page 198](#) or using TIBCO Designer.
The name of the instance is returned.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns	NameDescription	
	Instance name	Name of the instance that has stopped.

onInstanceChange()

Purpose Notify local TIBCO Hawk agents that a repository instance has changed.

Changing an instance means copying, modifying, creating, or deleting an instance. It does not mean changing the contents of an instance.

The return value includes the kind of operation performed and the name of the instance or instances.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns		
	Name	Description
	Operation	Name of the instance operation performed. Value is one of copy, modify, create, or delete.
	Instance name	Name of the instance that has been copied, modified, created, or deleted.
	Destination instance name	Name of the instance that is used for destination in a copy operation.

getConfigProperties()

Purpose This method returns properties used for initializing the server.

Most of properties are read from the `tibcoadmin_domain.tra` file this server is using. When a property is not specified explicitly in the `tibcoadmin_domain.tra` file, this method returns the default value that the server is currently using. See [Chapter 4, Administration Server Properties File](#).

This method returns one row consisting of multiple columns.

To retrieve the status of load-balancing mode, you can invoke the method [getStatus\(\) on page 181](#).

To retrieve information about TIBCO Rendezvous parameters of the server, you can use [getRvConfig\(\) on page 179](#).

To retrieve information about logging of the server, see [getLogConfig\(\) on page 184](#).

Type Open, Synchronous, METHOD_TYPE_INFO

Parameters None

Returns	Name	Description
	repo.ServerName	Name of the server.
	repo.lockTimeout	Number of seconds a repository instance is willing to wait without hearing a heartbeat from a client that has locked various resources, before automatically freeing those resources.
	repo.jdbcDriver	Name of the JDBC driver this repository process will use to connect its instances to their database.
	repo.jdbcURL	JDBC URL this repository process will use to connect its instances to their database.
	repo.username	JDBC database account user name this repository process will use to connect its instances to their database.
	repo.dbtype	Type of backend to connect to.
		The dbtype has to be set to one of the following: Sql, Oracle8, Oracle7, Db2, Sybase.

Name	Description
repo.directory	The directory which this repository process will search for repository instances.
repo.fileType	<p>The format of this repository process's instance data files.</p> <p>Valid values are BIN, XML, or VC.</p>
repo.encoding	Character encoding for repository communication, either ISO8859-1 or UTF-8.
repo.qThreadCnt	The value specifies how many threads are used to process read requests for each instance.
repo.master	<p>In the load-balancing mode, one server is designated as the primary and the other servers as secondary servers.</p> <p>The value is used to distinguish the primary server from secondary servers. The value of the <code>repo.master</code> parameter is either the host name or the IP address of the primary server.</p>
repo.hawkImplant	The value specifies whether the server creates a TIBCO Hawk microagent when it starts.
repo.hawkMaxThread	The value specifies how many threads should process TIBCO Hawk AMI requests for this server.

See Also [Chapter 4, Administration Server Properties File](#)

getRvConfig()

Purpose This method returns information related to the TIBCO Rendezvous session that the server uses for network communication.

Most of properties are read from the `tibcoadmin_domain.tra` file this server is using.

When a property is not specified explicitly in the `tibcoadmin_domain.tra` file, this method returns the default value that the server is currently using.

This method returns one row consisting of multiple columns.

To retrieve other properties of the server, use [getConfigProperties\(\) on page 177](#).

Type Open, Synchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
repo.rvDaemon	TIBCO Rendezvous daemon for this session.
repo.rvNetwork	Network used by this session.
repo.rvService	TIBCO Rendezvous service used by this session.
repo.instanceDiscoverySubject	Subject for the instance discovery subject.
repo.instanceMgmtSubjectPrefix	Instance management subject prefix on which this server listens for instance management requests such as create and copy.

See Also [TIBCO Rendezvous Properties on page 66](#)

getConfig()

Purpose This method returns information related to the server process.

It includes the Java version and the TIBCO Hawk AMI version used by the server. It also includes the path of the `tibcoadmin_domain.tra` file that the server uses for initialization properties.

Type Open, Synchronous, IMPACT_INFO

Parameters None

Returns	Name	Description
	Server name	Name of server.
	IP address of machine	IP address of the machine on which the repository is running.
	OS name	Operating system of the machine on which the repository is running.
	OS version	Operating system version of the machine on which the repository is running.
	Server start time	Time when the server started.
	JMV version	Version of the Java Virtual Machine of the machine on which the repository is running.
	Repository version	Version of the server.
	Hawk AMI version	Version of the Hawk AMI which the repository is using.
	RV version	Version of the TIBCO Rendezvous java package server is using for its network communication.
	Path of used tibcoadmin_domain.tra	Path of the <code>tibcoadmin_domain.tra</code> file which the server read for reading initialization properties.

getStatus()

Purpose This method returns information related to the current status of the server process.

It includes the number of repository instances and clients that this server is serving. It also includes memory usage of this server process.

The `Time while server has been up` represents the time that has passed since the server started, in seconds.

Type Open, Synchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
Number of instances	Number of project instances that the server is serving.
Number of currently connected read-only clients	Number of read-only clients currently connected to this server.
Number of currently connected read-write clients	Number of read-write clients currently connected to this server.
Number of total connections made till now	Number of connections which have been made through this server.
Total number of nodes	Total number of nodes in all instances connected to this server.
Number of locked nodes	Total number of nodes locked by instances connected to this server.
Total memory of this process	Total amount of memory in the Java Virtual Machine. The total amount of memory currently available for allocating objects, measured in bytes.
Free memory of this process	Amount of free memory in the system. An approximation to the total amount of memory currently available for future allocated objects, measured in bytes.

Name	Description
Time while server has been up	Time that has passed since the server started. Time is measured in seconds.

getQueueStats()

Purpose This method returns information related to the statistics of TIBCO Rendezvous queue lengths for the server.

An administration server is queueing requests from clients before processing those requests. If the queue lengths are big, that means the server is busy. A server maintains different queues for different kind of requests from the different instances.

This method provides information on these queues.

Calling this method too often will degrade server performance. Therefore, if less than 10 seconds have passed since the previous call, this method returns the previous result.

Type Open, Synchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
Sum of queue lengths	Sum of lengths of all TIBCO Rendezvous queues that this server currently maintains.
Maximum queue length	Maximum queue length for all TIBCO Rendezvous queues this server currently maintains. If the maximum value is high and the average is relatively low, only requests on a certain instance are queued before being processed.
Average queue length	Average of TIBCO Rendezvous queues lengths that this server currently maintains.
Total number of queues	Number of TIBCO Rendezvous queues on the server.

getLogConfig()

Purpose This method returns information about the log operation of the repository server component of TIBCO Administrator.

This method returns five columns, which represent the current log level as well as information about log files.

Users can determine whether the server writes the log to files or not, by checking the `Is log on` return column.

Type Open, Synchronous, IMPACT_INFO

Parameters None

Returns	Name	Description
	Is log on	The value specifies whether the server should write log into a file or not.
	Log level	<p>The value specifies the current level of logging.</p> <p>This value is to specify the level of log, which designates how detailed the content of log should be.</p> <p>The supported levels are error, warning, info, verbose</p> <ul style="list-style-type: none">• If the level is set to error, only error messages are logged.• If level is set to warning, warning and error messages are logged.• If level is set to info, informative, warning, and error messages are logged.• If level is set to verbose or debug, verbose, info, warning, and error messages are logged.
	Log directory	<p>This value specifies the directory where the log files are created.</p> <p>This value only has an effect if the <code>Is log on</code> return value is true.</p>

Name	Description
Log file max number	<p>This value specifies the maximum number of log files that are maintained in a log directory by the repository server.</p> <p>This value only describes log files created in the directory specified by the <code>Log directory</code> return value of this method.</p>
Log file max size	<p>The value is to specify the max size that log files normally grow only to.</p> <p>After reaching this limit, the log file is closed and a new file is opened.</p> <p>This value only describes log files created in the directory specified in the <code>Log directory</code> return value of this method.</p>

getTraceIds()

Purpose This method returns currently active trace IDs of repository server. Trace IDs are used for low-level debugging and should only be turned on at the request of TIBCO Support.

Type Open, Synchronous, IMPACT_INFO

Parameters None

Returns	Name	Type	Description
	Trace Ids	string	Current trace IDs of repository server.

onContentChange()

Purpose Notify local TIBCO Hawk agents of any content changes in any of the instances that this server is serving.

If you want to track a specific instance rather than all the instances, you can use instance name in your rule base.

Type Open, Asynchronous, IMPACT_INFO

Parameters None

Returns

Name	Description
Instance name	Name of repository instance in which the content change happened.
New commit number	Current commit number of the updated instance. This number is increased by one whenever any content change happens on the instance.
Client	Client which changed the instance.
Changes	Description of changes.

setLogConfig()

Purpose This method can control whether log is written to files or not, and controls the log level.

The repository server's logging is provided on both standard output and the log file if the log file is available.

Users can control the level for both file and standard output logging.

Users can decide whether server should write log or not, by setting the `Is Log On` parameter.

Users can change properties of log files with optional parameters `Log directory`, `Log file max number`, and `Log file max size`.

Type Open, Synchronous, METHOD_TYPE_ACTION

Parameters		
	Name	Description
	Is Log On	This parameter decides whether the server should write the log into a file or not.
	Log Level	<p>This parameter specifies the current level of logging, which determines how detailed the content of log should be.</p> <p>This parameter is optional. The user leave it empty or can specify a certain level. Default value is <code>info</code> for new log files.</p> <p>The supported levels are <code>error</code>, <code>warning</code>, <code>info</code>, <code>verbose</code></p> <ul style="list-style-type: none">• If the level is set to <code>error</code>, only error messages are logged.• If level is set to <code>warning</code>, warning and error messages are logged.• If level is set to <code>info</code>, informative, warning, and error messages are logged.• If level is set to <code>verbose</code> or <code>debug</code>, verbose, info, warning, and error messages are logged.

Name	Description
Log directory	<p>This parameter is optional. The user can specify a certain directory or leave it as empty.</p> <p>This value only has an effect if the <code>Is log on</code> parameter set <code>true</code>.</p> <p>When this parameter is specified as a directory name, the repository server creates log files in the specified directory.</p> <p>When this parameter is not specified (left as empty string), and a previous log file exists, then the server writes the log to the same file. But if no log file has been created by the server and the <code>repo.logDirectory</code> parameter in the <code>tibcoadmin_domain.tra</code> file is also not set, then the repository server uses default directory <code>./log</code> to create the log files.</p>
Log file max number	<p>This value specifies the maximum number of log files that are maintained in a log directory by the repository server.</p> <p>This value only describes log files created in the directory specified by the <code>Log directory</code> return value of this method.</p>
Log file max size	<p>This parameter is optional. The user can specify the size of each log file. If no value is specified, the size is zero, that is, no file is created.</p> <p>This value only has an effect when new log files are created under the directory specified in <code>Log directory</code> parameter of this method. Otherwise this parameter value is ignored.</p> <p>This parameter specifies the maximum size for log files. After reaching this limit, the log file is closed and a new file is opened.</p>

setTraceIds()

Purpose This method sets the current trace IDs for the repository server.

Type Open, Synchronous, IMPACT_INFO

Parameters	Name	Type	Description
	Trace Ids	string	Trace IDs of the repository server you want set.

doShutdownServer()

Purpose This method shuts down the repository server process.

The method sends a server shutdown notification. Its return value lists the names of clients who are connected to any instances of this server.

Time-out for this method is set to 20 seconds.

Type Open, Synchronous, METHOD_TYPE_ACTION_INFO

Parameters

Name	Description
Force shutdown	<p>This parameter determines whether the server is shut down forcibly if it is in the process of writing something to an instance.</p> <p>If the parameter is set to <code>true</code>, this method forces a server shutdown even if some clients had unsaved changes.</p> <p>If the parameter is set to <code>false</code>, and some clients had unsaved changes, this method throws an exception rather than shutting down the server.</p>
Release Instance Lock	<p>This parameter determines whether the server removes locks of its instances when it is shut down.</p> <p>If the parameter is set to <code>true</code>, this method not only kills the server process, but also releases all of its instance locks.</p> <p>If the parameter is set to <code>false</code>, instance locks remain even after the server process is killed, which encourages instances to be served by a server with the same name in the future.</p>

Returns

Name	Description
Client	Name of clients currently connected to any instances of this repository server.
Type	Type of client, either Read-only or Read-write.
Instance	Name of the repository instance to which this client is connecting.

getInstanceInfos()

Purpose This method returns the properties of a specified instance in the return values.

If you leave the input parameter Instance name empty, the method finds all instances that this repository server is serving.

If the specified instance is not available on this server, this method throws an exception.

Type Open, Synchronous, IMPACT_INFO

Parameters	Name	Description
	Instance Name	Instance name that you are interested in. Leave it empty if you want list all the instances that this server is serving.
Returns	Name	Description
	InstanceName	Name of the instance.
	InstanceType	Type of the instance. The value is either remoteRV or local File.
	serverName	Name of the server for this instance.
	displayName	Display name of the instance.
	description	Description of the instance.
	encoding	Encoding of the instance.
	fileType	File type of the instance. This value is only meaningful if you are dealing with a file repository.
	isDefault	Specifies whether this instance is the default instance or not. If the instance is a default instance, the value is true, otherwise false.
	requestSubjectPrefix	Prefix for the subject on which this repository instance listens for various types of requests.

Name	Description
responsiblePerson	Person responsible for the instance.
supportInfo	Support information for the instance.
version	Version of the instance.

getInstanceStatus()

Purpose This method returns information related to the current status of a repository instance.

It includes the number of clients and nodes, and also the current commit number for the specified instance.

The `Total number of connections made until now` return value counts the number of connections which have been made by any clients since the instance had started.

To retrieve the names of available instances, use [getInstanceInfos\(\)](#) on page 192.

- Exceptions** This method throws exceptions:
- If the client does not specify the instance name.
 - If the specified instance is not available on this server.

Type Open, Synchronous, IMPACT_INFO

Parameters	Name	Description
	Instance name	Name of the instance you are interested in.
Returns	Name	Description
	Instance name	Name of the instance.
	Number of currently connected read-only clients	Number of read-only clients that are connecting to the specified repository instance.
	Number of currently connected read-write clients	Number of read-write clients that are connecting to the specified repository.
	Total number of connections made until now	Number of connections which have been made by any clients since the instance started.
	Total number of nodes	Total number of nodes in the specified repository instance.
	Number of locked nodes	Number of the instance’s nodes that are locked by all repository clients.
	Commit number of instance	Number of commits made on the specified repository instance since its creation.

getSizeOfInstance()

Purpose This method returns current size of the data of the specified instance, in bytes. The method counts each character as two bytes and each reference to another node as four bytes.

Calling this method too often degrades repository server performance. This method therefore returns the previous result if less than 60 seconds has passed since the previous call.

You must specify an instance that exists in the server, otherwise this method throws an exception.

Type Open, Synchronous, IMPACT_INFO

Parameters	NameTypeDescription		
	Instance name	string	Name of the instance you are interested in.

Returns	NameTypeDescription		
	size in bytes	string	Size of the contents of this instance.

getListOfLockedNodes()

Purpose This method returns a list of names of all the nodes that are currently locked by any client.

You must specify an instance that exists for the server, otherwise this method throws an exception.

Type Open, Synchronous, IMPACT_INFO

Parameters	NameDescription	
	Instance name	Name of the instance you are interested in.

Returns	NameDescription	
	Node name	Names of all the objects that are currently locked by any client.
	Locker	Client that has locked each node.

getConnectedClients()

Purpose This method returns a list of names clients connected to the specified instance. If the instance name is not specified, it searches all the clients of all the instances that this server is serving. If the specified instance is not found in the server, then the server will throw an exception.

Type Open, Synchronous, IMPACT_INFO

Parameters

Name	Description
Instance name	Name of the instance you are interested in. If it is not specified, this method searches all the instances of this server.

Returns

Name	Description
Client	Name of the clients currently connected to the specified instance.
Type	Type of client, either read-only or read-write.
Instance	Name of the instance to which this client is connecting.

doStopInstance()

Purpose This method stops a repository instance being served by the monitored repository server.

The method sends a notification that the instance has been stopped to TIBCO Hawk. Its return value will list the names of clients who are connected to any instances of this server.

- Exceptions** The method throws exceptions:
- If the instance name is not specified or the specified named instance does not exist in the server.
 - If there are any nodes locked by connected clients.

Type Open, Synchronous, METHOD_TYPE_ACTION_INFO

Parameters	Name	Description
	Instance name	Name of the instance you are interested in. If the instance name is not specified or the specified named instance does not exist in the server, this method will throw an exception.
	ReleaseInstanceLock	<p>This parameter determines whether the method removes the lock on this instance when the instance on this server is stopped.</p> <p>If the parameter is set to <code>true</code>, this method stops the repository instance and release its instance lock.</p> <p>When the parameter is set to <code>false</code>, the instance lock will remain even after the instance is stopped, which encourages this instance to be served by a server of the same name in the future.</p>

Returns	Name	Description
	Client	Name of the clients currently connected to the specified instance.
	Type	Type of the client, either <code>Read-only</code> or <code>Read-write</code> .
	Instance	Name of the instance to which this client is connecting.

doStartInstance()

Purpose This method starts a repository instance, so that the specified instance becomes available through this repository server. The method will send a notification that the instance has been started to the TIBCO Hawk system.

When the server is operating in load-balancing mode, this method will throw an exception unless the global load-balancing mode of the repository servers is read-only. The timeout value for this method is therefore set to thirty minutes.

If the instance name is not specified or the specified named instance does not exist in the server, this method will throw an exception.

Type Open, Synchronous, IMPACT_ACTION

Parameters

Name	Description
Instance name	Name of the instance you want to start. If the instance name is not specified or the specified named instance does not exist in the server, this method will throw an exception.

doRefreshCache()

Purpose Refresh the content of the repository server's cache by re-reading the content of the specified instance.

Exceptions If the server is operating in load-balancing mode, this method throws an exception if the global load-balancing mode of the repository server is not read-only.

If the server has something to write in the cache, this method will throw an exception.

If the server is shutting down, this method throws an exception. Because reloading can be time consuming, the timeout for this method is set to 60 minutes.

Type Open, Synchronous, METHOD_TYPE_ACTION_INFO

Parameters	NameDescription	
	Instance Name	Name of the instance that you want to read again from the backend. If the instance name is not specified or the specified named instance does not exist in the server, this method throws an exception.
Returns	NameDescription	
	Client	Name of the clients currently connected to the specified instance.
	Type	Type of the client, either read-only or read-write.
	Instance	Name of the instance to which this client is connecting.

doRefreshSecurity()

Purpose Refresh the security objects of the Repository server.

Type Open, synchronous, METHOD_TYPE_ACTION_INFO

doTerminateClient()

- Purpose

This method disconnects the specified client from the repository instance
This means immediate rollback of any uncommitted data of the specified client, and removal of the client entry in the instance.
- Exceptions

If the client name is not specified, or the specified named client does not exist in the instance, this method will throw an exception.

If the instance name is not specified or the specified named instance does not exist in the server, this method will throw an exception.

Type Open, Synchronous, IMPACT_ACTION

Parameters	Name	Description
	Client	Name of client that you want to terminate. If it is not specified or specified named client does not exist in the instance, this method will throw an exception.
	Instance name	Name of the instance from which you want to disconnect a the client. If it is not specified or specified named instance does not exist in the server, this method will throw an exception.

Appendix C **Schema Files used when Exporting and Importing**

TIBCO Administrator includes import and export utilities. These utilities can be used in conjunction with predefined XML schema (.xsd) files to ensure that all data you have added to an XML project file are valid.

Syntax and parameters are listed in [Chapter 6, Command Line Utilities](#). This appendix gives some additional information about using the tools and about the schemas included.

Topics

- [XML Formats Used by TIBCO Projects, page 204](#)
- [Schema Files Included With TIBCO Runtime Agent, page 205](#)
- [Using RepoExport and RepoImport, page 206](#)
- [Creating a New Adapter Configuration Schema, page 208](#)
- [Including Custom Schemas During Import, page 209](#)

XML Formats Used by TIBCO Projects

When you use the tools to export TIBCO projects to XML file(s), or import repository XML files into TIBCO projects, the following formats are available:

- *Generic* XML format uses a generic TIBCO Repository format. This format is unchanged from TIBCO Repository 3.x. Although this format is efficient to process, it does not permit any validation of the content.
- *AEXML* format, introduced in ActiveEnterprise 4.0, permits validation of its content using XSD schema files included with the release. Additional schemas for validation may be defined as discussed below.
- *AEXML* multi-file format differs slightly from base *AEXML* format. You can, however, save multi-file projects as single-file projects and vice versa.

To find the XSD's that can be used to validate multi-file project files, unjar the file `tra/version/lib/TIBCOrt.jar`. Inside the file, all XSD files belonging to the package `\com\tibco\object\repo\mapper` can be used to verify resource files.



Even though *AEXML* format is similar to the format of multi-file projects but is not exactly the same. Files created with export cannot be used as projects; they must first be imported into a valid project.

Schema Files Included With TIBCO Runtime Agent

TIBCO Runtime Agent includes a number of schema files which you can use to validate AEXML files you import. The following files are included in the release:

- `Repository.xsd`— Basic repository object definitions.
- `AESchema.xsd`—AE class and sequence descriptions.
- `AEService.xsd`—Endpoint and session definitions
- `AESDK.xsd`—Other TIBCO Adapter SDK object definitions (tracing, monitoring, and so forth.)



To use the files, unjar the file `TRA_HOME\lib\TIBCOrt.jar`. You will see a number of XSDs. All XSDs belonging to the package `com\tibco\infra\repository\importexport` can be used for import/export validation.

The four files are dependent on each other as follows:

- `Repository.xsd` is required by the three other `.xsd` files.
- `AESDK.xsd` requires all of the other three `.xsd` files.



In the `schema\apps` directory, `ADB`, `peoplesoft`, and `MessageBroker` schema files are included.

The `RepoImport` and `RepoExport` tools always check the `Schemas.xml` file for the location of schemas when AEXML format is used.

Using RepoExport and RepoImport

You can export and import projects in one of two ways:

- Use the TIBCO Designer import and export facilities.
- Use the RepoExport or RepoImport utilities. See [RepoExport, page 137](#) and [RepoImport, page 139](#).

Simple export and import operations are useful, for example, if you want to archive your configuration in XML format. They are even more frequently used by developers who want to import an existing project (or part of it) into another project.

Example: Importing an Adapter into an Existing Project

To import an adapter into an existing project, follow these steps:

1. Export the adapter instance.

```
RepoExport -url url -format AEXML -nochase exportfile
/tibco/private/adapter/ADB/ACME
```

url is the location of the repository instance in which the adapter is stored.

/tibco/private/adapter/ADB/ACME is the root node of the adapter.

The command creates an XML file of the adapter. The file uses AEXML format so validation during import is possible. The file does not include the contents of referenced nodes. As a result, the adapter won't overwrite, for example, metadata or sessions already defined in the repository into which you want to import.

2. Import the adapter into the project, using validation.

```
RepoImport -url url importfile -continue
```

This command imports the adapter repository file *importfile* into the specified location inside a project. The import uses all schema files in the current directory for validation and continues import even when validation errors are found.

Example: Exporting, Changing, and Importing an Adapter

If you wanted to export an adapter to XML, make changes in the XML file, then the adapter into an existing project, follow these steps:

1. Export the adapter instance.

```
RepoExport -url url -format AEXML -nochase exportfile
/tibco/private/adapter/ADB/ACME
```

url is the location of the project in which the adapter is stored.

`/tibco/private/adapter/ADB/ACME` is the root node of the adapter.

The command creates an XML file of the adapter. The file uses AEXML format so validation during import is possible. The file does not include the contents of referenced nodes. As a result, the adapter won't overwrite, for example, metadata or sessions already defined in the project into which you want to import.

2. Make changes to the adapter XML file.
3. Import the adapter into the project, using validation.
`RepoImport -url url importfile`

This command imports the adapter project file *importfile* into the specified location inside a project. The import uses all schema files in the current directory for validation. The import process stops if errors are found.

Creating a New Adapter Configuration Schema

If you want to add new configuration elements to your adapter and benefit from validation, you must create a new `.xsd` schema file. Your custom schema file will then be used in conjunction with the four files discussed in [Schema Files Included With TIBCO Runtime Agent on page 205](#).

1. Create a new `.xsd` file that extends the AESDK schema and defines the new elements.
2. Include a namespace definition for your `.xsd`. If using XML Authority, choose **File > Schema Properties > Target Namespace**.

An example, `myAdapter.xsd`, which has the elements for a Clarify adapter, is included with the TIBCO Runtime Agent installation in the `schemas` folder.

3. Create an XML file for the adapter you want to define.
An example, `myAdapter.xml`, which specifies some attribute values for a Clarify adapter, is included with the TIBCO Runtime Agent installation.
4. Make sure that each element knows the namespace (that is, the `.xsd`) to which it belongs. The example uses two namespace (`xmlns`) elements:

- The default namespace, AESDK.
- The custom namespace, ClarifyAdapter.

5. If you later want to export the file using `RepositoryExport`, you must update the file `repo_home\bin\namespace.ini` file to include the custom namespace. For example, for the Clarify adapter, you would add:
`/tibco/private/adapter/ClarifyAdapter = myAdapter`

See Also [Schema Files used when Exporting and Importing on page 203](#)

Including Custom Schemas During Import

If you have created a custom schema and want RepoImport to use it, you have two options:

- Include the custom schema location(s) on the command line.
- Edit the file `Schemas.xml` to include the custom schema(s). The location of that file depends on the product you are using, for example, for TIBCO Runtime Agent, `Schemas.xml` is in `TRA_HOME\bin`. Adapters should have a `schemas` directory.

Appendix D **System Messages**

This appendix lists system messages sent by the administration server. .

Topics

- [AEREPO Messages, page 212](#)
- [POF Messages, page 246](#)
- [POFUTIL Messages, page 274](#)
- [POOL Messages, page 280](#)
- [PLUG-IN Messages, page 283](#)

AEREPO Messages

AEREPO-100001	Cannot start repository server as NT service: %1 Role: Error Category: System Resolution: Verify configuration of tibcoadmin_domain.tra file and verify the system environment variables are set correctly. Ensure that TIBCO Rendezvous and TIBCO Repository are in the CLASSPATH, and that a matching version of TIBCO Rendezvous is in the PATH.
AEREPO-100002	Cannot start repository server: %1 Role: Error Category: System Resolution: Verify proper configuration. If problem persists, contact TIBCO support.
AEREPO-100003	Cannot run instance: %1 Role: Error Category: System Resolution: Named instance is corrupted. If this is a file based instance: <ol style="list-style-type: none">1. Make a backup of the file.2. Delete the file.3. Create a new instance with the same name.4. Restore the instance by using RepositoryImport with a backup. If this is a database instance, contact TIBCO Support.
AEREPO-100004	Error in sending server discovery response: %1 Role: Error Category: Server

Resolution: Indicates a problem with the network. Ensure that server machine has sufficient memory and is communicating successfully with the local area network.

AEREPO-100005

%1

Role: Error

Category: Server

Resolution: Historical, no longer used.

AEREPO-100006

Error in processing server state change: %1

Role: Error

Category: Server

Resolution: Retry setting the server state. If this fails, edit the `tibcoadmin_domain.tra` files to set `repo.state` to the new desired value and shutdown all Load Balanced servers in the Load Balanced group, then restart the servers.

AEREPO-100007

Instances uninitialized for instance discovery response

Role: Error

Category: Server

Resolution: Clients are attempting to communicate with the server before it is initialized. Wait and retry the client connections.

AEREPO-100008

Error in sending instance discovery response: %1

Role: Error

Category: Server

Resolution: Indicates an problem with TIBCO Rendezvous. If the problem persists, reboot the machine. If the problem still persists, contact TIBCO support.

AEREPO-100009

Getting connected client names has found unmatched size of list

Role: Error

Category: HawkImplant

Resolution: Contact TIBCO support.

AEREPO-100010 **Bad initialization parameter for servlet, configFile parameter of the web.xml file must be filled.**
Role: Error
Category: Servlet
Resolution: Indicates a problem with tomcat configuration. Delete and recreate your domain.

AEREPO-100011 **Error in initializing servlet for repository: %1**
Role: Error
Category: Servlet
Resolution: Contact TIBCO support.

AEREPO-100012 **Error in running repository servlet: %1**
Role: Error
Category: Servlet
Resolution: Contact TIBCO support.

AEREPO-100013 **Error in create thread for checking db connection: %1**
Role: Error
Category: HawkImplant
Resolution: Contact TIBCO support.

AEREPO-100014 **Could not read InstanceInfo file %1**
Role: Error
Category: System
Resolution: Either the userid running the server does not have read access to the file, or the instance file is corrupted. In the former case, grant file system access to the specified file. In the latter case, delete the file, recreate the instance and restore its contents from a backup using RepositoryImport.

AEREPO-100015 Could not read server process property file %1

Role: Error

Category: System

Resolution: Make sure that the correct file is specified. Grant file system read privilege to the userid running the administration server.

AEREPO-100016 Error in creating instance with RepoCreateInstance tool: %1

Role: Error

Category: CreateInstanceTool

Resolution: Verify that configuration and environment variables are correct. Verify that database is up and communicating with the machine on which you are running the command. Verify the JDBC parameters. Verify that the specified userid has sufficient authority to drop and create the tables.

AEREPO-100017 Rendezvous error reading request or sending reply, no reply possible: %1

Role: Error

Category: TibRvComm

Resolution: Contact TIBCO support.

AEREPO-100018 Fatal error when reinitialization of security rules on instance %1 by user %2; Exception type:%3 content:%4

Role: Error

Category: Security

Resolution: Contact TIBCO support.

AEREPO-100019 Cannot start security enabled repository server with the user who does not have access right on the rule %1 on domain %2: %3

Role: Error

Category: Security

Resolution: Use TIBCO Domain Utility to set the user for the domain to the administrative super user. Never remove super user privileges from the admin user used to launch the server.

AEREPO-100020 Cannot start commit history: %1

Role: Error

Category: Configuration

Resolution: Verify that the directory specified for the property `repo.commitHistoryDirectory` in the `tibcoadmin_domain.tra` file exists.

AEREPO-100024 Invalid database type %1. Currently supported database types are %2

Role: Error

Category: Configuration

Resolution: Verify that the value specified for the `repo.dbtype` parameter in the `tibcoadmin_domain.tra` file is correct, has the correct capitalization and does not include extra spaces or punctuation. Supported values are: "Sql", "Oracle", "Db2", "Sybase"

AEREPO-100025 Cannot find one of parental nodes of the renamed node in the client cache while updating ACL.

Role: Error

Category: Security

Resolution: Your authorization information is out of sync. Restart the server. If problems persist, contact TIBCO support.

AEREPO-100026 Found duplicate repository instances of different formats (single-file and multi-file repository): %1. Neither of the instances will be started

Role: Error

Category: Configuration

Resolution: Make sure that the directory specified by the `repo.directory` parameter does not contain both a file `%1.dat` and a directory `%1`.

AEREPO-100027 Found duplicate VC Instances: %1. None of them will be started

Role: Error

Category: Configuration

Resolution: Contact TIBCO support.

AEREPO-100028 Cannot get environment variable TIBCO_ADMIN_HOME: %1

Role: Error

Category: Configuration

Resolution: Verify that your `tibcoadmin_domain.tra` file contains the property `java.property.TIBCO_ADMIN_HOME=XX` where XX is the directory where TIBCO Administrator is installed (for example, `C \:/tibco/administrator/5.1`).

AEREPO-100029 Cannot get environment variable TIBCO_REPOSITORY_VERSION: %1

Role: Error

Category: Configuration

Resolution: Contact TIBCO support.

AEREPO-100030 Expected repository URL after -url

Role: Error

Category: CreateInstanceTool

Resolution: Provide a valid `repoUrl` including server and instance names after the `-url` argument. On the command line type `CreateInstanceTool -help url` for a description of a valid `repo url`.

AEREPO-100031 Expected configFile name after -configFile

Role: Error

Category: CreateInstanceTool

Resolution: Provide a file name following the `-configFile` argument. This can be either an absolute or relative path.

AEREPO-100032 Unknown argument %1

Role: Error

Category: CreateInstanceTool
Resolution: Remove the identified argument and rerun the tool.

AEREPO-100033 **Expected -url to specify new repository to be created. Type 'RepoCreateInstance' for help**
Role: Error
Category: CreateInstanceTool
Resolution: Provide a valid repoUrl including server and instance names after the -url argument. On the command line type CreateInstanceTool -help url for a description of a valid repoUrl.

AEREPO-100034 **Cannot get instance name from the given url. Type 'RepoCreateInstance -help url' for information on the url format**
Role: Error
Category: CreateInstanceTool
Resolution: Add the instance name immediately following tibcr:// for server-based repositories, or provide the path to .dat file for file repositories.

AEREPO-100035 **You must specify the server parameter in the url. Type 'RepoCreateInstance -help url' for information on the URL format**
Role: Error
Category: CreateInstanceTool
Resolution: Must provide parameter server=xx as part of the repo url where xx is the name of the server or domain where you wish to create the instance.

AEREPO-100036 **Cannot find the repository server spiffiest in the url. Type 'RepoCreateInstance -help url' for information on the URL format**
Role: Error
Category: CreateInstanceTool
Resolution: Verify that the server name is specified correctly with the correct capitalization, and that it is running.

AEREPO-100037	<p>To connect security-enabled server, the <code>userName</code> and <code>password</code> and parameters must be specified in the url. Type '<code>RepoCreateInstance -help url</code>' for information on the URL format</p> <p>Role: Error</p> <p>Category: <code>CreateInstanceTool</code></p> <p>Resolution: Specify an administrative <code>userid</code> and <code>password</code> in the repo url.</p> <hr/>
AEREPO-200001	<p>Specified Master host %1 could not be found. Server running as slave</p> <p>Role: Warning</p> <p>Category:</p> <p>Resolution: May be safely ignored if this server was intended to be a slave. If the server was intended to be a primary server, correct the <code>hostname</code> specified on <code>repo.master</code> parameter in the <code>tibcoadmin_domain.tra</code> file.</p> <hr/>
AEREPO-200002	<p>HAWK Implant disabled: If you want to enable the hawk microagent, include the installed hawk jar files in the CLASSPATH environment variable used to launch the Repository Server. Cannot enable Hawk because unable to find the class %1</p> <p>Role: Warning</p> <p>Category: <code>System</code></p> <p>Resolution: If TIBCO Hawk monitoring is not required, set the parameter <code>repo.hawkImplant</code> in the <code>tibcoadmin_domain.tra</code> file to <code>false</code>. If TIBCO Hawk monitoring is required, install TIBCO Hawk and add the Hawk jar files as described in the <i>TIBCO Repository Installation and Administration Guide</i>.</p> <hr/>
AEREPO-200003	<p>Cannot start the specified instance %1 in the instance list in the tibcoadmin_domain.tra</p> <p>Role: Warning</p> <p>Category: <code>Configuration</code></p> <p>Resolution: Either correct or comment out the <code>repo.instanceNames</code> parameter in the <code>tibcoadmin_domain.tra</code> file.</p> <hr/>
AEREPO-200004	<p>Configuration specifies empty or invalid instance list, no instances started</p>

Role: Warning

Category: Configuration

Resolution: Either correct or comment out the `repo.instanceNames` parameter in the `tibcoadmin_domain.tra` file.

AEREPO-200005 Discovered another instance with same name on server %1; Instance %2 will not be started.

Role: Warning

Category: Configuration

Resolution: Indicates a configuration problem. Identify which of the servers is correctly configured and change the configuration for the other one. If Load Balancing mode is desired, make sure that all servers in a Load Balanced group have the same server name and that the `repo.master` parameter in the `tibcoadmin_domain.tra` file is uncommented and is non-empty for all servers.

AEREPO-200006 Warning: file instance %1 conflicts with existing database instance, ignoring file instance

Role: Warning

Category: Configuration

Resolution: Verify server configuration.

AEREPO-200007 Ignoring unknown incoming instance-management request %1 on %2

Role: Warning

Category: Server

Resolution: No action required. This indicates that a newer client is attempting to communicate with the server. The client will handle the server's response and retry in a way the server is able to handle.

AEREPO-200009 Detected another server with same name; This server is exiting.

Role: Warning

Category: Server

Resolution: Indicates a configuration problem. Identify which of the servers is correctly configured and change the configuration for the other one. If Load Balancing mode is desired, make sure that the `repo.master` parameter in the `tibcoadmin_domain.tra` file is uncommented and is non-empty for all servers.

AEREPO-200010 Detected another master for the LB group; This server is changing to slave

Role: Warning

Category: Server

Resolution: Indicates a configuration problem. Identify which of the servers is correctly configured and correct the configuration for the other one.

AEREPO-200011 State of this server is out of sync with other members of the LB group; This server is changing its state to: %1

Role: Warning

Category: Server

Resolution: If the new state was not expected, explicitly change the Load Balanced group with either TIBCO Designer or TIBCO Repository Manager to the desired state. If this happens frequently it indicates that your network is being periodically partitioned or is dropping packets.

AEREPO-200012 Server is shutting down cleanly

Role: Warning

Category: Server

Resolution: If server was not intended to be shutdown, restart as soon as it terminates. Set a server password to reduce the chance of inadvertent server shutdowns.

AEREPO-200013 Server is shutting down immediately, aborting unsaved changes

Role: Warning

Category: Server

Resolution: If server was not intended to be shutdown, restart as soon as it terminates. Set a server password to reduce the chance of inadvertent server shutdowns.

AEREPO-200014	Cannot shut down %1; Exception is thrown: %2 Role: Warning Category: Server Resolution: May have active clients that need to save their work. Use TIBCO Hawk console to get a list of active clients. If you intend to shutdown the server and the problem is something other than active clients, issue the kill command from UNIX or terminate with Task Manager if on Microsoft Windows.
AEREPO-200015	Exception in getting Queue Stats: %1 Role: Warning Category: HawkImplant Resolution: May be safely ignored.
AEREPO-200016	Exception in HAWK IMPLANT: notify On change of content: %1 Role: Warning Category: HawkImplant Resolution: Restart any TIBCO Hawk consoles that are monitoring the JDBC connection. If problems persist, contact TIBCO support.
AEREPO-200017	Exception in HAWK IMPLANT: notify On change of state of jdbc connection: %1 Role: Warning Category: HawkImplant Resolution: Restart any Hawk consoles that are monitoring the JDBC connection. If problems persist, contact TIBCO support.
AEREPO-200018	Exception in HAWK IMPLANT: notify On the event of server shutdown: %1 Role: Warning Category: HawkImplant Resolution: Restart Hawk console that issued the stop server request. If problems persist, contact TIBCO support.

AEREPO-200019	Exception in HAWK IMPLANT: notify On the event of stopping instance %1: %2 Role: Warning Category: HawkImplant Resolution: Restart Hawk console that issued the stop instance request. If problems persist, contact TIBCO support.
AEREPO-200020	Exception in HAWK IMPLANT: notify On the event of starting instance %1: %2 Role: Warning Category: HawkImplant Resolution: Restart Hawk console that issued the start instance request. If problems persist, contact TIBCO support.
AEREPO-200021	Exception in HAWK IMPLANT: notify On the event of deleting instance %1: %2 Role: Warning Category: HawkImplant Resolution: Restart Hawk console that issued the delete instance request. If problems persist, contact TIBCO support.
AEREPO-200022	Exception in HAWK IMPLANT: notify On the event of creating instance %1: %2 Role: Warning Category: HawkImplant Resolution: Restart Hawk console that issued the create instance request. If problems persist, contact TIBCO support.
AEREPO-200023	Exception in HAWK IMPLANT: notify On the event of modifying instance %1: %2 Role: Warning Category: HawkImplant

Resolution: Restart Hawk console that issued the modify instance request. If problems persist, contact TIBCO support.

AEREPO-200024 **Exception in HAWK IMPLANT: notify On the event of copying instance %1 to a new instance %2: %3**
Role: Warning
Category: HawkImplant
Resolution: Restart Hawk console that issued the copy instance request. If problems persist, contact TIBCO support.

AEREPO-200025 **Exception in HAWK IMPLANT: notify On change of global state of load balancing to "%1": %2**
Role: Warning
Category: HawkImplant
Resolution: Restart Hawk console that issued the state change request. If problems persist, contact TIBCO support.

AEREPO-200026 **Exception in sending unsolicited warning hawk message "%1": %2**
Role: Warning
Category: HawkImplant
Resolution: Contact TIBCO support

AEREPO-200027 **Exception in sending unsolicited error hawk message "%1": %2**
Role: Warning
Category: HawkImplant
Resolution: Contact TIBCO support

AEREPO-200028 **Exception in sending unsolicited info hawk message "%1": %2**
Role: Warning
Category: HawkImplant
Resolution: May be safely ignored

AEREPO-200029 Server will shutdown after 10 seconds by hawk request

Role: Warning

Category: HawkImplant

Resolution: A server shutdown was issued from a TIBCO Hawk console. If the server was not intended to be shut down, restart server after shutdown completes.

AEREPO-200030 Exception in doing rollback for user null: %1

Role: Warning

Category: Server

Resolution: Indicates a read only client issued a rollback. May be safely ignored.

AEREPO-200031 Exception in reading global variable substitution: %1

Role: Warning

Category: System

Resolution: Contact TIBCO support

AEREPO-200032 Exception in closing Rendezvous connection: %1

Role: Warning

Category: TibRvComm

Resolution: May need to manually kill the rvd daemon. If server cannot be restarted after this, reboot.

AEREPO-200033 Exception in migrating GSV from old schemas: %1

Role: Warning

Category: Database

Resolution: Only occurs with database backends. Indicates first time running a 4.x server on data created with a 3.x server.

1. Bring up the 3.x server. and export the data using the RepoExport script from a 4.x version.

- 2. Stop the 3.x server.
 - 3. Reinitialize the database with the appropriate DBInit script.
 - 4. Start the 4.x server.
 - 5. Recreate the instances and import the saved data using RepositoryImport.
-

AEREPO-200034 **Exception in closing result set during processing BLOB: %1**
Role: Warning
Category: Database
Resolution: If multiple occurrences of this warning appear, backup all instances and restart the database server. If they continue to appear, restart the repository server.

AEREPO-200035 **Exception in creating new statement from the buffer (binary array): %1**
Role: Warning
Category: Security
Resolution: Indicates a problem with the security data. Restart the TIBCO Administrator server. If problem persists, contact TIBCO support.

AEREPO-200036 **Custom rule[%1] on node %2 of instance %3 is invalid**
Role: Warning
Category: Security
Resolution: This may be safely ignored.

AEREPO-200037 **Custom rule[%1] on node %2 of instance %3 has an inconsistent name /%4**
Role: Warning
Category: Security
Resolution: This may be safely ignored.

AEREPO-200038 **Exception in checking whether the user is administrator or not: %1**
Role: Warning

Category: Security

Resolution: Server is in a confused state or network is running very slowly. If it persists, verify your network connection and restart the TIBCO Administrator server.

AEREPO-200039 Node does not exist for security rule %1

Role: Warning

Category: Security

Resolution: May be able to fix by restarting server. Alternatively, back up, delete, recreate and restore the repository listed in the message.

AEREPO-200040 Cannot find the created rule "%1" in the rollback information

Role: Warning

Category: Security

Resolution: May be safely ignored

AEREPO-200041 Exception in deleting rule %1 on the instance %2: %3

Role: Warning

Category: Security

Resolution: Eventually the extra rules will cause authentication to slow down, but otherwise they do no harm. To get rid of garbage rules, recreate and repopulate your domain.

AEREPO-200042 Exception in setting rule %1 on the instance %2: %3

Role: Warning

Category: Security

Resolution: Log out and then retry the action. If that fails, restart the server.

AEREPO-200043 Cannot find the deleted rule "%1" in the rollback information

Role: Warning

Category: Security

Resolution: May be safely ignored.

AEREPO-200045 Cannot read policy domain for instance %1: %2
Role: Warning
Category: Security
Resolution: Verify that the server was started with the current version of software.

AEREPO-200046 Invalid type of parent rule
Role: Warning
Category: Security
Resolution: Verify that the server was started with the current version of software.

AEREPO-200047 Invalid rule type of read rule
Role: Warning
Category: Security
Resolution: Verify that the server was started with the current version of software.

AEREPO-200048 Duplicated commit ?? the specified rule %1 is not locked by %2
Role: Warning
Category: Security
Resolution: Restart your server

AEREPO-200049 Cannot find any rules for the instance %1
Role: Warning
Category: Security
Resolution: If the specified project does not show up in the authorization console, backup, delete, recreate and restore the instance to recreate the security rules.

AEREPO-200050 Exception in rolling back security; problem in removing created but not committed rule: %1

Role: Warning

Category: Security

Resolution: If the problem persists, restart your server.

AEREPO-200051 **While rolling back security, we found unexpected IsUnderDeletion flag on rule %1**

Role: Warning

Category: Security

Resolution: If the problem persists, restart your server.

AEREPO-200052 **While rolling back security, we found unexpected IsUnderCreation flag on rule %1**

Role: Warning

Category: Security

Resolution: If problem persists, restart your server.

AEREPO-200053 **During committing security cache, user or instance is null; user:%1 instanceName:%2**

Role: Warning

Category: Security

Resolution: In memory security is inconsistent, restart your server.

AEREPO-200054 **Exception trying to find matched rule for %1 on the instance %2: %3**

Role: Warning

Category: Security

Resolution: If problem persists, contact TIBCO Support

AEREPO-200055 **Try to cache invalid User(null)**

Role: Warning

Category: Security

Resolution: May be safely ignored.

AEREPO-200055 File ""%1" is not a valid instance file; Ignoring it
Role: Warning
Category: Security
Resolution: Verify that file named %1.dat is an expected project. If not, move it to a different directory. If it is, restore the file.

AEREPO-200056 File ""%1" is not a valid instance file; Ignoring it
Role: Warning
Category: Security
Resolution: Verify that file named %1.dat is an expected project. If not, move it to a different directory. If it is, restore the file.

AEREPO-200057 Cannot create security policy domain and default rules for domain %1: %2
Role: Warning
Category: Security
Resolution: Check permission to the domain data directory, verify that the user running administrator has write permission.

AEREPO-200059 Cannot shut down %1, clients have unsaved changes
Role: Warning
Category: Security
Resolution: Request all clients actively modifying the specified instance to close their applications. May shutdown with the force option if clients are unavailable.

AEREPO-200060 Cannot persistent commit history %1 for the instance %2: %3
Role: Warning
Category: Security
Resolution: Verify that the user running the Administrator server has update rights to the commit history directory and all of its files.

AEREPO-200061	Cannot delete commit history for deleted instance %1: %2 Role: Warning Category: Security Resolution: Verify that the user running the Administrator server has update rights to the commit history directory and all of its files
AEREPO-200063	Cannot load deployment variables due to cause: %1 Role: Warning Category: Security Resolution: Verify that the directory specified by the parameter <code>repo.deploymentDirectory</code> exists and that the server has read access to the directory and all of its files.
AEREPO-200064	The number of returned ACL by getACL method is supposed to be more than 2 Role: Warning Category: Security Resolution: Verify that security rules exist for all instances.
AEREPO-200065	Read rule for root node is under being deleted Role: Warning Category: Security Resolution: Indicates that someone is trying to access a project that is being deleted. May be safely ignored.
AEREPO-200066	Inconsistency of Commit notification between rename and update Role: Warning Category: Client Resolution: May be safely ignored.

AEREPO-200067	Failing in process Null Ref during handling renamed reference Role: Warning Category: Client Resolution: Verify that client application is working correctly. Generally safe to ignore.
AEREPO-200068	repo.updateSecondaryServerPersistentStore property cannot be set to true when database backend is used Role: Warning Category: Configuration Resolution: Use database configuration to do replication, or switch to file storage if you wish repo secondary servers to maintain independent copies of the server projects.
AEREPO-200069	Administration Domain is not installed Role: Warning Category: Configuration Resolution: Administrator was explicitly installed without a domain. If an administration domain is desired, use TIBCO Domain Utility in the TIBCO Runtime Agent menu to configure an administration domain.
AEREPO-200070	Authorization: mismatch between number of resource %1 and number of statusCodes %2 Role: Warning Category: Security Resolution: Restart the administration server.
AEREPO-200071	Exception in findAllTheRules: %1 Role: Warning Category: Security Resolution: Restart the administration server

AEREPO-200072	AssocKey %1 has reference to empty object %2 Role: Warning Category: Security Resolution: May be safely ignored.
AEREPO-200073	Cannot synchronize security data from the master server. So this server will use locally cached security data which could be out of sync with the master server. Role: Warning Category: Security Resolution: Indicates that master server is unavailable. Verify network connectivity and make sure that the primary server is running. If the primary server is known to be down, this message is expected and can be safely ignored.
AEREPO-300001	Repository server started in load balancing mode Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300002	This server is designated as the master for the LB group %1 Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300003	Global state for the LB group:%1 Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300004	Local state for the LB group:%1 Role: Information

Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300005 Repository server "%1" is starting
Role: Information
Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300006 Repository server starting time %1
Role: Information
Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300007 Repository server "%1" is used for character encoding
Role: Information
Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300008 Repository server using %1 threads
Role: Information
Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300009 Repository server using TIB/Rendezvous Java package version %1
Role: Information
Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300010 **Security cache for previous authorization result will be refreshed every %1 seconds**

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300011 **Repository server "%1" is ready.**

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300012 **Server process specifies debug trace types: "%1"**

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300013 **Slave servers do not make file log in the load balancing mode.**

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300014 **Logging to files starting with %1 under the directory %2**

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300015 **HAWK Implant enabled with TIB/Hawk AMI version %1**

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300016	HAWK Implant disabled Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300017	starting file instance %1 Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300018	starting database instance %1 Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300019	Repository server verifying that no other servers named %1 are up. This may take some time. Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300020	NOTE: SERVER WILL BREAK ANY EXISTING INSTANCE LOCKS Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300021	RVD instance management session on daemon %1, Service %2 Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300022 Created Cm Queue transport for load balancing

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300023 Instance discovery subject is %1

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300024 Instance management subject is %1

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300025 Server discovery subject is %1

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300026 State change subject for load balancing is %1

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300027 Server global state changed to: %1

Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300028 Server local state changed to: %1
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300029 Server is stopped by server-based request
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300030 Another server with same name was trying to start; It is now exiting.
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300031 Another server with same name was trying to start as master of the LB group; It is now changing to slave.
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300032 Global state for the LB group:%1
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300033	Local state for the LB group:%1 Role: Information Category: Server Resolution: Information only, no resolution required.
AEREPO-300034	Shutting down instance %1 Role: Information Category: Server Resolution: Information only, no resolution required.
AEREPO-300035	Cannot shut down %1, clients have unsaved changes Role: Information Category: Server Resolution: Information only, no resolution required.
AEREPO-300036	Logging to the directory %1 by hawk request Role: Information Category: HawkImplant Resolution: Information only, no resolution required.
AEREPO-300037	The server role in the load balancing mode is changed to "master" from "slave" Role: Information Category: HawkImplant Resolution: Information only, no resolution required.
AEREPO-300038	The server role in the load balancing mode is changed to "slave" from "master" Role: Information Category: HawkImplant Resolution: Information only, no resolution required.

AEREPO-300039	Instance %1 stopped by hawk request Role: Information Category: HawkImplant Resolution: Information only, no resolution required.
AEREPO-300040	Client "%1" in the instance "%2" terminated by hawk request Role: Information Category: HawkImplant Resolution: Information only, no resolution required.
AEREPO-300041	RVD hawk communication session on daemon %1, Service %2 Role: Information Category: Configuration Resolution: Information only, no resolution required.
AEREPO-300042	Instance creation is completed Role: Information Category: CreateInstanceTool Resolution: Information only, no resolution required.
AEREPO-300043	Creating server-based database instance Role: Information Category: CreateInstanceTool Resolution: Information only, no resolution required.
AEREPO-300044	Creating server-based file instance Role: Information Category: CreateInstanceTool

Resolution: Information only, no resolution required.

AEREPO-300045 Creating local file instance

Role: Information

Category: CreateInstanceTool

Resolution: Information only, no resolution required.

AEREPO-300046 Instance listener subject prefix is %1

Role: Information

Category: CreateInstanceTool

Resolution: Information only, no resolution required.

AEREPO-300047 Instance %1 is ready at: %2

Role: Information

Category: CreateInstanceTool

Resolution: Information only, no resolution required.

AEREPO-300053 Instance %1 created by user %2

Role: Information

Category: Server

Resolution: Information only, no resolution required.

AEREPO-300054 Instance %1 copied to instance %2 by user %3

Role: Information

Category: Server

Resolution: Information only, no resolution required.

AEREPO-300055 Instance %1 deleted by user %2

Role: Information

Category: Server
Resolution: Information only, no resolution required.

AEREPO-300056 **Instance %1 modified by user %2**
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300057 **Security enabled**
Role: Information
Category: Security
Resolution: Information only, no resolution required.

AEREPO-300058 **Security disabled**
Role: Information
Category: Security
Resolution: Information only, no resolution required.

AEREPO-300059 **No deployment directory for server %1**
Role: Information
Category: Configuration
Resolution: Information only, no resolution required.

AEREPO-300060 **No deployment variable file for instance %1**
Role: Information
Category: Server
Resolution: Information only, no resolution required.

AEREPO-300061 **Starting VC Instance: %1**

Role: Information

Category: Server

Resolution: Information only, no resolution required.

AEREPO-300062 Instance %1 created successfully

Role: Information

Category: CreateInstanceTool

Resolution: Information only, no resolution required.

AEREPO-300063 No Regional subjects prefix will be used for read-operation for the load balancing server

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300064 Regional subject prefix %1 will be used for read-operation for the load balancing server

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300065 This secondary load balancing server will %1 persist data synchronized from master server in its own backend

Role: Information

Category: Configuration

Resolution: Information only, no resolution required.

AEREPO-300066 Dropped %1

Role: Information

Category: Mapper

Resolution: Information only, no resolution required.

AEREPO-300067 Succeed in authorization

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300068 Timeout in authorization

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300069 Statement expired

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300070 Other cause of authorization failure %1

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300071 Security authentication failure

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300072 Invalid user name or password

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300073 Timeout in authentication

Role: Information

Category: AUDIT

Resolution: Verify that server is working correctly and that the server machine is not overloaded. Retry client application that received the timeout.

AEREPO-300074 Other cause of authentication failure %1

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300075 Succeed in authentication

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

AEREPO-300076 Internal error

Role: Information

Category: AUDIT

Resolution: Information only, no resolution required.

POF Messages

POF-12001: Could not find entity with id %1 of class %2

Role: errorRole
Category: Application
Description:
Resolution:

POF-12002: Could not find entity with name %1 of class %2

Role: errorRole
Category: Application
Description:
Resolution:

POF-12003: Entity with name %1 was locked by %2 at %3

Role: errorRole
Category: Application
Description: The specified object is locked. You cannot edit the object until the specified owner releases the lock or you break the lock. Breaking the lock will prevent the previous owner's changes from being saved.
Resolution: TIBCO Administrator will prompt you for the option to break the lock. If you break the lock, then you can proceed to edit the object.

POF-12004: Could not authenticate user %1

Role: errorRole

Category: Application

Description: The username and password were not correct.

Resolution: Please check the username and password. Make sure the caps-lock key is not inadvertently pressed. Contact your system administrator if you cannot remember the password.

POF-12006: Domain with name %1 already exists

Role: errorRole

Category: Application

Description: A domain with the name specified already exists.

Resolution: Please choose another name for the domain you are creating.

POF-12007: Duplicate %1 with name "%2". Please choose another name.

Role: errorRole

Category: Application

Description: Cannot create a new user with the same name as an existing user.

Resolution: Please choose another name for the new user.

POF-12008: Duplicate %1 with name "%3" with parent "%2". Please choose another name.

Role: errorRole

Category: Application

Description: Cannot create a new role, application, or application folder with the same name as an existing role, application, or application folder with the same parent role or application folder.

Resolution: Please pick a new name for the role or application being saved.

POF-12009: A role name must be specified

Role: errorRole

Category: Application

Description: The role name field in TIBCO Administrator was empty.

Resolution: Please specify a role name.

POF-12010: A user name must be specified

Role: errorRole

Category: Application

Description: The user name field in TIBCO Administrator was empty.

Resolution: Please specify a user name.

POF-12012: Process %1 cannot support more than one deployment

Role: errorRole

Category: Application

Description: A container in one application was bound to a service in another application.

Resolution: TIBCO Administrator should prevent this from ever happening. Contact TIBCO support if this exception is encountered.

POF-12015: Entity store configuration is missing required property %1

Role: errorRole

Category: Application

Description: Properties in the `AdministrationDomain.properties` or `AuthorizationDomain.properties` are not specified correctly.

Resolution: Please check that the parameters `UserID`, `Credential` and `EntityStoreImplementation` are defined in the property files. If not, then copy them from another domain.

POF-12016: Can not save your changes because a missing lock for entity %1 indicates another user has broken your lock

Role: errorRole

Category: Application

Description: While changing an object, another user broke your lock on the object.

Resolution: All changes that have been made will be lost and have to be made again.

POF-12021: Missing connection pool for caching Groups, membership and users

Role: errorRole

Category: Application

Description: Domain utility failed to configure these parameters when enabling integration with the corporate Repository.

Resolution: Please contact TIBCO support.

POF-12023: Please close this browser instance to start new session.

Role: errorRole

Category: Application

Description: Authentication was configured to require starting a new browser session before the user can log in again.

Resolution: Please close this browser instance to start new session.

POF-12024: Could not find child object of type %1 in entity %2 of type %3

Role: errorRole

Category: Application

Description: Expected to find the specified object.

Resolution: Please contact TIBCO Support.

POF-12025: Invalid name (URI) specified for an access control list - must have a | in it

Role: errorRole

Category: Application

Description: This internal error specifies an illegal name for an ACL.

Resolution: Contact TIBCO support.

POF-12026: Please associate the machine with a region

Role: errorRole

Category: Application

Description: The machine being added was not added to a region.

Resolution: Contact TIBCO Support

POF-12027: Invalid interval of %1 requested. Interval must be less than the difference(%2) of the current values of FROM_TIME(%3) and TO_TIME(%4)

Role: errorRole

Category: Application

Description: This internal error specifies an invalid interval.

Resolution: Contact TIBCO Support

POF-12028: A name for entity of type %1 must be specified

Role: errorRole

Category: Application

Description: Name was not specified.

Resolution: Please specify a name for the object.

POF-12030: Product version for %1 must be of the form x.x was : %2

Role: errorRole

Category: Application

Description: The product major and minor version number was not specified. TIBCO Installer should set this attribute properly.

Resolution: Please contact TIBCO Support.

POF-12031: Product maintenance version for %1 must be set

Role: errorRole

Category: Application

Description: The product maintenance version number was not specified. TIBCO Installer should set this attribute properly.

Resolution: Please contact TIBCO Support.

POF-12032: Product is missing display name

Role: errorRole

Category: Application

Description: Product is missing display name. TIBCO Installer should set this attribute properly.

Resolution: Please contact TIBCO Support.

POF-12033: Short name for product %1 must be set.

Role: errorRole

Category: Application

Description: Missing short name for product. TIBCO Installer should set this attribute properly.

Resolution: Please contact TIBCO Support.

POF-12035: Objects of type %1 must have parent object of type %2: found unexpected parent of type %3

Role: errorRole

Category: Application

Description: An attempt at saving an object in an inappropriate place in the model was made.

Resolution: Please contact TIBCO Support.

POF-12036: Invalid fromTime of %1 requested. fromTime must be between 0 and %2

Role: errorRole

Category: Application

Description: An inappropriate number of milliseconds was specified for an interval.

Resolution: Please contact TIBCO Support.

POF-12037: Invalid toTime of %1 requested. toTime must be between 0 and %2

Role: errorRole

Category: Application

Description: An inappropriate number of milliseconds was specified for an interval.

Resolution: Please contact TIBCO Support.

POF-12038: Invalid time of day from time %1 to time %2 requested. The difference %3 of the current values of fromTime and toTime must be less than the current interval of %4

Role: errorRole

Category: Application

Description: An inappropriate number of milliseconds for the schedule interval was specified. The interval must fall between the fromTime and toTime.

Resolution: Please contact TIBCO Support.

POF-12039: Invalid time of day %1 was requested. All fire times must be between 0 and %2

Role: errorRole

Category: Application

Description: An inappropriate number of milliseconds since midnight was specified for a time of day.

Resolution: Please contact TIBCO Support.

POF-12040: Missing ContainerConfiguration for ServiceContainerBinding %1

Role: errorRole

Category: Application

Description: The attempt at saving the application failed because no process was associated with the specified binding. TIBCO Administrator should prevent this from happening.

Resolution: Please contact TIBCO Support.

POF-12041: Missing process name in component type %1

Role: errorRole

Category: Application

Description: TIBCO Installer failed to set a process name for product.

Resolution: Please contact TIBCO Support.

POF-12042: ContainerConfiguration %1 cannot have ServiceContainerBindings for ServiceConfiguration %2 that have two different versions(%3,%4) of the Product %5

Role: errorRole

Category: Application

Description: An attempt was made to have a single process run two different versions of a product. This function is not supported at this time. TIBCO Administrator should prevent this from happening.

Resolution: Please contact TIBCO Support.

POF-12043: ContainerConfiguration %1 cannot have ServiceContainerBindings for ServiceConfigurations (%2,%3) that have two different process names(%4,%5) in their Products (%6,%7)

Role: errorRole

Category: Application

Description: An attempt was made to binding multiple products with different process names to the same process in the model. TIBCO Administrator should prevent this from happening.

Resolution: Please contact TIBCO Support.

POF-12044: Product %1 cannot be added to Machine %2 because machine already has Product %3 of same version

Role: errorRole

Category: Application

Description: Cannot add a product of the same version more than once to the machine. TIBCO Installer should configure the version numbers appropriately.

Resolution: Please contact TIBCO Support.

POF-12046: Failed to start fault tolerance: %1

Role: errorRole

Category: Application

Description: An error was encountered when starting TIBCO Administrator's fault tolerance support.

Resolution: Please contact TIBCO support.

POF-12048: Could not find user profile for user: %1

Role: errorRole

Category: Application

Description: The user profile information does not exist for this user. This should happen automatically if a Corporate Repository is in use.

Resolution: Please contact TIBCO Support.

POF-12049: Created user %1 on the fly (Not saved yet)

Role: infoRole

Category: Application

Description: For information purposes only.

Resolution:

POF-12050: Warning getting domain for cookie: %1

Role: warnRole

Category: Application

Description: For information purposes only.

Resolution:

POF-12051: Missing group %1

Role: errorRole

Category: Application

Description: Missing group found when searching dynamic groups for membership.

Resolution: Please contact TIBCO Support.

POF-12052: Failure performing LDAP search: %1

Role: errorRole

Category: Application

Description: Failure while trying to integrate with the Corporate Repository.

Resolution: Make sure the search query is valid.

POF-12053: Could not find method with signature get%1()

Role: errorRole

Category: Application

Description: Could not find the specified method when trying to substitute the value returned by the method.

Resolution: Make sure the substitution variable was specified correctly.

POF-12054: Need to have a method with signature get%1() or methods with signatures get%2Names() and get%3(String)

Role: errorRole

Category: Application

Description: Could not find the specified iterator method when trying to substitute the value returned by the method.

Resolution: Make sure the substitution variable was specified correctly.

POF-12055: Another user broke the lock on entity: %1

Role: warnRole

Category: Application

Description: Another user decided to override the changes the current lock owner was making.

Resolution: Please redo all changes that were made.

POF-12056: *****

Role: infoRole

Category: Initialization

Description: For information purposes only.

Resolution:

POF-12057: %1 %2 started by %3 running on %4 %5 %6

Role: infoRole

Category: Initialization

Description: For information purposes only.

Resolution:

POF-12058: CLASSPATH=%1

Role: infoRole

Category: Initialization

Description: For information purposes only.

Resolution:

POF-12061: Received %1 event with a %2 attached to it

Role: warnRole

Category: Application

Description:

Resolution: For information purposes only.

POF-12065: User %1 %2

Role: infoRole
Category: Application
Description: For information purposes only.
Resolution:

POF-12066: No action specified. Must have: Alert, Email, and/or Command

Role: infoRole
Category: Application
Description: Please specify an alert, email and command for the rule.
Resolution:

POF-12067: No command specified

Role: infoRole
Category: Application
Description: Please specify a command for the rule.
Resolution:

POF-12068: No recipient specified in Email

Role: infoRole
Category: Application
Description: Please specify an email recipient for the alert
Resolution:

POF-12069: No mail server specified in Email

Role: infoRole

Category: Application

Description: Please contact TIBCO Support.

Resolution:

POF-12070: Starting FaultTolerance for component %1 of type %2

Role: infoRole

Category: Initialization

Description: For information purposes only.

Resolution:

POF-12071: Stopping FaultTolerance for component %1 of type %2

Role: infoRole

Category: Initialization

Description: For information purposes only.

Resolution:

POF-12072: Component %1 of type %2 received fault tolerance action %3

Role: infoRole

Category: FaultTolerance

Description: For information purposes only.

Resolution:

POF-12073: New User

Role: infoRole

Category: miscellaneous

Description: For information purposes only.

Resolution:

POF-12074: New Role

Role: infoRole

Category: miscellaneous

Description: For information purposes only.

Resolution:

POF-12075: Cannot make role "%1" a child or member of role "%2" or a circular dependency would be created.

Role: errorRole

Category: Application

Description: The role heirarchy is a directed acyclical graph. This means that the same role can appear in many places of the tree of roles presented in TIBCO Administrator. In order to prevent loops when authorizing, loops in the tree of roles are not allowed. Also, Ror containing other roles can create loops which are not allowed either.

Resolution: Please change group hierarchy in a Corporate Repository if that has cycles in it. Also, make sure you are not creating a cycle with changes made in the Role Tree tab or Role members tab or a combination of these.

POF-12076: Cannot remove the root role from the role tree.

Role: errorRole

Category: Application

Description: The root role cannot be removed.

Resolution: Authorization in TIBCO Administrator requires a single root role that cannot be removed.

POF-12077: Management Consoles

Role: infoRole

Category: Application

Description: For information purposes only.

Resolution:

POF-12078: Missing LDAP connection pool for verifying user authentication

Role: errorRole

Category: Application

Description: TIBCO Domain Utility should configure this pool correctly.

Resolution: Please contact TIBCO Support.

POF-12079: Missing LDAP connection pool for synchronizing with a Corporate Repository

Role: errorRole

Category: Application

Description: TIBCO Domain Utility should configure this pool correctly.

Resolution: Please contact TIBCO Support.

POF-12080: Service instance with name %1 already exists or has illegal characters. Please choose another name for the container that is unique and only uses alphanumeric characters, space, _ and -.

Role: errorRole

Category: Application

Description: Service instance names must be unique within an application. Also, they are used in Hawk Microagent names and therefore have the specified character restrictions.

Resolution: Please provide a valid service instance name.

POF-12081: Root role cannot be synchronized with a group. Trying to remove synchronization for a root role is also invalid.

Role: errorRole

Category: Application

Description: Cannot synchronize the root role with a group in the Corporate Repository.

Resolution: Please contact TIBCO Support.

POF-12082: Role with members cannot be converted to a synchronized role. This is only possible, if group being synchronized was its only member.

Role: errorRole

Category: Application

Description: Cannot have a role with the same name as that of a group being synchronized from the Corporate Repository.

Resolution: Please either remove the role in TIBCO Administrator and resynchronize with the Corporate Repository or remove the corresponding group from the Corporate Repository.

POF-12083: This application has a TIBCO Repository instance name "%1" that is identical to the instance name in application "%2". Please choose another instance name.

Role: errorRole

Category: Application

Description: Every deployed application creates a new TIBCO Repository instance in the TIBCO Repository Server embedded in TIBCO Administrator.

Resolution:

POF-12084: User %1 is a super user. Cannot be deleted.

Role: errorRole

Category: Application

Description: Cannot remove users that are super users.

Resolution: Please remove the user from the list of super users before removing the user.

POF-12085: Cannot remove all super users.

Role: errorRole

Category: Application

Description: There must be at least one super user.

Resolution: Please keep at least one super user in the list of users.

POF-12086: Illegal character in TIBCO Repository instance name "%1". Name can only contain alphanumeric characters [a-zA-Z1-9], dash [-] and underscore [_]

Role: errorRole

Category: Application

Description: Invalid TIBCO Repository instance name. This name will be used in a TIBCO Hawk microagent name so the characters are limited to the specified ones.

Resolution: Please use the appropriate characters in the name.

POF-12087: Cannot connect to domain. Please upgrade TIBCO Runtime Agent from version %2 to version %1

Role: errorRole

Category: Application

Description: The schema of the domain is older than that required for the installed product.

Resolution: Please consult product documentation on how to upgrade your domain.

POF-12088: Cannot connect to domain. Domain was created with TIBCO Runtime Agent version %2 which does allow clients of TIBCO Runtime Agent version %1 or higher

Role: errorRole

Category: Application

Description: The schema of the domain is newer than the schema required for the installed product.

Resolution: Please upgrade to the appropriate product version.

POF-12089: Cannot remove folder with name "%1", please undeploy the following applications first: %2

Role: errorRole

Category: Application

Description: All applications in the folder specified must be undeployed before the folder can be deleted.

Resolution: Please undeploy the applications from the Configuration console before deleting the folder.

POF-12090: Please undeploy application(s): %1 before deleting.

Role: errorRole

Category: Application

Description: The application must be undeployed before it can be deleted.

Resolution: Please undeploy the application from the Configuration console before deleting the application.

POF-12091: Cannot change or set a user property "%1" in Corporate Repositories.

Role: errorRole

Category: Application

Description: Some properties available from a user profile are synchronized from the Corporate Repository. These values must be changed in the CorporateRepository and not the user profile.

Resolution: Please contact TIBCO Support.

POF-12092: Cannot remove a user property "%1" from Corporate Repositories.

Role: errorRole

Category: Application

Description: Some properties available from a user profile are synchronized from the Corporate Repository. These values must be removed from the CorporateRepository and not from the user profile.

Resolution: Please contact TIBCO Support.

POF-12094: Cannot remove "%1", please undeploy the following applications that depend on this object first: %2

Role: errorRole

Category: Application

Description: This product or container is currently in use by deployed applications and should not be removed.

Resolution: Please undeploy or redeploy the specified applications without depending on the product or container.

POF-12095: No longer have locks for objects or objects are locked by someone else

Role: errorRole

Category: Application

Description: While changing an object, another user broke your lock on the object.

Resolution: All changes that have been made will be lost and need to be made again.

POF-12096: TIBCO Repository instance name is invalid. It must start with the characters "%1"

Role: errorRole

Category: Application

Description: TIBCO Repository instance names must start with the domain name followed by a dash. This ensures the instance name is unique on the network.

Resolution: Please change the instance name to start with the characters specified.

POF-12097: Cannot remove the guest role from the role tree.

Role: errorRole

Category: Application

Description: The guest role cannot be removed.

Resolution: Authorization in TIBCO Administrator requires a guest role that cannot be removed. The guest role is used to support unauthorized access to resources.

POF-12098: User %4 failed to lock object with name %1. Object was locked by %2 at %3

Role: errorRole

Category: Application

Description: The specified object is locked. You cannot edit the object until the specified owner releases the lock or you break the lock. Breaking the lock will prevent the previous owner's changes from being saved.

Resolution: TIBCO Administrator will prompt you with the option to break the lock. If you break the lock, you can proceed to edit the object.

POF-12099: The archive loaded contains data using encoding %1 which is not supported by the TIBCO Administrator that requires the archive encoding to be equivalent to or contained in encoding %2. Please change the value of parameters repo.encoding and tibcoadmin.client.encoding of the .tra file for TIBCO Administrator to %3, restart TIBCO Administrator and reload the archive.

Role: errorRole

Category: Application

Description:

Resolution: Each TIBCO Administrator domain can only support archives from a single encoding. Please choose an encoding that contains all the encodings for archives to be deployed.

POF-12100: Number of matching entries exceeds the Limit for searching from Domain. Please specify a more restrictive search pattern.

Role: warnRole

Category: Application

Description:

Resolution: Please narrow down the search pattern.

POF-12101: Role Membership Plug-in descriptor "%1" not found.

Role: errorRole

Category: Application

Description:

Resolution: Please upgrade the domain appropriately to create descriptors for all out-of-box and installed Role Membership Plug-ins.

POF-12102: Cannot uninstall this Role Membership Plug-in since there exists roles for this membership.

Role: infoRole

Category: Application

Description:

Resolution: Role Membership Plug-in cannot be uninstalled unless all the role defined for this membership are deleted.

POF-12103: You may need to migrate the domain to 5.2 release in order to use the 5.2 TRA client, due to the exception: %1

Role: errorRole

Category: Application

Description:

Resolution: TIBCO Administrator Domain can be upgraded by upgrading TRA and TIBCO Administrator or the machine installed with Primary TIBCO Administrator Server.

POF-12104: User %1 cannot delete itself.

Role: errorRole

Category: Application

Description:

Resolution: Other TIBCO Administrators or Super users should be able to delete you as a user. For this operation, remove yourself from the list of users to be deleted.

POF-12105: Cannot remove Role %1 from a group sunchronized parent role %2.

Role: errorRole

Category: Application

Description:

Resolution: This operation is not allowed because group hierarchy of group synchronized roles is governed by the Corporate LDAP.

POF-12106: Cannot remove Role %1 from the root role since it does not have any other parent.

Role: errorRole

Category: Application

Description:

Resolution: This operation is not allowed because it leads to orphan roles.

POF-12107: Deployment name must have a maximum of %1 characters: %2

Role: errorRole

Category: Application

Description:

Resolution: Please pick a shorter unique deployment name.

POF-12108: Authentication Failed: %1

Role: infoRole

Category: Application

Description:

Resolution: Please authenticate correctly as per Plug-in Authentication requirements.

POF-12109: The credential specified does not conform to the password policy being enforced

Role: errorRole

Category: Application

Description: The password policy may enforce various constraints on new passwords. The specified password violates one of these constraints. Please check password to find the reason that the password was not accepted.

Resolution: Please pick another password.

POF-12110: The user %1 failed to authenticate.

Role: errorRole

Category: Application

Description: Please see cause for more details. Either an invalid user was trying to authenticate, the password provided was incorrect, or the password has expired and needs to be changed.

Resolution: Please pick another password.

POF-12111: Deleting server based project %1.

Role: infoRole

Category: Application

Description: Cleanup as part of deployment of a locally based project. The server-based instance is being removed.

Resolution: This is a normal and expected activity.

POF-12112: The old credential for user %1 was specified incorrectly.

Role: errorRole

Category: Application

Description: The old password must be specified correctly before a new password can be set.

Resolution: The user should attempt to change the password again.

POF-12113: This application has a deployment name "%1" that is identical to the deployment name in application "%2". Please choose another name.

Role: errorRole

Category: Application

Description: Every application must have a unique deployment name.

Resolution: The user should change the deployment name.

POF-12114: Cannot add a group synchronized role %1, that has parent groups in LDAP, to the root role.

Role: errorRole

Category: Application

Description: Only those group-synchronized roles could be added to root role that do not have parent groups in LDAP. Any group synchronized role could be added to other regular roles though.

Resolution: The user should not attempt to add a group synchronized role to the root role if the group synchronized role has a parent group in LDAP.

POF-12115: Cannot remove a group synchronized role %1, that has no parent groups in LDAP, from the root role.

Role: errorRole

Category: Application

Description: Group synchronized roles could only be removed from root role if they had parent groups in LDAP. Any group synchronized role could be removed from other regular roles, though.

Resolution: The user should not attempt to remove a group synchronized role, that has no parent group in LDAP, from the root role.

POF-12116: A user with same name (%1) exists in LDAP. Creating or renaming this local user will remove the synchronized LDAP user including its security information in TIBCO Administration domain. Do you want to continue?"

Role: warnRole

Category: Application

Description: A user with same name (%1) exists in LDAP. Creating or renaming this local user will remove the synchronized LDAP user including its security information in TIBCO Administration domain."

Resolution: Either do not create, name, or rename the local user with that name, or ignore the warning and continue. This will remove the synchronized user including its security information from TIBCO Administration domain.

POF-12117: A local user with same name (%1) exists in TIBCO Administration domain. Creating or renaming this synchronized user will remove the local LDAP user including its security information in TIBCO Administration domain. Do you want to continue?"

Role: warnRole

Category: Application

Description: A local user with same name (%1) exists in the TIBCO Administration domain. Creating or renaming this synchronized user will remove the local LDAP user including its security information in TIBCO Administration domain."

Resolution: Either do not create, name, or rename the local user with that name, or renamed, or ignore the warning and continue. This will remove the local user including its security information in TIBCO Administration domain.

POF-12118: Please login using a valid local or an LDAP user.

Role: errorRole

Category: Application

Description: This domain requires LDAP usernames in a NetBIOS format, for example, "ACME_HQ\\jsmith".

Resolution:

POF-12119: A runtime exception is raised by the Role Membership component: %l.

Role: errorRole

Category: Application

Description:

Resolution:

POF-12120: This operation is not allowed on a read-only User object obtained from the User Manager.

Role: errorRole

Category: Application

Description:

Resolution:

POFUTIL Messages

POFUTIL-00001: Domain name must be specified after -domain flag

Role: Error

Category: ParseArgs

Description:

Resolution: After the -domain flag, add the domain name that you wish to use.

POFUTIL-00002: Encryption key must be specified after -encryptKey flag

Role: Error

Category: ParseArgs

Description: Not supported in 5.2.0

Resolution: After the encryptKey flag, specify the encryption key that you wish to use.

POFUTIL-00003: Date must be specified after -date flag

Role: Error

Category: ParseArgs

Description:

Resolution: Either remove the date flag, or follow the date flag to specify the date. Only entities modified after the specified date will be processed.

POFUTIL-00004: user must be specified after -user flag

Role: Error

Category: ParseArgs

Description:

Resolution: After the user flag, specify the administrative user who has authorization to read and write to the security information.

POFUTIL-00005: Password must be specified after -pw flag

Role: Error

Category: ParseArgs

Description:

Resolution: After the -pw flag, specify the password for the administrative user specified with the -user flag.

POFUTIL-00006: File path must be specified after -file flag

Role: Error

Category: ParseArgs

Description:

Resolution: Specify the fully qualified location of the file that will hold the exported security information.

POFUTIL-00007: Credential must be specified after -cred flag

Role: Error

Category: ParseArgs

Description:

Resolution: Credential file location must be specified after the -cred flag. Either remove this flag or specify the file path and name.

POFUTIL-00010: Could not find expected role {0}

Role: Error

Category: ExportDomainSecurity

Description: A role that was requested for export does not exist in the domain.

Resolution: This is informational. Remove the role from the list of requested roles to avoid seeing the warning message.

POFUTIL-00011: Property element must include the property name

Role: Error

Category: ImportDomainSecurity

Description: The XML representing the security information was malformed. A property element which should have had both a name and a value did not have a name element.

Resolution: Update the XML stanza adding name=....

POFUTIL-00012: Property {0} does not have a value defined

Role: Error

Category: ImportDomainSecurity

Description: The XML representing the security information was malformed. A property element which should have had both a name and a value did not have a value element.

Resolution: Update the XML stanza adding value=....

POFUTIL-00013: Error importing a child of user {0}

Role: Error

Category: ImportDomainSecurity

Description: Malformed XML describing User Profile information.

Resolution: Correct the XML.

POFUTIL-00014: Invalid credential for user {0}

Role: Error

Category: ImportDomainSecurity

Description: The Password specified for the user was not a valid obfuscated password.

Resolution: Provide a valid obfuscated password in the XML specifying the password for the listed user.

POFUTIL-00015: Invalid syntax for Synchronized Group ID of role {0}

Role: Error

Category: ImportDomainSecurity

Description: Malformed XML

Resolution: Correct XML specifying synchronzied group ID for specified role.

POFUTIL-00016: Expected parent missing for role {0}

Role: Error

Category: ImportDomainSecurity

Description: This is just a warning. A Role was specified with a name in the format a/b where a was supposed to be its parent. However, no such role exists. It will be implicitly created.

Resolution: Ignore the message or update the XML to explicitly create the parent role.

POFUTIL-00017: Invalid syntax for Synchronized Group ID of role {0}

Role: Error

Category: ImportDomainSecurity

Description: Malformed XML

Resolution: Correct the xml specifying the synchronized group ID.

POFUTIL-00018: Skipping role:

Role: Error

Category: ExportDomainSecurity

Description: A problem was encountered processing the specified role, most likely it was not defined in the domain.

Resolution: Correct the arguments and rerun.

POFUTIL-00019: Error exporting role: {0} -

Role: Error

Category: ExportDomainSecurity

Description: A problem was encountered processing the specified role. Possibly the domain is corrupt.

Resolution: Manually correct the exported XML to define the role or delete and recreate it in your domain.

POFUTIL-00020: Duplicate credential specified for user: {0} -

Role: Error

Category: ImportDomainSecurity

Description: The new password or other credential exactly matched another credential already stored for this user.

Resolution: This is informational only.

POFUTIL-00021: Missing argument after flag {0}

Role: Error

Category: ParseArgs

Description: Specified flag must have an associated value and there is not one.

Resolution: Provide the required argument on the command line.

POFUTIL-00022: Invalid key

Role: Error

Category: ImportDomainSecurity

Description:

Resolution:

POFUTIL-00023: Specified user must have Admin privileges

Role: Error

Category: ImportDomainSecurity

Description:

Resolution:

POFUTIL-00024: Node of [{0}] not find! This ACL can not imported.

Role: WRAN

Category: ImportDomainSecurity

Description:

Resolution:

POOL Messages

POOL-12100: Connect failed to address: %1 : %2

Role: errorRole
Category: Application
Description:
Resolution:

POOL-12101: Server not available at address %1

Role: errorRole
Category: Application
Description:
Resolution:

POOL-12102: Timeout getting connection for address %1

Role: errorRole
Category: Application
Description:
Resolution:

POOL-12103: Following potential problems are found in search parameters: %1

Role: warnRole
Category: Application
Description:
Resolution:

POOL-12104: No matching entries found for %1

Role: warnRole
Category: Application
Description:
Resolution:

POOL-12105: Attributes (%1) not found for %2

Role: warnRole
Category: Application
Description:
Resolution:

POOL-12106: User Search Filter is not specified on any of the search parameter sets

Role: warnRole
Category: Application
Description:
Resolution:

POOL-12107: Duplicate Search parameters sets are found

Role: warnRole
Category: Application
Description:
Resolution:

POOL-12108: Exception while Testing Search Parameters %1: %2

Role: warnRole
Category: Application
Description:
Resolution:

POOL-12109: Corporate LDAP server is throwing Critical Extension Unavailable exception.

Role: errorRole
Category: Application
Description:
Resolution: Narrowing down on the search criteria to return a smaller dataset will help. Otherwise additional indexes created on LDAP server may help avoid this situation.

PLUG-IN Messages

PLUG-12001: The singleton represented by the class %1 generates a circular references itself during construction.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12004: A component of the name %1 is already registered.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12005: A component of the name %1 could not be found.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12006: Unable to instantiate the class %1.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12007: Unable to access either the class or its constructor for class %1.

Role: errorRole
Category: Application
Description:
Resolution:

PLUG-12008: Difficult opening the JAR file %1.

Role: errorRole
Category: Application
Description:
Resolution:

PLUG-12009: Plug-in file does not contain the plug-in configuration for: %1.

Role: errorRole
Category: Application
Description:
Resolution:

PLUG-12010: Main class is not found for the plugin JAR %1.

Role: errorRole
Category: Application
Description:
Resolution:

PLUG-12011: The main class of plugin JAR %1 could not be instantiated because it is abstract or an interface.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12012: Unable to access either the class or its constructor for the main class of plugin %1.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12013: The main class of plugin JAR %1 does not implement the Plugin interface.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12014: The application failed to register its plugins: %1.

Role: errorRole

Category: Application

Description:

Resolution:

PLUG-12015: It is a class and not component factory interface.

Role: errorRole
Category: Application
Description:
Resolution:

PLUG-12016: Be not able to load %1 from the application archive classloader..

Role: errorRole
Category: Application
Description:
Resolution:

Index

Symbols

.logOperations parameter 67

A

adapter schema 208

Adding

Secondary Server Using Command Line Utility 49

administration domains

multiple 30

spanning subnets 32

argument (command-line)

clientvar 104

auto-discovery process 164

TIBCO Hawk 164

B

breaking the lock 102

C

cache configuration 96

cache statistics 97

category

message 159

standard 159

checkSecurityOnlyOnPolicyManagement
parameter 73

clientvar command-line argument 104

clientvar property 104

commitHistoryDirectory parameter 77

components, retrieving through TIBCO Hawk 169
Configuring Connection Pool Size for the Database
Server 89

creating adapter configuration schema 208

custom schema 209

custom schemas during import 209

customer support xx

D

data storage 103

database

storage parameters 64

database domain storage 7

database projects 7

dbType parameter 64

defining a URL 88

defining variables 103

delayForDBStartup parameter 64

deploymentDirectory parameter 76, 76, 76, 77

directory parameter 63

doRefreshCache() 200

doRefreshSecurity()() 201

doShutdownServer() 191

doStartInstance() 199

doStopInstance() 198

doTerminateClient() 202

E

EHCACHE 96

enabling SSL on the LDAP server 90

encapsulating locator string components 88

encoding parameter 66

ENV_NAME xvii

example
 exporting, changing, and importing an adapter 206
 importing an adapter into an existing repository 206

F

file
 xml import and export 105
 file storage parameters 63
 file.encoding 59
 fileType parameter 63
 forceInstanceLocks parameter 59

G

getcomponents, TIBCO Hawk method 169
 getConfig() 180
 getConfigProperties() 177
 getConnectedClients() 197
 getInstanceInfos() 192
 getInstanceStatus() 194
 getListofILockedNodes() 196
 getLogConfig() 184
 getQueueStats() 183
 getRvConfig() 179
 getSizeofInstance() 195
 getStatus() 181
 getTraceIds() 186
 getVersion() TIBCO Hawk method 168
 global variables 103, 103

H

hawkDaemon parameter 77
 hawkImplant parameter 77
 hawkMaxThread parameter 77
 hawkMicroAgentName parameter 77
 hawkNetwork parameter 77

hawkService parameter 77
 HTTP 82
 HTTPS 82
 URLfile 84

I

import
 custom schema 209
 instanceNames parameter 59
 invoking Microagent methods 165
 isSecurityEnabled 73

J

java.heap.size.initial 78
 java.heap.size.max 78
 java.thread.stack.size 78
 JDBC driver 64
 JDBC URL 64

L

LDAP server
 enabling SSL 90
 listing of utilities 106
 load balancing 25
 parameters 71
 load-balanced servers 28, 42
 Local Application Data 9
 local repository locator string 86
 locator string 86, 88
 lock 102
 locks 59, 102
 lockTimeout parameter 59
 logDirectory parameter 67
 logFilemaxNumber parameter 68
 logFileMaxSize parameter 68
 logFileNamePrefix parameter 68

logging levels [68](#)
 logging parameters [67](#)
 logLevel parameter [68](#)

M

message
 category [159](#)
 message trace description [159](#)
 methods [167](#)
 Microagent methods [167](#)
 minimizeHandshakeInterval parameter [60](#)
 multiple administration domains, single subnet [30](#)
 multiple load-balanced servers [28](#)
 multiple load-balanced servers spanning subnets [42](#)

O

onContentChange() [187](#)
 onCopyInstance() [169](#)
 onCreateInstance() [170](#)
 onDeleteInstance() [171](#)
 one administration domain, multiple load-balanced
 servers [28](#)
 one administration domain, one server in a single
 subnet [27](#)
 onInstanceChange() [176](#)
 onModifyInstance() [172](#)
 onServerShutdown() [173](#)
 onStartInstance() [174](#)
 onStopInstance() [175](#)

P

parameter
 database storage [64](#)
 parameters
 logging [67](#)

 logging parameters [67](#)
 persist commit history [76](#)
 password parameter [64](#)
 password policy [16](#)
 password policy choices [17](#)
 persist commit history [76](#)
 precedence [88](#)
 prefixRegionalSubject parameter [71](#)
 project locking [102](#)
 projects
 managed by TIBCO Administrator [103](#)
 substitution variables [103](#)
 property precedence [88](#)

Q

qThreadCnt parameter [60](#)

R

Removing
 Secondary Server Using the Command Line
 Utility [54](#)
 Secondary Server Using the GUI [54](#)
 repo.auditFileName [70](#)
 repo.auditFileNum [70](#)
 repo.auditFileSize [70](#)
 repo.checkSecurityOnlyOnPolicyManagement [73](#)
 repo.commitHistoryDirectory [76, 77](#)
 repo.commitHistoryFileMaxNumber [76](#)
 repo.dbType [64](#)
 repo.delayForDBStartup [64](#)
 repo.deploymentDirectory [76, 77](#)
 repo.directory [63](#)
 repo.encoding [66](#)
 repo.fileType [63](#)
 repo.forceInstanceLocks [59](#)
 repo.hawkDaemon [77](#)
 repo.hawkImplant [77](#)
 repo.hawkMaxThread [77](#)
 repo.hawkMicroAgentName [77](#)

- repo.HawkNetwork 77
- repo.hawkService 77
- repo.instanceNames 59
- repo.isRepoNavigatorEnabled 74
- repo.isSecurityEnabled 73
- repo.jdbcDriver 64
- repo.jdbcURL 64
- repo.lockTimeout 59
- repo.logDirectory 67
- repo.logFilemaxNumber 68
- repo.logFilemaxSize 68
- repo.logFileNamePrefix 68
- repo.logLevel 68
- repo.logOperations 67
- repo.master parameter 71
- repo.minimizeHandshakeInterval 60
- repo.password 64
- repo.prefixRegionalSubject 71
- repo.qThreadCnt 60
- repo.rvDaemon 66
- repo.rvNetwork 66
- repo.rvService 66
- repo.secureGuestPassword 73
- repo.secureGuestUsername 73
- repo.securePassword 74
- repo.secureStatementDuration 74
- repo.secureUsername 74
- repo.serverHeartbeatInterval 60
- repo.serverName 60
- repo.serverPassword 60
- repo.state parameter 71
- repo.updateSecondaryServerPersistentStore 72
- repo.username 64
- repo.windowsEventLogLogLevel 68
- repo.windowsEventLogLogLevel parameter 68
- RepoConvert 126, 126
- RepoDeleteInstance 134, 134
- RepoDiff 135, 135
- RepoExport 206
- RepoImport 139, 139, 206
- RepoPing 130, 130
- restrictions on database project repositories 7
- rvDaemon parameter 66
- rvNetwork parameter 66
- rvService parameter 66

S

- schema 208, 209
- schemas
 - included with TRA 205
- secureGuestPassword parameter 73
- secureGuestUsername parameter 73
- securePassword parameter 74
- secureStatementDuration parameter 74
- secureUsername parameter 74
- security 73
 - enable server 73
 - secureStatementDuration 74
 - server 73
- Security Considerations 12
- server
 - security 73
 - security enabled 73
 - TIBCO Hawk parameters 77
- server parameters for TIBCO Rendezvous 66
- server security 73
- serverHeartbeatInterval parameter 60
- serverName parameter 60
- serverPassword parameter 60
- setLogConfig() 188
- setTraceIds() 190
- setup tasks 33
- spanning subnets 32, 42
- starting
 - TIBCO Hawk on UNIX 163
 - TIBCO Hawk on Windows 163
- starting TIBCO Hawk software 163
- storage parameters 64
- subnets 32
- substitution variables 103
- support, contacting xx

T

- technical support xx
- TIBCO Administrator
 - managing projects 103
 - user password 74

- TIBCO Designer
 - defining variables [103](#)
 - global variables [103](#)
- TIBCO Hawk [164](#)
 - Microagent methods [167](#)
 - server parameters [77](#)
 - starting [163](#)
- TIBCO Hawk methods
 - getcomponents [169](#)
 - getversion() [168](#)
- TIBCO Hawk on Windows [163](#), [163](#)
- TIBCO Rendezvous [80](#)
 - server parameters [66](#)
- TIBCO Repository XML format [204](#)
- TIBCO_HOME [xvii](#)
- tibco.clientvar. property [104](#)
- TRA
 - included schemas [205](#)
- trace
 - message description [159](#)
- trace description [159](#)
- trace message format [151](#)
- traces
 - location [151](#)
- troubleshooting [41](#)

U

- Unicode support [92](#)
- updateSecondaryServerPersistentStore parameter [72](#)
- URL [86](#), [88](#), [88](#)
- URLfile accessed via HTTPS [84](#)
- user password [74](#)
- username parameter [64](#)
- using properties file [88](#)
- utilities [106](#)

V

- variable values [104](#)
- variables [103](#), [103](#)

X

- xml
 - file import and export [105](#)
- XML format [204](#)