Oracle® Business Transaction Management

Installation Guide

12.1.0.2

E27398-03

February 2, 2012

ORACLE®

Oracle Business Transaction Management Installation Guide, 12.1.0.2

E27398-03

Primary Author: Alan Davidson

Contents

17 The datastoreUtil Utility

Preface

Oracle Business Transaction Management Installation Guide explains how to install Business Transaction Management 12.1.0.2, including the central servers, monitors, and observers.

Audience

This document is intended for system administrators and others who want to install Business Transaction Management 12.1.0.2.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

This guide explains how to install Oracle Business Transaction Management. The information provided in this guide can help you install and configure a general purpose Business Transaction Management system suitable for experimenting with the system functions and use cases. The deployment configuration and resources required in a production environment can vary based on a variety of factors, such as anticipated throughput, message size and type, the number of applied monitoring policies, and so forth. An Oracle consultant can help you determine the appropriate configuration and resource requirements for your specific needs.

This chapter provides a high-level description of the product architecture, a description of the product packaging, and general deployment guidelines.

1.1 Architecture

At the highest level, Business Transaction Management consists of three types of components:

- **Central servers** – The central servers are application EAR files that you deploy to an application server. There are three central servers. You deploy only one instance of each of these servers, and for performance considerations you should deploy each to a separate application server. You must not deploy any of the central servers to an application server that hosts services or components you intend to monitor. The central servers are:
  - **Main Server** (btmMain.ear) – Contains all the central Business Transaction Management system services and user interface applications, including the *sphere*. The sphere is the Business Transaction Management component that manages the Business Transaction Management environment. In addition, btmMain.ear contains a subdeployment for the F5 intermediary.
  - **Performance Server** (btmPerformanceServer.ear) – Contains the service-level management components. Deploy btmPerformanceServer.ear on an application server other than where btmMain.ear or btmTransactionServer.ear are deployed.
  - **Transaction Server** (btmTransactionServer.ear) – Contains the transaction management components. Deploy btmTransactionServer.ear on an application server other than where btmMain.ear or btmPerformanceServer.ear are deployed.
- **Observers** – Observers are sets of libraries that you install into the application server that hosts the business applications you want to monitor. The observers monitor messages and calls between the components of your applications.

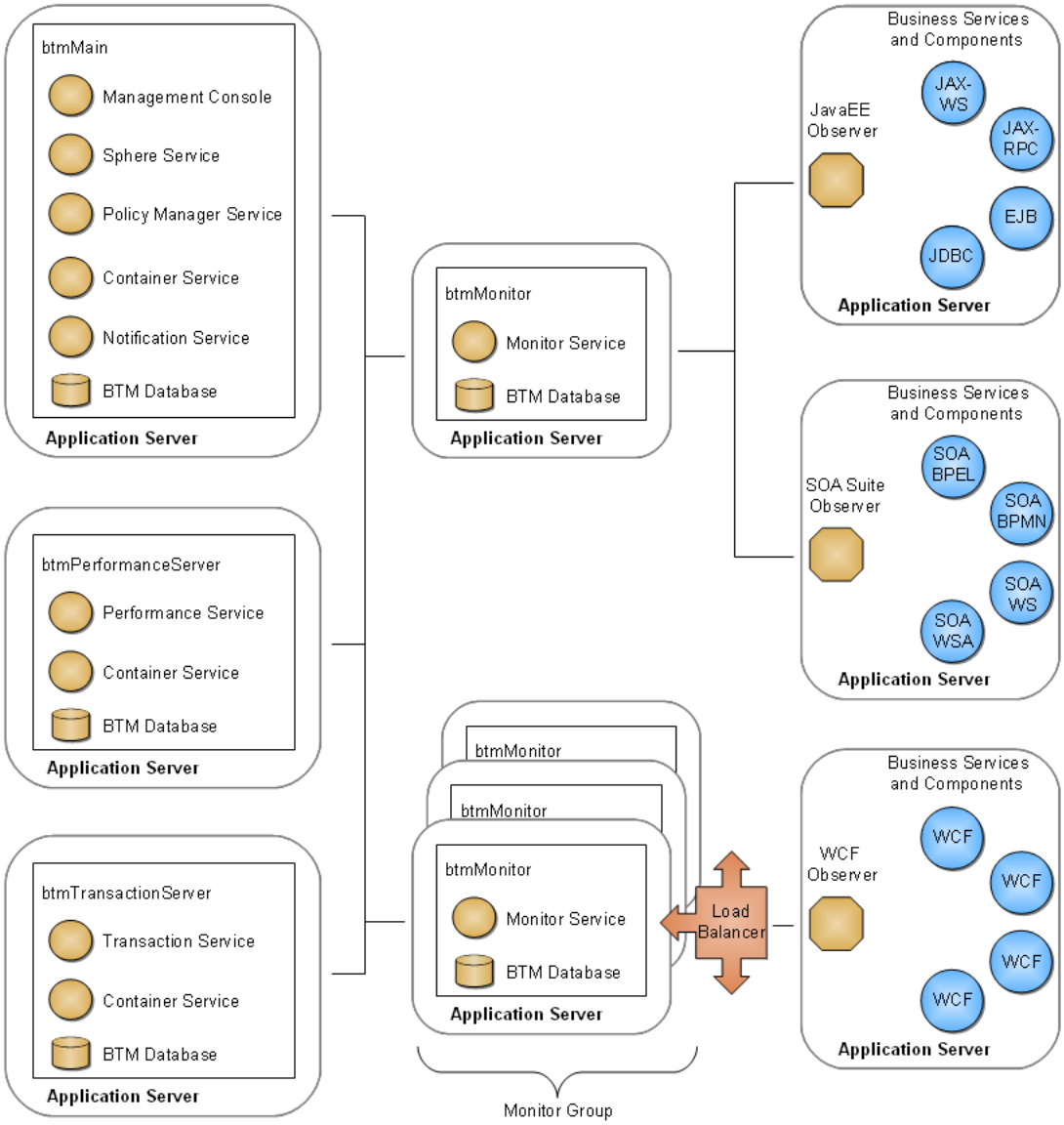
Observers are capable of monitoring many types of components, and are classified according to the type that they monitor, for example, JavaEE, OSB, WCF, etc.

- **Monitors** (btmMonitor.ear) – Monitors collect application performance and usage measurements from observers. The monitor is an application EAR file that you deploy to an application server. For large systems, you can deploy multiple monitors, either as singletons or replicates. For performance reasons, you should not deploy the monitor on an application server where the central servers are deployed.

Business Transaction Management also requires access to an Oracle RDBMS for storing performance measurements, logging messages, and maintaining the environment model and Business Transaction Management configuration.

The diagram below shows a typical distributed application environment, and the relationship of the Business Transaction Management components to that environment.

Figure 1–1 Deployment of Business Transaction Management components in a typical application environment



Business Transaction Management is designed for use in a distributed application environment in which the various Business Transaction Management components are deployed onto multiple machines and application servers.

Technically, you can install all the central servers into a single application server, but such a deployment scenario is not recommended for production environments. Installation in a single application server can be useful for demonstrations and for learning how to use the product, but this scenario might not scale successfully with a large number of business services or high volume of message traffic, just to name a few factors.

We recommend that you deploy each of the central servers to separate application servers. The Performance and Transaction components, in particular, typically perform a large amount of performance analysis computations. Dividing processes across application servers allows you to control memory and processor resources.

You should also deploy the monitor to an application server separate from the central servers. Depending on your monitoring requirements, you might need to deploy multiple monitors. You can deploy monitors either as singletons or as replicates behind a load balancer. For information about replicating the monitor, refer to Chapter 8, "Installing Monitors."

Observers must always be installed outside the application server hosting the central servers or monitors.

1.2 Packaging

Oracle distributes Business Transaction Management by way of ZIP files. The central servers and monitor are packaged together in BTM_Servers_*.zip (the * refers to the Business Transaction Management version number). The ZIP file's archive directory contains the central server and monitor deployments. Table 1–1 describes these deployments:

Table 1–1 The archives directory of BTM_Servers_*.zip contains the central servers and monitor in the following EAR files.

Deployment Name	Sub-deployments	Deployment strategy
btmMain.ear	btmcentral.war btmcontainer.war btmhelp.war btmui.war f5Intermediary.war	Deploy once per Business Transaction Management environment.
btmPerformanceServer.ear	btmcontainer.war btmperformance.war	Deploy once per Business Transaction Management environment on a separate application server from btmMain.ear and btmTransactionServer.ear.
btmTransactionServer.ear	btmcontainer.war btmtransaction.war	Deploy once per Business Transaction Management environment on a separate application server from btmMain.ear and btmPerformanceServer.ear.
btmMonitor.ear	btmmonitor.war	Deploy as many as needed on separate application servers from any of the central servers.

Observers are packaged in individual ZIP files according to platform and observer type. Table 1–2 lists the release 12.1.0.2 observers. The * in the ZIP file names refers to the observer version number, for example, 12.1.0.2.1.

Table 1–2 Observers distributed with release 12.1.0.2.

Observer ZIP File Name	Description
BTMObserver_Wls_10.3_JavaEE_*.zip	Contains the observer for JavaEE on WebLogic 10.3.
BTMObserver_Wls_10.3_Soa11gR1_*.zip	Contains the observer for Oracle SOA Suite on WebLogic 10.3.
BTMObserver_Wls_10.3_Osb11gR1_*.zip	Contains the observer for Oracle Service Bus 11gR1 on WebLogic 10.3.
BTMObserver_Iis_6-7_Wcf35_*.zip	Contains the observer for WCF 3.5 on Microsoft IIS version 6 or 7.
BTMObserver_Iis_6_Asp_*.zip	Contains the observer for ASP.NET on Microsoft IIS 6.

The observers listed in Table 1–3 are distributed as part of release 12.1.0.1. These observers are not included with, but are compatible with, release 12.1.0.2. You should install these observers if you require monitoring on their designated platforms. The * in the ZIP file names refers to the observer version number, for example, 12.1.0.1.0.

Table 1–3 Release 12.1.0.1 observers not included with, but compatible with, release 12.1.0.2.

Observer ZIP File Name	Description
BTMObserver_Wls_9.2_JavaEE_*.zip	Contains the observer for JavaEE on WebLogic 9.2.
BTMObserver_Wls_10.3_Osb10gR3_*.zip	Contains the observer for Oracle Service Bus 10gR3 on WebLogic 10.3.
BTMObserver_Was_6.1_JavaEE_*.zip	Contains the observer for JavaEE on WebSphere 6.1.
BTMObserver_Jboss_4.3_JavaEE_*.zip	Contains the observer for JavaEE on JBossEAP 4.3.

2

Upgrading Business Transaction Management

This chapter explains how to perform an in-place upgrade of Business Transaction Management from any version of release 11 or 12 to the current release. Performing an in-place upgrade means that you upgrade components by simply replacing them with new components, and without editing configuration settings.

A complete upgrade of Business Transaction Management consists of upgrading these components:

- all three central servers—these are, the Main server (btmMain.ear), the Performance server (btmPerformanceServer.ear), and the Transaction server (btmTransactionServer.ear)
- all monitors (btmMonitor.ear)
- all observers

If any of these components are older than release 11, do not attempt to perform an upgrade using the instructions in this chapter. Instead, enter a service request at My Oracle Support (http://support.oracle.com) for assistance in upgrading your installation.

If you choose to upgrade your system, you must upgrade all of the central servers and all of the monitors at the same time (a rolling upgrade is not supported between these components). However, you are not required to upgrade your observers, as long as the observers are at release 11 or higher. If you leave the observers at an older release version than the monitors, the monitors operate in compatibility mode in relation to the observers. Note, however, that if you do not upgrade the observers, you will not be able to take advantage of new functionality that depends on upgraded observers.

You must upgrade the Business Transaction Management central servers and monitors before upgrading the observers. Observers are permitted to be older than the central servers and monitors, but you must never install a version of the observer that is newer than the central servers and monitors.

Note: In comparing version numbers between the observer and the central servers and monitors, you need be concerned only with the digits up to the second point. For example, in "11.2.0.1", you need to consider only the "11" and the "2". Digits after the second point refer to patches and are not important in determining compatibility between the observer and the central servers and monitors.

To determine the version of your installed observer, open the observer's NanoAgent.log file and search for the line that displays the release number, for example:

```
INFO: Release 11.1.0.4: build 25237 of b21 (11.1.0.4/147754) on 2011-05-13
```

For information about locating the NanoAgent.log file, see Chapter 15, "Logging Observer Errors and Debugging Information."

2.1 Upgrading the Central Servers and Monitors

To upgrade your Business Transaction Management installation:

1. Back up your Business Transaction Management databases and configuration data.

   For information on how to perform this task, refer to the online help topics **Backup and Restore Concepts** and **Backing up BTM**. You can locate these topics by first choosing **Help > Help** in the Management console. After the online help opens, navigate to **Administering BTM > Backup and Restore** in the **Contents** pane.

   > **Note:** The central servers must be running in order for you to access the online help.

2. Back up the persistent storage directories for each of the central servers and monitors.

   For information on the locations of the persistent storage directories, refer to the online help topic **About Persistent Storage Directories**. You can locate this topic by first choosing **Help > Help** in the Management console. After the online help opens, navigate to **Administering BTM > Persistent Data** in the **Contents** pane.

3. Locate the distribution archive that contains the Business Transaction Management central servers and monitor and unzip it into a directory (referred to as henceforth *Install_Dir*).

   The distribution archive is named BTM_Servers_*.zip, where * is the Business Transaction Management version number.

4. *Optional security step for UNIX-like operating systems* – If you want to set permissions on the files that make up the distribution to the most restrictive level that still maintains functionality, complete this step:

   a. Locate setPermissions.sh at the top level of *Install_Dir*.

      This script contains commands for setting file permissions of all regular files to Owner – read/delete; all directories to Owner – read/execute/delete; and all scripts to Owner – read/execute/delete.

      > **Note:** These permission levels are extremely restrictive, for example, only the owner can read the files.

   b. On a command line, at the top level of *Install_Dir*, run this command:

      ```
      source setPermissions.sh
      ```

This command runs the commands in the script file and sets permissions for all files and directories in the expanded archive.

5. Shut down all of the central servers and monitors (btmMain.ear, btmPerformanceServer.ear, btmTransactionServer.ear, and btmMonitor.ear).

   It is essential that you shut them all down.

   For information about shutting down Business Transaction Management components, see Chapter 7, "Starting and Shutting Down Business Transaction Management."

6. Using your application server's deployment tools, redeploy each of the central servers and monitors using your new EAR files located in *Install_Dir*\archives.

7. Restart the central servers and monitors.

   Restarting the central servers and monitors triggers the in-place upgrade of all Business Transaction Management data. During this process, the system might suffer from reduced performance and some of the data might be temporarily unavailable. This process should complete within 30 minutes.

   For information about starting Business Transaction Management components, see Chapter 7, "Starting and Shutting Down Business Transaction Management."

8. Notify all Business Transaction Management users to flush their web browser caches.

   The Management Console contains a number of Adobe Flash widgets. Web browsers normally cache these widgets and will continue to use the older cached widgets until you either flush the cache or restart your web browser.

After you are finished with upgrading the central servers and monitors, and the system has settled down, you can optionally upgrade your observers by following the instructions in the next section.

You should also read the online help topic named **Post-Upgrade Configuration and Issues** for information about further upgrade-related tasks you might have to perform and changes in behavior to expect, such as the default values of fields. You can locate this topic by first choosing **Help > Help** in the Management console. After the online help opens, navigate to **Overview of BTM** in the **Contents** pane.

2.2 Upgrading Observers

You must upgrade the Business Transaction Management central servers and monitors, as described in the preceding section, before upgrading the observers.

The procedure for upgrading observers is specific to the application server and observer type. Refer to the following sections for detailed instructions on upgrading observers:

- section 2.2.1.1, "Upgrading the JavaEE Observer for WebLogic 10.3 on Node Manager-Configured Servers"
- section 2.2.2, "Upgrading Observers on WebSphere"
- section 2.2.3, "Upgrading Observers on JBoss"
- section 2.2.4, "Upgrading the Observer for WCF"
- section 2.2.5, "Upgrading the Observer for ASP.NET"

Note: For detailed information about a specific observer's compatibility and functionality, refer to the README.txt file located in the observer's nanoagent directory after you expand the observer ZIP file.

2.2.1 Upgrading Observers on WebLogic

This section describes how to upgrade an observer installed into a WebLogic application server. Separate procedures are provided for three different upgrade paths. The procedure you use depends on the type of observer you are upgrading, your version of WebLogic server, and whether your server is Node-Manager configured or script-configured. These are the procedures to choose from:

- section 2.2.1.1, "Upgrading the JavaEE Observer for WebLogic 10.3 on Node Manager-Configured Servers"
- section 2.2.1.2, "Upgrading the JavaEE Observer for WebLogic 10.3 on Script-Configured Servers"
- section 2.2.1.3, "Upgrading all Other Observers for WebLogic"

2.2.1.1 Upgrading the JavaEE Observer for WebLogic 10.3 on Node Manager-Configured Servers

1. Locate the distribution ZIP file for the JavaEE observer for WebLogic 10.3 (for example, BTMObserver_Wls_10.3_JavaEE_12.1.0.2.zip).
2. Open the WebLogic Administration Console (the default URL is http://*machine_name*:7001/console).
3. Remove *WL_HOME*/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar from your managed server's classpath (this setting was required by the previous release of the observer but must be removed for the current release):
   a. Using the Domain Structure pane (on the left), navigate to **Environment > Servers**.
   b. In the **Servers** table, click your managed server.
   c. Display the **Configuration / Server Start** tab.
   d. Click **Lock & Edit**.

   Note: These instructions assume you are operating in a production environment and that your WebLogic server's **Automatically Acquire Lock and Activate Changes** setting is therefore disabled. However, if this setting is enabled as it might be in a development environment, you do not have to click **Lock & Edit** in order to make changes and you do not have to activate changes after saving them.

   e. Delete the entry from the **Class Path** field as follows:

      For Windows systems, delete this string:

      ```
      WL_HOME\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar
      ```

      For UNIX-like systems, delete this string:

      ```
      WL_HOME/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
      ```

f. Keep this page open.

4. With the **Configuration / Server Start** tab still displayed, edit your WebLogic startup arguments as follows:

   a. Remove JVM arguments from the **Arguments** field as follows (these settings were required by the previous release of the observer but must be removed for the current release):

      For Windows systems, remove these arguments:

      ```
      -Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APA
      spectPreProcessor -javaagent:
      WL_HOME\nanoagent\lib\bootstrap\aspectwerkz-jdk5-2.0.jar
      ```

      For UNIX-like systems, remove these arguments:

      ```
      -Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APA
      spectPreProcessor -javaagent:
      WL_HOME/nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar
      ```

   b. Configure the observer bootstrap module into your server by adding a JVM argument to the **Arguments** field as follows.

      For Windows systems, add this argument:

      ```
      -javaagent:WL_HOME\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar
      ```

      For UNIX-like systems, add this argument:

      ```
      -javaagent:WL_HOME/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
      ```

5. Click **Save** and then click **Activate Changes**.

   You should receive the following status:

   "All changes have been activated. No restarts are necessary. Settings updated successfully."

6. Shut down the WebLogic managed server in which the observer is installed.

7. Delete the *WL_HOME*\nanoagent directory from your WebLogic managed server.

   The string *WL_HOME* refers to the server's home directory, which is the wlserver_10.3 directory located in your server's installation directory. For the remainder of this procedure, replace the string *WL_HOME* with the actual path to the WebLogic home directory.

8. Unpack the observer distribution ZIP file into *WL_HOME*.

   Unpacking the ZIP file creates a directory named nanoagent that contains three subdirectories bin, config, and lib.

9. Ensure that the user account running the WebLogic server has at least the following privileges:

   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory

10. If you inserted <filter> and <filter-mapping> elements into the web.xml files of your web applications during your original installation of the observer, you must now remove those elements.

The elements to remove from the web.xml files are these:

```
<filter>
  <filter-name>ORACLE_BTM_WEB_APP_OBSERVER</filter-name>
  <filter-class>
   com.amberpoint.nanoagent.bootstrap.servlet.FilterHandlerBootstrap
  </filter-class>
</filter>
<filter-mapping>
  <filter-name>ORACLE_BTM_WEB_APP_OBSERVER</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

11. Restart your WebLogic server.

2.2.1.2 Upgrading the JavaEE Observer for WebLogic 10.3 on Script-Configured Servers

> **Note:** When you originally installed the observer, you either installed it into all servers defined in the WebLogic installation or into servers of specific domains. The term *global install* refers to installing the observer into all servers, and the term *domain install* refers to installing the observer into a specific domain.

In the following procedure, you will replace the nanoagent directory and its contents. If you installed the observer as a global install, the nanoagent directory is located inside your WebLogic server's home directory, which is the wlserver_10.3 directory located in your WebLogic installation directory.

If you installed the observer as a domain install, the nanoagent directory is located inside your domain directory. In this case, you need to repeat this procedure for each domain in which you installed the observer.

1. Shut down the WebLogic application server in which the observer is installed.
2. Make a backup copy of your observer script file.

   The observer script file is the nanoEnvWeblogic.cmd or nanoEnvWeblogic.sh file located inside the nanoagent/bin directory.
3. Delete the nanoagent directory.
4. Unpack the observer distribution ZIP file (for example, BTMObserver_Wls_10.3_JavaEE_12.1.0.2.zip) into the directory from which you deleted the nanoagent directory.

   Unpacking the ZIP file creates a directory named nanoagent that contains three subdirectories bin, config, and lib.
5. Ensure that the user account running WebLogic has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
6. Open your new observer script file for editing and open your backup copy so you can copy settings from it.

7. Ensure that the values of the NANOAGENT_HOME and NANOAGENT_CONFIGURATION_URL variables in your new observer configuration file match the values in your backup copy.
8. If you inserted <filter> and <filter-mapping> elements into the web.xml files of your web applications during your original installation of the observer, you must now remove those elements.

   The elements to remove from the web.xml files are these:

```
<filter>
  <filter-name>ORACLE_BTM_WEB_APP_OBSERVER</filter-name>
  <filter-class>
   com.amberpoint.nanoagent.bootstrap.servlet.FilterHandlerBootstrap
  </filter-class>
</filter>
<filter-mapping>
  <filter-name>ORACLE_BTM_WEB_APP_OBSERVER</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

9. Restart your WebLogic server.

2.2.1.3 Upgrading all Other Observers for WebLogic

This section explains how to upgrade any type of observer installed in a WebLogic server other than the JavaEE Observer installed in a WebLogic 10.3 server.

> **Note:** Observers for some older platforms (such as WebLogic 9.2) are distributed with release 12.1.0.1 rather than release 12.1.0.2. Refer to section 1.2, "Packaging" for details.

1. Shut down the WebLogic application server in which the observer is installed.
2. Make a backup copy of the *WL_HOME*\nanoagent directory.

   The string *WL_HOME* refers to your WebLogic server's home directory, which is the weblogic92, wlserver_10.0, or wlserver_10.3 directory located in your WebLogic installation directory.
3. Delete the *WL_HOME*\nanoagent directory.
4. Unpack the observer distribution ZIP file (for example, BTMObserver_Wls_10.3_Soa11gR1_12.1.0.2.zip) into *WL_HOME*.

   Unpacking the ZIP file creates a directory named nanoagent that contains three subdirectories bin, config, and lib.
5. Ensure that the user account running WebLogic has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
6. If you originally installed the observer by editing script files rather than by using the Node Manager, replace your new observer script file with the observer script file located in your backup copy of the nanoagent directory. If you use the Node Manager, you can skip this step.

   On Windows systems, the observer script file is located at:

WL_HOME\nanoagent\bin\nanoEnvWeblogic.cmd

On UNIX-like systems, the observer script file is located at:

WL_HOME/nanoagent/bin/nanoEnvWeblogic.sh

7. Restart your WebLogic server.

2.2.2 Upgrading Observers on WebSphere

This section explains how to upgrade an observer installed in a WebSphere application server.

> **Note:** The most recently released observer for this platform is distributed as part of release 12.1.0.1. It is not included with, but is compatible with, release 12.1.0.2.

1. Shut down the WebSphere server in which the observer is installed.
2. Make a backup copy of the *WAS_INSTALL_ROOT*/nanoagent directory.

   WAS_INSTALL_ROOT refers to your WebSphere installation root directory. The default location of this directory on Windows systems is:

   C:\Program Files\IBM\WebSphere\AppServer

   On UNIX-like systems, the default location is:

   /opt/IBM/WebSphere/AppServer

3. Delete the *WAS_INSTALL_ROOT*/nanoagent directory.
4. Unpack the observer distribution ZIP file (BTMObserver_Was_6.1_JavaEE_*.zip) into *WAS_INSTALL_ROOT*.

   Unpacking the ZIP file creates a directory named nanoagent that contains two subdirectories—config and lib.
5. Ensure that the user account running WebSphere has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
6. Restart your WebSphere server.

2.2.3 Upgrading Observers on JBoss

This section explains how to upgrade an observer installed in a JBossEAP application server.

> **Note:** The most recently released observer for this platform is distributed as part of release 12.1.0.1. It is not included with, but is compatible with, release 12.1.0.2.

1. Shut down the JBoss server in which the observer is installed.

2. Make a backup copy of the *JBOSS_HOME*/nanoagent directory.

   JBOSS_HOME refers to the value of your JBOSS_HOME environment variable.

   ```
   C:\Program Files\IBM\WebSphere\AppServer
   ```

   On UNIX-like systems, the default location is:

   ```
   /opt/IBM/WebSphere/AppServer
   ```

3. Delete the *JBOSS_HOME*/nanoagent directory.
4. Unpack the observer distribution ZIP file (BTMObserver_Jboss_4.3_JavaEE_*.zip) into *JBOSS_HOME*.

   Unpacking the ZIP file creates a directory named nanoagent that contains three subdirectories—config, jaxws, and lib.

5. Ensure that the user account running WebSphere has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
6. Restart your JBoss server.

2.2.4 Upgrading the Observer for WCF

This section explains how to upgrade an observer for WCF.

1. Unpack the observer distribution file (BTMObserver_Iis_6-7_Wcf35_*.zip) into a temporary directory (referred to henceforth as *observer_temp*).

   Unpacking the ZIP file creates a nanoagent directory containing two subdirectories—config and lib. The lib directory contains the observer DLL files.

2. Make a note of the version number of the new DLLs.

   To find the version number, open the Windows Properties dialog box for one of the DLL files and click the Version tab.

3. Use gacutil.exe or a Windows Explorer to copy all of the DLL files from *observer_temp*\nanoagent\lib to the global application cache (GAC; normally located at C:\WINDOWS\assembly).
4. Using a text editor, open the application configuration file that contains the observer configuration code that you added when you originally installed the observer.

   The file is either the machine.config, or a web.config file.

5. In the application configuration file, locate the two occurrences of the Version attribute that refer to the version number of the observer DLLs.

   The elements containing the Version attributes are inside a </behaviorExtensions> element and should look similar to this:

   ```
   <add name="APEPInterceptor"
   type="AmberPoint.NanoAgent.DotNet.Wcf.APEPBehaviorExtnElem,
   AmberPoint.NanoAgentWCF, Version=64000.64000.25233.19024, Culture=neutral,
   PublicKeyToken=d8685c0afbb35893"/>

   <add name="APServiceInterceptor"
   type="AmberPoint.NanoAgent.DotNet.Wcf.APServiceBehaviorExtnElem,
   ```

```
AmberPoint.NanoAgentWCF, Version=64000.64000.25233.19024, Culture=neutral,
PublicKeyToken=d8685c0afbb35893"/>
```

6. Edit the setting of the Version attribute so that it matches the version number of the new observer DLLs, and then save and close the file.

> **Note:** If you configured the observer into multiple web.config files on the machine, edit the Version attributes in each file.

7. *Optional* – Remove the old observer DLLs from the GAC (unless they are being used by another observer on the machine).

   The name of each observer DLL begins with the string "AmberPoint". To remove a DLL, right-click it and choose **Uninstall**.

> **Note:** The observer for ASP.NET uses many of the same DLLs as the observer for WCF. If you have the observer for ASP.NET installed on the machine, you must not remove the version of the DLLs that are being used by it.

2.2.5 Upgrading the Observer for ASP.NET

This section explains how to upgrade an observer for ASP.NET.

1. Unpack the observer distribution file (BTMObserver_Iis_6_Asp_*.zip) into a temporary directory (referred to henceforth as *observer_temp*).

   Unpacking the ZIP file creates a nanoagent directory containing two subdirectories—config and lib. The lib directory contains the observer DLL files.

2. Make a note of the version number of the new DLLs.

   To find the version number, open the Windows Properties dialog box for one of the DLL files and click the Version tab.

3. Use gacutil.exe or a Windows Explorer to copy all of the DLL files from *observer_temp*\nanoagent\lib to the global application cache (GAC; normally located at C:\WINDOWS\assembly).

4. Using a text editor, open the web.config file that contains the observer configuration code that you added when you originally installed the observer.

5. In the web.config file, locate the Version attribute that refers to the version number of the observer DLLs.

   Here is an example of the XML code containing the version number:

```
<system.web>
 <webServices>
  <soapExtensionTypes>
   <add type="AmberPoint.NanoAgent.DotNet.AspNet.Handlers.SoapExtensionHandler,
    AmberPoint.NanoAgentAspNet, Version=64000.64000.25642.24032,
    Culture=neutral, PublicKeyToken=d8685c0afbb35893" priority="1" group="0" />
   </soapExtensionTypes>
  </webServices>
</system.web>
```

6. Edit the setting of the Version attribute so that it matches the version number of the new observer DLLs, and then save and close the file.

Note: If you configured the observer into multiple web.config files on the machine, edit the Version attributes in each file.

7. *Optional* – Remove the old observer DLLs from the GAC (unless they are being used by another observer on the machine).

   The name of each observer DLL begins with the string "AmberPoint". To remove a DLL, right-click it and choose **Uninstall**.

   Note: The observer for WCF uses many of the same DLLs as the observer for ASP.NET. If you have the observer for WCF installed on the machine, you must not remove the version of the DLLs that are being used by it.

3

Installation Overview

This chapter provides an overview of the entire Business Transaction Management installation procedure.

3.1 Installation Overview

1. Installation and initial configuration of the central servers:
   a. *Optional* – Configure security for the central servers. You can also perform the security configuration for the monitors and observers at this time, if you wish.

      See Chapter 4, "Configuring Security."

   b. Ensure that you have a supported web browser installed.

      Section 5.1, "Web Browser Requirements."

   c. Configure the application servers that will host your central servers.

      See section 5.2, "Setting up your WebLogic Environment" or Section 5.3, "Setting up your WebSphere Environment," depending on your application server.

   d. Set up the Business Transaction Management databases.

      See Section 5.4, "Setting up Business Transaction Management Databases."

   e. *Optional* – Configure the persistent storage directories.

      See Section 6.2, "Configuring Persistent Storage Directories."

   f. Deploy the central servers.

      See Section 6.3, "Deploying the Central Servers."

   g. If needed, remap Business Transaction Management user roles in your application server.

      See Section 6.4, "Mapping Users to Business Transaction Management Application Roles."

   h. Perform initial configuration of Business Transaction Management.

      After deploying the central servers, you configure them using the browser-based configuration wizard (see Section 6.5, "Initial Configuration of Business Transaction Management.").

      Alternatively, you can use a command line script (see Chapter 16, "Scripted Configuration of Oracle Business Transaction Management."). For first-time configuration, however, we recommend that you use the browser-based

wizard. The wizard produces an XML output file that can be used with the command line script for subsequent configurations.

2. Installation of monitors and configuration of the Observer Communication policy (see Chapter 8, "Installing Monitors."):
   a. *Optional* – Configure security for the monitors if you haven't already done so.

      See Chapter 4, "Configuring Security."
   b. Deploy monitors.
   c. Configure monitor-observer communication by way of the Observer Communication policy.
3. Installation of observers (see Chapter 9, "Installing Observers Overview"):
   a. *Optional* – Configure security for the observers if you haven't already done so.

      See Chapter 4, "Configuring Security."
   b. Install the observer libraries.
   c. Configure the observers to locate a monitor.

4

Configuring Security

This chapter explains how to configure security for Business Transaction Management. You should read this chapter thoroughly, and, if you decide to configure security, you should perform the security configuration for the relevant execution environment before installing any Business Transaction Management components into it.

You can configure security for Business Transaction Management using the following mechanisms:

- Network-level security, in which you configure your application servers to communicate over HTTPS.

  For management components that communicate using HTTPS, this security mechanism provides one-way or mutual authentication, message integrity, and encryption of on-the-wire messaging data. Refer to section 4.2, "Setting up Network-Level Security" for details.

- Business Transaction Management's built-in security for HTTP communications between Business Transaction Management components.

  This security mechanism provides trust between management components that communicate using HTTP(S) and encryption of sensitive elements of their messaging data prior to serialization (that is, before being sent over the wire or stored to disk). Refer to section 4.3, "Configuring the Business Transaction Management Assertion Secret and Encryption Key" for details.

- Business Transaction Management's built-in security for the transmission of observation messages from observers to monitors over a secure socket (SSL).

  This security mechanism provides server authentication between an observer and monitor, message integrity, and encryption of observation messages sent over the wire. Refer to section 4.4, "Setting up a Secure Socket (SSL) for Observation Messages" for details.

4.1 Communication Protocols and Deployment Scenarios

Business Transaction Management is designed for use in a distributed environment. As such, some components might be deployed in remote data centers, in a DMZ, or in a network without access to a centralized database. The following table describes the type of access required between the various Business Transaction Management components and between those components and your business services.

Table 4–1 Business Transaction Management Communication Protocols

From this Component	Required Protocol	To this Component	Description	Index to Diagrams
Each central server (the Main, the Transaction, and the Performance server)	HTTP or HTTPS	Any other central server	The central servers must be able to communicate with each other. The usual deployment topology will place each of these in the same network zone.	A
Each central server	HTTP or HTTPS	All monitors	Each of the central servers must be able to communicate with all monitors. This includes monitors that are deployed outside of the central network zone.	B
Main server	TCP, HTTP or HTTPS	All monitored business services	The Main server should be able to communicate with the monitored business services to determine whether they are "alive" (up or down). Note that the precise protocol used to determine aliveness can be configured on a per-service basis. (For information on configuring this protocol, enter a service request at My Oracle Support.)	C
Each monitor	HTTP or HTTPS	All central servers	Monitors must be able to communicate with each of the central servers.	D
Each monitor	JDBC	A Message Log Database (messageLogDB)	Each monitor must be able to write to, and read from, its Message Log Database using JDBC. The Message Log Database can be a single database inside a trusted network that is used by all monitors. Alternatively, if you have monitors in a DMZ or in a network without access to a centralized database, you can deploy one or more databases for the remote monitors.	E
Each observer	TCP and HTTP or HTTPS	Associated monitor	Each observer requires two connections to its associated monitor. The observer uses an HTTP(S) connection to download its configuration from the monitor. To transmit observation messages to the monitor, the observer connects using Observation Protocol (OP) over TCP or Observation Protocol Secure (OPS) over a secure socket. For monitor groups, these connections are made to the load balancer that fronts the monitor group. The socket port number is configured by way of the Observer Communication policy.	F
Transaction server	JDBC	All Message Log Databases	Optional. The Transaction server does not require direct access to the Message Log Databases. However, by providing such access, you can improve the performance of message log queries. You should enable this communication channel whenever possible.	G
Main server	JDBC	Sphere database (sphereDB)	The Main server must be able to write to the Sphere database using JDBC.	H

Table 4–1 (Cont.) Business Transaction Management Communication Protocols

From this Component	Required Protocol	To this Component	Description	Index to Diagrams
Performance	JDBC	Measurement database (measurementDB)	The Performance server must be able to write to the Measurement database using JDBC.	I
Transaction	JDBC	Transaction database (transactionDB)	The Transaction server must be able to write to the Transaction database using JDBC.	J
Monitor (in a monitor group)	JDBC	Monitor group database	Each monitor in a particular monitor group must be able to write to the same monitor group database using JDBC.	K

The following diagrams illustrate the type of access required by Business Transaction Management components in a variety of deployment scenarios. Note that the circled letters in the diagrams cross reference Table 4–1, " Business Transaction Management Communication Protocols".

- Figure 4–1, "Deployed Business Transaction Management Components and Business Services"
- Figure 4–2, "Communication Connections Among the Central Servers and Between the Central Servers and Monitors"
- Figure 4–3, "Connections Between Business Transaction Management Components and Databases"
- Figure 4–4, "Communication Connections Between the Monitors and Observers"
- Figure 4–5, "Communication Connections Between the Main Server and Business Services"

Figure 4–1 Deployed Business Transaction Management Components and Business Services

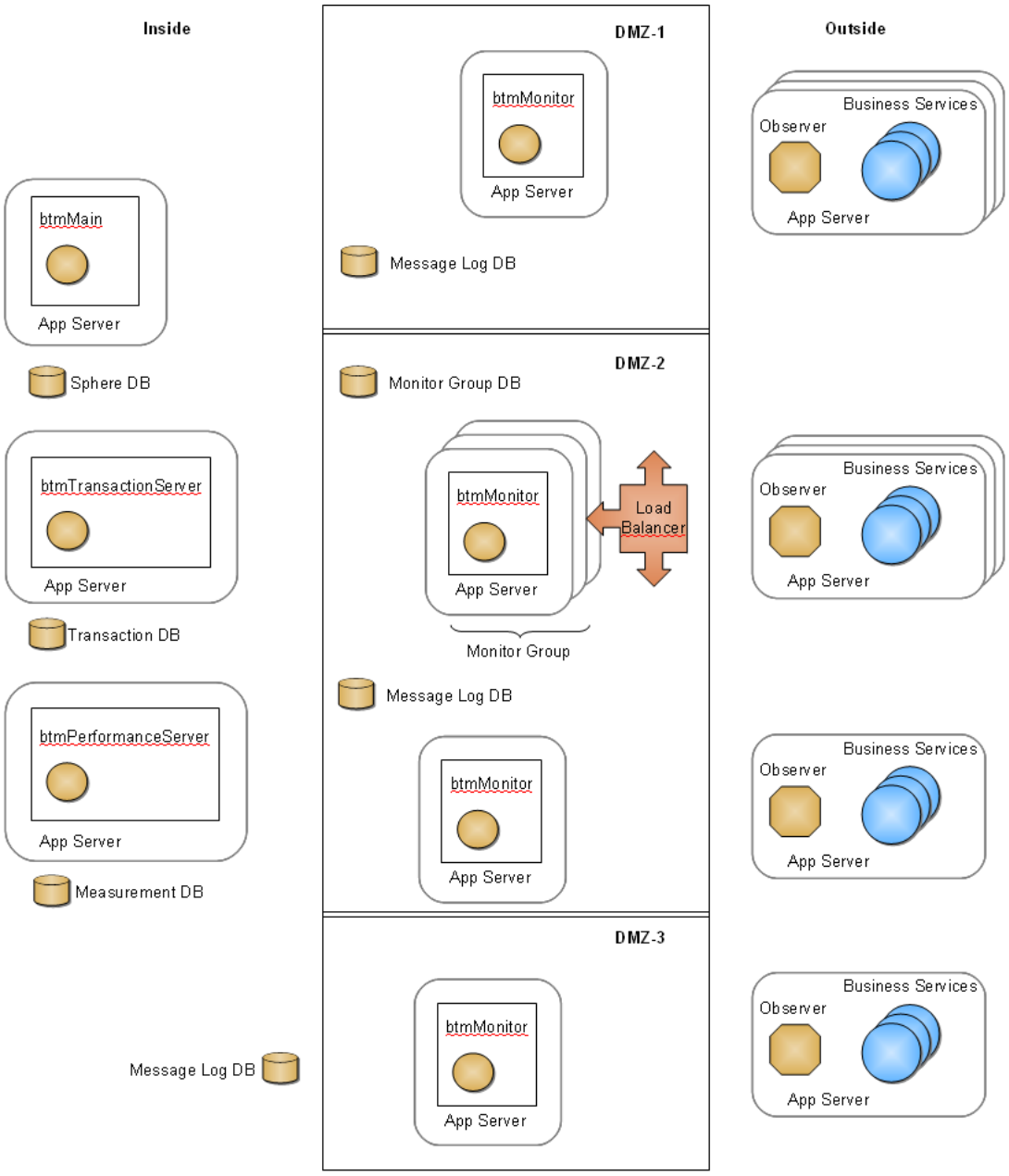


Figure 4–2 Communication Connections Among the Central Servers and Between the Central Servers and Monitors

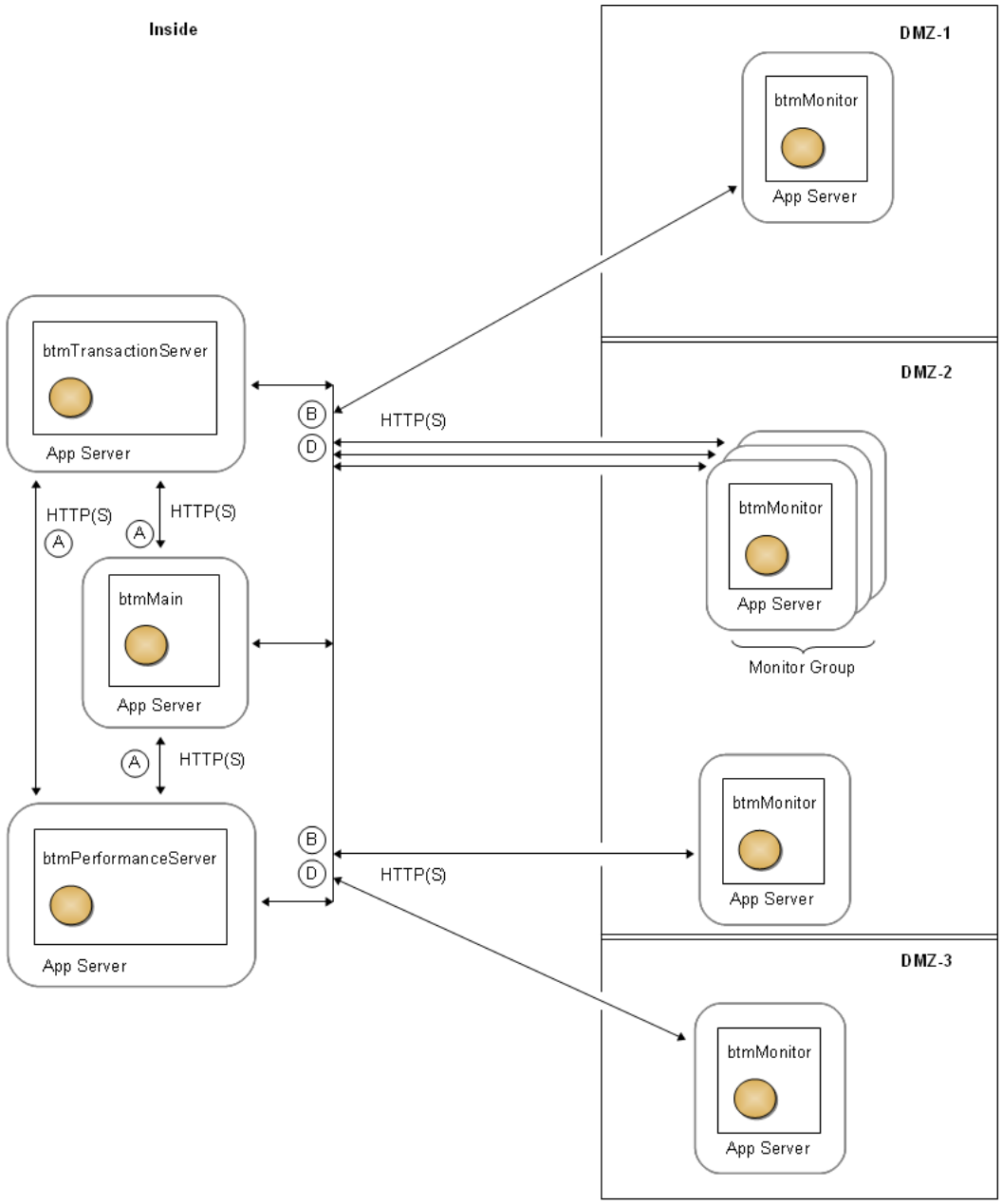


Figure 4–3 Connections Between Business Transaction Management Components and Databases

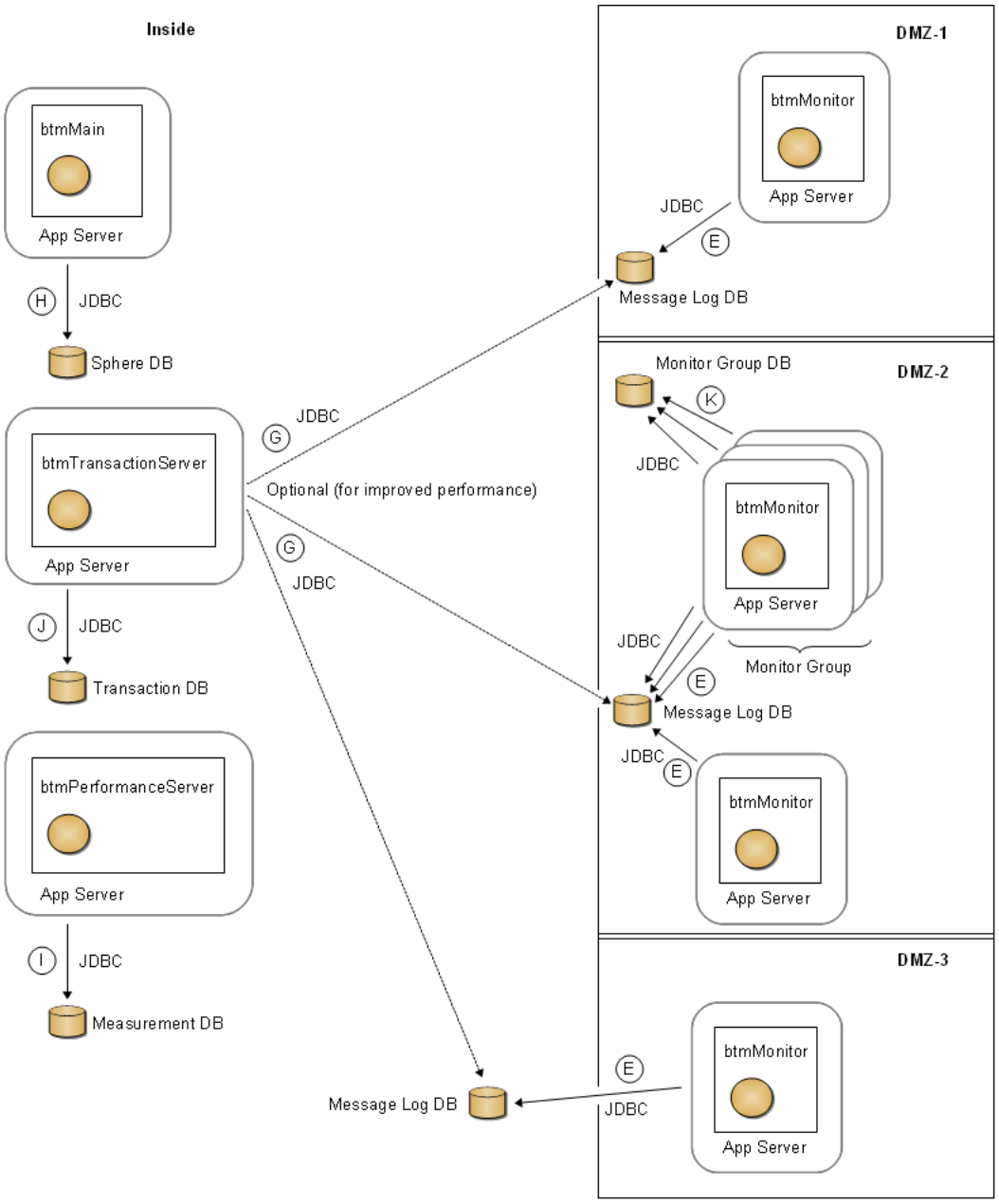


Figure 4–4 Communication Connections Between the Monitors and Observers

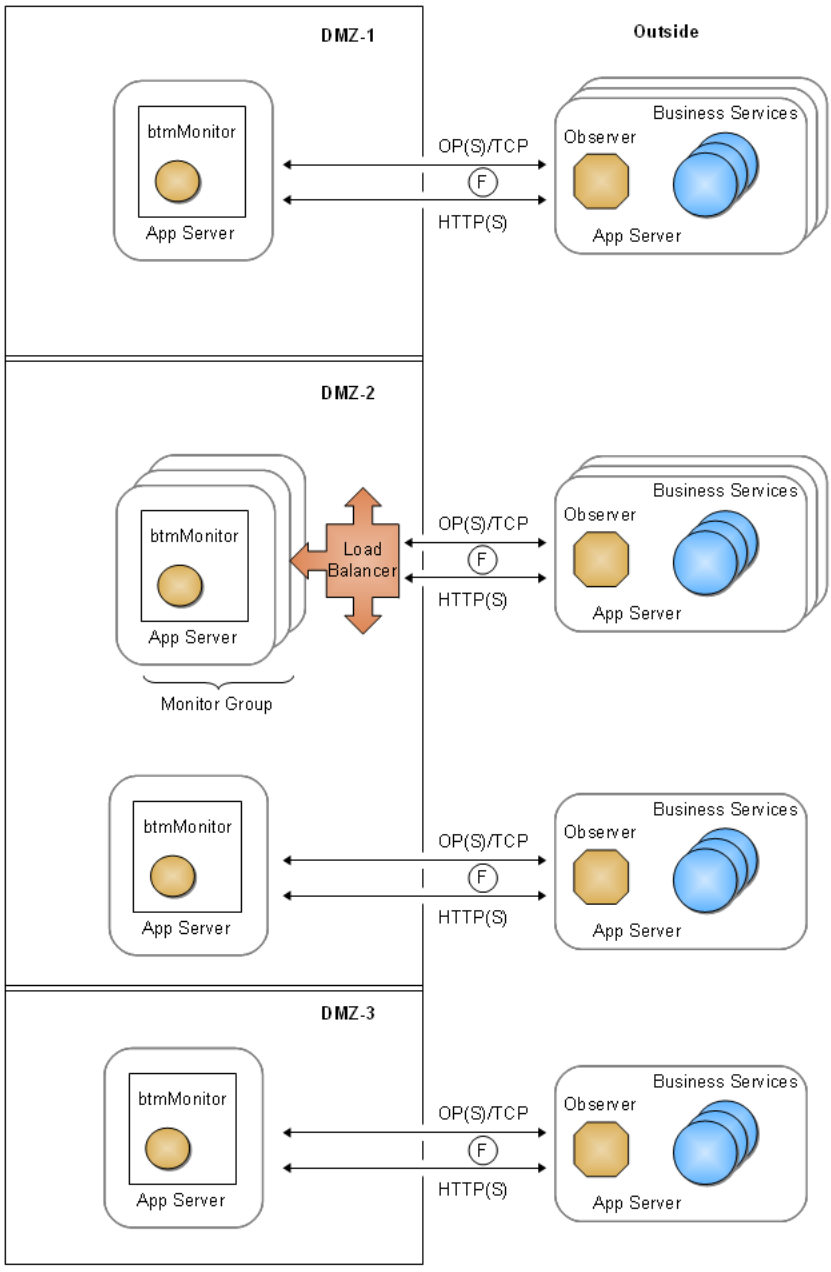
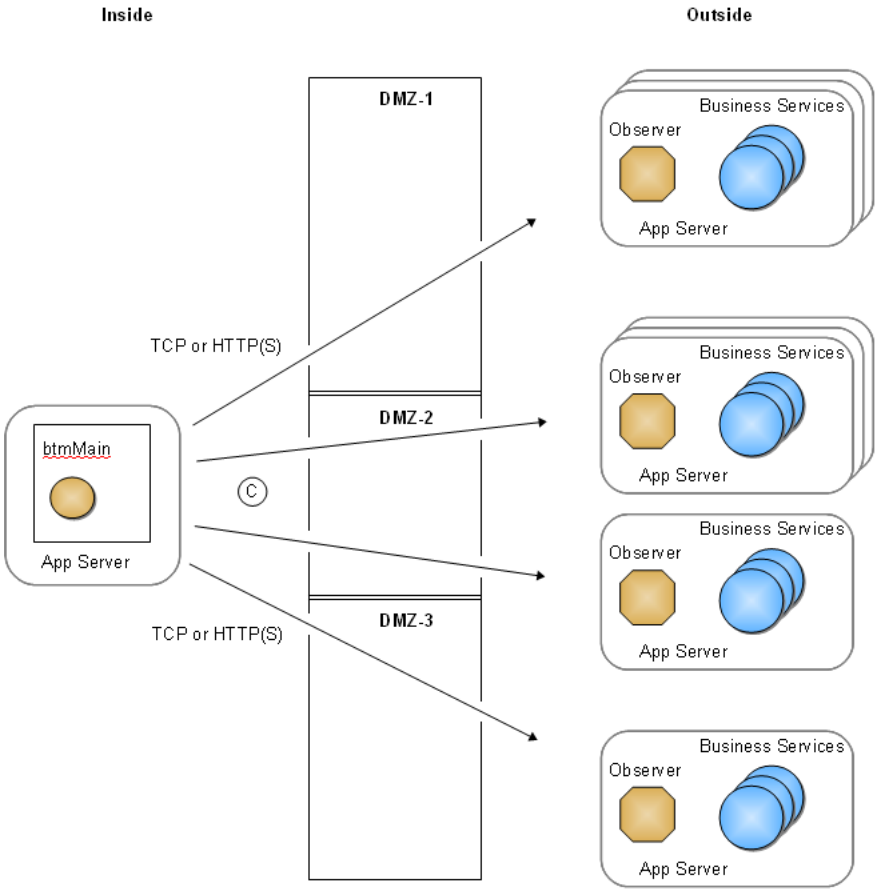


Figure 4–5 Communication Connections Between the Main Server and Business Services



4.2 Setting up Network-Level Security

If you want to enable network-level security for Business Transaction Management components, you can do so using SSL/TLS provided by the application server in which the components are deployed. SSL provides message integrity and confidentiality for communication among distributed management components, encrypting management traffic as it flows from one server to another. When configured with client certificates, SSL also provides mutual authentication between management components. Business Transaction Management provides no specialized support for SSL and does not interfere with standard SSL configurations. Therefore, any configuration supported by your application server can be used to secure Business Transaction Management traffic. It is strongly recommended that you first ensure that your application servers are properly configured for SSL before you install any Business Transaction Management components.

Once you have configured and tested SSL on the application servers where Business Transaction Management will be deployed, you will install Business Transaction Management and then configure it using the browser-based Initial Configuration wizard. During initial configuration, you will provide the SSL address and port number for your installation. The Business Transaction Management components will automatically communicate over the secured transports.

4.2.1 Configuring HTTPS

This section pertains to the central servers.

If you are installing the central servers into an HTTPS-only environment (that is, an environment in which there is no HTTP traffic between components), you must do one of the following:

- Ensure that the application servers hosting the central servers are listening only on HTTPS ports.

  Or

- Modify each web.xml file packaged in each of the central server WAR files so as to specify that only HTTPS access is allowed. You can do this by simply uncommenting the following <security-constraint> element that is provided in those files:

```
<security-constraint>
  <display-name>Require SSL communication</display-name>
  <web-resource-collection>
    <web-resource-name>All AmberPoint system services</web-resource-name>
    <url-pattern>/*</url-pattern>
  </web-resource-collection>
  <user-data-constraint>
    <transport-guarantee>CONFIDENTIAL</transport-guarantee>
  </user-data-constraint>
</security-constraint>
```

4.2.2 Configuring Firewalls

This section pertains to the central servers, monitors, and observers.

If you are using a firewall, you must configure it to allow access to each port on which a Business Transaction Management component receives communications. These are:

- the ports on which the hosting application servers listen.
- the sockets on which the monitors receive observation messages from the observers.
- the ports used for JDBC connections to Business Transaction Management databases. These databases are discussed in Section 5.4, "Setting up Business Transaction Management Databases."

4.3 Configuring the Business Transaction Management Assertion Secret and Encryption Key

---

Note: If you choose to perform the configuration described in this section, then you must do so on each application server that hosts a Business Transaction Management central server, monitor, or observer. You must also perform this configuration for the execution environment in which you use the Business Transaction Management command line interface (CLI).

Communications between Business Transaction Management components are secured by way of trusted assertions. This means that for your Business Transaction Management components to communicate with each other, and for your Business Transaction Management installation to function properly, every Business Transaction

Management component must be configured with an assertion secret of the same value.

Business Transaction Management also encrypts sensitive data contained in the communications between its components. It encrypts this data for both on-the-wire communications and storage in the Business Transaction Management databases.

These security mechanisms are enabled by default, and all Business Transaction Management components are preconfigured with a default value for both the assertion secret and the encryption key. This default security configuration fully enables the security mechanisms and, at the same time, simplifies the installation of Business Transaction Management.

However, because every Business Transaction Management installation uses the same default values, using the default values is a potential security threat. For demonstration purposes, and perhaps for development environments, using the default values might be adequate. But, in production environments, you should tighten security by providing your own unique values. You should also use your own values in your test environment before deploying Business Transaction Management into your production environment. If you intend to provide your own values for the assertion secret and encryption key, you should perform that configuration on each application server that hosts a Business Transaction Management component before you deploy the component.

For components deployed to WebLogic servers, you have a choice as to the method you use for configuring and storing these security settings—you can use either Oracle Wallet or Java system properties. (Oracle Wallet is an implementation of Oracle Credential Store Framework, or OCSF, and is a component of Java Platform Security, or JPS.) For all other Java application servers you use Java system properties. And, for observers deployed to .NET execution environments, you use environment variables.

You should use Oracle Wallet, if possible, because this method provides a tighter level of security. Oracle Wallet stores the assertion secret and encryption key in files that are protected by the operating system's file system security. Java system properties and environment variables, on the other hand, are visible to any user that has access to the machine. If necessary, you can mix these security configuration methods. For example, you could use Oracle Wallet on some application servers but use Java system properties (or environment variables for .NET observers) on other application servers.

4.3.1 Configuring Security Using Oracle Wallet

This section pertains to central servers, monitors, and observers that will be deployed to WebLogic application servers, only. It explains how to configure the assertion secret and encryption key for your Business Transaction Management components using Oracle Wallet. You must repeat this procedure for each central server, monitor, and observer for which you want to configure the assertion secret and encryption key using Oracle Wallet.

1. Ensure that the WebLogic domain in which the Business Transaction Management component will be installed includes the Java Runtime Files (JRF) template.

   If the domain does not yet exist, be sure to add the JRF template when you create the domain. If the domain already exists but doesn't include the JRF template, extend the domain and add the template.

   The JRF template includes the JPS JAR files that are referred to in later steps. (JPS is a component of Oracle Platform Security Services.)

2. Decide on names for the credentials that will hold your assertion secret and encryption key.

You are free to choose any name you want, but each of the two credentials must have a different name, and you must use the same two credential names for all of your Business Transaction Management components.

3. Decide on values for your issuer name and issuer assertion secret.

   You are free to choose any values you want for these strings, but you must use the same values for all of your Business Transaction Management components.

4. Locate the distribution archive for the Business Transaction Management central servers and expand it into a directory on the machine that hosts the central server, monitor, or observer for which you want to configure security (this directory is henceforth referred to as *BTM_Central_Expanded*).

   The distribution archive is named BTM_Servers*.zip, where * represents the version number.

5. Configure the setBtmOverriderEnv_via_CredStore script file by completing the following substeps.

   Configuring the setBtmOverriderEnv_via_CredStore script file accomplishes these tasks: adds the JPS JAR files to your application server's classpath; overrides the default credential store; and specifies the names of the credentials that hold the shared secret and encryption key (note that you will create the credentials in a later step).

   a. Locate the setBtmOverriderEnv_via_CredStore script file in BTM_Central_Expanded/security_add_ons.

      For Windows systems, use setBtmOverriderEnv_via_CredStore.cmd; for UNIX-like systems, use setBtmOverriderEnv_via_CredStore.sh.

   b. Copy the script file to your WebLogic server:

      If you want to configure security for all domains, copy the script file to your WebLogic server's home directory. The home directory is the weblogic92 or wlserver_10.3 directory located in your WebLogic installation directory, for example, C:\bea\wlserver_10.3.

      If you want to configure security for a particular domain, copy the script file to the top level of that domain's directory.

   c. Open the script file in a text editor.

   d. Specify the name of the credential that holds the assertion secret by replacing the string **>>> YOUR_ISSUER_SECRET_CREDENTIAL_NAME_HERE <<<** with the credential name you chose for your assertion secret.

   e. Specify the name of the credential that holds the encryption key by replacing the string **>>> YOUR_ENCRYPT_KEY_CREDENTIAL_NAME_HERE <<<** with the credential name you chose for your encryption key.

   f. Save and close the script file.

6. Configure your WebLogic domain startup scripts to call the setBtmOverriderEnv_via_CredStore script file:

   Note: This step assumes that you haven't modified your startWebLogic scripts. If you have modified your scripts, you might also have to modify the installation procedure accordingly.

a. Navigate to the bin directory of one of the WebLogic domains for which you want to configure security and open the startup script in a text editor (open bin\startWebLogic.cmd for Windows systems or bin/startWebLogic.sh for UNIX-like systems; do not edit the startup script located directly within the domain directory).

b. Locate the following line (the first line is for Windows and the second for UNIX-like systems):

```
call "%DOMAIN_HOME%\bin\setDomainEnv.cmd"

. ${DOMAIN_HOME}/bin/setDomainEnv.sh
```

c. Directly after that line, add a line that calls the setBtmOverriderEnv_via_CredStore script file:

If you are configuring security for all domains on a Windows system, add this line:

```
call "%WL_HOME%\setBtmOverriderEnv_via_CredStore.cmd"
```

If you are configuring security for all domains on a UNIX-like system, add this line (note the initial period and space):

```
. ${WL_HOME}/setBtmOverriderEnv_via_CredStore.sh
```

If you are configuring security for a particular domain on a Windows system, add this line:

```
call "%DOMAIN_HOME%\setBtmOverriderEnv_via_CredStore.cmd"
```

If you are configuring security for a particular domain on a UNIX-like system, add this line (note the initial period and space):

```
. ${DOMAIN_HOME}/setBtmOverriderEnv_via_CredStore.sh
```

d. Repeat this step for each domain for which you want to configure security.

7. Add permission grants for Business Transaction Management components to the policy store:

a. Navigate to the **config\fmwconfig** directory of one of the WebLogic domains for which you are configuring security.

This directory was created when you extended the domain to include the JRF template and is the default location where the oracle.security.jps.config property looks.

b. Open the file **system-jazn-data.xml** in a text editor.

c. For each deployment in the domain for which you want to configure security, add a **<grant>** element as a child of the **<jazn-data>/<jazn-policy>** element, replacing the string **Your Deployment Name Here** with the name of the deployment, as shown in the following example:

```
<jazn-data>
 <jazn-policy>
 (... other <grant> elements, etc. ...)

  <!-- Begin of Business Transaction Management grants -->
  <grant>
   <grantee>
    <codesource>
```

```
     <url>file:${domain.home}/servers/${weblogic.Name}/tmp/_WL_user/Your
Deployment Name Here/-</url>
    </codesource>
   </grantee>
   <permissions>
    <permission>
     <class>
      oracle.security.jps.service.credstore.CredentialAccessPermission
     </class>
     <name>context=SYSTEM,mapName=BTM,keyName=*</name>
     <actions>read,write</actions>
    </permission>
   </permissions>
  </grant>
  <!-- End of Business Transaction Management grants -->

 (... other <grant> elements, etc. ...)
 </jazn-policy>
</jazn-data>
```

Table 1–1 lists the deployment names for the central servers and monitors. For observers, use the name of the deployment that contains the business application you want to monitor.

d. Save and close the file.

e. Repeat this step for each domain for which you want to configure security.

8. Use the Business Transaction Management command line interface (CLI) to create a "Trusted Issuer and Secret" credential that contains your assertion secret and add it to the Oracle Wallet credential store associated with the policy store that you just configured:

In the following substeps, you will first set up the environment for the CLI, then point the CLI to the Oracle Wallet credential store, and finally use the CLI to create the credential for your assertion secret and add it to the store.

a. Open a command shell or window and ensure that its JAVA_HOME and JPS_11_HOME environment variables are set properly.

These variables can be set at the system or the shell/window level. The JAVA_HOME variable must point to a JDK that is version 6 or higher. The JPS_11_HOME variable must point to your oracle.jps file, for example, to:

```
oracle\Middleware\oracle_common\modules\oracle.jps_11.1.1
```

b. Open the file *BTM_Central_Expanded*/**security_add_ons/jps_config_dir/jps-config.xml** in a text editor and locate the following line:

```
<serviceInstance name="credstore" provider="credstoressp" location="./">
```

c. Change the value of the **location** attribute from **./** to the **/config/fmwconfig/** directory under the WebLogic domain directory, for example:

```
<serviceInstance name="credstore" provider="credstoressp"
location="C:/Oracle/Middleware/user_projects/domains/my_
domain/config/fmwconfig/">
```

The value of the **location** attribute must match the location of the %DOMAIN_HOME%\config\fmwconfig directory where you configured the policy store for your domain, in step 7.

d. In your command shell/window, navigate to the *BTM_Central_Expanded*/tools directory and issue the following command (replacing *my_credential_name* with the name you chose for this credential in step 5d):

```
btmcli credStoreTool -createCred my_credential_name -credType is
```

The CLI then prompts you for the Trusted Assertion Issuer and the Trusted Assertion Secret. The latter is masked with asterisks as you type.

e. Input the values for the Trusted Assertion Issuer and the Trusted Assertion Secret (and keep the command shell/window open for the next step).

Use the values you chose in step 3.

Note: If you prefer not to use the interactive mode, you can provide the issuer name and assertion secret by appending **-credValue** *my_issuer*:*my_secret* to the command line. However, using interactive mode is more secure because of the masking.

The credential for your assertion secret has now been created and added to the Oracle Wallet credential store.

9. In the same command shell/window, use the CLI to create a credential for your encryption key by issuing the following command (replacing *my_credential_name* with the name you chose for this credential in step 5e):

```
btmcli credStoreTool -createCred my_credential_name -credType bin -genKey
AES:128
```

This command creates a credential with the given name, generates an AES, 128-bit, random encryption key, and adds it to the Oracle Wallet credential store.

10. Retrieve the encryption key for use in configuring the other machines in your system by issuing the following command (replacing *my_credential_name* with the name of your credential):

```
btmcli credStoreTool -getCred my_credential_name -credType bin -showSecret
```

This command returns the string value of your encryption key. You will need to copy this string to the other machines in your system as you configure them.

11. Repeat the pertinent steps of this procedure for each central server, monitor, and observer for which you want to configure the assertion secret and encryption key using Oracle Wallet, but on repetition make the following changes:

- Skip steps 2 and 3 (you will use the same credential names and issuer name and issuer assertion secret values that you used on the first machine).
- Issue the following CLI command instead of the commands shown in steps 9 and 10 (replacing *my_credential_name* with the name of your credential, and *my_encryption_key_string* with the string you retrieved in step 10):

  ```
  btmcli credStoreTool -createCred my_credential_name -credType bin
  -credValue my_encryption_key_string
  ```

  This command creates a credential with the given name and encryption key, and adds it to the Oracle Wallet credential store.

4.3.2 Configuring Security Using Java System Properties

This section pertains to the central servers and monitors, and to observers deployed in Java environments. It explains how to configure the assertion secret and encryption key using Java system properties.

4.3.2.1 Configuring the Assertion Secret Using Java System Properties

You must configure the assertion secret on each application server that hosts a Business Transaction Management component. To configure the assertion secret, you must set two Java system properties and, optionally, a third Java system property. You set these system properties in the server hosting the Business Transaction Management component.

This example shows how to set the first required system property:

```
-Dcom.amberpoint.SimpleIdentityAssertion.TrustedIssuerOverriderClassName=com.amber
point.wsclient.TrustedIssuerOverriderByExtProp
```

This example shows how to set the second required system property:

```
-Dcom.amberpoint.SimpleIdentityAssertion.TrustedIssuerSecretOverride=MySecret
```

where *MySecret* is your own secret string.

By default, the name of the issuer of the security assertion is AmberPoint. You can override the default issuer's name by creating a third system property. This example shows how to set this property:

```
-Dcom.amberpoint.SimpleIdentityAssertion.TrustedIssuerNameOverride=MyIssuerName
```

where *MyIssuerName* is the name of the issuer of the security assertion.

4.3.2.2 Configuring the Encryption Key Using Java System Properties

You must configure the encryption key on each application server that hosts a Business Transaction Management component. To configure the encryption key, create a Java system property named com.amberpoint.security.encryption.aes.defaultKey in the server and set its value to your encryption key, for example:

```
-Dcom.amberpoint.security.encryption.aes.defaultKey=MyEncryptionKey
```

where *MyEncryptionKey* is a base 64-encoded, AES, 128-bit key.

After generating your encryption key, you can copy and paste it in order to set the value of your `com.amberpoint.security.encryption.aes.defaultKey` property. If your key includes special characters, you should enclose it in double quotes, for example:

```
-Dcom.amberpoint.security.encryption.aes.defaultKey="oylJKoTGXTHasOYwtjwA7g=="
```

4.3.3 Configuring Security for Observers Deployed in .NET Environments

This section pertains only to observers deployed in .NET environments. It explains how to configure the assertion secret and encryption key using environment variables. You must perform these configurations for each .NET environment that hosts an observer.

4.3.3.1 Configuring the Assertion Secret

To configure the assertion secret, you must set two environment variables and, optionally, a third environment variable as follows:

1. Create an environment variable named:

   `com.amberpoint.SimpleIdentityAssertion.TrustedIssuerOverriderClassName`

   and set its value to:

   `AmberPoint.Wsutility.Wsclient.TrustedIssuerOverriderByExtProp`

2. Create an environment variable named:

   `com.amberpoint.SimpleIdentityAssertion.TrustedIssuerSecretOverride`

   and set its value to your assertion secret string.

3. *Optional* – By default, the name of the issuer of the security assertion is AmberPoint. You can override the default issuer's name by creating an environment variable named:

   `com.amberpoint.SimpleIdentityAssertion.TrustedIssuerNameOverride`

   and setting its value to the issuer name.

4.3.3.2 Configuring the Encryption Key

Configure the encryption key as follows:

1. Generate a base 64-encoded, AES, 128-bit encryption key.
2. On the machine hosting the observer, create an environment variable named:

   `com.amberpoint.security.encryption.aes.defaultKey`

   and set its value to your generated encryption key.

4.4 Setting up a Secure Socket (SSL) for Observation Messages

Observers send observation messages to their associated monitor by way of a socket connection. You can secure your observation messages by setting up a secure socket (SSL) for this communication channel.

Setting up a secure socket between a monitor and its associated observers requires a properly configured key store that is accessible to the monitor and a properly configured trust store that is accessible to the observers. Business Transaction Management provides built-in, preconfigured key and trust stores so as to minimize the setup required to enable SSL (a preconfigured server certificate is also provided for .NET-based observers). For demonstration purposes, and perhaps for development environments, using these built-in security stores might be adequate. But, in production environments, you should tighten security by providing your own security stores.

The following procedure describes how to create and deploy your own key store and trust store containing a self-signed certificate:

1. Prepare a key store and trust store containing an appropriate certificate and private key:
   a. Locate the keytool application in your JDK.
   b. Generate a key store containing a certificate-private key pair.

      For example, the following keytool command generates a keystore containing a certificate-private key pair in a file named "mykeystore.ks". The alias for the certificate-private key pair is "myks", the common name is "MyMonitor", the

algorithm is RSA, the password for accessing the certificate and private key is "mycertandprivkeypass", and the password for accessing the key store is "mykeystorepass" (this command creates the file if it does not yet exist):

```
keytool -genkey -alias myks -dname "CN=MyMonitor" -keyalg RSA -keypass
mycertandprivkeypass -storepass mykeystorepass -keystore mykeystore.ks
```

c. Export the certificate from your key store to an external file.

For example, the following keytool command exports the certificate created in the previous example to a file named "mycertificate.cer":

```
-export -alias myks -storepass mykeystorepass -file mycertificate.cer
-keystore mykeystore.ks
```

d. Import the certificate (without the associated private key) into the trust store.

For example, the following keytool command imports the certificate created in the previous example into the trust store contained in the file "mytruststore.ks" (this command creates the trust store file if it does not yet exist).

```
-import -v -trustcacerts -alias myks -file mycertificate.cer -keystore
mytruststore.ks -keypass mycertandprivkeypass -storepass mykeystorepass
```

2. Deploy the key store to the machines that will host the monitors.

Copy the key store (for example, mykeystore.ks) either to the machines that will host the monitors or to a location accessible to the monitors by way of HTTP GET.

3. Deploy the trust store to the machines that will host your Java-based observers (ignore this step if you don't have Java-based observers).

Perform this task in one of these two ways:

- Copy the trust store (for example, mytruststore.ks) either to the machine that will host the observers or to a location accessible to the observers by way of HTTP GET.
- Copy the trust store (for example, mytruststore.ks) either to the machines that will host the monitors or to a location accessible to the monitors by way of HTTP GET. By using this second method, you can configure the monitor to automatically dispatch the trust store to the observers by enabling the **Auto Dispatch Trust Store to Java Observers** field when you configure the Observer Communication policy (see Section 8.5).

4. Deploy the certificate to the machines that will host your .NET-based observers (ignore this step if you don't have .NET-based observers).

Using the Windows Certificate Import Wizard, import the certificate (for example, mycertificate.cer) to the Trusted Root Certification Authorities folder of the certificate store on the machines that will host the .NET-based observers.

If you decide to use the built-in security stores rather than your own, and you are going to use Java-based observers only, you do not have to perform any of the preliminary setup described in the previous procedure.

If you are going to use the built-in security stores with .NET-based observers, you need only to deploy the preconfigured certificate to the machines that will host the .NET observers (as described in step 4 of the previous procedure). You can find the preconfigured certificate at nanoagent\config\ssl\server.cer in the ZIP files containing the .NET-based observers.

5

Prerequisite Requirements and Preliminary Setup

This chapter describes prerequisite requirements and preliminary setup that you must satisfy before you begin installing Business Transaction Management. Business Transaction Management is composed of several types of components. Some of the following requirements pertain to all Business Transaction Management components while other requirements pertain to a subset of components.

5.1 Web Browser Requirements

The requirements in this section pertain to the web browser that you use to perform the initial configuration of Business Transaction Management and to access the Business Transaction Management Console.

- The web browser requires the Adobe Flash plugin version 10.1 or higher.
- If you are using Internet Explorer as your web browser, you must configure it to allow the Flash player's Active X control. Consult your Internet Explorer documentation for instructions on enabling this setting.

5.2 Setting up your WebLogic Environment

> **Note:** If you are using WebLogic Node Manager, you will use the WebLogic Administrative Console to adjust settings rather than edit scripts.

1. Ensure that the appropriate database driver for your Oracle RDBMS is in the WebLogic server's classpath for each central server and monitor.

   Drivers are supplied in the jdbc directory of BTM_Servers_*.zip. Use ojdbc5.jar with JDK 1.5 and ojdbc6.jar with JDK 1.6.

2. Ensure that each WebLogic server in which you install a Business Transaction Management central server or monitor is uniquely identified so that the central servers and monitors can reliably connect to each other. You can perform this task in either of the following ways:

   - Ensure that any and all IP addresses assigned to the host machine uniquely identify that machine.
   - Ensure that the WebLogic server's **Listen Address** property is set to a hostname or IP address that uniquely identifies the server.

To set this property, navigate in the WebLogic Administration Console to **Environment > Servers**, then click your server and display the **Configuration/General** tab.

Note: If the machine has an IP address that is shared with another machine on the network, or the machine has multiple IP addresses that are treated as separate virtual machines, you must set the domain's **Listen Address** property as described above.

---

3. For central servers and monitors, ensure that memory allocation for your WebLogic server is set appropriately. Two methods of setting the Java options that control memory allocation are described below. Use the method that is appropriate for how you start and stop your managed servers:

   If you use the Node Manager to remotely start and stop your managed servers, set the Java memory options for your server using this method:

   a. Open the WebLogic Administration Console.

   b. Select your managed server.

   c. Select the **Configuration > Server Start** tab.

   d. Enter the following Java options in the **Arguments** field, making sure to separate all entries in the field with a space:

      ```
      -Xms256m -Xmx768m -XX:MaxPermSize=256m
      ```

      These are the minimum recommended settings. Depending on your environment, you might have to set them higher.

   If you start and stop your managed servers by executing local script files, set the Java memory options for your server using this method:

   a. Open the setDomainEnv script file for your domain in a text editor.

      On Windows systems, open setDomainEnv.cmd; on UNIX-like systems, open setDomainEnv.sh. These script files are located in the user_projects\domains*domain_name*\bin directory of your WebLogic installation.

   b. Locate the following settings and ensure that they are set to at least the values indicated (depending on your environment, you might have to set them higher.):

      ```
      MEM_ARGS=-Xms256m -Xmx768m

      -XX:MaxPermSize=256m
      ```

      There are several of these entries; set them all.

      Depending on your version of WebLogic, you might also see separate 32-bit and 64-bit settings like these:

      ```
      set WLS_MEM_ARGS_64BIT=-Xms256m -Xmx512m
      set WLS_MEM_ARGS_32BIT=-Xms256m -Xmx512m
      set MEM_MAX_PERM_SIZE_64BIT=-XX:MaxPermSize=256m
      set MEM_MAX_PERM_SIZE_32BIT=-XX:MaxPermSize=128m
      ```

      In this case, set them all to at least the minimum recommended settings.

4. Set up an administrative user on the WebLogic server in which you will install the Main server (btmMain.ear).

Business Transaction Management maps roles defined in WebLogic to its own application roles. See section 6.4.2, "Mapping WebLogic Users to Business Transaction Management Roles" for more information.

5.3 Setting up your WebSphere Environment

1. Ensure that the appropriate database driver for your Oracle RDBMS is in your WebSphere server's classpath for each central server and monitor.

   If you edit the classpath while the server is running, be sure to restart WebSphere before configuring Business Transaction Management.

   Drivers are supplied in the jdbc directory of BTM_Servers_*.zip. Use ojdbc5.jar with JDK 1.5 and ojdbc6.jar with JDK 1.6.

2. Use the WebSphere Administrative Console to specify Initial Heap Size of no less than 256, and Maximum Heap Size of no less than 1024.

   Perform this task on each WebSphere instance in which you install Business Transaction Management components.

3. Enable global security on WAS, including application and Java 2 security.

   Perform this task on each WebSphere instance in which you install Business Transaction Management components.

4. Set up an administrative user on the WebSphere server on which you will install the Main server (btmMain.ear).

   Business Transaction Management maps roles defined in WebSphere to its own application roles. See section 6.4.3, "Mapping WebSphere Users to Business Transaction Management Roles" for more information.

5.4 Setting up Business Transaction Management Databases

Several Business Transaction Management system services use a database to store persistent information and log messages. You must use an Oracle 10g or 11g RDBMS for these databases, and it must be configured to support SQL authentication mode and TCP/IP connections.

Before you configure Business Transaction Management, create the following database users (these are suggested names):

- sphereDB
- measurementDB
- transactionDB
- messageLogDB

You can create the database users in the same Oracle instance or in separate instances. You must create these users before starting configuration of Business Transaction Management. When you configure Business Transaction Management (see section 6.5, "Initial Configuration of Business Transaction Management"), the system will automatically create the appropriate database tables.

If you prefer to create the schemas manually for the first three of these databases (sphereDB, measurementDB, and transactionDB), your DBA can create them beforehand (see the following note). If you intend to let the system automatically create these tables and indexes, the database users must have create table, create index, create view, and analyze privileges. You cannot create the fourth schema

(messageLogDB) beforehand because the system must be able to create and drop tables dynamically in response to changes in your monitored applications. For this database, the user must have create table, drop table, create index, create view, and analyze privileges. (Note: It is not sufficient to assign the privileges to the roles associated with the user. You must explicitly assign the privileges to the user.)

Note: Your DBA can manually create the tables and indexes for the sphereDB, measurementDB, and transactionDB databases using the `datastoreUtil` utility. This utility generates the appropriate schema definitions. Documentation on using this utility to generate the schema definitions is provided in Chapter 17, "The datastoreUtil Utility."

6

Installing and Configuring the Central Servers

This chapter describes how to install and perform the initial configuration of the Business Transaction Management central servers on the following application servers:

- Oracle WebLogic 10.3.2 through 10.3.5
- IBM WebSphere 7.0

6.1 Overview of Installing and Configuring the Central Servers

1. *Optional* – Configure security for the central servers. You can also perform the security configuration for the monitors and observers at this time, if you wish (see Chapter 4, "Configuring Security").
2. Ensure that all prerequisite requirements and setup described in Chapter 5 are satisfied, including:
   - section 5.2, "Setting up your WebLogic Environment" or section 5.3, "Setting up your WebSphere Environment"
   - section 5.4, "Setting up Business Transaction Management Databases"
3. *Optional* – Configure the persistent storage directories (see Section 6.2).
4. Deploy the central servers (see Section 6.3).
5. Review the default mapping of users to Business Transaction Management application roles and make adjustments if necessary (see Section 6.4).
6. Perform the initial configuration of Business Transaction Management (see Section 6.5).

6.2 Configuring Persistent Storage Directories

At initial startup, Business Transaction Management creates a set of persistent storage directories to collect system output log entries and store user preferences for the system deployments. By default, the persistent storage directories are created within the application server's installation directory at the following location:

- On WebLogic servers:

  WL_install_dir/user_projects/domains/*domain_name*/servers/*server_name*/btmstorage/*
- On WebSphere servers:

  WAS_install_dir/profiles/*profile_name*/btmstorage/*node_name*/*server_name*/*

Your company's in-house procedures and rules for persistent storage might require you to place the persistent storage directories in a different location. In such a case, you can reconfigure the location of the persistent storage directories.

An installed Business Transaction Management system is composed of a set of deployments (EAR files), which are themselves composed of subdeployments (WAR files). Each subdeployment has an associated persistent storage directory of the same name, minus the ".war". The following table lists the names of the deployments, subdeployments, and persistent storage directories.

Table 6–1 Business Transaction Management deployments, subdeployments, and persistent storage directories

Deployments (EARs)	Subdeployments (WARs)	Persistent storage directories
btmMain	btmui	btmui
	btmcentral	btmcentral
	btmcontainer	btmcontainer
btmPerformanceServer	btmcontainer	btmcontainer
	btmperformance	btmperformance
btmTransactionServer	btmcontainer	btmcontainer
	btmtransaction	btmtransaction
btmMonitor	btmmonitor	btmmonitor

6.2.1 Reconfiguring the Location of Persistent Storage Directories

1. Create the persistent storage directories in your file system, in the location that you want them.

   You must name the directories using the default names, as listed in Table 6–1. Leave the directories empty.

2. Locate the distribution archive for the Business Transaction Management central servers and unzip it into a directory.

   The distribution archive is named BTM_Servers*.zip, where * represents the version number. This archive also contains the Business Transaction Management monitor.

3. Modify the persistent storage directory location as specified in each subdeployment's web.xml file:

   a. Locate and expand the WAR file for the deployment whose storage directory location you want to change.

   b. Within the expanded WAR file, open the WEB-INF/web.xml file in a text or XML editor.

   c. Set the new location for the storage directory by editing the value of the storageDirectory parameter:

      As shown in the following example, the default value of the parameter is **AmberPointDefault**:

```
<!-- PERSISTENT STORAGE DIRECTORY
To set the persistent storage area to some value, change the value of
param-value to some EXISTING directory where you want things stored.
-->

<context-param>
```

```
<param-name>com.amberpoint.storageDirectory</param-name>
<param-value>AmberPointDefault</param-value>
</context-param>
```

Delete the value **AmberPointDefault** and replace it with the path to the storage directory you created, as shown in the following examples:

On Windows systems – If you want the persistent storage directory for btmcentral to be C:\btm_data\btmcentral, change the default entry within your btmcentral web.xml file to the following:

```
<context-param>
  <param-name>com.amberpoint.storageDirectory</param-name>
  <param-value>C:\btm_data\btmcentral</param-value>
</context-param>
```

On Unix-like systems – If you want the persistent storage directory for btmcentral to be opt/webserviceapplogs/btm_data/btmcentral, change the default entry within your btmcentral web.xml file to the following:

```
<context-param>
  <param-name>com.amberpoint.storageDirectory</param-name>
  <param-value>/opt/webserviceapplogs/btm_data/btmcentral</param-value>
</context-param>
```

d. Repeat this step for each persistent storage directory whose location you want to reconfigure.

Note: If you are going to reconfigure the location of persistent storage directories for your monitors (as well as for your central servers), this might be a convenient time to do that.

---

4. Repackage your WAR and EAR files.

You should document the location of your persistent storage directories because when you upgrade or reinstall the central servers, you will need to once again define the location of your persistent storage directories for these deployments.

6.3 Deploying the Central Servers

Note: These instructions assume that if you are installing onto WebLogic servers, you are using managed instances of WebLogic.

---

1. Locate the distribution archive for the Business Transaction Management central servers and unzip it into a directory (henceforth referred to as *Install_Dir*).

The distribution archive is named BTM_Servers*.zip, where * represents the version number. This archive also contains the Business Transaction Management monitor.

Note: If you configured the persistent storage directories, as described in Section 6.2, you will have already completed this step.

---

2. *Optional security step for UNIX-like operating systems* – If you want to set file permissions on the files that make up the distribution to the most restrictive level that still maintains functionality, complete this step:

   a. Locate setPermissions.sh at the top level of *Install_Dir*.

      This script contains commands for setting file permissions of all regular files to Owner – read/delete; all directories to Owner – read/execute/delete; and all scripts to Owner – read/execute/delete.

      Note: These permission levels are extremely restrictive. For example, only the owner can read the files.

   b. On a command line, at the top level of *Install_Dir*, run this command:

      ```
      source setPermissions.sh
      ```

      This command runs the commands in the script file and sets permissions for all files and directories in the expanded archive.

3. Using your application server's administration console, deploy each of the following applications (located in *Install_Dir*\archives):

   - btmMain
   - btmPerformanceServer
   - btmTransactionServer

   Deploy only one instance of each of these servers, and for performance considerations you should deploy each to a separate application server instance. You must not deploy any of these servers to an application server instance that hosts services or components you intend to monitor. For more information about the central server applications, see Chapter 1.1, "Architecture."

4. *For WebLogic* – Start all managed servers where you deployed Business Transaction Management components.

5. Start the deployments.

6. If you reconfigured the locations of your persistent storage directories (as described in Section 6.2), confirm that system output log entries have been written in the new locations.

7. If this is a reinstallation of the central servers, notify all Business Transaction Management users to flush their web browser caches.

   The Management Console contains a number of Adobe Flash widgets. Web browsers normally cache these widgets and will continue to use the older cached widgets until you either flush the cache or restart your web browser.

6.4 Mapping Users to Business Transaction Management Application Roles

This section describes Business Transaction Management application roles and the default mappings of Business Transaction Management users to these roles. If necessary, you can reconfigure these mappings using your system administration facilities.

Business Transaction Management applications (the central servers and monitors) rely on the application server in which they are deployed for the authentication of users and the association of application roles with users.

By default, authentication is enabled for the Management Console. If you want to disable authentication, use whatever tool or procedure is appropriate for the application server you are using. If you disable authentication, users of the Management Console must still log in. However, they can log in using any user name and are not required to provide a password. Note that all UI personalizations, such as edits to the Navigator, filters, and column sets are stored as preferences and associated with the user name.

6.4.1 Business Transaction Management Application Roles

The Business Transaction Management Console uses roles to authorize access to various features of the user interface. In order to log into the Management Console, you must use credentials that are mapped to at least one of theses BTM application user roles: btmAdmin, btmUser, or btmObserver. In order to perform the initial configuration of Business Transaction Management, explained in Section 6.5, you must log in as a user with the btmAdmin role.

6.4.1.1 Primary Roles

Each Business Transaction Management user must be assigned at least one primary role. The primary roles are:

btmAdmin – Users with this role are granted all privileges. These users can use all tools and facilities provided by the Management Console, including the ability to view and create sensitive properties and to view all message content.

btmUser – Users with this role have most privileges needed to configure basic monitoring. For example, they can configure monitors; create, edit, and delete policies (does not include system policies); register services; set registry attributes on services and endpoints; and create and edit transactions and conditions. They also have all the privileges of btmObserver. This role does not grant the privilege to modify the Business Transaction Management environment, access message content, or view or edit sensitive properties.

btmObserver – Users with this role have privileges to use most of the basic monitoring facilities. They can view summary, dependency, and administrative information about the monitoring system, but are not allowed to configure any of the policies or settings related to it. They can also view transactions and conditions, but are not allowed to create or edit them. This role does not allow users to modify the Business Transaction Management environment, access message content, or view or edit sensitive properties.

Note: All navigation and views in the Management Console are available to all primary roles. However, some roles cannot access certain menus and menu items and the tools associated with them.

6.4.1.2 Auxiliary Role

In addition to the primary roles, Business Transaction Management defines an auxiliary role. The auxiliary role provides additional privileges that you might want to assign certain users. For example, you might want to let a user access message content but not want to give that user full administrative privileges. You could do this by

assigning the user a primary role of btmUser and an auxiliary role of btmInspector. The auxiliary role is:

btmInspector – Users with this role can view message content and view and create properties, including sensitive properties.

> **Note:** The btmAdmin role has all of the privileges of btmInspector.

6.4.2 Mapping WebLogic Users to Business Transaction Management Roles

In a WebLogic server, the role of Business Transaction Management administrator (btmAdmin) is automatically mapped to the Administrators group defined in your WebLogic server. The role of Business Transaction Management user (btmUser) is mapped to the groups Operators and Monitors. The role of Business Transaction Management observer (btmObserver) is mapped to the group Everyone, granting all authenticated users observer privileges. The following table lists the Business Transaction Management application roles and their default mappings to application server groups:

Table 6–2 Role Mapping in WebLogic

Business Transaction Management Application Role	Application Server Group
btmAdmin	Administrators
btmUser	Operators, Monitors
btmObserver	Everyone
btmInspector	btmInspectors

> **Note:** By default, the role btmInspector is mapped to a group named btmInspectors. Because this group does not exist, by default, the application server administrator must create the group and assign it to the appropriate users.

You can modify these default mappings in the WebLogic deployment descriptor file (weblogic.xml). If you change the mappings, you should change them for all Business Transaction Management applications. The names of the applications (EAR files) begin with the string “btm”. Refer to your WebLogic documentation for further information about editing the default mappings.

6.4.3 Mapping WebSphere Users to Business Transaction Management Roles

In a WebSphere server, the application role of Business Transaction Management administrator (btmAdmin) is automatically mapped to the Administrators group on Windows systems, or the adm, sys, and bin groups on Unix-like systems. The roles of Business Transaction Management user (btmUser) and observer (btmObserver) are mapped to all authenticated users on your system. The following table lists the Business Transaction Management application roles and their default mappings to your operating system groups:

Table 6–3 Role Mapping in WebSphere

Business Transaction Management Application Role	Operating System Group
btmAdmin	Administrators (Windows); adm, sys, and bin (Unix-like systems)
btmUser, btmObserver	all authenticated users
btmInspector	btmInspectors

Note: The role btmInspector is, by default, mapped to a group named btmInspectors, but the application server administrator must create a group named btmInspectors and assign it to the appropriate users.

These initial mappings assume that WebSphere security is configured to use Local OS Registry. If you are using a different security setup, you might have to remap users/groups based on your authentication domain settings.

You can change these mappings in the WebSphere Administrative Console, on the "Mapping Users to Roles" page. If you change the mappings, you should change them for all Business Transaction Management applications. The names of the applications (EAR files) begin with the string "btm". Refer to your WebSphere documentation for further information about editing the default mappings.

6.5 Initial Configuration of Business Transaction Management

Before you can access the facilities of Business Transaction Management, you must perform an initial configuration of the central servers. If this is the first time you have installed Business Transaction Management, you should use the browser-based Configuration wizard, as described below. However, for subsequent installations, you might want to use the Command Line Interface (CLI) to perform a scripted configuration of Business Transaction Management. For more information, see "Scripted Configuration of Oracle Business Transaction Management" on page 16-1.

1. Ensure that the central servers are running.
2. Open the Management Console by pointing a web browser at the server that hosts the btmMain deployment.

   Use a URL in the form of:

   `http://host_name:port_number/btmui`

   The Management Console login page opens.
3. Log in as a user that is in the btmAdmin role (see section 6.4, "Mapping Users to Business Transaction Management Application Roles" for information about this role).
4. The introductory page of the Configuration wizard opens.

   Click **Next**, and the **Database Type** page opens.
5. Choose **External Database** (the embedded database is not supported for production systems).

   When finished, click **Next**, and the **External Database Configuration** page opens.

6. Provide the connection string to your Oracle instance and the user names and passwords for the database users you created in section 5.4, "Setting up Business Transaction Management Databases".

   If you created the users on separate Oracle instances, first select **Custom** so that you can provide multiple connection strings. If you are using a single connection string for all databases, each database requires a distinct user name.

   When finished, click **Next**, and the **Sphere URL** page opens.

7. Ensure that the URL for the sphere is correct—in most cases the default should be correct (unless you are running in an HTTPS environment).

   Note: If you click the Test Sphere URL link and the sphere URL is correct, the system displays a Sphere Status page that indicates that the sphere is not initialized. This is expected and the sphere service will be initialized at the completion of the configuration. If the URL is incorrect, the browser displays a "page not found" error.

   When finished, click **Next**, and the **Local Container Setup** page opens.

8. This page lets you optionally specify DNS aliases for the network node on which you installed btmMain.

   The use of aliases helps avoid the creation of duplicate endpoints when users register services manually or when the system observes message traffic at an alias address. Use a comma to separate multiple addresses.

   When finished, click **Next**, and the **Performance Monitoring Components** page opens.

9. Enter the URL at which you deployed btmPerformanceServer in the form of http://*HostName:Port*/btmcontainer/container/.

   This URL must end with "/btmcontainer/container/". Make sure that you deploy apPerformanceServer before you exit this screen.

   When finished, click **Next**, and the **Local Container Setup** page opens again (unless you deployed btmPerformanceServer to the same machine as btmMain).

10. This page lets you optionally specify DNS aliases for the network node on which you installed btmPerformanceServer.

    When finished, click **Next**, and the **Transaction Monitoring Components** page opens.

11. Enter the URL at which you deployed btmTransactionServer in the form http://*HostName:Port*/btmcontainer/container/.

    This URL must end with "/btmcontainer/container/". Make sure that you deploy btmTransactionServer before you exit this screen.

    When finished, click **Next**, and the **Local Container Setup** page opens again (unless you deployed btmTransactionServer to the same machine as btmPerformanceServer or btmMain).

12. This page lets you specify DNS aliases for the network node on which you installed btmTransactionServer.

    When finished, click **Next**, and the **Summary of Configuration** page opens.

13. Ensure that the configuration information is correct and click **Finish**.

Business Transaction Management validates the information you supplied during configuration, and if all the information is valid, configuration completes successfully and the Business Transaction Management Console appears.

If any configuration information cannot be validated, Business Transaction Management displays an error message as well as a check box with the following text: “Ignore these errors and proceed with a potentially incompletely configured system.” In general you should attempt to correct any configuration errors. However, you have the option of proceeding with Business Transaction Management partially configured by enabling the checkbox and clicking **Finish**. If you choose to access the partially configured system, you will need to correct the configuration errors before you can successfully use the product. For example, if your database connection information was not accurate, you will need to re-configure it from within the Management Console. For information about configuring databases after initial configuration, refer to the Business Transaction Management online help.

7

Starting and Shutting Down Business Transaction Management

This chapter explains how to:

- start and shutdown Business Transaction Management components
- shutdown and restart members of a monitor group without losing observations
- log in to and out of the Management Console
- access the online help

7.1 Starting Business Transaction Management Components

Business Transaction Management components start automatically when the application server in which they are installed starts. Because the central servers and monitor are deployed applications, you can also start them manually using your application server's management facilities. Observers, however, are installed into your business applications and start along with those applications.

If all of the components of your system are shut down, the best order in which to start them is from the center out, that is:

1. Central servers
2. Monitors
3. Observers

However, you can start them in any order and achieve a fully functioning system after all the components have recognized each other and configured themselves.

Note: If the observers, and therefore your business applications, are running, but the monitors are not, the observers' outgoing queue will eventually fill up. At that point, either observation messages will be dropped from the queue or your business applications will stall. The default behavior is for observation messages to be dropped from the queue, which ensures that the performance of your business applications is not degraded. This behavior is configurable by way of the Observer Communication policy. However, this is an advanced option and you should leave it at its default setting unless you are instructed by the Oracle support team to edit it. If you require the ability to shut down monitors without the risk of dropping observations, you should replicate your monitors as described in Chapter 8, "Installing Monitors."

7.2 Shutting Down Business Transaction Management Components

Business Transaction Management components shut down when the application server in which they are installed shuts down. Because the central servers and monitor are deployed applications, you can also shut them down them using your application server's management facilities. Observers, however, are installed into your business applications and shut down along with those applications.

If you want to shut down all of the components of your system, the best order to shut them down is from the outside to the center, that is:

1. Observers
2. Monitors
3. Central servers

However, you can shut down and then restart individual components without harming the system. After all the components have recognized each other and reconfigured themselves, you should once again have a fully functioning system.

The note in Section 7.1 that describes a possible loss of observations, however, also applies in this case.

7.3 Shutting Down and Restarting Monitor Group Members

This section explains how to shut down individual monitor replicates in a monitor group without losing in-flight data. It also explains how to properly restart the monitor replicates.

7.3.1 Shutting Down Monitor Group Members

1. Disable the monitor's associated pool member in the load balancer's socket virtual server.
2. Wait for all in-flight data to clear from the monitor.

   You can check the status of in-flight data as follows:

   a. Display the monitor's status page in either of the following ways:

      In the Business Transaction Management Console's Navigator, choose **Administration > Monitors**, select the monitor in the main area, and click the **Status** tab.

Or, access the following URL in a web browser, where *host:port* are the host name and port number where the monitor is running:

```
http://host:port/btmmonitor/agent/agent?status
```

b. Scroll to the **Nano Observation Queue** section.

c. All in-flight data has cleared from the monitor if both of the following equations are true:

```
perf.SimpleMeasurementsProcessed = perf.SimpleMeasurementsQueued

Perf.RequestsQueued = Perf.ResponsesQueued + Perf.QueueSize
```

3. Use your application server's administration tools to shut down the monitor deployment.

7.3.2 Restarting Monitor Group Members

1. Use your application server's administration tools to start the monitor deployment.
2. Enable the monitor's associated pool member in the load balancer's socket virtual server.

7.4 Logging in to the Management Console

You administer and access the facilities of Business Transaction Management using a web-based interface called the Business Transaction Management Console (sometimes referred to in this document simply as the Management Console).

In order to access the Management Console, the Main server (btmMain.ear) must be running. To log in to the Management Console, open a URL of the following form in a web browser:

```
http://hostname:port/btmui
```

Note: The web browser must meet the requirements listed in Section 5.1, "Web Browser Requirements."

Replace *hostname:port* with the name of the machine on which you installed the Main server, and the port number on which it is accessible.

The web browser will display a log-in page. Log in using the appropriate credentials, which are described in section 6.4, "Mapping Users to Business Transaction Management Application Roles".

7.5 Logging out of the Management Console

To log out of the Management Console, click the **Log out** link in the upper-right corner of the page.

7.6 Online Help

You can access Business Transaction Management online help by clicking the **Help** menu in the Management Console and choosing **Help**, or by clicking the **Help** button

from within a tool dialog box. Once the online help opens, use the navigation facilities on the left to locate the appropriate help topic.

8

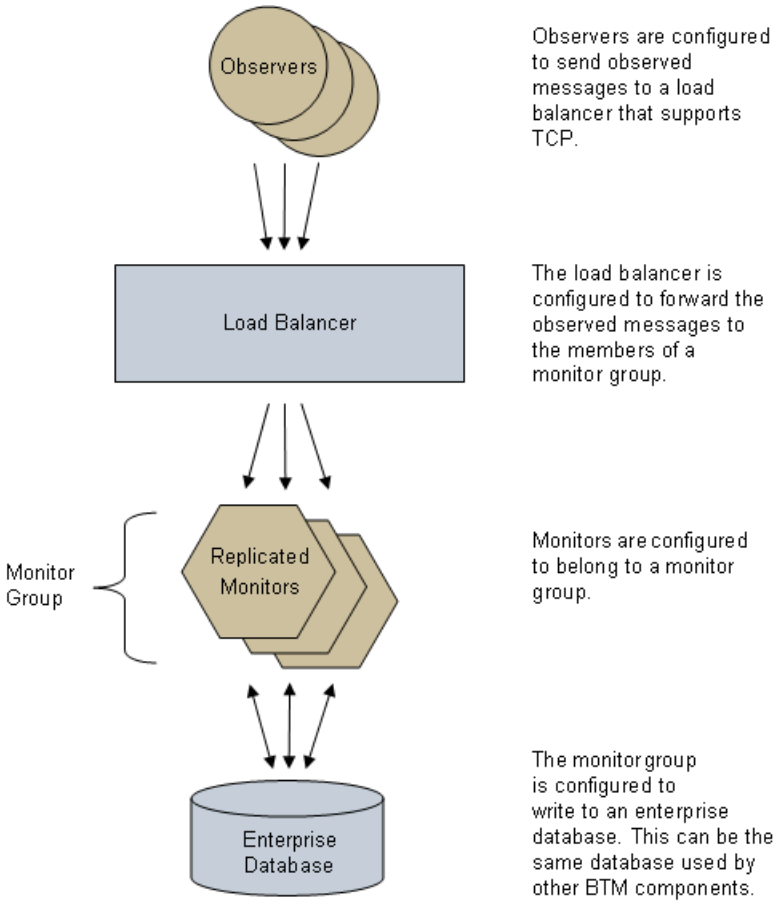
Installing Monitors

This chapter describes how to install Business Transaction Management monitors. Instructions for installing both singleton and replicated monitors is provided.

You can deploy as many monitors as required for your monitoring needs. You can deploy monitors as singleton instances or as groups of replicated instances behind a load balancer. Such a group of replicated monitors is referred to as a *monitor group*. You can also deploy a mixture of singleton monitors and monitor groups.

Using monitor groups gives you the ability to scale your monitoring system and make it more fault tolerant. Monitor groups require a third-party load balancer that supports TCP for routing messages from your observers to your monitors. Monitor groups also require access to an Oracle 10g or 11g database, which is used to share data between the replicated instances of the monitor. This database can be the same database used by other Business Transaction Management components. Although the replicated instances of any one monitor group can be widely dispersed across your network, they must all have access to this same database. A generalized deployment of a monitor group is shown below.

Figure 8–1 Generalized deployment of a monitor group



8.1 Overview of Installing Monitors

This section provides an overview of the tasks you must perform to install monitors. Subsequent sections provide details on how to perform these tasks. To ensure a properly configured system, you must perform the steps in the order shown.

1. Configure security if needed (see Chapter 4, "Configuring Security").

> **Note:** If you configured the assertion secret and encryption key on your central servers, then you must also configure your monitors to use the same assertion secret and encryption key. If, on the other hand, you are using the default security configuration on your central servers, then you must use the default security configuration on your monitors (see Section 4.3, "Configuring the Business Transaction Management Assertion Secret and Encryption Key").

2. Ensure that all prerequisite requirements and setup described in Chapter 5 are satisfied, including:
   - Section 5.2, "Setting up your WebLogic Environment" or Section 5.3, "Setting up your WebSphere Environment"
   - Section 5.4, "Setting up Business Transaction Management Databases"

3. *Optional* – Configure the persistent storage directories for the monitors. Use the same procedure as described for the central servers in Section 6.2.
4. Deploy and register the monitors (see Section 8.2).
   a. Ensure that the central servers are installed and running.
   b. Deploy monitors to the target application servers using your application server's deployment tools.
   c. Register your monitors using the registerMonitor CLI command.
5. *For replicated monitors only* – Set up a monitor group (see Section 8.3).
   a. Create a monitor group.
   b. Assign monitors to the monitor group.
6. *For replicated monitors only* – Configure your load balancer (see Section 8.4).
7. Apply an Observer Communication Policy (see Section 8.5).

8.2 Deploying and Registering Monitors

This section pertains to both singleton and replicated monitors. You must perform this task for all installation scenarios.

1. Ensure that the Business Transaction Management central services are installed and running.
2. Using your application server's deployment tools, install the btmMonitor.ear deployment package on each application server that will host a monitor.

> **Notes:** Do not deploy more than one monitor per application server. Also, do not deploy the monitor into an application server that is hosting any of the Business Transaction Management central services.
>
> (An application server is sometimes referred to as a container in the Business Transaction Management Console and elsewhere in the documentation.)

3. Start the monitor deployments.
4. Use the registerMonitor CLI command to register each of the monitors that you deployed, for example:

```
btmcli registerMonitor -e http://my_monitor_host:8080/btmmonitor/agent/agent
                       -fn My_BTM_Monitor
                       -s http://localhost:8080/btmcentral/sphere
                       -l my_admin_username:my_admin_password
```

Table 8–1 Flags for the registerMonitor CLI command

Flag	Description
-e | -endpointUrl	Required. Specify the URL of the monitor to register, for example: `http://my_monitor_host:8080/btmmonitor/agent/agent` Replace the host name and port number with appropriate values. This URL always ends with **btmmonitor/agent/agent**.

Table 8–1 (Cont.) Flags for the registerMonitor CLI command

Flag	Description
-fn | -friendlyName	Optional. Specify a friendly name for the endpoint of the monitor.
-s | -sphereUrl	Required unless the AP_SPHERE_URL environment variable is set. Specify the URL of the sphere, for example: `http://mySphereHost:8080/btmcentral/sphere/` Replace the host name and port number with appropriate values. This URL always ends with **btmcentral/sphere/**.
-l | -userLogin	Required unless the AP_USER_LOGIN environment variable is set. Specify the credentials of a user belonging to the **btmadmin** role in the format: *username:password*. You can encrypt passwords using the encryptPassword CLI command, for example: `btmcli encryptPassword -password "myPassword"`

Refer to the Business Transaction Management online help for more information about CLI commands.

5. Verify that all monitors and system services are running properly.

8.3 Setting Up a Monitor Group

This section pertains to replicated monitors, only. You must perform this task if you plan to use replicated monitors.

1. Create a monitor group:
   a. In the Business Transaction Management Console, choose **Admin > Create System Policy > Monitor Agent Group**.
   b. Enter a name for your monitor group (you can optionally provide descriptive information in the **Version** and **Notes** fields).
   c. Specify the connection string for your database.

      If you use the default string, replace the text within the curly braces (and the curly braces themselves) with values appropriate for your database. Each member of the monitor group must have access to this database.
   d. Specify a user name and password for accessing your database.
   e. Click **Apply**.
2. Assign monitors to the monitor group you just created:

Note: If your monitor will not be using the default Observer Communication policy, you must complete Section 8.5, "Applying an Observer Communication Policy" before continuing with this section. One scenario in which this would be necessary is if multiple monitors are running on the same machine, in which case they would have to listen on different socket ports and would therefore require separate Observer Communication policies.

a. In the Business Transaction Management Console's Navigator, choose **Administration > Monitors**.

b. In the main area, select the monitor that you want to assign to a group.

c. Choose **Modify > Edit Profile for Monitor Agent**.

d. Type the name of the monitor group in the **Monitor Group** field.

   This name must match the value of the **Name** field in the monitor group policy you used to register the monitor group.

e. Click **Apply**.

8.4 Configuring Your Load Balancer

This section pertains to replicated monitors, only. You must perform this task if you plan to use replicated monitors.

1. Configure your load balancer to communicate with the observers by defining both an HTTP virtual server and a socket virtual server.

   Your load balancer requires two observer communication channels—one referred to as the *HTTP virtual server*, and the other referred to as the *socket virtual server*:

   - The HTTP virtual server is used for administrative purposes. For example, observers retrieve their configurations from any one of the replicated monitors by way of the HTTP virtual server.
   - The observers transmit observational data to the socket virtual server, and the load balancer distributes this data across the replicated monitors. We refer to the messages containing such data as observation messages.

   Note: The replicated monitors communicate with the Business Transaction Management central servers and database directly—not by way of the load balancer.

2. Take note of the IP address and port of your socket virtual server.

   You will need this information when you configure the Observer Communication policy (see Section 8.5).

3. Configure your load balancer to distribute observation messages across the replicated monitors as follows:

   a. Create a pool for the socket virtual server.

   b. Assign each monitor to the pool by assigning the port on which the monitor will receive observations.

4. Take note of the monitor port number(s) that you assign to the socket virtual server's pool.

   You will need this information when you configure the Observer Communication policy (see Section 8.5).

   Note: Configuring the Observer Communication policy will be easier if all the monitors listen on the same port number.

5. Configure your load balancer to distribute administrative messages across the replicated monitors as follows:

   a. Create a pool for the HTTP virtual server.

   b. Assign each monitor to the pool by assigning the port on which the monitor's application server listens.

6. Ensure that the application servers that host the observers have their AP_NANO_CONFIG_URL Java system property or AmberPoint:NanoConfigUrl Windows key (depending on the platform) set to the URL of the monitor by way of the load balancer's HTTP virtual server.

   For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then the URL of the monitor by way of the HTTP virtual server would be:

   ```
   http://10.147.46.152:5072/btmmonitor/agent/agent/
   ```

   For more information on this topic, see Chapter 9, "Installing Observers Overview".

8.5 Applying an Observer Communication Policy

This section pertains to both singleton and replicated monitors. You must perform this task for all installation scenarios. However, the way in which you complete this task depends on whether you are using singleton or replicated monitors, and whether your monitors all listen on the same port.

The Observer Communication policy sets up communication between observers and a monitor or monitor group. By default, an Observer Communication policy is applied to all monitors in the Business Transaction Management sphere. Depending on the monitor deployment scenario you employ, you might be able to simply edit the default policy, or you might have to apply additional policies.

The following scenarios require only a single Observer Communication policy. In these scenarios, you can simply edit the default policy:

- All of your monitors are singleton, you want to configure them all to receive observations on the same port number, and (if you want to enable SSL) they all use the same private key/certificate and the security stores are in the same relative locations for all monitors and associated observers.
- All of your monitors are replicated behind a single load balancer, you want to configure them all to receive observations on the same port number, and (if you want to enable SSL) they all use the same private key/certificate and the security stores are in the same relative locations for all monitors and associated observers.

The following scenarios require you to apply additional Observer Communication policies:

- You want to configure your monitors to receive observations on different port numbers. In this case, you must apply a separate policy for each port number.
- Your monitors are replicated behind multiple load balancers. In this case, you must apply a separate policy for each load balancer. In addition, if you want to configure the individual monitors to receive observations (from the load balancer) on different port numbers, you must apply a separate policy for each port number, per load balancer.
- You have a mix of singleton and replicated monitors. In this case, you must apply one policy per port number for the singleton monitors and one policy per port number per load balancer for the replicated monitors.
- You intend to enable SSL, and different monitors will use different private key/certificate pairs. In this case, you must apply a different for each private key/certificate pair.
- You intend to enable SSL, and the location of the key store is different for different monitors, or the location of the trust store is different for different observers. In this case, you must apply a different policy for each location.

The following procedure describes how to apply the Observer Communication policy:

1. Open the Observer Communication policy you will use for configuring your observer and its associated monitor. Depending on your monitor deployment scenario, either edit the default policy instance or create a new policy instance.
   - To edit the default Observer Communication policy instance: Select **Administration > System Policies** in the Navigator, select the **Default Observer Communication Policy** in the main area, and then choose **Modify > Edit Definition for Default Observer Communication Policy**.
   - To create a new instance of an Observer Communication policy: Choose **Admin > Create System Policy > Observer Communication**.
2. In the **Active Probes** section, choose the types of business components that you want to discover and monitor.

   This section lets you activate and deactivate probes.

   Note: A probe is the component within an observer that is responsible for discovering and monitoring a particular type of business component. Most types of observers have multiple probes. For more information about probes, see Section 1.1, "Architecture".

   In almost all cases, you can accept the default settings for this section. By default all probes, except JAVA, are activated.

   Note: The JAVA probe monitors local Java calls, which in most cases is not needed and can be distracting because of the typically large number of local Java calls that occur. In order to use the JAVA probe, you must first deploy and configure it. For information about deploying and configuring the JAVA probe, enter a service request at My Oracle Support (support.oracle.com).

   There is no need to explicitly deactivate probes that are not installed—neither for the sake of performance nor for any other reason (uninstalled probes are inherently not activated). The only reason to deactivate a probe is if: (1) the probe

is installed, AND (2) you do not want to monitor the type of business component the probe monitors. Furthermore, if you do deactivate (or activate) probes, you must do so in groups for some types of probes. These are the two groups of probes that you must deactivate/activate as a group: JavaEE probes (EJB, JAXRPC, JAXWS, JMS, and RMI) and the SOA Suite probes (SOA_BIZRULE, SOA_BPEL, SOA_BPMN, SOA_MEDIATOR, SOA_SPRING, SOA_WS, SOA_WSA).

Select/deselect the **Enable Discovery** checkbox to activate/deactivate the discovery mechanism for the associated component type. Components of that type are then discovered and displayed in the Management Console the next time they receive a message or call.

Select/deselect the **Monitor Upon Discovery** checkbox for a component type if you want to immediately begin/stop monitoring components of that type as they are discovered.

Note: If you enable discovery but not monitoring and then later edit the policy and enable monitoring, the system will not begin monitoring previously discovered components. The system will begin monitoring only the components discovered after you enable monitoring. For information on enabling monitoring for previously discovered components, see the topic "Start and Stop Monitoring of Endpoints" in the online help.

---

3. The way you perform the next steps depends on whether your monitors are singleton or replicated.

   For singleton monitors:

   a. Set the **Communication path** field to: **Direct to monitor** (this is the default setting).

   b. Specify the port number on which your monitors listen in the **Monitor port number** field.

      This is the port to which the observers send observations and at which the monitors receive observations. This setting configures both the monitors and observers.

      Note: You do not have to specify the host name for the monitor. The host name is obtained from the AP_NANO_CONFIG_URL Java system property or AmberPoint:NanoConfigUrl Windows key. You will set this system property/key on the application servers on which you install observers (see Section 9.2).

      ---

   c. In the **Criteria** section, ensure that the policy is applied to the correct monitors.

      Note: You can apply only one Observer Communication policy to any one monitor. If you are applying an additional policy, you must ensure that the **Criteria** section of any other Observer Communication policies either do not include, or explicitly exclude the monitors to which you are applying this policy.

   For replicated monitors:

a. Set the **Communication path** field to: **Through router to monitor group**

b. Specify the IP address and port number of your load balancer's socket virtual server in the **Router IP address** and **Router port number** fields.

These settings configure your observers to send their observations to your load balancer's incoming address (that is, to the load balancer's socket virtual server that you configured in Section 8.4).

c. Specify the port number on which your monitors will receive observations in the **Monitor port number** field.

This setting configures the monitors to listen on the specified port. This port number should coincide with the monitor port number that you assigned in the pool of your load balancer's socket virtual server in Section 8.4.

d. In the **Criteria** section, ensure that the policy is applied to the appropriate monitors.

You can apply the policy to all monitors in a particular monitor group by adding a **Monitor Group** clause, for example:

Note: If you later add or delete a monitor to or from a monitor group, you must reconfigure your load balancer accordingly.

4. If you want to use a secure connection for transporting observation messages from observers to monitors, enable the **Enable SSL** checkbox, otherwise click **Apply** now and exit the policy editor.

Enabling the **Enable SSL** checkbox displays additional fields.

5. If you have enabled SSL, specify the protocol for the connection in the **Protocol** field.

Choose **TLSv1** to use TLS 1.0.

Choose **SSLv3** to use SSL 3.0 (**SSLv3** is not supported by the .NET-based observers.).

Choose **Any** to let the components decide on the best protocol at runtime.

6. By default, built-in, preconfigured security stores are used when you enable SSL. If you want to use these built-in security stores, proceed to substep d.

If you want to use your own security stores, disable the **Use Default Stores** checkbox (in which case, additional fields are displayed) and complete all of the following substeps:

a. Specify the information for the Monitor's key store as follows:

Key Store Location – The location of the key store. You can specify this location as either an absolute path, if the key store file is local to your monitor, or as an HTTP(S) URL, if the file is accessible by HTTP GET.

Key Store Password – The password for accessing the key store.

Key Store Type – The type of the key store, for example, JKS, JCEKS, or PKCS12.

Key Name – The certificate and private key. You can specify a key alias or a certificate attribute of the form CN=*value*, UID=*value*, etc.

Key Password – The password for accessing the certificate and private key. If unspecified, the password for the key store is used.

b. If you want the monitor to automatically dispatch the trust store to your Java-based observers, enable the **Auto Dispatch Trust Store to Java Observers** checkbox.

c. Specify the information for the trust store as follows:

Trust Store Location – The location of the trust store. You can specify this location as either an absolute path, if the key store file is local to your observer (or local to your monitor, if you are using auto dispatch), or as an HTTP(S) URL, if the file is accessible by HTTP GET.

Trust Store Password – The password for accessing the trust store.

Trust Store Type – The type of the trust store, for example, JKS, JCEKS, or PKCS12.

d. If you are using .NET-based observers, ensure that you have deployed the appropriate certificate to the machines hosting those observers as described in Section 4.4, "Setting up a Secure Socket (SSL) for Observation Messages."

7. Click **Apply**.

This installation guide does not describe the policy's advanced settings. For information about the advanced settings, refer to the Business Transaction Management online help.

8.6 Adding and Removing Monitors to/from a Monitor Group

After setting up a monitor group, you might find that you need to add or remove monitor replicates to or from the monitor group. You are free to do this, but you must perform the steps of the procedure as described in this section.

8.6.1 Adding Monitors to a Monitor Group

1. Deploy, and then register the monitor as described in Section 8.2, "Deploying and Registering Monitors".
2. Assign the monitor to the monitor group as described in Section 8.3, "Setting Up a Monitor Group".
3. Assign the monitor to your load balancer's socket virtual server pool as described in Section 8.4, "Configuring Your Load Balancer".

8.6.2 Removing Monitors from a Monitor Group

1. Remove the monitor from your load balancer's socket virtual server pool.
2. Remove the monitor from the monitor group as follows:
   a. In the Business Transaction Management Console's Navigator, choose **Administration > Monitors**.
   b. In the main area, select the monitor that you want to remove from the monitor group.

c. Choose **Modify > Edit Profile for** *Your_Monitor*, where Your_Monitor is the name of your monitor.

   The **Edit Profile** tool opens.

d. Delete the name of the monitor group from the **Monitor Group** field.

e. Click **Apply**.

f. If you want to remove the monitor only from the monitor group and leave it as part of your Business Transaction Management system, stop now. If you want to remove the monitor completely from your system, perform the remaining steps.

3. Use your application server's management tools to undeploy the monitor.

4. Unregister the monitor as follows:

   a. In the Business Transaction Management Console's Navigator, choose **Administration > Monitors**.

   b. In the main area, select the monitor.

   c. Choose **Modify > Delete** Your_Monitor **Registration**, where Your_Monitor is the name of your monitor.

      The **Delete Monitor Registration** tool opens.

   d. Click **Delete**.

9 Installing Observers Overview

This chapter describes the procedure for installing Business Transaction Management observers. Installation of observer libraries are covered in separate chapters and are cross referenced from this chapter.

9.1 Prerequisite and Preliminary Setup Checklist

- Ensure that the Business Transaction Management central servers are installed and running (see Chapter 6, "Installing and Configuring the Central Servers").
- Ensure that at least one monitor is installed, registered, and running (see Chapter 8, "Installing Monitors").
- Verify that all system services are running properly.

  You can verify that the system services are running properly by choosing **Administration > System Services** in the Navigator. The summary area then lists all system services. A round, green icon should be displayed in the **System Service Status** column for each system service to indicate that it is running properly.
- If there is a firewall between the observer and its associated monitor (the monitor from which the observer retrieves its configuration and sends observations), ensure that the firewall is configured to allow the observer access to the monitor:
  - In step 4 of Section 9.2, you will associate the observer with a monitor by configuring the observer to retrieve its configuration at a particular URL. You must configure the firewall so that the observer can access this URL.
  - When you set up the Observer Communication policy, you specified a host and port to which the observer will send observations. You must configure the firewall so that the observer can access this host/port. For more information about this host/port, see step 3 of Section 8.5, "Applying an Observer Communication Policy".
- If you are using Spring Framework, enter a service request at My Oracle Support to receive special instructions on how to install the observer into an application server that contains the Spring Framework JAR files. Conflicts might arise between the observer JAR files and Spring Framework JAR files.

9.2 General Steps for Installing Observers

1. Ensure that the assertion secret and encryption key for your observer are set to the same values that are used by the other components of your Business Transaction Management system.

The components of your Business Transaction Management system use an assertion secret and an encryption key in order to secure communications between themselves. If you have configured your Business Transaction Management server components to use nondefault values for these settings, you must, likewise, configure each JVM or .NET execution environment that hosts observers to use identical values for these settings. For an in-depth explanation of these security settings and how to configure them, see Section 4.3, "Configuring the Business Transaction Management Assertion Secret and Encryption Key".

2. *Optional* – Configure SSL security on execution platforms that will host observers for the observer-to-monitor transport of observation messages. If you have configured SSL security for any of your monitors, then you must also configure it for the associated observers.

   See Section 4.4, "Setting up a Secure Socket (SSL) for Observation Messages."

3. Ensure that an Observer Communication policy is set up.

   See Section 8.5, "Applying an Observer Communication Policy".

4. Install the observer libraries and configure them into your application server.

   The procedure for performing this task is specific to the application server and observer type. See the following chapters for detailed instructions:

   - Chapter 10, "Installing Observer Libraries on WebLogic".
   - Chapter 11, "Installing Observer Libraries on WebSphere".
   - Chapter 12, "Installing Observer Libraries on JBoss".
   - Chapter 13, "Installing Observer Libraries for WCF".
   - Chapter 14, "Installing Observer Libraries for ASP.NET".

   Note: For detailed information about a specific observer's compatibility and functionality, refer to the README.txt file located in the observer's nanoagent directory after you expand the observer ZIP file.

5. Set the AP_NANO_CONFIG_URL Java system property or AmberPoint:NanoConfigUrl Windows key on the application server.

   This property/key associates the observer with the monitor whose URL you specify in the property/key. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor. Following is an example of the URL. Edit only the host name and port number:

   ```
   http://my_host:8080/btmmonitor/agent/agent/
   ```

   Setting this property/key is also explained in the application server-specific instructions for installing the observer libraries.

Note: If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

```
http://10.147.46.152:5072/btmmonitor/agent/agent/
```

For more information about the load balancer's HTTP virtual server, see Section 8.4.

6. Restart your application server.

Notes: *For WebSphere* – After installing the observer, if you deploy a new application, you must restart the application server for the services to be correctly monitored by the observer.

9.3 Specifying the Observer Library Location

This section applies only to Java application servers (WCF and ASP .NET observer libraries are installed in the GAC).

By default, the observer library is located at AP_NANO_HOME/lib. If you want to place the observer library in a different location, create a Java system property in your server named AP_NANO_CLASSLOADER_BASEDIR and set its value to the location of the library.

The observer uses the following order of precedence to locate its library (from highest to lowest):

1. the value of the system property AP_NANO_CLASSLOADER_BASEDIR
2. the value of the deprecated system property apclassloader.basedir
3. AP_NANO_HOME/lib

10

Installing Observer Libraries on WebLogic

This chapter provides instructions for installing and uninstalling observer libraries for monitoring:

- JavaEE running in WebLogic application servers versions 9.2, 10.0, and 10.3
- Oracle Service Bus 10*g*R3 running in WebLogic 10.3 application servers
- Oracle Service Bus 11*g*R1 running in WebLogic 10.3 application servers
- Oracle SOA Suite 11*g*R1 running in WebLogic 10.3 application servers

Separate instructions are provided for:

- Installation on Node Manager-Configured Servers
- Installation on Script-Configured Servers

10.1 The Observer Distribution Files

The Business Transaction Management observers are distributed by way of ZIP files. Each ZIP file contains one type of observer that is suitable for installation into a particular application server. The ZIP files suitable for installing observers into WebLogic application servers are as follows:

- **BTMObserver_Wls_10.3_JavaEE_*.zip** – Use this ZIP file to install the observer for JavaEE into a WebLogic 10.3 server.
- **BTMObserver_Wls_9.2_JavaEE_*.zip** – Use this ZIP file to install the observer for JavaEE into a WebLogic 9.2 or 10.0 server (this observer is distributed with release 12.1.0.1 rather than release 12.1.0.2).
- **BTMObserver_Wls_10.3_Soa11gR1_*.zip** – Use this ZIP file to install the observer for Oracle SOA Suite into a WebLogic 10.3 server.
- **BTMObserver_Wls_10.3_Osb11gR1_*.zip** – Use this ZIP file to install the observer for Oracle Service Bus 11*g*R1 into a WebLogic 10.3 server.
- **BTMObserver_Wls_10.3_Osb10gR3_*.zip** – Use this ZIP file to install the observer for Oracle Service Bus 10*g*R3 into a WebLogic 10.3 server (this observer is distributed with release 12.1.0.1 rather than release 12.1.0.2).

Note: In the complete ZIP file names, the asterisk (*) is replaced with the observer version number.

10.2 Installation on Node Manager-Configured Servers

Note: You must not install more than one observer per application server.

1. Locate the observer distribution ZIP file that is appropriate for your version of WebLogic and the type of traffic you want to monitor.
2. Unpack the observer ZIP file into your WebLogic managed server's home directory (*WL_HOME*).

   The home directory is the weblogic92 or wlserver_10.3 directory located in your WebLogic installation directory, for example, C:\bea\wlserver_10.3. For the remainder of this procedure, replace the string *WL_HOME* with the actual path to the WebLogic home directory.

   Unpacking the ZIP file creates a directory named nanoagent that contains three subdirectories bin, config, and lib.

   Note: By default, the observer looks in the lib directory for its libraries. For information on overwriting this default location, see Section 9.3, "Specifying the Observer Library Location."

3. Ensure that the user account running WebLogic has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
4. If your WebLogic server accesses an Oracle database using either the ojdbc5.jar or ojdbc6.jar database driver, then you must add oraclepki.jar to your server's *WL_HOME*/server/lib directory.

   You can download oraclepki.jar from the downloads section of Oracle Technology Network (http://www.oracle.com/technetwork/index.html). This JAR file is necessary for the JDBC probe to collect observations correctly.
5. Open the WebLogic Administration Console (the default URL is http://*machine_name*:7001/console).
6. *Do not perform this step if you are installing the JavaEE observer for WebLogic 10.3 (BTMObserver_Wls_10.3_JavaEE_*.zip), but do perform it for all other observers.* – Add *WL_HOME*/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar to your managed server's classpath:
   a. Using the Domain Structure pane (on the left), navigate to **Environment > Servers**.
   b. In the **Servers** table, click your managed server.
   c. Display the **Configuration / Server Start** tab.
   d. Click **Lock & Edit**.

Note: These instructions assume you are operating in a production environment and that your WebLogic server's **Automatically Acquire Lock and Activate Changes** setting is therefore disabled. However, if this setting is enabled as it might be in a development environment, you do not have to click **Lock & Edit** in order to make changes and you do not have to activate changes after saving them.

e. Add the observer libraries to your server's class path by adding the observer bootstrap file to the **Class Path** field as follows:

For Windows systems, add this string, using a semicolon (";") to separate entries in the field:

```
WL_HOME\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar
```

For UNIX-like systems, add this string, using a colon (":") to separate entries in the field:

```
WL_HOME/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
```

Note: If the **Class Path** field is initially blank, it means that the server is using the default classpath. In this case, adding the observer JAR file to the field will override the default classpath and cause the server not to start. You must, therefore, explicitly specify the default classpath in the field and then append the observer JAR file to it. You can find the value of the default classpath in the your server's log file. The log file is named after the server (for example, server_name.log) and is located in the server's directory under the domain directory. When you open the log file, search for **java.class.path**. The **Class Path** field permits spaces, forward slashes, and abbreviated Windows 8.3 names (that use the ~ symbol) within paths.

f. Keep this page open.

7. With the **Configuration / Server Start** tab displayed, edit your WebLogic startup arguments as follows:

a. Create an AP_NANO_HOME system property and set it to the location of the observer's nanoagent directory by adding the following string to the **Arguments** field:

Add this string for Windows systems, using spaces to separate entries:

```
-DAP_NANO_HOME=WL_HOME\nanoagent
```

Add this string for UNIX-like systems, using spaces to separate entries:

```
-DAP_NANO_HOME=WL_HOME/nanoagent
```

Note: Do not use new lines to separate argument entries.

b. Create an AP_NANO_CONFIG_URL system property by adding the following string to the **Arguments** field:

```
-DAP_NANO_CONFIG_URL=http://Host:Port/btmmonitor/agent/agent/
```

Replace *Host:Port* with the host name and port number of the monitor to which the observer will forward messages.

This property associates the observer with the monitor whose URL you specify. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor.

Note: If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

```
http://10.147.46.152:5072/btmmonitor/agent/agent/
```

For more information on this topic, see Section 8.4.

---

c. *Perform this step only if you are installing the JavaEE observer for WebLogic 10.3 (BTMObserver_Wls_10.3_JavaEE_*.zip).* – Configure the observer bootstrap module into your server by adding one of the following JVM arguments to the **Arguments** field.

Add this argument for Windows systems:

```
-javaagent:WL_HOME\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar
```

Add this argument for UNIX-like systems:

```
-javaagent:WL_HOME/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
```

d. *Do not perform this step if you are installing the JavaEE observer for WebLogic 10.3 (BTMObserver_Wls_10.3_JavaEE_*.zip), but do perform it for all other observers.* – Configure the AspectWerkz module into your server by adding one of the following sets of JVM arguments to the **Arguments** field.

Add these arguments for Windows systems:

```
-Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APA
spectPreProcessor -javaagent:
WL_HOME\nanoagent\lib\bootstrap\aspectwerkz-jdk5-2.0.jar
```

Add these arguments for UNIX-like systems:

```
-Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APA
spectPreProcessor -javaagent:
WL_HOME/nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar
```

8. Ensure that the user under which your WebLogic server is running has permission to write to the observer's error log directory.

   By default, the observer's error log directory is the WebLogic domain directory. For information about configuring error logging, see Chapter 15, "Logging Observer Errors and Debugging Information".

9. Click **Save** and then click **Activate Changes**.

10. Restart your managed server.

11. Ensure that the monitor to which your observer forwards messages has an Observer Communication policy applied to it.

For information on applying an Observer Communication policy, see Section 8.5, "Applying an Observer Communication Policy."

10.3 Installation on Script-Configured Servers

Note: Depending on how you perform this procedure, you can install the observer into either all servers defined in the WebLogic installation or into servers of a specific domain. If you choose to install the observer into a specific domain, you are permitted to install one package of observer types into one domain and a different package of observer types into a different domain. For example, you could install BTMObserver_Wls_10.3_JavaEE_*.zip into one domain in order to observe JavaEE services, install BTMObserver_Wls_10.3_Soa11gR1_*.zip into a second domain in order to observe SOA services, and install BTMObserver_Wls_10.3_Osb11gR1_*.zip into a third domain in order to observe OSB services. In this procedure, the term *global install* refers to installing the observer into all servers, and the term *domain install* refers to installing the observer into a specific domain.

1. Locate the observer distribution ZIP file that is appropriate for your version of WebLogic and the type of traffic you want to monitor.
2. Unpack the observer ZIP file into either your WebLogic server's home directory (to perform a global install) or into one of its domain directories (to perform a domain install).

   The home directory is the weblogic92 or wlserver_10.3 directory located in your WebLogic installation directory, for example, C:\bea\wlserver_10.3. For the remainder of this procedure, replace *WL_HOME* with the actual path to the WebLogic home directory.

   Unpacking the ZIP file creates a directory named nanoagent that contains three subdirectories bin, config, and lib.

   Note: By default, the observer looks in the lib directory for its libraries. For information on overwriting this default location, see Section 9.3, "Specifying the Observer Library Location."

3. Ensure that the user account running WebLogic has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
4. If your WebLogic server accesses an Oracle database using either the ojdbc5.jar or ojdbc6.jar database driver, then you must add oraclepki.jar to your server's *WL_HOME*/server/lib directory.

   You can download oraclepki.jar from the downloads section of Oracle Technology Network (http://www.oracle.com/technetwork/index.html). This JAR file is necessary for the JDBC probe to collect observations correctly.
5. Configure your WebLogic domain startup scripts to call the observer script file:

Note: This step assumes that you haven't modified your startWebLogic scripts. If you have modified your scripts, you might also have to modify the installation procedure accordingly.

a. Navigate to the bin directory of one of the WebLogic domains whose services you want to monitor and open the startup script in a text editor (open bin\startWebLogic.cmd for Windows systems or bin/startWebLogic.sh for UNIX-like systems; do not edit the startup script located directly within the domain directory).

b. Locate the following line (the first line is for Windows and the second for UNIX-like systems):

```
call "%DOMAIN_HOME%\bin\setDomainEnv.cmd"

. ${DOMAIN_HOME}/bin/setDomainEnv.sh
```

c. Directly after that line, add a line that calls the observer script file:

If you are performing a global install on a Windows system, add this line:

```
call "%WL_HOME%\nanoagent\bin\nanoEnvWeblogic.cmd"
```

If you are performing a global install on a UNIX-like system, add this line (note the initial period and space):

```
. ${WL_HOME}/nanoagent/bin/nanoEnvWeblogic.sh
```

If you are performing a domain install on a Windows system, add this line:

```
call "%DOMAIN_HOME%\nanoagent\bin\nanoEnvWeblogic.cmd"
```

If you are performing a domain install on a UNIX-like system, add this line (note the initial period and space):

```
. ${DOMAIN_HOME}/nanoagent/bin/nanoEnvWeblogic.sh
```

Note: If you performed the security-related configuration described in Section 4.3.1, "Configuring Security Using Oracle Wallet," you will have already added a call to the setBtmOverriderEnv_via_CredStore script in this location. The relative order in which these scripts are called does not matter.

d. Perform this step for each domain whose services you want to monitor.

6. Open the observer script file in a text editor.

The observer script file is the nanoEnvWeblogic.cmd or nanoEnvWeblogic.sh file that your startWebLogic script file calls.

7. If you are performing a domain install, make the following change (otherwise, skip to the next step):

For Windows systems, locate the line:

```
set NANOAGENT_HOME=%WL_HOME%\nanoagent
```

and change it to:

```
set NANOAGENT_HOME=%DOMAIN_HOME%\nanoagent
```

For UNIX-like systems, locate the line:

```
NANOAGENT_HOME=$WL_HOME/nanoagent
```

and change it to:

```
NANOAGENT_HOME=$DOMAIN_HOME/nanoagent
```

Keep the file open.

8. Associate your observer with a monitor.

Note: The observer script file uses the NANOAGENT_CONFIGURATION_URL variable to set the value of the system property AP_NANO_CONFIG_URL. This system property associates the observer with the monitor whose URL you specify. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor.

---

   a. In the observer script file, locate the following variable definition (the first line is for Windows and the second for UNIX-like systems):

```
set NANOAGENT_CONFIGURATION_URL=http://HOST:PORT/btmmonitor/agent/agent/

NANOAGENT_CONFIGURATION_URL=http://HOST:PORT/btmmonitor/agent/agent/
```

   b. Replace HOST:PORT with the host name and port number of the monitor to which you want the observer to send observations, for example:

```
set NANOAGENT_CONFIGURATION_URL=http://myhost:7002/btmmonitor/agent/agent

NANOAGENT_CONFIGURATION_URL=http://myhost:7002/btmmonitor/agent/agent"
```

Note: If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

```
http://10.147.46.152:5072/btmmonitor/agent/agent/
```

For more information about the load balancer's HTTP virtual server, see Section 8.4.

---

   c. If you are performing a domain install, perform steps 7 and 8 for each domain.

9. Ensure that the user under which your WebLogic server is running has permission to write to the observer's error log directory.

   By default, the observer's error log directory is the WebLogic domain directory. For information about configuring error logging, see Chapter 15, "Logging Observer Errors and Debugging Information".

10. Restart your server.

11. Ensure that the monitor to which your observer forwards messages has an Observer Communication policy applied to it.

    For information on applying an Observer Communication policy, see Section 8.5, "Applying an Observer Communication Policy."

10.4 Uninstalling Observer Libraries for WebLogic

This section describes how to uninstall observer libraries from a WebLogic 9.2 or 10.3 server. The procedure differs according to whether you configure your server using the Node Manager or local scripts.

- If you configure your server using the Node Manager, see Section 10.4.1, "Uninstallation from a Managed Server Configured by the Node Manager."
- If you configure your server using local scripts, see Section 10.4.2, "Uninstallation from a Server Configured by a Local Script."

10.4.1 Uninstallation from a Managed Server Configured by the Node Manager

1. Stop your WebLogic managed server.
2. Delete the nanoagent folder located in your WebLogic home directory (*WL_HOME*).

   The home directory is the weblogic92 or wlserver_10.3 directory located in your WebLogic installation directory, for example, C:\bea\wlserver_10.3. For the remainder of this procedure, replace *WL_HOME* with the actual path to the WebLogic home directory.
3. Open the WebLogic Administration Console (the default URL is http://*Machine_Name*:7001/console).
4. Remove *WL_HOME*/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar from your WebLogic managed server's classpath:
   a. Using the navigation pane (on the left), navigate to **Environment > Servers**.
   b. In the **Servers** table, click your managed server.
   c. Display the **Configuration / Server Start** tab.
   d. Click **Lock & Edit**.
   e. Remove the following string from the **Class Path** field.

      ```
      WL_HOME/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
      ```

   f. Keep this page open.
5. Remove all observer-related startup arguments:

   With the **Configuration / Server Start** tab displayed, remove any and all of the following strings from the **Arguments** field:

   ```
   -DAP_NANO_HOME=WL_HOME/nanoagent

   -DAP_NANO_CONFIG_URL=http://HOST:PORT/btmmonitor/agent/agent/

   -Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APAspectPreProcessor

   -javaagent:WL_HOME/nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar
   ```

6. Click **Save** and then click **Activate Changes**.
7. Restart your WebLogic server.

10.4.2 Uninstallation from a Server Configured by a Local Script

1. Stop your WebLogic server.
2. Delete the nanoagent folder located in your WebLogic home directory (*WL_HOME*).

   The home directory is the weblogic92 or wlserver_10.3 directory located in your WebLogic installation directory, for example, C:\bea\wlserver_10.3. For the remainder of this procedure, replace *WL_HOME* with the actual path to the WebLogic home directory.
3. Navigate to the bin directory of the WebLogic domain whose services you no longer want to monitor and edit the startWebLogic script (startWebLogic.cmd for Windows or startWebLogic.sh for UNIX-like systems). Delete the following line from the script (the first line is for Windows and the second for UNIX-like systems):

   ```
   call "%WL_HOME%\nanoagent\bin\nanoEnvWeblogic.cmd"

   . ${WL_HOME}/nanoagent/bin/nanoEnvWeblogic.sh
   ```
4. Restart your WebLogic server.

11

Installing Observer Libraries on WebSphere

This chapter provides instructions for installing and uninstalling observer libraries for monitoring JavaEE running in WebSphere 6.1 application servers.

11.1 The Observer Distribution Files

The Business Transaction Management observers are distributed by way of ZIP files. Each ZIP file contains one type of observer that is suitable for installation into a particular application server. The ZIP file suitable for installing an observer into a WebSphere application server is BTMObserver_Was_6.1_JavaEE_*.zip (this observer is distributed with release 12.1.0.1 rather than release 12.1.0.2).

Note: In the actual ZIP file, the asterisk (*) is replaced with the observer version number.

11.2 Installing the Observer Libraries on WebSphere

1. Locate the observer distribution ZIP file that is appropriate for your version of WebSphere.

   This ZIP file contains the observer for WebSphere 6.1.

2. Unpack the observer ZIP file into your WebSphere installation root directory (*WAS_INSTALL_ROOT*).

   The default location of the WebSphere installation root directory on Windows systems is:

   ```
   C:\Program Files\IBM\WebSphere\AppServer
   ```

   On UNIX-like systems, the default location is:

   ```
   /opt/IBM/WebSphere/AppServer
   ```

   For the remainder of this procedure, replace *WAS_INSTALL_ROOT* with the actual path to the WebSphere installation root directory.

   Unpacking the ZIP file creates a directory named nanoagent that contains two subdirectories—config and lib.

Note: By default, the observer looks in the lib directory for its libraries. For information on overwriting this default location, see Section 9.3, "Specifying the Observer Library Location."

---

3. Ensure that the user account running WebSphere has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
4. Move ap-nano-jaxrpc.jar from nanoagent/lib/bootstrap to *WAS_INSTALL_ROOT*/lib/ext.
5. Restart WebSphere and log in to the WebSphere Administrative Console for your profile.
6. In the WebSphere Administrative Console, navigate to **Servers > Application servers >** *server1* **> Server Infrastructure > Java and Process Management > Process Definition > Java Virtual Machine** (you might have to substitute a different server name for server1).
   a. Add the observer libraries to your server's class path by adding the observer bootstrap file to the **Classpath** field as follows:

      For Windows systems, add this string:

      ```
      ${WAS_INSTALL_ROOT}\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar
      ```

      For UNIX-like systems, add this string:

      ```
      ${WAS_INSTALL_ROOT}/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
      ```

   b. Add the following arguments to the **Generic JVM arguments** field:

      For Windows systems, add this string:

      ```
      -javaagent:${WAS_INSTALL_
      ROOT}\nanoagent\lib\bootstrap\aspectwerkz-jdk5-2.0.jar
      ```

      For UNIX-like systems, add this string:

      ```
      -javaagent:${WAS_INSTALL_
      ROOT}/nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar (Linux/UNIX)
      ```

      Note: This property configures the AspectWerkz module. You do not need to set it if you plan to monitor only JAX-RPC services.

7. Add custom properties as follows:
   a. Click the **Custom Properties** link, and then for each property you are adding, click **New**, input the name, value, and description, and then click **OK**. The properties to add follow:
   b. Add a property that specifies the observer home directory:

      Name = AP_NANO_HOME

      Value = *WAS_INSTALL_ROOT*/nanoagent

Note: Replace *WAS_INSTALL_ROOT* with the actual path to your WebSphere installation root directory. You must uses backslashes for Windows. Follow these examples:

For Windows systems:

Value = C:\Program Files\IBM\WebSphere\AppServer

For UNIX-like systems:

Value = /opt/IBM/WebSphere/AppServer

Description = The observer home directory

c. Add a property that associates the observer with a monitor:

Name = AP_NANO_CONFIG_URL

Value = http://*Host:Port*/btmmonitor/agent/agent/

Notes: Replace *Host:Port* with the host name and port number at which the monitor is available.

If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

```
http://10.147.46.152:5072/btmmonitor/agent/agent/
```

For more information about the load balancer's HTTP virtual server, see Section 8.4.

Description = Associates the observer with the monitor whose URL you specify. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor.

d. Add a property that configures the AspectWerkz module:

Name = aspectwerkz.classloader.preprocessor

Value = com.amberpoint.nanoagent.plugins.APAspectFilterPreProcessor

Description = Business Transaction Management observer class preprocessor

---

Note: You do not need to set this property if you plan to monitor only JAX-RPC services.

---

e. Click **Save** to save your master configuration.

8. Shutdown WebSphere and modify the Java policies as described in Section 11.3, "Java Policy Modifications."

If Java 2 security is enabled for WebSphere, you must make all of the modifications described in the section. If Java 2 security is not enabled, you need modify only the server.policy file.

9. Ensure that the user under which your WebSphere server is running has permission to write to the observer's error log directory.

   By default, the observer's error log directory is the WebSphere profile directory. For information about configuring error logging, see Chapter 15, "Logging Observer Errors and Debugging Information."

10. Restart WebSphere.

11. Ensure that the monitor to which your observer forwards messages has an Observer Communication policy applied to it.

    For information on applying an Observer Communication policy, see Section 8.5, "Applying an Observer Communication Policy."

11.3 Java Policy Modifications

You must modify your WebSphere server's server.policy file as described in Section 11.3.1. If Java 2 security is enabled on your WebSphere server, you must also modify your app.policy files as described in Section 11.3.2.

11.3.1 Modifications to the WebSphere server.policy file

Add the following policy definition to the WebSphere server.policy file and save it (the server.policy file is located at *WAS_INSTALL_ROOT*/profiles/*Profile_Name*/properties/server.policy):

```
grant codeBase "file:${was.install.root}/nanoagent/lib/-" {
  permission java.security.AllPermission;
};
```

Note: You must use forward slashes in the path, even on Windows.

11.3.2 Modifications to the app.policy files

Add permissions to the app.policy file(s) as follows:

WebSphere has an app.policy file for each node. You must make the modification for each node that you want to monitor. The app.policy files are located in:

```
WAS_INSTALL_ROOT/profiles/profile_name/config/cells/cell_name/nodes/node_name
```

Add the following permissions to each policy definition in the app.policy file:

```
permission java.lang.RuntimePermission "getClassLoader";
permission java.lang.RuntimePermission "accessDeclaredMembers";
permission java.lang.RuntimePermission "modifyThreadGroup";
permission java.lang.RuntimePermission "modifyThread";
permission java.lang.RuntimePermission "shutdownHooks";
permission java.io.FilePermission "<<ALL FILES>>", "read,write,delete";
permission java.lang.RuntimePermission "accessClassInPackage.*";
permission java.util.logging.LoggingPermission "control";
permission java.lang.RuntimePermission "createClassLoader";
permission java.lang.RuntimePermission "reflectionFactoryAccess";
permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
permission com.ibm.websphere.security.WebSphereRuntimePermission
"AdminPermission";
```

The following permissions are required in each policy definition in the app.policy file except the application policy definition (the first policy definition in the file). Your app.policy file probably has these permissions already declared. If not, add them.

```
permission java.net.SocketPermission "*", "connect";
permission java.util.PropertyPermission "*", "read";
```

Your app.policy file should look something like this when you have finished:

```
grant codeBase "file:${application}" {
  // The following are required by Java mail
  permission java.io.FilePermission "${was.install.root}${/}lib${/}mail-impl.jar",
"read";
  permission java.io.FilePermission
"${was.install.root}${/}lib${/}activation-impl.jar",
"read";
  //Permissions Added for BTM observers
  permission java.lang.RuntimePermission "getClassLoader";
  permission java.lang.RuntimePermission "accessDeclaredMembers";
  permission java.lang.RuntimePermission "modifyThreadGroup";
  permission java.lang.RuntimePermission "modifyThread";
  permission java.lang.RuntimePermission "shutdownHooks";
  permission java.io.FilePermission "<<ALL FILES>>", "read,write,delete";
  permission java.lang.RuntimePermission "accessClassInPackage.*";
  permission java.util.logging.LoggingPermission "control";
  permission java.lang.RuntimePermission "createClassLoader";
  permission java.lang.RuntimePermission "reflectionFactoryAccess";
  permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
  permission com.ibm.websphere.security.WebSphereRuntimePermission
"AdminPermission";
};
grant codeBase "file:${jars}" {
  permission java.net.SocketPermission "*", "connect";
  permission java.util.PropertyPermission "*", "read";
  //Permissions Added for BTM observers
  permission java.lang.RuntimePermission "getClassLoader";
  permission java.lang.RuntimePermission "accessDeclaredMembers";
  permission java.lang.RuntimePermission "modifyThreadGroup";
  permission java.lang.RuntimePermission "modifyThread";
  permission java.lang.RuntimePermission "shutdownHooks";
  permission java.io.FilePermission "<<ALL FILES>>", "read,write,delete";
  permission java.lang.RuntimePermission "accessClassInPackage.*";
  permission java.util.logging.LoggingPermission "control";
  permission java.lang.RuntimePermission "createClassLoader";
  permission java.lang.RuntimePermission "reflectionFactoryAccess";
  permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
  permission com.ibm.websphere.security.WebSphereRuntimePermission
"AdminPermission";
};
grant codeBase "file:${connectorComponent}" {
  permission java.net.SocketPermission "*", "connect";
  permission java.util.PropertyPermission "*", "read";
  //Permissions Added for BTM observers
  permission java.lang.RuntimePermission "getClassLoader";
  permission java.lang.RuntimePermission "accessDeclaredMembers";
  permission java.lang.RuntimePermission "modifyThreadGroup";
  permission java.lang.RuntimePermission "modifyThread";
  permission java.lang.RuntimePermission "shutdownHooks";
  permission java.io.FilePermission "<<ALL FILES>>", "read,write,delete";
  permission java.lang.RuntimePermission "accessClassInPackage.*";
  permission java.util.logging.LoggingPermission "control";
```

```
  permission java.lang.RuntimePermission "createClassLoader";
  permission java.lang.RuntimePermission "reflectionFactoryAccess";
  permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
  permission com.ibm.websphere.security.WebSphereRuntimePermission
"AdminPermission";
};
grant codeBase "file:${webComponent}" {
  permission java.io.FilePermission "${was.module.path}${/}-", "read, write";
  permission java.lang.RuntimePermission "loadLibrary.*";
  permission java.lang.RuntimePermission "queuePrintJob";
  permission java.net.SocketPermission "*", "connect";
  permission java.util.PropertyPermission "*", "read";
  //Permissions Added for BTM observers
  permission java.lang.RuntimePermission "getClassLoader";
  permission java.lang.RuntimePermission "accessDeclaredMembers";
  permission java.lang.RuntimePermission "modifyThreadGroup";
  permission java.lang.RuntimePermission "modifyThread";
  permission java.lang.RuntimePermission "shutdownHooks";
  permission java.io.FilePermission "<<ALL FILES>>", "read,write,delete";
  permission java.lang.RuntimePermission "accessClassInPackage.*";
  permission java.util.logging.LoggingPermission "control";
  permission java.lang.RuntimePermission "createClassLoader";
  permission java.lang.RuntimePermission "reflectionFactoryAccess";
  permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
  permission com.ibm.websphere.security.WebSphereRuntimePermission
"AdminPermission";
};
grant codeBase "file:${ejbComponent}" {
  permission java.lang.RuntimePermission "queuePrintJob";
  permission java.net.SocketPermission "*", "connect";
  permission java.util.PropertyPermission "*", "read";
  //Permissions Added for BTM observers
  permission java.lang.RuntimePermission "getClassLoader";
  permission java.lang.RuntimePermission "accessDeclaredMembers";
  permission java.lang.RuntimePermission "modifyThreadGroup";
  permission java.lang.RuntimePermission "modifyThread";
  permission java.lang.RuntimePermission "shutdownHooks";
  permission java.io.FilePermission "<<ALL FILES>>", "read,write,delete";
  permission java.lang.RuntimePermission "accessClassInPackage.*";
  permission java.util.logging.LoggingPermission "control";
  permission java.lang.RuntimePermission "createClassLoader";
  permission java.lang.RuntimePermission "reflectionFactoryAccess";
  permission java.lang.reflect.ReflectPermission "suppressAccessChecks";
  permission com.ibm.websphere.security.WebSphereRuntimePermission
"AdminPermission";
};
```

11.4 Uninstalling the Observer Libraries from WebSphere

1. Log in to the WebSphere Administrative Console for your profile.
2. Navigate to **Servers > Application Servers >** *server1* **> Server Infrastructure >Java and Process Management > Process Definition >Java Virtual Machine**.

   You might have to substitute a different server name for *server1*.
3. Remove the following line from the **Classpath** field:

   ```
   ${WAS_INSTALL_ROOT}/nanoagent/lib/bootstrap/ap-nano-bootstrap.jar
   ```
4. Remove the following line from the **Generic JVM** arguments field:

```
-javaagent:${WAS_INSTALL_ROOT}/nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar
```

5. Navigate to **Servers > Application Servers >** *server1* **> Server Infrastructure > Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties**.
6. Remove any and all of the following custom properties (you might not have all of these properties set):
   - AP_NANO_CONFIG_URL
   - AP_NANO_HOME
   - aspectwerkz.classloader.preprocessor
7. Click **Save** to save these changes to your master configuration.
8. Stop the WebSphere server.
9. Remove ap-nano-jaxrpc.jar from *WAS_INSTALL_ROOT*\lib\ext.

   The default location of the WebSphere installation root directory (*WAS_INSTALL_ROOT*) on Windows systems is:

   ```
   C:\Program Files\IBM\WebSphere\AppServer
   ```

   On UNIX-like systems, the default location is:

   ```
   /opt/IBM/WebSphere/AppServer
   ```

10. Delete the nanoagent directory from *WAS_INSTALL_ROOT*, including all subdirectories and files.
11. Restart WebSphere.

12

Installing Observer Libraries on JBoss

This chapter provides instructions for installing and uninstalling observer libraries for monitoring JavaEE running in JBossEAP 4.3 application servers.

12.1 The Observer Distribution File

The Business Transaction Management observers are distributed by way of ZIP files. Each ZIP file contains one type of observer that is suitable for installation into a particular application server. The ZIP file suitable for installing an observer into a JBoss server is BTMObserver_Jboss_4.3_JavaEE_*.zip (this observer is distributed with release 12.1.0.1 rather than release 12.1.0.2).

> **Note:** In the complete ZIP file name, the asterisk (*) is replaced with the observer version number.

12.2 Installing the Observer Libraries on JBossEAP 4.3

1. Shut down your JBoss server.
2. Locate the observer distribution ZIP file for JBoss (BTMObserver_Jboss_4.3_JavaEE_*.zip).
3. Unpack the ZIP file into *JBOSS_HOME*.

   JBOSS_HOME refers to the value of your JBOSS_HOME environment variable

   Unpacking the ZIP file creates a nanoagent directory containing three subdirectories—config, jaxws, and lib.

   > **Note:** By default, the observer looks in the lib directory for its libraries. For information on overwriting this default location, see Section 9.3, "Specifying the Observer Library Location."

4. Ensure that the user account running JBoss has at least the following privileges:
   - read permission on the nanoagent/config and nanoagent/lib directories (on UNIX-like systems traverse permission is also required)
   - read permission on all JAR files in the lib directory
5. If you want to monitor JAX-WS services:

a. Move ap-nano-jaxws.jar from *JBOSS_HOME*/nanoagent/jaxws to *JBOSS_HOME*/server/*server_config*/lib, where *server_config* stands for the server configuration directory (for example, JBOSS_HOME/server/default/lib).

b. Configure the observer into the message flow of the JAX-WS services that you want to manage by inserting the observer into your services' post handler chains. These post handler chains are defined in *JBOSS_HOME*/server/*server_config*/deploy/jbossws.sar/META-INF/standard-jaxws-endpoint-config.xml and standard-jaxws-client-config.xml.

Note: Before editing your standard-jaxws-endpoint-config.xml and standard-jaxws-client-config.xml files, save a backup copy to use in case you want to uninstall the observer.

If you are using the default standard-jaxws-client-config.xml and standard-jaxws-endpoint-config.xml files provided by JBoss, you can simply replace them with the like-named files provided with Business Transaction Management and located in *JBOSS_HOME*/nanoagent/jaxws. In this case, you don't need to edit the files. Otherwise, edit the files according to the following description.

The file standard-jaxws-endpoint-config.xml defines a number of endpoint configurations that control processing of inbound requests. By default, all of your JAX-WS endpoints are associated with the configuration named Standard Endpoint. Likewise, the file standard-jaxws-client-config.xml defines a number of client configurations that control processing of outbound requests. By default, all of your JAX-WS clients (that is, your services acting as clients) are associated with the configuration named Standard Client.

You must insert the observer as a handler into the post handler chain of each configuration associated with the services you want to manage. If you have not explicitly associated particular endpoints and clients with non-default configurations, then you need to edit only the Standard Endpoint and Standard Client configurations.

Place the observer-handler in front of any other handlers that might already exist in the handler chain.

The following example shows the observer-handler inserted into the Standard Endpoint configuration:

```
<endpoint-config>
 <config-name>Standard Endpoint</config-name>
 <post-handler-chains>
  <javaee:handler-chain>
   <javaee:handler>
    <javaee:handler-name>EndpointGlobalHandler</javaee:handler-name>
    <javaee:handler-class>com.amberpoint.jaxws.jboss.handlers.JBossJaxwsEnd
pointHandler</javaee:handler-class>
   </javaee:handler>
  </javaee:handler-chain>
 </post-handler-chains>
</endpoint-config>
```

The following example shows the observer-handler inserted into the Standard Client configuration:

```
<client-config>
 <config-name>Standard Client</config-name>
```

```
 <post-handler-chains>
  <javaee:handler-chain>
   <javaee:handler>
    <javaee:handler-name>ClientGlobalHandler</javaee:handler-name>
    <javaee:handler-class>com.amberpoint.jaxws.jboss.handlers.JBossJaxwsCal
lSideHandler</javaee:handler-class>
   </javaee:handler>
  </javaee:handler-chain>
 </post-handler-chains>
 <feature>http://org.jboss.ws/dispatch/validate</feature>
 <property>
  <property-name>http://org.jboss.ws/http#chunksize</property-name>
  <property-value>2048</property-value>
 </property>
</client-config>
```

For more code examples showing insertion of the observer-handler, refer to the annotated standard-jaxws-client-config.xml and standard-jaxws-endpoint-config.xml files provided with Business Transaction Management and located in *JBOSS_HOME*/nanoagent/jaxws.

6. Modify the ClassPath and JVM arguments for your JBoss server.

   Your edit to the JVM arguments includes adding a system property named AP_NANO_CONFIG_URL. This property associates the observer with the monitor whose URL you specify in the property. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor. Following is an example of the URL. Edit only the host name and port number:

   ```
   http://my_host:8080/btmmonitor/agent/agent/
   ```

   Note: If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

   ```
   http://10.147.46.152:5072/btmmonitor/agent/agent/
   ```

   For more information about the load balancer's HTTP virtual server, see Section 8.4.

   Here are the platform-specific details:

   - *(Windows systems)* – Locate the following line in *JBOSS_HOME*\bin\run.bat:

     ```
     set JBOSS_ENDORSED_DIRS=%JBOSS_HOME%\lib\endorsed
     ```

     Directly after that line, insert the following code (replacing *Monitor_URL*, in the next-to-last line, with the monitor's URL):

     ```
     rem Set up the observer.

     set JBOSS_CLASSPATH=%JBOSS_
     CLASSPATH%;..\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar

     set AW_OPTS=-javaagent:..\nanoagent\lib\bootstrap\aspectwerkz-jdk5-2.0.jar
     ```

```
set JAVA_OPTS=%JAVA_OPTS% %AW_OPTS% -DAP_NANO_HOME=..\nanoagent -DAP_NANO_
CONFIG_URL=Monitor_URL
-Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APA
spectPreProcessor
```

- *(UNIX-like systems)* – Locate the following line in *JBOSS_HOME*/bin/run.sh:

  ```
  JBOSS_ENDORSED_DIRS="$JBOSS_HOME/lib/endorsed"
  ```

  Directly after that line, insert the following code (replacing *Monitor_URL*, in the next-to-last line, with the URL of the monitor):

  ```
  #Set up the observer.

  JBOSS_CLASSPATH=$JBOSS_
  CLASSPATH:../nanoagent/lib/bootstrap/ap-nano-bootstrap.jar

  AW_OPTS="-javaagent:../nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar"

  #LD_LIBRARY_PATH=$LD_LIBRARY_PATH:../nanoagent/lib/bootstrap

  #export LD_LIBRARY_PATH

  #Note: set LD_LIBRARY_PATH as $LD_LIBRARY_
  PATH:../nanoagent/lib/bootstrap/solaris/x86
  for Solaris Intel

  #and $LD_LIBRARY_PATH:../nanoagent/lib/bootstrap/solaris/sparc for Solaris
  Sparc.

  JAVA_OPTS="$JAVA_OPTS $AW_OPTS -DAP_NANO_HOME=../nanoagent -DAP_NANO_
  HOME=..\nanoagent
  -DAP_NANO_CONFIG_URL=Monitor_URL
  -Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APA
  spectPreProcessor"
  ```

  Note: If your server is running on Solaris, set your LD_LIBRARY_PATH variable as indicated in the commented lines. Otherwise, you can omit the commented lines.

7. Ensure that the user under which your JBoss server is running has permission to write to the observer's error log directory.

   By default, the observer's error log directory is *JBOSS_HOME*/bin. For information about configuring error logging, see Chapter 15, "Logging Observer Errors and Debugging Information."

8. Restart your JBoss server.

9. Ensure that the monitor to which your observer forwards messages has an Observer Communication policy applied to it.

   For information on applying an Observer Communication policy, see Section 8.5, "Applying an Observer Communication Policy."

12.3 Uninstalling the Observer Libraries from JBossEAP 4.3

1. Shut down your JBoss server.
2. Remove ap-nano-jaxws.jar from *JBOSS_HOME*/server/*server_config*/lib.

JBOSS_HOME refers to the value of your JBOSS_HOME environment variable and *server_config* refers to the server configuration directory (for example, *JBOSS_HOME*/server/default/lib).

> **Note:** This JAR file is used for monitoring JAX-WS services and will be present only if you set up your observer to monitor JAX-WS services.

3. Replace the files *JBOSS_HOME*/server/server_config/deploy/jbossws.sar/META-INF/standard-jaxws-client-config.xml and standard-jaxrws-endpoint-config.xml with the backup copies you made of the original files.
4. Delete the *JBOSS_HOME*/nanoagent directory.
5. Unset all observer-related system properties:

   (Windows systems) – Remove any and all of the following lines from *JBOSS_HOME*/bin/run.bat:

```
rem Set up observer.
set JBOSS_CLASSPATH=%JBOSS_
CLASSPATH%;..\nanoagent\lib\bootstrap\ap-nano-bootstrap.jar

set AW_OPTS=-javaagent:..\nanoagent\lib\bootstrap\aspectwerkz-jdk5-2.0.jar

set JAVA_OPTS=%JAVA_OPTS% %AW_OPTS% -DAP_NANO_HOME=..\nanoagent
-Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APAspec
tPreProcessor
```

   (UNIX-like systems) – Remove any and all of the following lines from *JBOSS_HOME*/bin/run.sh:

```
#Set up observer.

JBOSS_CLASSPATH=$JBOSS_
CLASSPATH:../nanoagent/lib/bootstrap/ap-nano-bootstrap.jar

AW_OPTS="-javaagent:../nanoagent/lib/bootstrap/aspectwerkz-jdk5-2.0.jar"

#LD_LIBRARY_PATH=$LD_LIBRARY_PATH:../nanoagent/lib/bootstrap

#export LD_LIBRARY_PATH

#Note: set LD_LIBRARY_PATH as $LD_LIBRARY_
PATH:../nanoagent/lib/bootstrap/solaris/x86 for Solaris Intel and

#$LD_LIBRARY_PATH:../nanoagent/lib/bootstrap/solaris/sparc for Solaris Sparc.

JAVA_OPTS="$JAVA_OPTS $AW_OPTS -DAP_NANO_HOME=../nanoagent
-Daspectwerkz.classloader.preprocessor=com.amberpoint.nanoagent.plugins.APAspec
tPreProcessor"
```

6. Restart your JBoss server.

13

Installing Observer Libraries for WCF

This help topic provides instructions for installing and uninstalling observer libraries for monitoring WCF 3.5 services. The observer requires that you have the .NET 3.5 Framework installed on your machine. This observer monitors only IIS 7.5-hosted, WCF Services.

13.1 The Observer Distribution File

The Business Transaction Management observers are distributed by way of ZIP files. Each ZIP file contains one type of observer that is suitable for installation into a particular application server. The ZIP file containing the observer for monitoring WCF 3.5 services is BTMObserver_Iis_6-7_Wcf35_*.zip.

Notes: In the complete ZIP file name, the asterisk (*) is replaced with the observer version number.

In spite of the ZIP file name, this observer is not supported on IIS 6.

13.2 Installing the Observer Libraries for WCF 3.5

1. Locate the observer distribution ZIP file for WCF (BTMObserver_Iis_6-7_Wcf35_*.zip).
2. Unpack the ZIP file into a temporary directory (referred to henceforth as observer_temp).

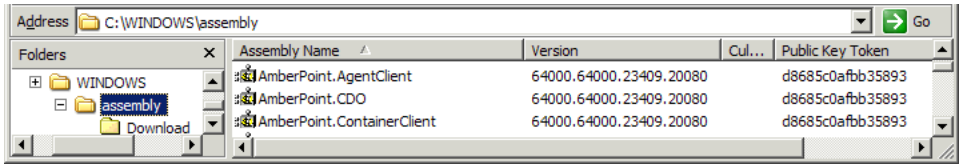
   Unpacking the ZIP file creates a nanoagent directory containing two subdirectories—config and lib.
3. Use gacutil.exe to copy all of the DLL files from observer_temp\nanoagent\lib to the global application cache (GAC; normally located at C:\WINDOWS\assembly).
4. Configure the observer into your applications by editing the application configuration file as follows:

   Caution: Make a backup copy before editing your application configuration file.

   To monitor all WCF applications running on the machine, edit the machine.config file (see Section 13.3).

   To monitor one or more specific web applications, edit the web.config file appropriate for each web application (see Section 13.4).

5. Ensure that the version numbers of the AmberPoint assemblies in the GAC match the version number in your application configuration file, for example:




```
<soapExtensionTypes>
  <add type="AmberPoint.NanoAgent.DotNet.AspNet.Handlers.SoapExtensionHandler,
   AmberPoint.NanoAgentToolkit, Version=64000.64000.23409.20080,
   Culture=neutral,
   PublicKeyToken=d8685c0afbb35893" priority="1" group="0" />
</soapExtensionTypes>
```

   The version numbers must match for the assemblies to be found.

6. In your application configuration file, set the value of the AmberPoint:NanoConfigUrl key to the URL of the monitor to which you want the observer to forward messages.

   Use the following form, replacing *Host:Port* with the host name and port number of the monitor:

```
<add key="AmberPoint:NanoConfigUrl"
value="http://Host:Port/btmmonitor/agent/agent"/>
```

   This key associates the observer with the monitor whose URL you specify in the key. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor.

   As an alternative to setting the monitor URL in the AmberPoint:NanoConfigUrl key, you can create an environment variable named AP_NANO_CONFIG_URL and use it to set the monitor URL.

   Notes: If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

```
http://10.147.46.152:5072/btmmonitor/agent/agent/
```

   For more information about the load balancer's HTTP virtual server, see Section 8.4.

7. Also in your application configuration file, set the value of the AmberPoint:NanoLogBaseDir key to the location in which you want error logs created. Specify the location as an absolute path. If you want the directory created in case it doesn't exist, set the value of the AmberPoint:NanoCreateLogBaseDir key to true. For example:

```
<add key="AmberPoint:NanoLogBaseDir" value="C:/Inetpub/AmberPoint"/>
<add key="AmberPoint:NanoCreateLogBaseDir" value="true"/>
```

The AmberPoint:NanoLogBaseDir key does not have a default value. If it is set to null, log files will not be generated.

In order for the observer to generate the log files, ensure that the user under which the observer is running has permission to write to the log directory. The observer runs as the user named NETWORK SERVICE.

For information about configuring error logging, see Chapter 15, "Logging Observer Errors and Debugging Information."

8. Ensure that the monitor to which your observer forwards messages has an Observer Communication policy applied to it.

   For information on applying an Observer Communication policy, see Section 8.5, "Applying an Observer Communication Policy."

13.3 Editing the machine.config File

Caution: Make a backup copy before editing your machine.config file.

To monitor all WCF applications running on the machine, edit the machine.config file. This file is typically located at C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\CONFIG.

An annotated example of a machine.config file is provided at observer_temp\nanoagent\config\Machine.config.part. Use this file as a guide in making your edits.

13.4 Editing the web.config File

Caution: Make a backup copy before editing your web.config file.

You can monitor all web applications or only specific web applications depending on which web.config file you edit. For example, if you want to monitor all the applications installed in the web site, edit the web.config file located in the web site's home directory. If you want to monitor only specific web applications, edit the web.config file located in the directories of those specific applications.

An annotated example of a web.config file is provided at observer_temp\nanoagent\config\Web.config.part. Use this file as a guide in making your edits.

13.5 Uninstalling the Observer Libraries for WCF 3.5

1. Delete the apobserver.configuration file from the location specified in your application configuration file as the value of the AmberPoint:NanoConfig key.

   Your application configuration file is either machine.config or web.config.

2. Revert your application configuration file to the state it was in before you installed the observer.

   The observer distribution file (BTMObserver_Iis_6-7_Wcf35_*.zip) contains annotated examples of application configuration files that explain how to edit an

application configuration file to insert the observer. You can also use these as a guide in reverting your application configuration file.

If you need to revert your machine.config file, use the config\Machine.config.part file as a guide. If you need to revert your web.config file, use the config\Web.config.part file as a guide.

3. Remove the observer DLLs from the GAC (unless they are being used by another observer on the machine).

   The name of each observer DLL begins with the string "AmberPoint". To remove a DLL, right-click it and choose **Uninstall**.

> **Note:** The observer for ASP.NET uses many of the same DLLs as the observer for WCF. If you have the observer for ASP.NET installed on the machine, you must not remove the DLLs that are being used by it.

14

Installing Observer Libraries for ASP.NET

This help topic provides instructions for installing and uninstalling observer libraries for monitoring ASP.NET services hosted in IIS 6.x.

14.1 The Observer Distribution File

The Business Transaction Management observers are distributed by way of ZIP files. Each ZIP file contains one type of observer that is suitable for installation into a particular application server. The ZIP file containing the observer for monitoring ASP.NET services is BTMObserver_Iis_6_Asp_*.zip.

Note: In the complete ZIP file name, the asterisk (*) is replaced with the observer version number.

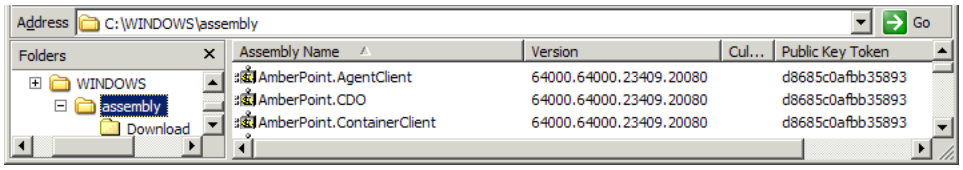
14.2 Installing Observer Libraries for ASP.NET

1. Locate the observer distribution ZIP file for ASP.NET (BTMObserver_Iis_6_Asp_*.zip).
2. Unpack the ZIP file into a temporary directory (referred to henceforth as observer_temp).

   Unpacking the ZIP file creates a nanoagent directory containing two subdirectories—config and lib.
3. Use gacutil.exe to copy all of the DLL files from C:\observer_temp\nanoagent\lib to the global application cache (GAC; normally located at C:\WINDOWS\assembly).
4. Copy C:\observer_temp\nanoagent\config\Web.config to your web site's home directory (the default home directory for the default web site is C:\Inetpub\wwwroot).

   If you already have a Web.config in this directory, copy the contents of the provided Web.config into your existing Web.config. You should also back up your original Web.config in case you decide to uninstall the observer.

   Copying the Web.config (or its contents) into the web site's home directory, enables monitoring on all of the applications installed in the web site. If you want to monitor only specific applications, copy the Web.config (or its contents) into the directories of those specific applications.
5. Open Web.config in a text editor and ensure that the version numbers of the AmberPoint assemblies in the GAC match the version number in Web.config, for example:




```
<system.web>
 <webServices>
  <soapExtensionTypes>
   <add type="AmberPoint.NanoAgent.DotNet.AspNet.Handlers.SoapExtensionHandler,
     AmberPoint.NanoAgentAspNet, Version=64000.64000.23409.20080,
     Culture=neutral, PublicKeyToken=d8685c0afbb35893" priority="1" group="0" />
   </soapExtensionTypes>
  </webServices>
</system.web>
```

The version numbers must match for the assemblies to be found.

6. In Web.config, set the value of the AmberPoint:NanoConfigURL key to the URL of the monitor to which you want the observer to forward messages.

   Use the following form, replacing *Host:Port* with the host name and port number of the monitor:

   ```
   <add key="AmberPoint:NanoConfigURL"
   value="http://Host:Port/btmmonitor/agent/agent"/>
   ```

   This key associates the observer with the monitor whose URL you specify in the key. At startup, the observer retrieves its configuration from the specified monitor and begins sending observations to the monitor.

   As an alternative to setting the monitor URL in the AmberPoint:NanoConfigURL key, you can create an environment variable named AP_NANO_CONFIG_URL and use it to set the monitor URL.

   Notes: If you are using replicated monitors, you must set the host and port portion of the URL to the host and port of your load balancer's HTTP virtual server. For example, if the HTTP virtual server's IP address is 10.147.46.152, and its port number is 5072, then you would set the URL to:

   ```
   http://10.147.46.152:5072/btmmonitor/agent/agent/
   ```

   For more information about the load balancer's HTTP virtual server, see Section 8.4.

7. Also in Web.config, set the value of the AmberPoint:NanoLogBaseDir key to the location in which you want error logs created. Specify the location as an absolute path. If you want the directory created in case it doesn't exist, set the value of the AmberPoint:NanoCreateLogBaseDir key to true. For example:

   ```
   <add key="AmberPoint:NanoLogBaseDir" value="C:/Inetpub/AmberPoint"/>
   <add key="AmberPoint:NanoCreateLogBaseDir" value="true"/>
   ```

   The AmberPoint:NanoLogBaseDir key does not have a default value. If it is set to null, log files will not be generated.

In order for the observer to generate the log files, ensure that the user under which the observer is running has permission to write to the log directory. The observer runs as the user named NETWORK SERVICE.

For more information about configuring error logging, see Chapter 15, "Logging Observer Errors and Debugging Information."

8. Ensure that the monitor to which your observer forwards messages has an Observer Communication policy applied to it.

   For information on applying an Observer Communication policy, see Section 8.5, "Applying an Observer Communication Policy."

14.3 Uninstalling the Observer Libraries for ASP.NET

1. Delete apobserver.configuration from the location specified in Web.config as the value of the AmberPoint:NanoConfig key (this key might use the deprecated name AmberPoint:NanoAgentConfigFile, depending on your version of the observer).
2. Revert your Web.config file to the state it was in before you installed the observer.

   To perform this step, remove all observer-related content from the file. The observer-related content is:

   - The <sectionGroup> element with name AmberPoint
   - The <AmberPoint> element
   - The AmberPoint <add> element within the <soapExtensionTypes> element, for example:

     ```
     <add
     type="AmberPoint.NanoAgent.DotNet.AspNet.Handlers.SoapExtensionHandler,
     AmberPoint.NanoAgentAspNet, Version=64000.64000.23409.20080,
     Culture=neutral, PublicKeyToken=d8685c0afbb35893" priority="1" group="0" />
     ```

   If the Web.config contains only observer-related content, you can delete the entire file.
3. Remove the observer DLLs from the GAC (unless they are being used by another observer on the machine).

   The name of each observer DLL begins with the string "AmberPoint". To remove a DLL, right-click it and choose **Uninstall**.

> **Note:** The observer for WCF uses many of the same DLLs as the observer for ASP.NET. If you have the observer for WCF installed on the machine, you must not remove the DLLs that are being used by it.

15

Logging Observer Errors and Debugging Information

The observer writes error and debugging information to the following log files:

NanoAgentStartupErrors.log – contains configuration-related errors. This file is recreated on each restart of the server. The file will be empty if no errors were encountered. It's maximum size is 5 MB.

NanoAgent.log – contains runtime error and debugging information (you can adjust this logger's settings using the Enable trace logging option in the Observer Communication policy.)

AWTrace.log – contains trace-level messages. This file is created only for aspect-based observer probes and is recreated on each restart of the server. (All probes are aspect-based except for the JAX-RPC probe) The maximum size of this log file is 10 MB.

> **Note:** You can also configure the observer to log observed messages. For information on this topic, refer to the online help for information about the **Log Observed Messages to File** field in the Observer Communication policy.

The default location of the log files is:

- the JBOSS_HOME/bin directory of a JBoss server
- the domain directory of a WebLogic server
- the profile directory of a WebSphere server
- c:/temp/NanoAgentBaseDir for WCF and ASP.NET

> **Note:** The default log location for WCF and ASP.NET is not a true default. It is simply the default setting of the AmberPoint:NanoLogBaseDir key. If you set this key to null, log files will not be created.

If you want the log files generated in a different directory, set the AP_NANO_LOG_BASEDIR Java property or AmberPoint:NanoLogBaseDir Windows key. For Java application servers, you can set the property to either an absolute path or a path that is relative to the default log directory. For WCF and ASP.NET, you must set the key to an absolute path. The following examples illustrate how to set this property or key:

- On JBoss, edit your server startup script JBOSS_HOME/bin/run. In the options section of the file, add **set JAVA_OPTS=-DAP_NANO_LOG_BASEDIR="my_log_dir".** This relative path would generate the log files in the directory JBOSS_HOME/bin/my_log_dir.
- On WebLogic, if you configure your server by editing local scripts, edit the nanoEnvWeblogic script located in WL_HOME/nanoagent/bin directory. In the options section of the file, add **-DAP_NANO_LOG_BASEDIR="my_log_dir"** to the end of the NANOAGENT_JAVA_OPTIONS. This relative path would generate the log files in the directory my_log_dir under your domain directory.

  If you configure you WebLogic server using the Node Manager, open the WebLogic Administration Console, select your server, and display the **Configuration / Server Start** tab. Then add **-DAP_NANO_LOG_BASEDIR=my_log_dir** to the **Arguments** field. This relative path would generate the log files in the directory my_log_dir under your domain directory.
- On WebSphere, in the WebSphere Administrative Console, navigate to Servers > Application servers > server1 > Server Infrastructure > Java and Process Management > Process Definition > Java Virtual Machine > Custom Properties (you might have to substitute a different server name for server1). Create a custom property named **AP_NANO_LOG_BASEDIR** and set it's value to **my_log_dir**. This relative path would generate the log files in the directory my_log_dir under your profile directory.
- For WCF or ASP.NET, edit the application configuration file (for example, Web.config) and set the value for the AmberPoint:NanoLogBaseDir key to **C:/Inetpub/wwwroot/my_log_dir**. This absolute path would generate the log files in the directory my_log_dir under your default web site directory, for example:

```
<configuration>
 <configSections>
  ...
 </configSections>
 <AmberPoint>
  <NanoAgentDataSection>
   <add key="AmberPoint:NanoConfig"
value="c:/temp/NanoAgentLogBaseDir/nanoagentDiscovery.CONFIGURATION"/>
   <add key="AmberPoint:NanoLogBaseDir" value="c:/Inetpub/wwwroot/my_log_dir"/>
   <add key="AmberPoint:NanoCreateLogBaseDir" value="false"/>
  </NanoAgentDataSection>
 </AmberPoint>
 <system.web>
  ...
 </system.web>
</configuration>
```

In order for the observer to generate the log files, ensure that the user under which the observer is running has permission to write to the log directory. For Java observers, the user is the user that is running the application server. For IIS observers (WCF and ASP.NET), the user is as follows:

- IIS 5.x – the observer user is ASPNET
- IIS 6.x and 7.x – the observer user is NETWORK SERVICE

By default, the directory specified by the AP_NANO_LOG_BASEDIR property is automatically created if it does not exist. If you do not want this directory to be automatically created, set the property AP_NANO_CREATE_LOG_BASEDIR to **false**. In this case, you must create the directory yourself. Set this property in the same way you set AP_NANO_LOG_BASEDIR.

Notes: *For Java application servers* – If the log directory does not exist and AP_NANO_CREATE_LOG_BASEDIR is set to false, runtime errors might occur and the observer might not initialize.

For IIS – If the NanoAgentLogBaseDir Windows key set to null, log files are not created.

16

Scripted Configuration of Oracle Business Transaction Management

When you configure Business Transaction Management for the first time, we recommend that you use the browser-based Configuration Wizard. For users who want to later execute various configuration tasks from the command line, a command line interface (CLI) is provided.

The CLI provides command-line equivalents to many configuration tasks. The CLI command relevant to this document is the configure command. This command provides an alternative to the wizard approach to configuration.

You can find complete documentation for the configure command, as well as the entire CLI, in the Business Transaction Management online help.

16.1 The configure Command

The configure command takes an XML configuration file as an argument. This configuration file specifies all the setup information for Business Transaction Management, including database type and connection information, deployment credentials, and so on.

You can develop this configuration input file by using the generated configuration file that is output when you perform initial configuration of Business Transaction Management using the browser-based Configuration Wizard. The wizard produces a configuration file that contains all the configuration information (sphere URL, database connection information, performance server location, authentication credentials, and so on). You can edit this configuration file and use it as input to the configure command. The generated configuration file is named essentialConfiguration.xml.

On WebLogic servers, essentialConfiguration.xml is located inside the WebLogic installation directory at:

`user_projects/domains/`*`MyDomain`*`/servers/`*`MyServer`*`/btmstorage/btmMain /globalPreferences`

On WebSphere servers, essentialConfiguration.xml is located inside the WebSphere installation directory at:

`profiles/`*`MyProfile`*`/btmstorage/`*`MyNode`*`/`*`MyServer`*`/btmMain/globalPreferences`

16.2 Invoking the CLI

The CLI executable is located in the *Install_Dir*/tools directory—btmcli.bat for Windows and btmcli.sh for Unix-like systems.

Command syntax for use with the configure command:

```
btmcli configure -i inputFile -s sphereUrl -l username:password
```

You can avoid placing the username and password on the command line by setting the AP_USER_LOGIN environment variable before executing the CLI, for example:

```
set AP_USER_LOGIN=MyUsername:MyPassword
```

17

The datastoreUtil Utility

Section 5.4, "Setting up Business Transaction Management Databases," instructed you to create database users for the sphere, performance, and transaction databases using the following suggested names:

- **sphereDB** (for the sphere database)
- **measurementDB** (for the performance database)
- **transactionDB** (for the transaction database)

This chapter assumes you are using the suggested names. If not, substitute your names as appropriate.

When you configure Business Transaction Management (see section 6.5, "Initial Configuration of Business Transaction Management"), the system automatically creates the appropriate database tables for these users, unless you choose to create them beforehand with the datastoreUtil Utility.

The datastoreUtil utility enables you to generate the DDL that you can use as input (with sqlplus) to create the necessary tables and views for Business Transaction Management. You can alternatively use datastoreUtil to connect to a database instance and create the tables and views directly.

17.1 Usage

To invoke the datastoreUtil utility, use a command window or shell to navigate to the tools directory of your Business Transaction Management installation and execute the datastoreUtil script that is appropriate for your operating system (either datastoreUtil.bat or datastoreUtil.sh).

After starting the utility, you can run any of the commands listed in Section 17.2, "Commands." For commands that have multiple arguments, you must call the arguments in the order described.

17.2 Commands

The datastoreUtil utility provides the following commands:

- generateSchema (or generate)
- connect
- createSchema (or create)
- close
- exit

- help

17.2.1 generateSchema (or generate)

Generates a DDL of the specified schema definition. You can use this command without being connected to a database.

```
generateSchema schemaType databaseType [[directory] targetSchema]
-partition|-nopartition
```

- *schemaType* – Specify one of the following schema types:
  - **sphere** – Creates a schema for the sphere database (the sphereDB user).
  - **exm** – Creates a schema for the transaction database (the transactionDB user).
  - **performance** – Creates a schema for the performance database (the measurementDB user).
  - **monitorgroup** – Creates a schema for a monitor group.
  - **msglog** – Creates a schema for the system message log.
- *databaseType* – Specify **oracle**. This is the only supported value.
- *directory* – Specify a location to generate the DDL file (defaults to the local directory).
- *targetSchema* – Optional. This argument scopes the generated schema to a specific user, for example, **sphereDB**, **transactionDB**, or **measurementDB**.
- **-partition** or **-nopartition** – This flag is required if your specified *schemaType* is **performance** or **monitorgroup**. If your specified *schemaType* is any other value, this flag is not required and is ignored if you use it.

  If you are using Oracle Enterprise Edition, you can create a performance or monitorgroup schema that takes advantage of Oracle's partitioning feature by specifying the **-partition** flag. If you do not want to take advantage of this feature, or if your Oracle edition does not provide the partitioning feature, you must specify the **-nopartition** flag (if you are creating a performance or monitorgroup schema).

17.2.2 connect

Connect to a database using the user-specified connection information.

```
connect databaseType|filename
```

- *databaseType* – Specify **oracle**. This is the only supported value.
- *filename* – Specify a file output by the **saveConnection** command.

Use the **connect** command to enter database connection information and connect to the database. You must have the following information for the database to which you want to connect:

- driver name
- username
- password
- URL connection string

Once connected, you might issue the **saveConnection** command to save the connection information within a file. The next time you want to connect to the same database, you can provide the file name with the connect command. If you provide the database type, the utility automatically selects the corresponding default driver.

17.2.3 createSchema (or create)

Create the specified schema within the connected database.

```
createSchema schemaType -partition|-nopartition
```

- *schemaType* – Specify one of the following schema types:
  - **sphere** – Creates a schema for the sphere database (the sphereDB user).
  - **exm** – Creates a schema for the transaction database (the transactionDB user).
  - **performance** – Creates a schema for the performance database (the measurementDB user).
  - **monitorgroup** – Creates a schema for a monitor group.
  - **msglog** – Creates a schema for the system message log.
- **-partition** or **-nopartition** – This flag is required if your specified *schemaType* is **performance** or **monitorgroup**. If your specified *schemaType* is any other value, this flag is not required and is ignored if you use it.

  If you are using Oracle Enterprise Edition, you can create a performance or monitorgroup schema that takes advantage of Oracle's partitioning feature by specifying the **-partition** flag. If you do not want to take advantage of this feature, or if your Oracle edition does not provide the partitioning feature, you must specify the **-nopartition** flag (if you are creating a performance or monitorgroup schema).

17.2.4 close

Close a connection previously opened through the connect command.

```
close
```

17.2.5 exit

Exit the utility.

```
exit
```

17.2.6 help

Use the help command to view help for all commands, or enter a command name after help to receive help for a single command.

```
help | help command
```