TEAMCENTER

Web Application Deployment

Teamcenter 2412

SIEMENS

About Siemens Digital Industries Software

Siemens Digital Industries Software is a global leader in the growing field of product lifecycle management (PLM), manufacturing operations management (MOM), and electronic design automation (EDA) software, hardware, and services. Siemens works with more than 100,000 customers, leading the digitalization of their planning and manufacturing processes. At Siemens Digital Industries Software, we blur the boundaries between industry domains by integrating the virtual and physical, hardware and software, design and manufacturing worlds. With the rapid pace of innovation, digitalization is no longer tomorrow’s idea. We take what the future promises tomorrow and make it real for our customers today. Where today meets tomorrow. Our culture encourages creativity, welcomes fresh thinking and focuses on growth, so our people, our business, and our customers can achieve their full potential.

Support Center: **support.sw.siemens.com**

Send Feedback on Documentation: **support.sw.siemens.com/doc_feedback_form**

Contents

1. Getting started deploying web applications

Deployment considerations

Deployment of your Teamcenter web applications is an important step in setting up your Teamcenter environment. How you deploy the web application is determined by how you intend to use Teamcenter and can affect the application's performance.

1. Consider the high-level **requirements** of your deployment.

2. Review the different supported **deployment configurations** to determine which is best for your enterprise.

3. Determine your application server. The application server you use may impact your **deployment configuration**. Not all configurations are supported for all application servers. Global Services web applications are supported for basic deployments only.

   For the information about the versions of application servers certified for your platform, see the Hardware and Software Certifications knowledge base article on Support Center.

   Siemens Digital Industries Software certifies third-party software applications with the latest patches available when the certification testing is performed. If you encounter problems deploying a Teamcenter web application, ensure that you have installed the latest patches for your application server.

For versions of web applications supported by Teamcenter, see the Hardware and Software Certifications knowledge base article on Support Center.

Support for IPv6 requires a dual stack application server host and a dual stack Teamcenter server host. See **https://www.cisco.com** for information about IPv6 and dual stack networks. Information about supporting IPv6 and dual stack networks on your application server host can be found in your Windows or Linux server documentation.

Requirements for deploying web applications

Prerequisites	You must have administrator privileges to use the application servers administration tools. You must have performed web application installation

	as described in the appropriate Teamcenter server installation guides (for Windows or Linux).
Enable a web application	The web tier application is enabled by deploying it in the application server and, depending on your configuration, its associated proxy component in the web server.
Configure a web application	Teamcenter web applications are configured during installation and in the application server after deployment.
Start a Teamcenter web application	Once your Teamcenter web application is deployed, it is running. If you need to stop, start, or restart the application at a later time, you must use the application server administration tools to perform these actions.

Determining your requirements

How you configure your servers that run your Teamcenter web tier application depends on your enterprise requirements for scalability (concurrent users and processes) and data availability (server fail over). An HTTP front-end cluster provides better performance for static web content. Clustering application servers provides better performance for dynamic content and ensures availability because the Teamcenter application has multiple instances that allow a particular application server to fail without causing the Teamcenter data to be inaccessible.

To determine the best configuration for your installation you must be familiar with the installation, use, and performance tuning of the servers you choose for deploying the web tier application.

For information about server performance, see the documentation provided with your server.

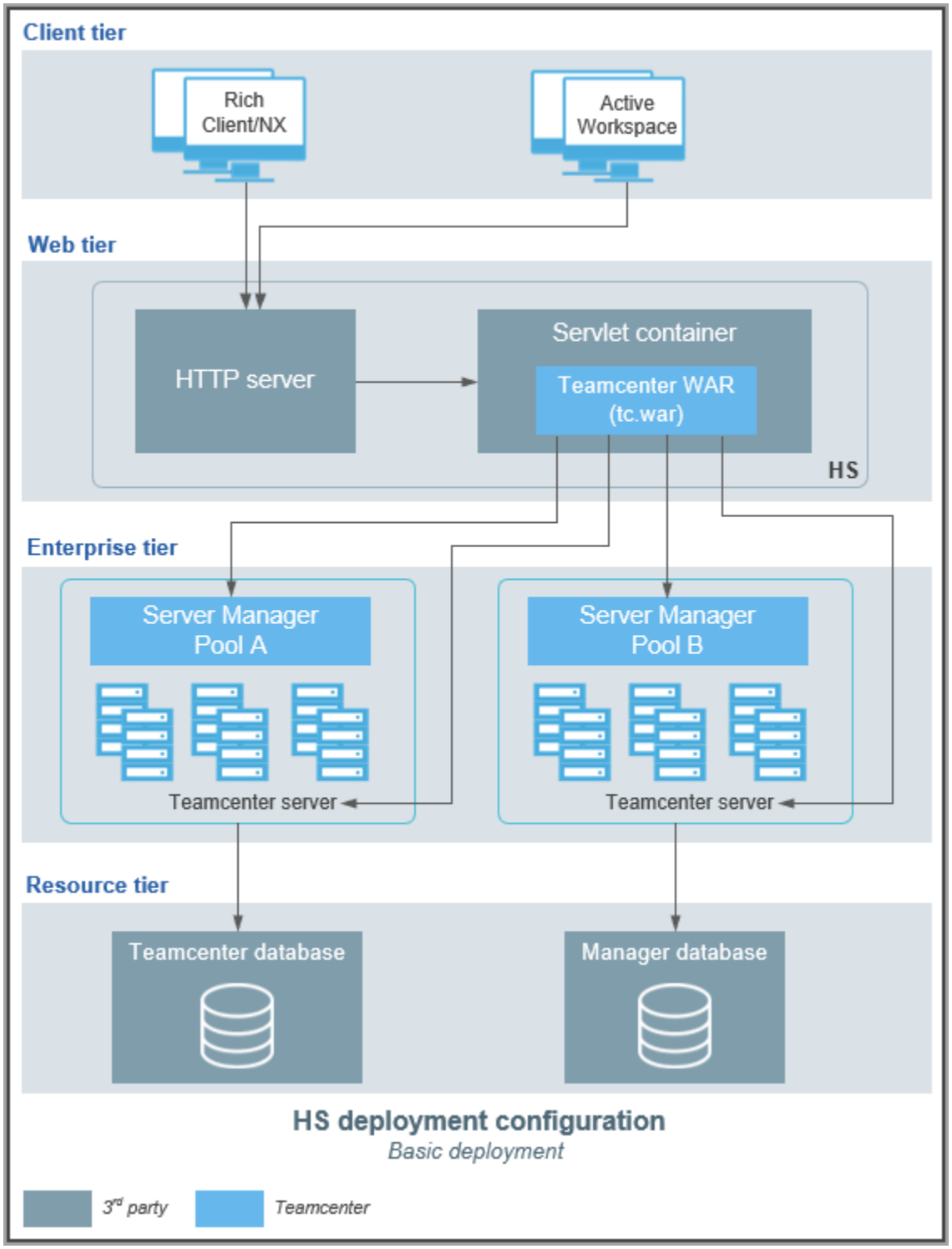
Basic concepts of Teamcenter web application deployment

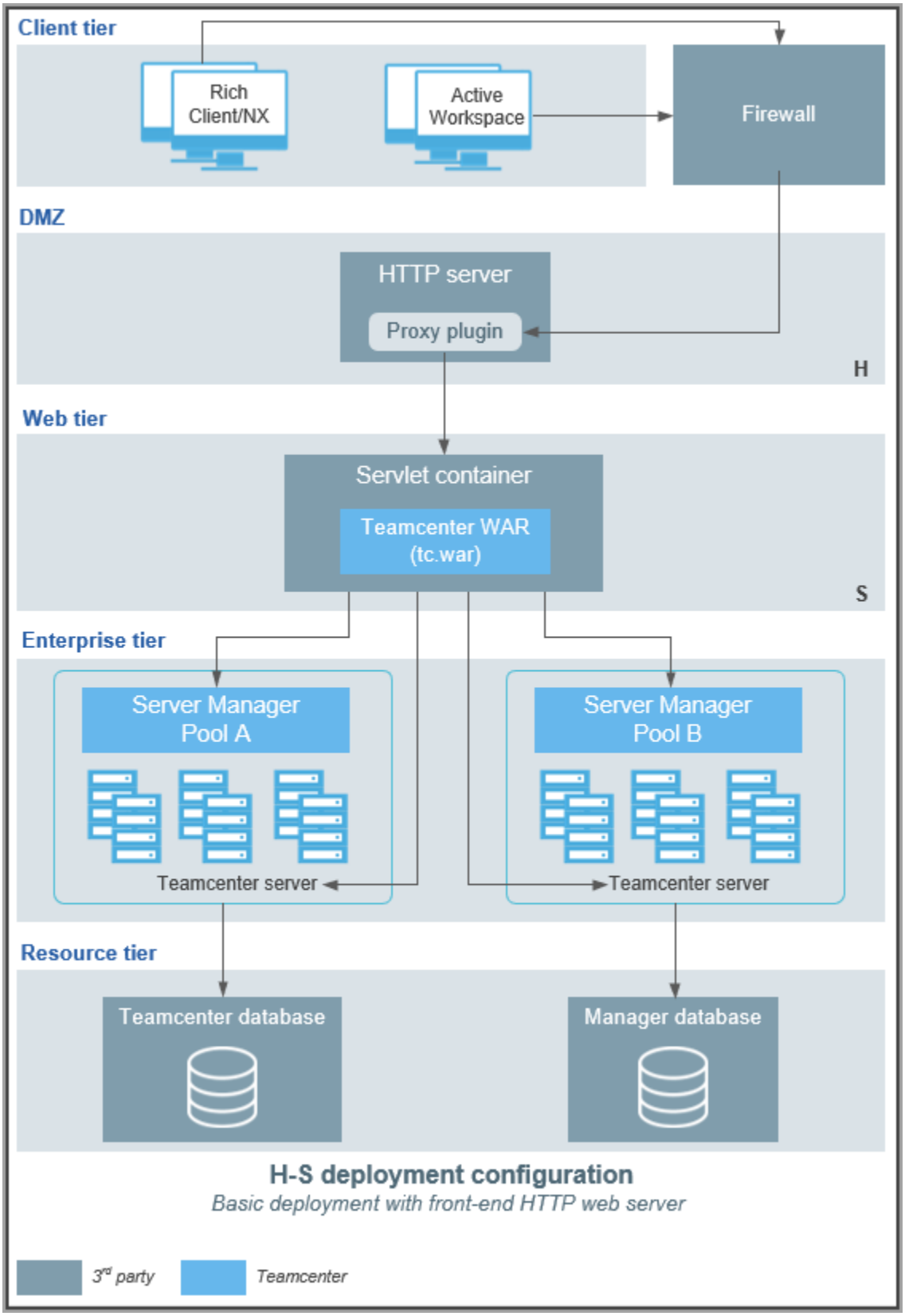
You should understand the following terms.

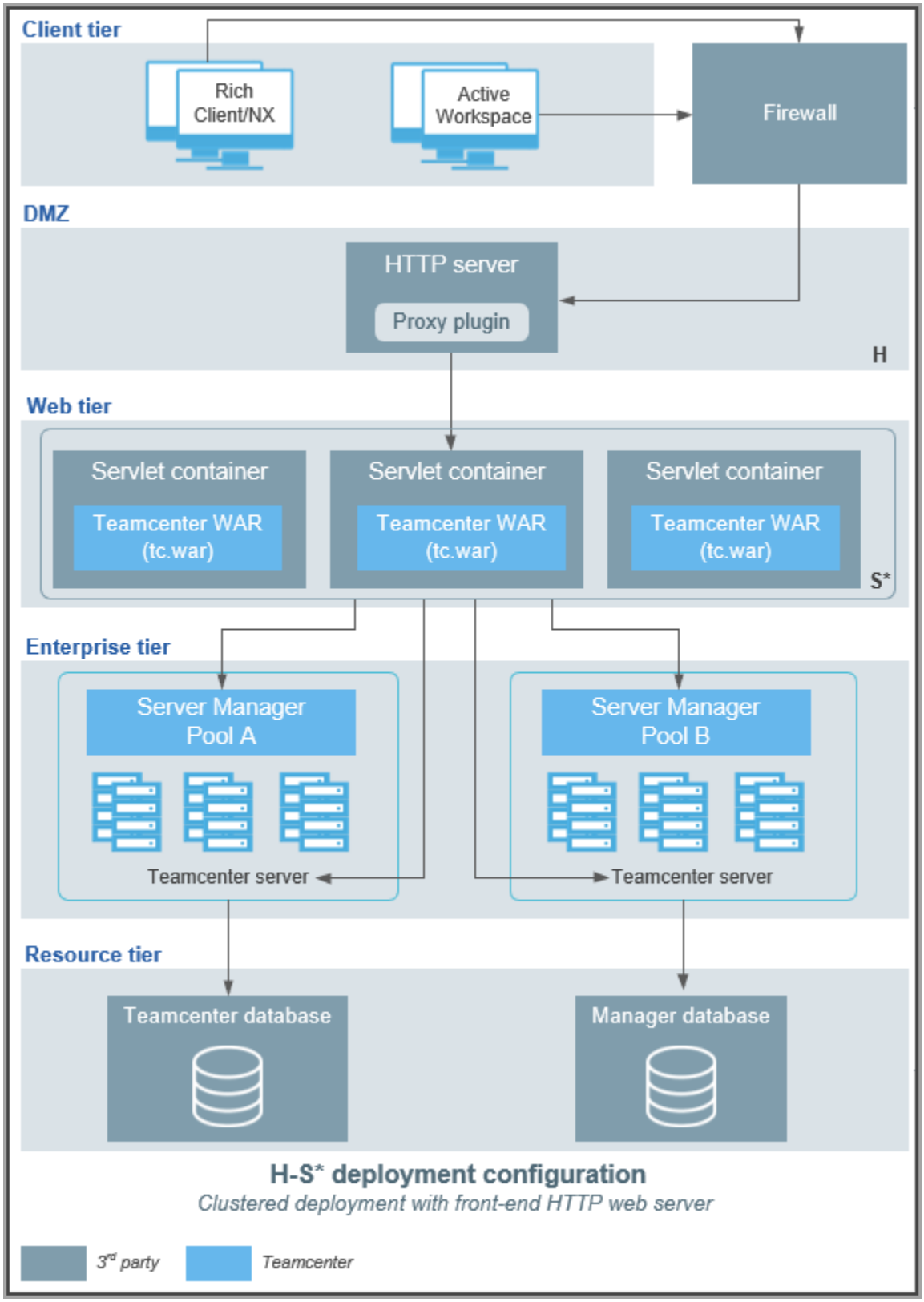
Term	Definition
Basic deployment (HS)	Basic deployment on an enterprise (Java EE) application server. The HTTP web server (H) and servlet container (S) are provided on the same platform as part of the same process. The Teamcenter web tier application (WAR file) is deployed on a Java EE application server that has a built-in HTTP listener, such as JBoss (WildFly) Application Server, and IBM WebSphere Application Server.

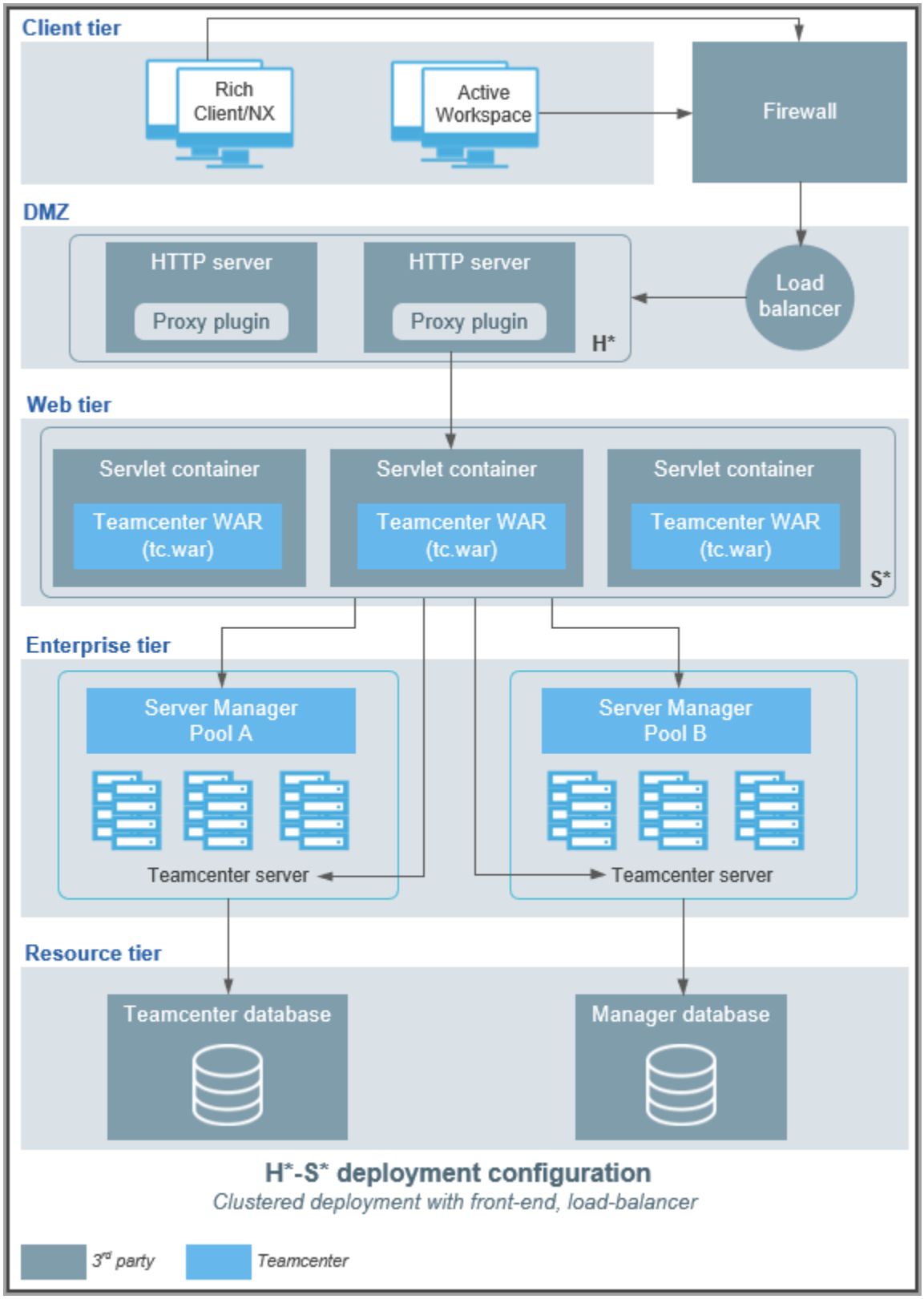
Term	Definition
	Deploying a separate HTTP web server to listen to the incoming request is not required.
Basic deployment with front-end HTTP web server (H-S)	A stand-alone HTTP web server is configured as the front-end to a Java EE application server.
Clustered deployment with front-end HTTP web server (H-S*)	A stand-alone HTTP web server is configured with a cluster of web application server instances. The HTTP web server routes requests to a cluster of Java web application servers. The Teamcenter web tier application (WAR file) is deployed in each application server instance in the cluster.
Clustered deployment with front-end, load-balancer (LB*-HS*)	A load balancer configured with a cluster of Java web application server instances. A load balancer balances the load for incoming requests and routes the request to the cluster of Web application servers. In this configuration, the Teamcenter web tier application (WAR file) is deployed in each application server instance in the cluster.
Network load balancing (NLB)	HTTP web servers are configured to allow each HTTP web server in the load balanced cluster (see web server farm) to respond to a virtual IP address. Requests to this virtual IP are intercepted and routed to a machine running one of the web servers in the cluster.
Web archive (WAR)	A web application that requires an HTTP web server and servlet engine.
Web server farm	Multiple HTTP web servers are configured as self contained (redundant) servers in a cluster. The web servers serve a single IP address that allows any of the servers that are available to handle a request to the address. This provides improved performance and reliability.

The following figures show each of the deployment configurations for Teamcenter web tier applications.









2. Teamcenter web application deployment

Introduction to Teamcenter web application deployment

All of the deployment procedures assume that you have installed your application server per the instructions provided with the application server and that you have created the required Teamcenter and Security Services web applications (WAR files) as described in the appropriate Teamcenter installation guide for Deployment Center or TEM (Windows or Linux).

For web tier deployment recommendations and sample deployments, see *Teamcenter Deployment Reference Architecture* in the Teamcenter downloads area on Support Center, under **Support White Papers Teamcenter Deployment Reference Architecture**.

Basic deployment

Introduction to basic deployment

The following example instructions for basic deployment of the Teamcenter web tier application (WAR file) on selected Java application servers are for general reference only. Please refer to vendor documentation for instructions on the exact version used in your deployment. For information about versions of operating systems, third-party software, Teamcenter software, and system hardware certified for your platform, see the Hardware and Software Certifications knowledge base article on Support Center.

- **JBoss EAP/WildFly**
- **Tomcat**
- **WebSphere**

Instructions for enabling secure socket layer (SSL) on an application server are provided in the application server documentation.

Deploy on a JBoss EAP (WildFly) application server (HSE)

The following procedure assumes that you have JBoss 7.4.0 installed, and that you are using the stand-alone server location for deploying your Teamcenter web application.

> Caution:
>
> Certain versions of JBoss (WildFly) configure the Java virtual machine (JVM) to prefer the IPv4 stack. This can cause **socket errors** when the server manager starts due to a mismatch in protocols between the web tier and server manager hosts.

1. Copy the Teamcenter WAR (by default, **tc.war**) file to the following directory:

*jboss-EAP-7.4.0-install-root-folder***standalone\deployments**

2. Define JMX as a global module. JMX enables monitoring the web tier using the Teamcenter management console.

   a. Expand the **configuration** directory:

      *jboss-EAP-7.4.0-install-root-folder***standalone\configuration**

   b. Open the **standalone.xml** file in and editing application.

   c. Locate the **subsystem** element for the **urn:jboss:domain:ee** subsystem, and add the following **global-modules** element content:

```
<subsystem xmlns="urn:jboss:domain:ee:4.0">
  <global-modules>
    <module name="org.jboss.as.jmx" slot="main"/>
  </global-modules>
...
</subsystem>
```

      Locate the **deployment-scanner** element and add the **deployment-timeout** attribute with a value of **600** as follows:

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:2.0">
    <deployment-scanner path="deployments"
         relative-to="jboss.server.base.dir"
         scan-interval="5000"
         deployment-timeout="600"
         runtime-failure-causes-rollback=
         "$
{jboss.deployment.scanner.rollback.on.failure:false}”/>
</subsystem>
```

   d. To make the service public in your environment (IPv4 and IPv6), locate the **interface** element for the **public** interface and modify its contents as follows:

```
<interface name="public">
  <any-address/>
</interface>
```

3. Update the logging format to match Teamcenter's log format. (This is required when using a log aggregating system.)

```
<subsystem xmlns="urn:jboss:domain:logging:8.0">
  <!-- Add these elements to remove extra formatting of application
(Teamcenter) log statements that are directed to stdout -->
```

```
  <console-handler name="STDOUT-CONSOLE" autoflush="true">
    <level name="ALL"/>
    <formatter>
      <pattern-formatter pattern="%s%n"/>
    </formatter>
  </console-handler>
  <logger category="stdout" use-parent-handlers="false">
    <handlers>
      <handler name="STDOUT-CONSOLE"/>
    </handlers>
  </logger>

  <!-- Modify these elements to match Teamcenter's log format -->
  <formatter name="PATTERN">
    <pattern-formatter pattern="%z{UTC}%d{YYYY/MM/dd-HH:mm:ss.SSS} UTC
- %-5p - - - Wildfly   - - %s%e%n"/>
  </formatter>
  <formatter name="COLOR-PATTERN">
    <pattern-formatter pattern="%K%z{UTC}%d{YYYY/MM/dd-HH:mm:ss.SSS}
UTC - %-5p - - - Wildfly   - - %s%e%n"/>
  </formatter>
```

4. Define a dependency to allow the JBoss (WildFly) connector module to use JMX MBeans.

   a. Expand the **main** directory:

      *jboss-EAP-7.4.0-install-root-folder***modules\system\layers\base\org\jboss\as\connector\main**

   b. Open the **module.xml** file.

   c. Locate the **dependencies** element, and add the following **module** element:

      ```
      <module name="org.jboss.as.jmx"/>
      ```

5. To allow the Teamcenter web application to listen to nonloopback addresses, configure JBoss (WildFly) using the information in the JBoss (WildFly) documentation available at **https://docs.jboss.org**. Specifically, see the *Command line parameters* and *Interfaces and ports* sections of that documentation.

6. If you require IPv6 support, open the **standalone_conf** script file in your JBoss (WildFly) installation **bin** directory and add the following settings:

   ```
   -Djava.net.preferIPv4Stack=false
   -Djava.net.preferIPv6Addresses=false
   ```

7. If you use Requirements Management, open the **standalone.xml** file and remove the following lines if they are present:

```
<locking isolation="REPEATABLE_READ"/>
<transaction mode="BATCH"/>
```

Doing so ensures that Requirements Management microservices functions correctly.

8. Applications servers often limit the size of an HTTP POST (all service requests). This may result in service requests failing to be sent to the Teamcenter server.

   For JBoss and Wildfly, you can adjust this limit by adding or modifying the **max-post-size** attribute in the *standalone.xml* file as follows:

```
<subsystem xmlns="urn:jboss:domain:undertow:12.0" default-server="default-server"
   default-virtual-host="default-host" default-servlet-container="default"
   default-security-domain="other" statistics-enabled=
   "${wildfly.undertow.statistics-enabled:${wildfly.statistics-enabled:false}}">
     <buffer-cache name="default"/>
     <server name="default-server">
        <http-listener name="default" max-post-size="524288000"
           socket-binding="http" redirect-socket="https" enable-http2="true"/>
        <https-listener name="https" socket-binding="https"
           security-realm="ApplicationRealm" enable-http2="true"/>
        <host name="default-host" alias="localhost">
           <location name="/" handler="welcome-content"/>
           <http-invoker security-realm="ApplicationRealm"/>
        </host>
     </server>
   ...
```

9. Open a command shell and ensure you have defined the **JAVA_HOME** environment variable, and set it to the location of your Java installation.

10. Start the server by typing **standalone** (**standalone.sh** on Linux) **-b** *host-name* in the command shell.

    You must start the application server instance with the bind option to enable connections from clients running on a host different from the application server host. The simplest way to do this is to start the server with the **-b** *host-name* option. Substitute the host name or IP address of the local host for *host-name*. However, this has some security implications.

    For information about JBoss (WildFly) security, see the documentation available at **https://developer.jboss.org**.

If the web tier encounters errors obtaining JCA connections under peak activity, increase the **Max_Capacity** context parameter value for your Teamcenter web application.

Deploy on a Tomcat application server

1. Update the logging format to match Teamcenter's log format. (This is required when using a log aggregating system.) Update the file *tomcat-root/conf/logging.properties* as follows:

```
java.util.logging.ConsoleHandler.formatter
=java.util.logging.SimpleFormatter
java.util.logging.SimpleFormatter.format=%1$tY/%1$tm/%1$td-
%1$tH:%1$tM:%1$tS.%1$tL %1$tZ - %4$-5s - - - Tomcat    - - %5$s%6$s%n

The time stamps in logging files
are in local time, to change to UTC time update the file
<tomcat-root>/bin/catalina.bat and/or <tomcat-root>/bin/catalina.sh:

catalina.bat

set "JAVA_OPTS=%JAVA_OPTS% -Duser.timezone=UTC"

catalina.sh

JAVA_OPTS=$JAVA_OPTS -Duser.timezone=UTC
```

2. Deploy Tomcat using one of the following methods.

   Ensure the Tomcat service logs on as "Local System" or as a local or domain user account.

Autodeploy

1. Copy the Teamcenter WAR (by default, **tc.war**) file to the following directory:

   apache-tomcat-*version***\webapps**

2. Start Tomcat by running the following command:

   *installed-location***\apache-tomcat-***version***\bin\startup.bat**

   When you start Tomcat, Teamcenter is loaded at the default Tomcat port (**8080**).

3. To verify that the WAR file is loaded, open the following test URL:

   http://*host-name***:8080/tc/controller/test**

Deploy from the application manager page

1. Start Tomcat by running the following command:

   *installed-location***\apache-tomcat-***version***\bin\startup.bat**

2. Click the **Manager App** button and log on to the **Tomcat Web Application Manager** page.

Tip:

To set the manager user name and password, add the **manager-gui** role to the following fie:

*installed-location***apache-tomcat-******conf\tomcat-users.xml**

For example, to make the manager user name **TomcatMgr** and the manager password **T0mc4tM4n4g3r**, add the following to the file:

```
<role rolename="manager-gui"/>
<user username="TomcatMgr" password="T0mc4tM4n4g3r"
roles="manager-gui"/>
```

3. Click the **Browse** button in the **Select WAR file to upload** box and select the Teamcenter WAR file (by default, **tc.war**).

4. Click the **Deploy** button.

   The Teamcenter application (for example, **tc**) is deployed and is displayed in the list of deployed applications.

5. To verify that the WAR file is loaded, open the following test URL:

   http://*hostname***:8080/tc/controller/test**

Deployment on a WebSphere application server

Deploy on a WebSphere application server (HS)

This procedure deploys one instance of WebSphere Application Server hosting the Teamcenter web tier application (WAR file).

Please refer to the complete WebSphere documentation for your WebSphere version available at **https://www.ibm.com**.

1. Install the WebSphere application server by itself on a single machine. This enables the internal HTTP transport train suitable for handling a low level of web requests.

2. Start the WebSphere integrated solutions console.

3. In the navigation tree, expand **Applications** and click **Install New Application**.

4. In the **Preparing for the application installation** pane, type the path to, or browse to, the location of the Teamcenter web tier WAR file in the **Full path** box.

Select **Prompt me only when additional information is required** and click **Next**.

5. Accept the default **Select installed options** for enterprise applications and modules and click **Next**.

6. In the **Map modules to servers** pane, if you have multiple server instances, select the check boxes for all modules and map them to the same server instance. Click **Next** again.

7. In the summary pane, click **Finish**. Wait for WebSphere to complete the application deployment.

8. Click **Apply**, scroll to the top of the page, and click **Save**.

   Your application is now deployed and can be started.

9. In the **Enterprise Applications** pane, select the Teamcenter web application check box and click **Start.**

If you deploy a web application that contains the **Teamcenter - Online Help** solution, set the context root for the web application in WebSphere to the enterprise tier ID for the web application. This is the value of the **Deployable File Name** context parameter or the **Enterprise Application Lookup ID** context parameter. Make sure you include the file name (*file-name***.war**) when specifying the context root.

Provide HTTP session isolation for multiple applications

If you deploy multiple applications in the same application server instance, HTTP session cookies may be overwritten by browsers connecting to different applications. To avoid this, configure the application server to provide separate cookie paths:

1. Log on to the Integrated Solution Console, expand **Applications** in the navigation tree, and click **Enterprise Applications**.

2. In the **Enterprise Applications** pane, click the Teamcenter application link.

3. Click **Session Management** under **Web Modules Properties**.

4. Click **Override session Management** under **General Properties**.

5. Click the **Enable cookies** link and type a slash (**/**) followed by the Teamcenter web application name. For example, if you use the default web application name, type **/tc**.

Basic deployment with front-end HTTP (Web) server

About application servers and HTTP (Web) servers

Each of the supported applications servers can be configured to use a front-end HTTP server. The HTTP servers that you can use vary according to the application server you are using.

Deployment on a JBoss (WildFly) application server with IIS front end (H-S)

Deploying on JBoss (WildFly) application server with IIS front end (H-S)

This procedure:

- Deploys the Teamcenter web tier application (WAR file) on the JBoss (WildFly) Application Server.
- Installs and configures the Tomcat ISAPI Redirector on a Windows Server host.
- Configures the Microsoft Internet Information Services (IIS) as the front-end listener (web server) on a Microsoft Windows Server host or a Windows Server host.

> Note:
>
> As a precondition, the **ISAPI Extensions** feature of the IIS Application must be activated to allow integration with the Tomcat ISAPI redirector.

Deploy on a JBoss (WildFly) application server (H-S)

This procedure assumes that the JBoss 7.4.0 final version is installed, and that you are using the stand-alone server location for deploying your Teamcenter web application.

> Caution:
>
> Please refer to vendor instructions for your specific version of JBoss (WildFly).
>
> Certain versions of JBoss (WildFly) configure the Java virtual machine (JVM) to prefer the IPv4 stack. This can cause **socket errors** when the server manager starts due to a mismatch in protocols between the web tier and server manager hosts.

1. Copy the Teamcenter WAR (by default, **tc.war**) file to the following directory:

   jboss-as-7.4.0.Final\standalone\deployments

2. Define JMX as a global module.

   a. Expand the **configuration** directory:

      jboss-as-7.4.0.Final\standalone\configuration

   b. Open the **standalone.xml** file.

   c. Change the HTTPS protocol to **ALL**. (The default protocol is **TLSv1**.) Locate the **subsystem** element for the **urn:jboss:domain** subsystem, and add the following **connector** element content:

```
<subsystem xmlns="urn:jboss:domain:ee:1.0">
<connector name="https" scheme="https" protocol="HTTP/1.1" socket
-binding="https" enable-lookups="false" secure="true">
      <ssl name="jbossSSL-SSL" password="private" protocol="ALL" key-
      alias="jbossSSL" certificate-key-file="D:\ssl\jbossSSL.keystore" />
</connector></subsystem>
```

d. Locate the **subsystem** element for the **urn:jboss:domain** subsystem, and add the following **global-modules** element content:

```
<subsystem xmlns="urn:jboss:domain:ee:1.0">
  <global-modules>
    <module name="org.jboss.as.jmx" slot="main"/>
   </global-modules>
</subsystem>
```

Locate the **deployment-scanner** element and add the **deployment-timeout** attribute with a value of **600** as follows:

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:1.1">
    <deployment-scanner path="deployments"
                        relative-to="jboss.server.base.dir"
                        scan-interval="5000"
                        deployment-timeout="600"/>
</subsystem>
```

e. If you require IPv6 support, locate the **interface** element for the **public** interface and modify its contents as follows:

```
<interface name="public">
  <any-address/>
</interface>
```

3. Microsoft IIS uses the AJP 1.3 protocol to forward requests to JBoss (WildFly). Perform the following steps to enable the AJP 1.3 protocol:

a. Open the *JBoss-installation***/server/default /deploy/jbossweb.sar/server.xml** file. Add or modify the following **Connector** element:

```
<!-- A AJP 1.3 Connector on port 8009 -->
<Connector protocol="AJP/1.3" port="8009" address="$
{jboss.bind.address}"
       tomcatAuthentication="false" emptySessionPath="true"
enableLookups="false" redirectPort="8443"  />
```

IIS forwards requests to JBoss (WildFly) using the AJP 1.3 protocol on the specified port. This must be set to allow access to the remote user name (**getRemoteUser**) method.

b. Open the *JBoss-installation***/standalone/ /configuration/standalone.xml** file and add the AJP connector as the child resource of the **jboss:domain:web** subsystem:

```
<subsystem xmlns="urn:jboss:domain:web:1.1" default-virtual-server="default-
host" native="false">
    <connector name="http" protocol="HTTP/1.1" scheme="http" socket-
binding="http"/>
    <connector name="ajp13" protocol="AJP/1.3" scheme="http" socket-binding="ajp"/>
    <virtual-server name="default-host" enable-welcome-root="true">
        <alias name="localhost"/>
        <alias name="example.com"/>
    </virtual-server>
</subsystem>
```

c. Set or verify the port for the AJP protocol:

```
<socket-binding name="ajp" port="8009" />
```

If the default port for the AJP 1.3 protocol (**8009**) is not available on your host running JBoss (WildFly), set this value to an available port.

Record the port value for use when you configure the redirector.

> Note:
>
> If Windows Authentication is enabled in IIS 7 (which is a supported use case for Security Services), you cannot use JBoss 7.4 for the Security Services login service.

4. Define a dependency to allow the JBoss (WildFly) connector module to use JMX MBeans.

a. Expand the **main** directory:

jboss-as-7.4.0.Final\modules\org\jboss\as\connector\main

b. Open the **module.xml** file.

c. Locate the **dependencies** element, and add the following **module** element:

```
<module name="org.jboss.as.jmx"/>
```

5. To allow the Teamcenter web application to listen to nonloopback addresses, configure JBoss (WildFly) using the information in the documentation available at **https://docs.jboss.org**. Specifically, see the sections *Command line parameters* and *Interfaces and ports* in that documentation.

6. If you use Requirements Management, open the **standalone.xml** file and remove the following lines if they are present:

```
<locking isolation="REPEATABLE_READ"/>
<transaction mode="BATCH"/>
```

Doing so ensures that Requirements Management microservices functions correctly.

7. If you require IPv6 support, open the **standalone_conf** script file in your JBoss (WildFly) installation **bin** directory and add the following settings:

   ```
   -Djava.net.preferIPv4Stack=false
   -Djava.net.preferIPv6Addresses=false
   ```

8. Open a command shell and ensure you have defined the **JAVA_HOME** environment variable, and set it to the location of your Java installation. The Teamcenter web application requires Java 1.7.

9. Start the server by typing **standalone** (**standalone.sh** on Linux) **-b** *host-name* in the command shell.

   You must start the application server instance with the bind option to enable connections from clients running on a host different from the application server host. The simplest way to do this is to start the server with the **-b** *host-name* option. Substitute the host name or IP address of the local host for *host-name*. However, this has some security implications.

   For information about JBoss (WildFly) security, see the documentation available at **https://developer.jboss.org**.

If the web tier encounters errors obtaining JCA connections under peak activity, increase the **Max_Capacity** context parameter value for your Teamcenter web application.

Install and configure the Tomcat ISAPI Redirector

Please refer to the documentation provided by Apache and Microsoft to configure IIS as a front end HTTP Server. The following reference instructions may not be completely applicable to your specific version.

You must install the Tomcat ISAPI Redirector and configure the Windows registry for the redirector. If you are installing on a Windows Server host, you must **install the redirector**. You must also create the **workers.properties** and **uriworkermap.properties** files for the redirector.

For additional information about the settings in these files, see the Tomcat connectors documentation available at **https://tomcat.apache.org**.

1. Create a directory (for example, **iis75-jboss7**) for the redirector in a location accessible to Microsoft IIS that contains the following directories:

   - **bin**
   - **conf**
   - **log**
   - **wwwroot**

2. Download the 32-bit or 64-bit (as needed for your host) ISAPI Redirector from a mirror site for the Apache Tomcat web site. Only the DLL file (**isapi_redirector–1.2.35.dll** or later version) is required.

   Record the name and location of the Tomcat ISAPI Redirector installation directory for later use.

3. Configure Windows registry settings on the host where IIS and ISAPI Redirector are installed.

   a. In the ISAPI Redirector installation directory, create a file with a **.reg** extension. The name of this file is discretionary (**isapi_redirector.reg** is recommended).

   b. Add the following contents to the **.reg** file:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Apache Software Foundation\
    Jakarta Isapi Redirector\1.0]
"extension_uri"="/jakarta/isapi_redirect.dll"
"log_file"="D:\\iis75–jboss7\\logs\\jk_iis.log"
"log_level"="debug"
"worker_file"="D:\\iis75–jboss7\\workers.properties"
"worker_mount_file"="D:\\iis75–jboss7\\uriworkermap.properties"
"uri_select"="unparsed"
```

   Siemens Digital Industries Software recommends that you use **debug** for the **log_level** entry when you initially configure the redirector to get all messages. You can change this after you have tested your installation and determined that it is working properly. The following table provides a brief description of these entries:

Name	Description
extension_uri	Represents the IIS virtual directory including the ISAPI Redirector file.
log_file	Defines the name and location of the ISAPI Redirector log file.
log_level	Defines the level of debug messages written to the ISAPI Redirector log file. Valid values are **debug**, **info**, **error**, and **emerg**.
worker_file	Defines the location of the ISAPI redirector **worker.properties** file.
worker_mount_file	Defines the location of the ISAPI redirector **uriworkermap.properties** file.

See these registry settings in the *Apache Tomcat Connector – Reference Guide* available at **https://tomcat.apache.org**.

c. Change the following lines in the **.reg** file to reflect your directory settings:

A. For **log_file**, enter the location of the **logs** directory you created and the name of the log file.

The log file itself is created later by the ISAPI Redirector.

B. For **worker_file**, enter a location for the worker definition file. It is recommended that you create this file in the directory where you installed the Tomcat ISAPI Redirector. You create this file later.

C. For the **worker_mount_file**, enter a location for the worker-URI map file. You create this file later.

D. For the **extension_uri**, enter **tomcat**.

d. In the ISAPI Redirector installation directory, right-click the **isapi_redirector.reg** file and choose **Merge**.

e. After receiving a confirmation message from Windows, check the ISAPI Redirector settings using the Microsoft Registry Editor program (**regedit.exe**) to ensure the registry settings are correct. For information about using the Microsoft Registry Editor, see the Microsoft Windows online help.

4. Create a text file with contents similar to the following:

```
# Define node1 (one node required for H_SE)
worker.list=node1
worker.node1.port=8009
worker.node1.host=host-name1
worker.node1.type=ajp13
```

The default port is **8009**. If you changed this AJP port number in JBoss (WildFly) configuration when you configured the Tomcat ISAPI Redirector, use that value. The *host-name* value is the host where you run JBoss (WildFly).

5. Save the file as **workers.properties** in the directory you defined for it in the registry file.

6. Create a text file with contents similar to the following:

```
# Send all /tc requests to node1
/tc/*=node1
```

Replace *tc* with the name of your Teamcenter web application (**tc** by default). This configures the redirector to forward all requests with the **/tc/*** signature to **node1**.

7. Save the file as **uriworkermap.properties**. Save this file in the same directory as the **workers.properties** file.

Install and configure the Tomcat ISAPI Redirector on Windows Server platforms

You must install the Tomcat ISAPI Redirector and configure the Windows registry for the redirector. You must also create the **workers.properties** and **uriworkermap.properties** files for the redirector.

For additional information about the settings in these files, see the Tomcat connector documentation available at **https://tomcat.apache.org**.

1. Create a directory (for example, **iis75-jboss7**) for the redirector in a location accessible to Microsoft IIS that contains the following directories:

   - **bin**
   - **conf**
   - **log**
   - **wwwroot**

2. Download the 32-bit or 64-nit (as appropriate for your host) ISAPI Redirector from a mirror site for the Apache Tomcat web site. Rename the downloaded file to **isapi_redirect.dll**. Only the DLL (**isapi_redirector-1.2.35.dll** or later version) file is required.

   Record the name and location of the Tomcat ISAPI Redirector installation directory for later use.

3. Configure Windows registry settings on the Windows Server host.

   a. In the ISAPI Redirector installation directory, create a file with a **.reg** extension. The name of this file is discretionary (**isapi_redirector.reg** is recommended).

   b. Add the following contents to the **.reg** file:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Apache Software Foundation\
    Jakarta Isapi Redirector\1.0]
"extension_uri"="/jakarta/isapi_redirect.dll"
"log_file"="D:\\iis75–jboss7\\logs\\jk_iis.log"
"log_level"="debug"
"worker_file"="D:\\iis75–jboss7\\workers.properties"
"worker_mount_file"="D:\\iis75–jboss7\\uriworkermap.properties"
"uri_select"="unparsed"
```

   Siemens Digital Industries Software recommends that you use **debug** for the **log_level** entry when you initially configure the redirector to get all messages. You can change this after you have tested your installation and determined that it is working properly. The following table provides a brief description of these entries:

Name	Description
extension_uri	Represents the IIS virtual directory including the ISAPI Redirector file.
log_file	Defines the name and location of the ISAPI Redirector log file.
log_level	Defines the level of debug messages written to the ISAPI Redirector log file. Valid values are **debug**, **info**, **error**, and **emerg**.
worker_file	Defines the location of the ISAPI redirector **worker.properties** file.
worker_mount_file	Defines the location of the ISAPI redirector **uriworkermap.properties** file.
uri_select	Determines how the forwarded URI is handled. Unparsed indicates the original request URI is forwarded. Siemens Digital Industries Software recommends this option. Rewriting the URI and forwarding the rewritten URI does not work correctly.

See these registry settings in the *Apache Tomcat Connector – Reference Guide* available at **https://tomcat.apache.org**.

c. Change the following lines in the **.reg** file to reflect your directory settings:

A. For **log_file**, enter the location of the **logs** directory you created and the name of the log file.

The log file itself is created later by the ISAPI Redirector.

B. For **worker_file**, enter a location for the worker definition file. Siemens Digital Industries Software recommends that you create this file in the directory where you installed the Tomcat ISAPI Redirector. You create this file later.

C. For the **worker_mount_file**, enter a location for the worker-URI map file. You create this file later.

D. For the **extension_uri**, enter **tomcat**.

d. In the ISAPI Redirector installation directory, right-click the **isapi_redirector.reg** file and choose **Merge**.

e. After receiving a confirmation message from Windows, check the ISAPI Redirector settings using the Microsoft Registry Editor program (**regedit.exe**) to ensure the registry settings are correct. For information about using the Microsoft Registry Editor, see the Microsoft Windows online help.

4. Create a text file with contents similar to the following:

```
# Define node1 (one node required for H_SE)
worker.list=node1
worker.node1.port=8009
worker.node1.host=host-name1
worker.node1.type=ajp13
```

The default port is **8009**. If you changed this AJP port number in JBoss (WildFly) configuration when you configured the Tomcat ISAPI Redirector, use that value. The *host-name* value is the host where you run JBoss (WildFly).

5. Save the file as **workers.properties** in the directory you defined for it in the registry file.

6. Create a text file with contents similar to the following:

```
# Send all /tc requests to node1
/tc/*=node1
```

Replace *tc* with the name of your Teamcenter web application (**tc** by default). This configures the redirector to forward all requests with the **/tc/*** signature to **node1**.

7. Save the file as **uriworkermap.properties**. Save this file in the same directory as the **workers.properties** file.

Configure Microsoft Internet Information Services on Windows Server platforms

1. Open the IIS Manager and choose **Start→Administrative Tools→Internet Information Services (IIS) Manager**.

2. In the **Connections** pane, expand your computer name until you see **Sites**.

3. Add a new web site for your deployment:

a. Right-click **Sites** and choose **Add Web Site**.

b. In the **Add Web Site** dialog box type a name for the site in the **Site name** box, for example, **iis75-jboss75**.

c. In the **Physical path** box, type or browse to the location of the **wwwroot** directory you created when you **installed the Tomcat ISAPI Redirector**.

d. In the **Port** box, type a value for the binding port, for example, **8028** .

e. Clear the **Start Web site immediately** check box and click **OK**.

4. Add a virtual directory:

a. In the **Connections** pane, right-click your new site name and choose **Add Virtual Directory**.

b. In the **Alias** box, type **jakarta**.

c. In the **Physical path** box, type the path or browse to the **bin** directory you created when you **installed the Tomcat ISAPI Redirector** and click **OK**.

5. Configure a handler mapping:

a. In the **Connections** pane, select you new site name.

b. Right-click **Handler Mappings** and select **Open Feature**.

c. In the **Handler Mappings** pane, double-click **ISAPI-dll**.

d. In the **Edit Module Mapping** dialog box, type an asterisk (*****) character in the **Request path** box

e. Click the browse button next to the **Executable** box and browse to the location of the **isapi_redirector.dll** file.

f. Click **Request Restrictions** and clear the **Invoke handler only if request is mapped to** check box on the **Mapping** tab.

g. Click the **Verbs** tab and ensure the **All verbs** option is selected.

h. Click the **Access** tab, ensure the **Execute** option is selected, and click **OK**.

i. In the **Connections** pane, select you new site name and click Start in the **Actions** pane (on the right side under **Manage Web Site**).

To access the web site, enter a URL in the following format:

`http://<host-name>:<port-number>/console/login/LoginForm.jsp`

Deployment on JBoss application server with Apache front end (H-S)

Deploying on JBoss (WildFly) application server with Apache front end (H-S)

The following procedure is provided for general reference. The versions may not match the versions in your deployment. The procedure does the following:

- Deploys the Teamcenter web tier application (WAR file) on JBoss (WildFly) Application Server.
- Installs and configures the Tomcat connector.
- Configures the Apache HTTP front end web server.

Deploy the Teamcenter web application on JBoss (WildFly) (H-S)

This procedure assumes that the JBoss 7.1.1 final version is installed, and that you are using the stand-alone server location for deploying your Teamcenter web application.

> Caution:
>
> Certain versions of JBoss (WildFly) configure the Java virtual machine (JVM) to prefer the IPv4 stack. This can cause **socket errors** when the server manager starts due to a mismatch in protocols between the web tier and server manager hosts.

1. Copy the Teamcenter WAR (by default, **tc.war**) file to the following directory:

   jboss-as-7.1.1.Final\standalone\deployments

2. Define JMX as a global module.

   a. Expand the **configuration** directory:

      jboss-as-7.1.1.Final\standalone\configuration

   b. Open the **standalone.xml** file.

   c. Change the HTTPS protocol to **ALL**. (The default protocol is **TLSv1**.) Locate the **subsystem** element for the **urn:jboss:domain** subsystem, and add the following **connector** element content:

```
<subsystem xmlns="urn:jboss:domain:ee:1.0">
<connector name="https" scheme="https" protocol="HTTP/1.1" socket
-binding="https" enable-lookups="false" secure="true">
      <ssl name="jbossSSL-SSL" password="private" protocol="ALL" key-
      alias="jbossSSL" certificate-key-file="D:\ssl\jbossSSL.keystore" />
</connector></subsystem>
```

d. Locate the **subsystem** element for the **urn:jboss:domain** subsystem, and add the following **global-modules** element content:

```
<subsystem xmlns="urn:jboss:domain:ee:1.0">
  <global-modules>
    <module name="org.jboss.as.jmx" slot="main"/>
   </global-modules>
</subsystem>
```

Locate the **deployment-scanner** element and add the **deployment-timeout** attribute with a value of **600** as follows:

```
<subsystem xmlns="urn:jboss:domain:deployment-scanner:1.1">
    <deployment-scanner path="deployments"
                        relative-to="jboss.server.base.dir"
                        scan-interval="5000"
                        deployment-timeout="600"/>
</subsystem>
```

e. If you require IPv6 support, locate the **interface** element for the **public** interface and modify its contents as follows:

```
<interface name="public">
  <any-address/>
</interface>
```

3. Microsoft IIS uses the AJP 1.3 protocol to forward requests to JBoss (WildFly). Perform the following steps to enable the AJP 1.3 protocol:

a. Open the *JBoss-installation***/server/default /deploy/jbossweb.sar/server.xml** file. Add or modify the following **Connector** element:

```
<!-- A AJP 1.3 Connector on port 8009 -->
<Connector protocol="AJP/1.3" port="8009" address="$
{jboss.bind.address}"
       tomcatAuthentication="false" emptySessionPath="true"
enableLookups="false" redirectPort="8443"  />
```

IIS forwards requests to JBoss (WildFly) using the AJP 1.3 protocol on the specified port. This must be set to allow access to the remote user name (**getRemoteUser**) method.

b. Open the *JBoss-installation***/standalone/ /configuration/standalone.xml** file and add the AJP connector as the child resource of the **jboss:domain:web** subsystem:

```
<subsystem xmlns="urn:jboss:domain:web:1.1" default-virtual-server="default-
host" native="false">
    <connector name="http" protocol="HTTP/1.1" scheme="http" socket-
binding="http"/>
```

```
        <connector name="ajp13" protocol="AJP/1.3" scheme="http" socket-binding="ajp"/>
        <virtual-server name="default-host" enable-welcome-root="true">
            <alias name="localhost"/>
            <alias name="example.com"/>
        </virtual-server>
    </subsystem>
```

c. Set or verify the port for the AJP protocol:

```
<socket-binding name="ajp" port="8009" />
```

If the default port for the AJP 1.3 protocol (**8009**) is not available on your host running JBoss (WildFly), set this value to an available port.

Record the port value for use when you configure the redirector.

4. Define a dependency to allow the JBoss (WildFly) connector module to use JMX MBeans.

a. Expand the **main** directory:

jboss-as-7.1.1.Final\modules\org\jboss\as\connector\main

b. Open the **module.xml** file.

c. Locate the **dependencies** element, and add the following **module** element:

```
<module name="org.jboss.as.jmx"/>
```

5. To allow the Teamcenter web application to listen to nonloopback addresses, configure JBoss (WildFly) using the information in the documentation available at **https://docs.jboss.org**. Specifically, see the *Command line parameters* and *Interfaces and ports* sections of that documentation.

6. If you use Requirements Management, open the **standalone.xml** file and remove the following lines if they are present:

```
<locking isolation="REPEATABLE_READ"/>
<transaction mode="BATCH"/>
```

Doing so ensures that Requirements Management microservices functions correctly.

7. If you require IPv6 support, open the **standalone_conf** script file in your JBoss (WildFly) installation **bin** directory and add the following settings:

```
-Djava.net.preferIPv4Stack=false
-Djava.net.preferIPv6Addresses=false
```

8. Open a command shell and ensure you have defined the **JAVA_HOME** environment variable, and set it to the location of your Java installation. Use a version of Java supported by the version of JBoss (WildFly).

9. Start the server by typing **standalone** (**standalone.sh** on Linux) **-b** *host-name* in the command shell.

   You must start the application server instance with the bind option to enable connections from clients running on a host different from the application server host. The simplest way to do this is to start the server with the **-b** *host-name* option. Substitute the host name or IP address of the local host for *host-name*. However, this has some security implications.

   For information about JBoss (WildFly) security, see the documentation available at **https://developer.jboss.org**.

If the web tier encounters errors obtaining JCA connections under peak activity, increase the **Max_Capacity** context parameter value for your Teamcenter web application.

Install and configure the Tomcat connector

Setting up of Tomcat connector has the following phases:

- Obtaining the Tomcat connector and placing it in the proper location.
- Configuring Apache to load the connector.
- Configuring the worker nodes for the connector.

Set up the Tomcat connector as follows:

1. Download the Tomcat connector, available at **http://www.apache.org**.

2. Change the connector file name to **mod_jk_so** and copy it to the **modules** directory in the *APACHE_HOME* (installation) directory.

3. Add the following line to the **httpd.conf** file in the *APACHE_HOME***/conf** directory:

   ```
   #Include mod_jk specific configuration file
   Include conf/mod_jk.conf
   ```

4. Create a **mod_jk.conf** text file in the *APACHE_HOME***/conf** directory with contents similar to the following:

   ```
   # Load mod_jk module
   # Specify the filename of the mod_jk lib
   LoadModule jk_module modules/mod_jk.so
   ```

```
# Where to find workers.properties
JkWorkersFile conf/workers.properties
# Where to put jk logs
JkLogFile logs/mod_jk.log
# Set the jk log level [debug/error/info]
JkLogLevel info
# Select the log format
JkLogStampFormat  "[%a %b %d %H:%M:%S %Y]"
# JkOptions indicates to send SSK KEY SIZE
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories
# JkRequestLogFormat
JkRequestLogFormat "%w %V %T"

# Mount your applications
JkMount /tc/* node1
# You can use external file for mount points.
# It will be checked for updates each 60 seconds.
# The format of the file is: /url=worker
# /examples/*=loadbalancer
JkMountFile conf/uriworkermap.properties
# Add shared memory.
# This directive is present with 1.2.10 and
# later versions of mod_jk, and is needed for
# for load balancing to work properly
JkShmFile logs/jk.shm

# Add jkstatus for managing runtime data
<Location /jkstatus/>
    JkMount status
    Order deny,allow
    Deny from all
    Allow from 127.0.0.1
</Location>
```

The **LoadModules** directive must reference the connector library file (**mod_jk.so**) with the **modules** directory prefix. The **JkMount** directive determines which URLs Apache forwards to the connector module. The **/tc/*** entry indicates that all requests to Teamcenter are sent to **node1**, assuming that the default application context root (**/tc**) is used.

You can also use the **JkMountFile** directive to specify a mount points configuration file (**uriworkermap.properties**). The format for entries in this file is */url=worker-name*. For example:

```
# Simple worker configuration file
# Mount the Servlet context to the ajp13 worker
/jmx-console=node1
/jmx-console/*=node1
/web-console=node1
/web-console/*=node1
```

5. Create a **workers.properties** text file in the *APACHE_HOME*/**conf** directory that identifies the location of the servlet container. For example, the following is a worker properties file with a single node:

```
# Define list of workers that will be used
# for mapping requests
worker.list=node1
# Define Node1
# modify the host as your host IP or DNS name.
worker.node1.port=8009
worker.node1.host=ahla6002
worker.node1.type=ajp13
worker.node1.cachesize=10
```

   By convention, each node is defined as **worker.***name***.***attribute***=***value*. You can use any *name* value to designate the servlet container with the specified host or IP address and port number of the AJP13 connector running in the servlet container. The **cachesize** attribute defines the size of the thread pool (number of concurrent requests allowed) for the servlet container.

Deployment on a WebSphere application server (H-S)

Deploy on a WebSphere application server (H-S)

This procedure deploys the Teamcenter web tier application (WAR file) on IBM WebSphere Application Server and configures IBM HTTP server as the front-end web server.

> Note:
>
> If you deploy multiple applications in the same application server instance, HTTP session cookies may be overwritten by browsers connecting to different applications. To avoid this, **configure the application server to provide separate cookie paths**.

You must set up the following WebSphere components for this configuration.

This sequence is recommended by the WebSphere Launchpad program. The process assumes that you have separate hosts for the application server (host A) and the web server (host B). WebSphere provides installation wizards to aid you in this process. The wizards are accessed from the **launchpad.exe** application. See the complete **WebSphere documentation** for additional details.

1. Install the WebSphere application server on host A. Use the installation wizard for WebSphere Application Server.

2. Install the IBM HTTP server with the required plug-in on host B. Use the installation wizard for IBM HTTP Server. If using a different web server, skip this wizard and install the web server per the vendors instructions on host B.

3. If using a previously installed IBM HTTP server or a different web server, install the web server plug-in on host B. Use the installation wizard for web server plug-ins.

4. Copy the **configure***Web-server-name* script file from the *plugins-install-root***/bin** directory on host B to the *profiles-install-root***/***profile-name***/bin** directory on host A.

5. In a command shell, run the **configure***Web-server-name* script. This creates a web server definition file for the integrated solutions console. You can now use the console to manage the web server.

6. Start the WebSphere application server.

7. Start the WebSphere integrated solutions console.

8. Propagate the web server plug-in file and configure the web server to accept all content.

   - For an IBM HTTP server:

     a. In the navigation tree, expand **Servers** and select **Web servers**.

     b. In the **Web servers** pane, click **Propagate Plug-in**.

     c. Expand **Servers→Web Servers→***Web-server-name***→Plug-in properties**.

        If you have load-balanced clustered web servers in your configuration, you must update the plug-in configuration on each web server. You can also locate the **plugin-cfg.xml** file for your web server, manually set the **AcceptAllContent** value to **true**, and push the change to the other web servers. For information about the IBM HTTP web server configuration, see the IBM documentation at *ftp://ftp.software.ibm.com/*.

     d. Select **AcceptAllContent** from the **Accept content for all requests** list and click **OK**.

   - For most other web servers, you must manually apply the web server plug-in file to the web server environment, However, It may be possible to propagate some other web server plug-in files in this manner.

9. In the navigation tree, expand **Applications** and click **Install New Application**.

10. In the **Preparing for the application installation** pane, type the path to, or browse to, the location of the Teamcenter web tier WAR file in the **Full path** box.

    Select **Prompt me only when additional information is required** and click **Next**.

11. Accept the default **Select installed options** for enterprise applications and modules and click **Next**.

12. In the **Map modules to servers** pane, if you have multiple server instances, select the check boxes for all modules and map them to the same server instance. Click **Next** again.

13. In the summary pane, click **Finish**. Wait for WebSphere to complete the application deployment.

14. Click **Apply**, scroll to the top of the page, and click **Save**.

If you deploy a web application that contains the **Teamcenter - Online Help** solution, set the context root for the web application in WebSphere to the enterprise tier ID for the web application. This is the value of the **Deployable File Name** context parameter or the **Enterprise Application Lookup ID** context parameter. Make sure you include the file name (*file-name*.**war**) when specifying the context root.

Configure the HTTP web server

1. Open the Teamcenter **site_specific.properties** file and modify the following properties:

   ```
   portalCommunicationTransport=http
   HTTP_SERVER_1.URI=http://host-name:port-number/tc-name/controller/test
   ```

   Replace *host-name* and *port-number* with the WebSphere application server host name and HTTP listening port number. Replace *tc-name* with your Teamcenter web application name; by default, this value is **tc**.

2. In the Integrated Solutions Console navigation tree, expand **Environment→Virtual Host** and click **default_host**.

3. Click **Host Aliases** under **Additional Properties** and click **New**.

4. Type the web server listening port number in the **Port** box and click **OK**.

5. In the navigation tree, expand **Environment→Update global web server plug-in configuration** and click **OK**.

6. Propagate the plug-in configuration file to the web server. The web server plug-in configuration service propagates the **plugin-cfg.xml** file automatically for IBM HTTP server. For all other web servers, propagate the plug-in configuration file manually. For information about propagating the plug-in configuration file, see IBM's **WebSphere application server documentation**.

   > Note:
   >
   > If the plug-in configuration service does not propagate the configuration file properly for an IBM HTTP server, you must manually copy the file to the web server plug-in directory.

   a. Copy the **plugin-cfg.xml** file from the *profile-root*/**config/cells/** *cell-name*/**nodes/***web-server-name-node*/**servers/***web-server-name* directory on the host where your WebSphere application server is installed.

   b. Paste the file into the *plugins-root*/**config/***web-server-name* directory on the host where the web server is installed.

c. Restart the web server.

Clustered deployment with front-end HTTP server

Overview of clustered deployment

Setting up an application server cluster can be a very complex process and can vary depending on your particular hardware, performance requirements, or availability requirements. The following instructions provide information specific to the Teamcenter web tier application. The application server documentation available from the vendor provides the best source for the cluster set up process and is referenced at several points in the following procedures.

Siemens Digital Industries Software does not support clustered deployment of Teamcenter web applications on JBoss (WildFly).

Deploy WebSphere application server cluster with HTTP (Web) server

This configuration is similar to **deploying on WebSphere application server (H-S)** with the additional requirement that you have the optional WebSphere application server Deployment Manager.

1. Ensure the WebSphere application server, including the optional IBM HTTP server or Sun web server and its corresponding plug-in, and the optional WebSphere application server deployment manager, are installed.

   See the following topics in the WebSphere Application Server documentation available at **https://www.ibm.com**:

   - *Installing your application serving environment*
   - *Balance workloads by clustering application servers*
   - *Establishing high availability (HA) for failover*

2. **Deploy on WebSphere application server (H-S).**

3. Ensure the Teamcenter WAR file and all its modules are deployed to all cluster instances.

4. Ensure the plug-in configuration file is propagated to all cluster members and the HTTP server side.

Deploying clustered with front-end load-balanced HTTP servers

Overview of clustered deployment with front-end load-balanced HTTP servers

This configuration requires that you setup an **H-S deployment**, and then configure an external load balancer for the HTTP (Web) servers to create a web server farm. There are various external load balancers available and each has to be configure according to the vendors instructions. Therefore, Siemens Digital Industries Software cannot provide instructions for all possible configurations. You can use the Microsoft IIS load balancing instructions as a guide.

Configure Microsoft IIS load balancing

This procedure provides instructions for configuring the network load balancing mechanism provided with Microsoft IIS 6.0. Ensure that each host is self-sufficient with resources duplicated on each one. The Teamcenter database, whether a single or distributed database, must be on host separate from the web and application servers.

Network load balancing (NLB) aids in the process of creating a farm. A *farm* is a redundant cluster of several web servers serving a single IP address. Each machine can be configured to route the requests to the Java EE application server where your web tier is deployed. Each server in the cluster is fully self-contained, which means it is able to function without requiring any other server in the cluster. If any machine in the cluster is unavailable, NLB rebalances the incoming requests to the running servers in the cluster. The servers in the cluster must be able to communicate with each other to exchange information about their current processor and network load and to determine when a server is unavailable. NLB can provide reasonably close to 1:1 performance improvement for each server added to the cluster.

NLB requires a minimum of two servers running Windows Server 2003. Each server must have at least one network card (NIC) and a fixed IP address. For best performance, Siemens Digital Industries Software recommends you have two adapters in each server; one mapped to the real IP address (dedicated IP) and one mapped to the virtual IP address (cluster IP). NLB uses advanced networking features of network adapters. Therefore, low end adapters, especially those for nonserver hosts, may not support the required NDIS protocols.

1. Select an available IP address on the same class C network segment as the fixed IP addresses for the virtual IP address.

2. On any server, start the Network Load Balancing Manager in one of these ways:

   - Choose **Start→Administrative Tools→Network Load Balancing Manager**.

   - At a command prompt, type **NLBmgr**.

3. In the **Network Load Balancing Manager** dialog box, right-click the **Network Load Balancing Clusters** root node and choose **New cluster**.

4. Define the cluster parameters:

   a. Type the virtual IP address you selected for the cluster in the **IP address** box.

   b. Type a subnet mask in the **Subnet mask** box. You must use the same subnet mask for all servers in the cluster.

      The **Full Internet name** value is only for reference and is used primarily for displaying the name of the server. However, if you have a domain configured for the server you may use that domain name.

   c. If your server has more than one network adapter, click **Unicast** for the **Cluster operation mode**. If you are using a single adapter, Siemens Digital Industries Software recommends that you select **Multicast** to allow both the NLB traffic and the native IP traffic to move through the same network adapter. While Multicast is slower than Unicast as both kinds of traffic must be handled by the network adapter, it is the only way to remotely configure all machines centrally for servers with one network adapter.

   d. Clear the **Allow Remote Control** check box and click **Next**. If you need this functionality, enable it after you have the cluster running.

5. Click **Next** in the **Cluster IP addresses** dialog box.

6. Define the standard port rules:

   a. Click **Add**.

   b. Select the **All** check box and type **80** in both the **From** and **To** boxes.

   c. Click **Both** for **Protocols**.

   d. Click **Multiple hosts** for **Filtering mode** and **None** for **Affinity**.

   e. Click **OK**.

7. Define the secure port rules:

   a. Click **Add**.

   b. Select the **All** check box and type **443** in both the **From** and **To** boxes.

   c. Click **Both** for **Protocols**.

   d. Click **Multiple hosts** for **Filtering mode** and **Single** for **Affinity**.

   e. Click **OK**.

8. Connect the master (primary) host as a node in the cluster:

    a. Type the IP address of the host you want as the primary in the **Host** box.

    Node 1 is the primary, which means that it receives requests and acts as the routing manager. Although when the load is high on this node, other machines may take over for the primary.

    b. Click **Connect**.

    c. Click **Next**.

    d. In the **Host Parameters** dialog box, select **1** from the **Priority** list. Priority sets a unique ID for each node in the cluster. The lower the number the higher the priority.

    e. Click **Finish**.

    The Network Load Balancing Manager configures your network adapter. The network connection flashes on and off a few times during this configuration process on the sever you are configuring as a host. When the configuration is complete, the **Status** column displays **Converged** for the node.

9. In the Network Load Balancing Manager, right-click the cluster domain and choose **Connect**.

10. Repeat step 8 until all nodes have been added to the cluster.

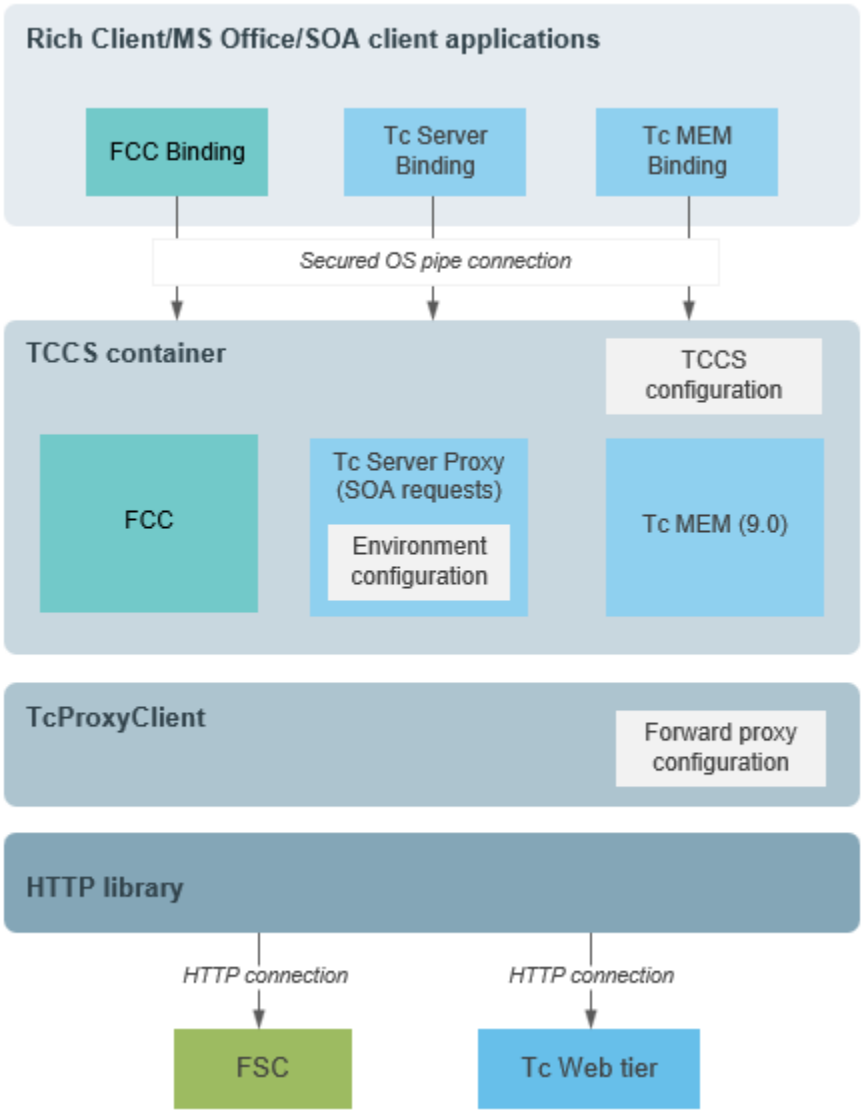
A. Teamcenter client communication system and proxy server configuration

Overview of TCCS and proxy server configuration

Teamcenter currently supports IBM WebSEAL and CA SiteMinder commercial single sign-on (SSO) products for reverse proxy servers. Security Services is required when using these reverse proxy servers.

Teamcenter provides the Teamcenter client communication system (TCCS) application that contains the **TcProxyClient** component to support forward and reverse proxy servers. This component detects form-based and 401-based challenges from reserve proxy servers. It uses the criteria defined in the **reverseproxy_config.xml** file to identify form-based challenges from a reverse proxy and uses the Apache HTTP client library to detect 401-based challenges. If the **reverseproxy_config.xml** file is not available, the component uses default criteria defined for the type of reverse proxy server (only WebSEAL is supported if the configuration file does not exist).

The following figure shows the TCCS architecture.



The **TcServerProxy** (TSP) manages HTTP communications for Teamcenter server (tcserver) requests. It accepts client requests over secured pipes using a proprietary protocol and submits the requests over HTTP to the web tier endpoint. You can use the **tspstat** utility to administer and obtain runtime statistics from the **TcServerProxy** component.

The FMS client cache (FCC) runs within the TCCS container. The TCCS application is started when you start the FCC (**startfcc** command). The FCC accepts client requests over secure pipe connections and submits them to the appropriate FMS server cache (FSC) process. The FCC uses the **TcProxyClient** component and forward proxy configuration to support forward and reverse proxy servers. Hooks to the **java.net** package are used to integrate the forward proxy library and the Jakarta Commons HTTP state into the **java.net** processing.

The Teamcenter model event manager (TcMEM) component manages event synchronization across SOA clients sharing the same Teamcenter server instance.

For form-based challenges, the **TcProxyClient** component examines the response for content type. For a 200 response, if the content type is not **text/html** the component does no further processing.

When the **TcProxyClient** component detects a challenge from a reverse proxy server, it passes the URL for the reverse proxy server to Teamcenter Security Services which returns a cookie corresponding to a valid session for the reverse proxy. The cookie patterns for the proxy servers are defined in the **tcsso_rp_cookienamepattern** context parameter during the TCCS installation process as part of the Security Services configuration.

The **TcProxyClient** component also supports one-way and two-way SSL using smart card client certificate or soft client-certificate authentication. Client-certificate authentication is more secure than any of the other supported forms of authentication. A client certificate can be either of the following:

- A smart card containing a certificate that complies with the PKCS#11 standard. Smart-card authentication is an example of two-factor authentication (2FA). Two-factor authentication requires the presentation of *something the user knows* and *something the user has*.

  Smart-card authentication is supported only for a 32-bit Java Runtime Environment (JRE). It is not supported for a 64-bit JRE.

- A file containing a certificate that complies with the PKCS#12 standard. Commonly used file extensions are **.p12**, **.pfx**, and **.jks**. Teamcenter supports soft certificates for both 32- and 64-bit JREs.

Teamcenter client communication system (TCCS) supports client-certificate authentication for the rich client, Client for Office, Lifecycle Visualization, stand-alone Electronic Design Automation (EDA), Solid Edge, and NX applications.

You can configure the server to display a *notice and consent logon banner* when a user connects to a Teamcenter client using smart card or soft certificate authentication. Teamcenter displays the notice defined by the **banner.txt** file in the login service WAR file. This file is located in the root folder of the Login Service WAR file. If the **banner.txt** file is empty or contains only whitespace characters, Teamcenter does not display the notice. The consent to log on dialog box provides a cancel button. If the user clicks **Cancel**, the connection to Teamcenter is prohibited.

The pattern is defined as a case-insensitive string that can contain wildcard characters (*****) for matching one or more characters at their position in the string. The literal ***** character can be include by preceding it with the backslash (****) escape character. You can include a wildcard at the beginning or end of the string or both. The following examples are valid patterns:

```
*string
string*
*string*
stri\*ng
```

You can also include a wildcard character within a string, for example:

*coo*kie
co*ok*ie

About reverse proxy servers

Teamcenter client communication system (TCCS) supports form-based challenge from reverse proxy servers:

- IBM WebSEAL
- CA SiteMinder

Security Services supports cookie sharing in both of these reverse proxy servers, but you must enable the feature for the given server before you can use it.

Enabling File Management System (FMS) URL path extensions

FMS URL path extensions are always enabled. The configuration elements that require additional path information can include:

- **parentfsc** address in the **fcc.xml** file.

  This value can be only entered from the **fcc_only** installer. No other FCC installer supports this; therefore, the **fcc.xml** file must be modified manually.

- **fscmaster** address in the **fsc.xml** file.
- **multisite fsc** addresses in the **fmsmaster.xml** file.
- **Fms_BootStrap_Urls** preference values.

> Note:
>
> The **/tc/fms/***fmsenterpriseid* path extension is not configurable. Reverse proxies must be configured to map to this path extension.

FMS server cache (FSC) SSL client credentials (two-way SSL)

FMS SSL configuration is not fully supported by the installers. Additional steps are required to generate certificates and configure the FSC property and keystore files.

Two-way SSL configuration can be enabled only after first successfully configuring for SSL.

Caution:

The password specified for the **com.teamcenter.fms.servercache.keystore.password** property and the **com.teamcenter.fms.servercache.keystore.ssl.certificate.password** property must be identical. These properties are contained in the **fsc.properties** files.

The **com.teamcenter.fms.allowuntrustedcertificates** property cannot be used with two-way SSL. This property can only be used for trusting one-way SSL self-signed certificates.

Additional configuration steps for enabling two-way SSL

The following additional steps are required to configure two-way SSL:

1. Modify the **fmsmaster** FSC address and/or the **connection** element to add the following value:

   ```
   <fsc id="FSC_fscmidzone_tcdba"
        address="https://fscmidzone.yourcompany.com:4544
        options="needclientauth">
   ```

   or

   ```
   <connection id="another2waySSLconn"
               protocol="https" port="4545"
               options="needclientauth"/>
   ```

2. Uncomment or add the following properties in the **fsc.properties** file to point to the existing keystore that was created to support the initial SSL configuration:

   ```
   javax.net.ssl.keyStore=${FMS_HOME}/keystore
   javax.net.ssl.keyStorePassword=keystorepassword
   javax.net.ssl.trustStore=${FMS_HOME}/keystore
   javax.net.ssl.trustStorePassword=keystorepassword
   ```

3. Add trusted certificates to the keystore that can validate the clients that are allowed to connect.

   The trusted certification from the CA, for example the **thawte premium server CA** certificate, is required in addition to the client certificate.

File Management System (FMS), reverse proxy, and two-way SSL configuration details

Overview of FMS, reverse proxy, and two-way SSL configuration

This section describes how to configure an FMS system with the following characteristics:

- All client traffic is directed to a reverse proxy server.
- All client traffic uses one-way SSL.
- Several logical and/or physical zones exist behind the reverse proxy. These are separated by firewalls.
- Reverse proxy sends traffic to an FMS Server Cache (FSC) located within the same zone (using one-way SSL).
- Another FSC in another zone hosts the real volumes.
- FSC-to-FSC communication across the zones requires two-way SSL.

Basic File Management System (FMS) configuration

Introduction to basic FMS configuration

This example describes the basic FMS server caches (FSCs), groups, and client maps.

The target configuration consists of two FSCs behind a reverse proxy server. All clients are in front of the reverse proxy.

There are three zones in this example:

- **Client zone**

  All clients are on one side of the reverse proxy server. All communication is routed through the reverse proxy to the backend servers. The only resource the clients communicate with is the reverse proxy server.

- **Middle zone**

  The location of the reverse proxy, web tier, first FSC, and LDAP.

- **Resource zone**

  The second FSC, volumes, and Oracle.

The following **fmsmaster_FSC_fscmidzone.xml** file is the primary configuration file used in this example.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fmsworld SYSTEM "fmsmasterconfig.dtd">
<fmsworld>
  <fmsenterprise id="471539747">
    <fscgroup id="midzone">
      <!-- the following fsc element is a caching FSC -->
```

```
      <fsc id="FSC_fscmidzone_tcdba"
           address="http://fscmidzone.yourcompany.com:4544" />
      <!-- the following fsc element represents the reverse proxy -->
      <fsc id="FSC_reverseproxy_tcdba"
           address="http://reverseproxy.yourcompany.com:80" />
      <!-- the following clientmap element maps all clients
           to the reverse proxy -->
      <clientmap subnet="127.0.0.1" mask="0.0.0.0">
        <assignedfsc fscid="FSC_proxy_tcdba" />
      </clientmap>
    </fscgroup>
    <fscgroup id="reszone">
      <!-- the following fsc element is the FSC that hosts the volumes -->
      <fsc id="FSC_fscreszone_tcdba"
           address="http://fscreszone.yourcompany.com:4544">
        <volume id="139747566d871c1b2023"
                root="/mnt/disk1/tcapps/tceng2005sr1mp5/TC_VOL/volume1" />
        <transientvolume id="ce8399515feada2dee4c3e79b955d8ba"
          root="/tmp/transientVolume_tceng2005sr1mp5_tcdba" />
      </fsc>
    </fscgroup>
  </fmsenterprise>
</fmsworld>
```

Configuration element details

Element	Definition
fscgroup	Describes either a group of FSCs on a LAN or a network of FSCs that have defined entry and exit FSCs. This configuration is simple because there is only one real FSC in each group; therefore, declared entries and exits are not required. There are two defined fscgroups: **midzone** — Represents the middle zone. **reszone** — Represents the resource zone.
FSC	The FSC for each zone is defined within the groups and one FSC is defined to represent the reverse proxy server, as follows: **FSC_fscmidzone_***tc-admin-user* — The FSC in the middle tier acts as a cache and performs the role of an FSC configuration primary. This means it serves the primary configuration file. **FSC_fscreszone_***tc-admin-user* — The FSC in the resource tier mounts the volume and it is a configuration secondary to the **FSC_fscmidzone***tc-admin-user* FSC. **FSC_proxy_***tc-admin-user* — This FSC represents the reverse proxy server. It is required so that the **clientmap** elements can point to the FSC (address) for assignment. Clients

Element	Definition
	should be assigned to the reverse proxy address, not to any of the real FSC servers.
clientmap	Clients are to be mapped to a single FSC (WebSEAL or SiteMinder); therefore, only a single comprehensive **clientmap** that assigns all clients to the reverse proxy is required. There are no volumes in the assigned group; therefore, you do not have to turn off direct routing to prevent the FCC from attempting to reach FSCs hosting volumes directly within the group.

FSC configuration files

The following configuration files are associated with the real FSCs:

- **FSC_fscmidzone_infodba**

  The FSC that is the FMS primary configuration.

  - **$FSC_HOME/fmsmaster_FSC_fscmidzone_infodba.xml**

    Primary FMS configuration file.

  - **$FSC_HOME/FSC_fscmidzone_infodba.xml**

    FSC configuration file that specifies the **fscid** and primary/secondary state.

- **FSC_fscreszone_infodba**

  - **$FSC_HOME/fmsmaster_FSC_fscreszone_infodba.xml**

    Local copy of the primary FMS configuration file.

  - **$FSC_HOME/FSC_fscreszone_infodba.xml**

    FSC configuration file that specifies the **fscid** and primary/secondary state.

Configuration file content – bootstrap references

Bootstrap references must be changed to point to the reverse proxy (FSC) rather than to any of the real backend FSCs.

You must also add the default URL context to all of the bootstrap references in the **site context** form:

protocol`://`*host*`[:`*port*`]/tc/fms/`*fmsenterpriseid*

$FMS_HOME/fcc.xml

```
<parentfsc
  address="http://reverseproxy.yourcompany.com:80/tc/fms/471539747"/>
...
...
```

$FSC_HOME/FSC_fscmidzone_infodba.xml

```
...
       <fscmaster serves="true"/>
...
```

$FSC_HOME/FSC_fscreszone_infodba.xml

This is the secondary FSC **fsc.xml** file that points to the primary FSC. This is on the same side of the reverse proxy; therefore, a direct reference is used here.

```
...
   <fscmaster serves="false"
       address="http://fscmidzone.yourcompany.com:4544/tc/fms/471539747"/>
...
```

Fms_BootStrap_Urls preference

This value is used to bootstrap other FMS client integrations. The value must be appropriate for clients outside of the WebSEAL or SiteMinder reverse proxy; therefore, it points to the reverse proxy.

For example, for a WebSEAL reverse proxy:

`http://webseal.`*yourcompany*`.com:80/tc/fms/471539747`

For example, for a SiteMinder reverse proxy:

`http://siteminder.`*yourcompany*`.com:80/tc/fms/471539747`

One-way SSL configuration

Introduction to one-way SSL configuration

This section describes how to configure one-way SSL between the clients, the reverse proxy, and the FSC servers.

The following **fmsmaster_FSC_fscmidzone.xml** file is the primary configuration file used in this example. The example uses purchased certificates.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fmsworld SYSTEM "fmsmasterconfig.dtd">
<fmsworld>
  <fmsenterprise id="471539747">
    <fscgroup id="midzone">
      <!-- the following fsc element is a caching FSC,
           the default connection now uses SSL -->
      <fsc id="FSC_fscmidzone_tcdba"
           address="https://fscmidzone.yourcompany.com:4544" />
      <!-- the following fsc element represents the WebSEAL proxy -->
      <fsc id="FSC_webseal_tcdba"
           address="https://reverseproxy.yourcompany.com:443" />
      <!-- the following clientmap element maps all clients
           to the WebSEAL proxy -->
      <clientmap subnet="127.0.0.1" mask="0.0.0.0">
        <assignedfsc fscid="FSC_proxy_tcdba" />
      </clientmap>
    </fscgroup>
    <fscgroup id="reszone">
      <!--the following fsc element is the FSC that hosts the volumes,
          the default connection now uses SSL-->
      <fsc id="FSC_fscreszone_tcdba"
           address="https://fscreszone.yourcompany.com:4544">
        <volume id="139747566d871c1b2023"
                root="/mnt/disk1/tcapps/tceng2005sr1mp5/TC_VOL/volume1" />
        <transientvolume id="ce8399515feada2dee4c3e79b955d8ba"
           root="/tmp/transientVolume_tceng2005sr1mp5_tcdba" />
      </fsc>
    </fscgroup>
  </fmsenterprise>
</fmsworld>
```

One-way SSL configuration element details

Element	Definition
FSC	The addresses defined for the FSCs specify **https**. This causes the listener to be configured for SSL. The port on the FSC representing the reverse proxy is changed to use **443** rather than **80**.

One-way SSL FSC configuration files

The following configuration files are associated with the real FSCs:

- **FSC_fscmidzone**

  Specifies the primary FMS configuration.

- **$FSC_HOME/fmsmaster_FSC_fscmidzone.xml**

  The primary FMS configuration file.

- **$FSC_HOME/FSC_fscmidzone.xml**

  FSC configuration file that specifies the **fscid** and primary/secondary state.

- **$FSC_HOME/fsc.FSC_fscmidzone.properties**

  Additional properties for this FSC used to configure the keystore.

- **$FSC_HOME/keystore.FSC_fscmidzone.jks**

  Keystore for this FSC.

- **FSC_fscreszone**

  Specifies the secondary FMS configuration.

  - **$FSC_HOME/fmsmaster_FSC_fscreszone.xml**

    Local copy of the primary FMS configuration file.

  - **$FSC_HOME/FSC_fscreszone.xml**

    FSC configuration file that specifies the **fscid** and primary/secondary state.

  - **$FSC_HOME/fsc.FSC_fscreszone.properties**

    Additional properties for this FSC used to configure the keystore.

  - **$FSC_HOME/keystore.FSC_fscreszone.jks**

    The keystore for this FSC.

One-way SSL configuration file changes – bootstrap references

Bootstrap references must be changed to use the new port on the reverse proxy (FSC) and to configure the keystores.

$FMS_HOME/fcc.xml

```
...
        <parentfsc address="https://reverseproxy.yourcompany.com:443/tc/fms/
```

```
471539747"/>
...
```

Fms_BootStrap_Urls preference

This value is used to bootstrap other FMS client integrations. The value must be appropriate for clients outside of the WebSEAL or SiteMinder reverse proxy; therefore, it points to the reverse proxy.

For example, for WebSEAL:

```
http://webseal.yourcompany.com:443/tc/fms/471539747
```

For example, for SiteMinder:

```
http://siteminder.yourcompany.com:443/tc/fms/471539747
```

One-way SSL sew configuration files – property and keystore files

When editing FMS configuration files, always use Linux-style path separators (/) and refer only to **$FMS_HOME** (not to **$FSC_HOME**).

$FSC_HOME/fsc.FSC_fscmidzone.properties

The property file used to configure the keystore.

```
# fsc.FSC_fscmidzone.properties
    com.teamcenter.fms.servercache.keystore.file=$<FMS_HOME}/keystore.FSC_fscmidzone.jks
    com.teamcenter.fms.servercache.keystore.password=keystore.FSC_fscmidzone.password

com.teamcenter.fms.servercache.keystore.ssl.certificate.password=keystore.FSC_fscmidzone.pass
word
    # these are not needed for 1-way SSL
    # javax.net.ssl.keyStore=$<FMS_HOME}/keystore.FSC_fscmidzone.jks
    # javax.net.ssl.keyStorePassword=keystore.FSC_fscmidzone.password
    # javax.net.ssl.trustStore=$<FMS_HOME}/keystore.FSC_fscmidzone.jks
# javax.net.ssl.trustStorePassword=keystore.FSC_fscmidzone.password
```

$FSC_HOME/keystore.FSC_fscmidzone

The keystore for this FSC. The keystore must contain the private key and certificate for the local machine.

```
fscmidzone> keytool -list -v -keystore keystore.FSC_fscmidzone.jks
-storepass keystore.FSC_fscmidzone.password

    Keystore type: jks
    Keystore provider: SUN
    Your keystore contains 1 entries

    Alias name: fscmidzone.yourcompany.com
    Creation date: Jan 23, 2008
```

```
    Entry type: keyEntry
    Certificate chain length: 2
    Certificate[1]:
    Owner: CN=fscmidzone.yourcompany.com, OU=QA, O=YOUR  Corp, L=Plano, ST=Texas, C=US
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number:
485099dcc36d1ea9d773ba153022a951
Valid from: Thu Jan 10 16:44:38 CST 2008 until: Thu Mar 27 13:20:25 CDT 2008 Certificate
fingerprints:
    MD5:  86:7E:16:59:99:E6:6F:B6:27:9B:92:19:E7:65:EB:A2
    SHA1: 6A:D1:64:7A:0A:E1:CB:62:D3:EF:91:BF:E9:A0:CE:AF:A3:3D:E4:1E
    Certificate[2]:
    Owner: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number: 1
 Valid from: Wed Jul 31 19:00:00 CDT 1996 until: Thu Dec 31 17:59:59 CST 2020 Certificate
fingerprints:
    MD5:  06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
    SHA1: 62:7F:8D:78:27:65:63:99:D2:7D:7F:90:44:C9:FE:B3:F3:3E:FA:9A

    *******************************************
*******************************************
```

$FSC_HOME/fsc.FSC_fscreszone.properties

The property file used to configure the keystore.

```
# fsc.FSC_fscreszone.properties
    com.teamcenter.fms.servercache.keystore.file=$<FMS_HOME}/keystore.FSC_fscreszone.jks
    com.teamcenter.fms.servercache.keystore.password=keystore.FSC_fscreszone.password

com.teamcenter.fms.servercache.keystore.ssl.certificate.password=keystore.FSC_fscreszone.pass
word
    # these are not needed for 1-way SSL
    # javax.net.ssl.keyStore=$<FMS_HOME}/keystore.FSC_fscreszone.jks
    # javax.net.ssl.keyStorePassword=keystore.FSC_fscreszone.password
    # javax.net.ssl.trustStore=$<FMS_HOME}/keystore.FSC_fscreszone.jks
# javax.net.ssl.trustStorePassword=keystore.FSC_fscreszone.password
```

$FSC_HOME/keystore.FSC_fscreszone

The keystore for this FSC. The keystore must contain the private key and certificate for the local machine.

```
fscreszone> keytool -list -v -keystore keystore.FSC_fscreszone.jks
 -storepass keystore.FSC_fscreszone.password

    Keystore type: jks
    Keystore provider: SUN
    Your keystore contains 1 entries
```

```
    Alias name: fscreszone.yourcompany.com
    Creation date: Jan 23, 2008
    Entry type: keyEntry
    Certificate chain length: 2
    Certificate[1]:
    Owner: CN=fscreszone.yourcompany.com, OU=QA, O=YOUR  Corp, L=Plano, ST=Texas, C=US
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number:
485099dcc36d1ea9d773ba153022a951
Valid from: Thu Jan 10 16:44:38 CST 2008 until: Thu Mar 27 13:20:25 CDT 2008 Certificate
fingerprints:
    MD5:  86:7E:16:59:99:E6:6F:B6:27:9B:92:19:E7:65:EB:A2
    SHA1: 6A:D1:64:7A:0A:E1:CB:62:D3:EF:91:BF:E9:A0:CE:AF:A3:3D:E4:1E
    Certificate[2]:
    Owner: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number: 1
Valid from: Wed Jul 31 19:00:00 CDT 1996 until: Thu Dec 31 17:59:59 CST 2020 Certificate
fingerprints:
    MD5:  06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
    SHA1: 62:7F:8D:78:27:65:63:99:D2:7D:7F:90:44:C9:FE:B3:F3:3E:FA:9A

    ****************************************
****************************************
```

Configuring two-way SSL between FMS server caches (FSCs)

Overview of two-way SSL between FSCs

Building on the one-way SSL configuration example, this section describes how two-way SSL is configured exclusively for FSC to FSC traffic.

The following **fmsmaster_FSC_fscmidzone.xml** file is the primary configuration file used in this example.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE fmsworld SYSTEM "fmsmasterconfig.dtd">
<fmsworld>
  <fmsenterprise id="471539747">
    <fscgroup id="midzone">
      <!-- the following fsc element is a caching FSC,
           the default connection now uses 2-way SSL -->
      <fsc id="FSC_fscmidzone_tcdba"
           address="https://fscmidzone.yourcompany.com:4545"
           options="needclientauth">
        <!-- the following connection element adds an additional
             connection supporting SSL(1-way) to this FSC -->
        <connection id="1waysslcon" protocol="https" port="4544"/>
      </fsc>
```

```
        <!-- the following fsc element represents the WebSEAL proxy -->
        <fsc id="FSC_webseal_tcdba"
             address="https://reverse_proxy.yourcompany.com:443"/>
        <!-- the following clientmap element is used to map particular
             clients (e.g. DAK server, Web Application Server) within the
                    midzone to the 1-way SSL connection of the midzone FSC -->
        <clientmap subnet="146.122.69.94" mask="255.255.255.255">
          <assignedfsc fscid="FSC_fscmidzone_tcdba"
                       connectionid="1waysslcon"/>
        </clientmap>
        <!-- the following clientmap element maps all (remaining)
             clients to the WebSEAL proxy -->
        <clientmap subnet="127.0.0.1" mask="0.0.0.0">
          <assignedfsc fscid="FSC_proxy_tcdba"/>
        </clientmap>
      </fscgroup>
      <fscgroup id="reszone">
        <!-- the following fsc element is the FSC that hosts the volumes,
             the default connection now uses 2-way SSL -->
        <fsc id="FSC_fscreszone_tcdba"
             address="https://fscreszone.yourcompany.com:4545"
             options="needclientauth">
          <volume id="139747566d871c1b2023"
                  root="/mnt/disk1/tcapps/tceng2005sr1mp5/TC_VOL/volume1"/>
          <transientvolume id="ce8399515feada2dee4c3e79b955d8ba"
                       root="/tmp/transientVolume_tceng2005sr1mp5_tcdba"/>
        </fsc>
      </fscgroup>
    </fmsenterprise>
  </fmsworld>
```

Two-way SSL configuration element details

Element	Definition
FSC	The **FSC** elements specify **options="needclientauth"**. This causes the default connection to require a two-way SSL handshake. The default connection is defined in the address attribute of the **fsc** element. In this example, the port number is changed to **4545**.
connection	A new **connection** element is added (using the original SSL port number **4544**) to the **FSC_fscmidzone** FSC to continue to support the one-way SSL connection that reverse proxy is configured to use.
clientmap	There is an additional **clientmap** element to map clients that are already inside the midzone to the one-way SSL connection of the midzone FSC. (The Teamcenter Engineering Data Integration Services Adapter is one such client.)

Two-way SSL FSC configuration files

The following configuration files are associated with the real FSCs:

- **FSC_fscmidzone**

  Specifies the FMS primary configuration.

  - **$FSC_HOME/fmsmaster_FSC_fscmidzone.xml**

    Primary FMS configuration file.

  - **$FSC_HOME/FSC_fscmidzone.xml**

    FSC configuration file that specifies the **fscid** and primary/secondary state.

  - **$FSC_HOME/fsc.FSC_fscmidzone.properties**

    Additional properties for this FSC used to configure the keystore.

  - **$FSC_HOME/keystore.FSC_fscmidzone.jks**

    The keystore for this FSC.

- **FSC_fscreszone**

  Specifies the secondary FMS configuration.

  - **$FSC_HOME/fmsmaster_FSC_fscreszone.xml**

    Local copy of the primary FMS configuration file.

  - **$FSC_HOME/FSC_fscreszone.xml**

    FSC configuration file that specifies the **fscid** and primary/secondary state.

  - **$FSC_HOME/fsc.FSC_fscreszone.properties**

    Additional properties for this FSC used to configure the keystore.

  - **$FSC_HOME/keystore.FSC_fscreszone.jks**

    The keystore for this FSC.

Two-way SSL configuration file changes

Bootstrap references

None of the bootstrap references change; they continue to point to the reverse proxy HTTPS address.

Property and keystore files

When editing FMS configuration files, always use Linux-style path separators (/) and refer only to **$FMS_HOME** (not to **$FSC_HOME**).

$FSC_HOME/fsc.FSC_fscmidzone.properties

Property file used to configure the keystore.

```
# fsc.FSC_fscmidzone.properties
com.teamcenter.fms.servercache.keystore.file=${FMS_HOME}/keystore.FSC_fscmidzone.jks
com.teamcenter.fms.servercache.keystore.password=keystore.FSC_fscmidzone.password
com.teamcenter.fms.servercache.keystore.ssl.certificate.password=keystore.FSC_fscmidzone.pass
word
# these are not needed for 1-way SSL
    javax.net.ssl.keyStore=${FMS_HOME}/keystore.FSC_fscmidzone.jks
    javax.net.ssl.keyStorePassword=keystore.FSC_fscmidzone.password
    javax.net.ssl.trustStore=${FMS_HOME}/keystore.FSC_fscmidzone.jks
javax.net.ssl.trustStorePassword=keystore.FSC_fscmidzone.password
```

$FSC_HOME/keystore.FSC_fscmidzone

The keystore for this FSC. The keystore just contain the private key and certificate for the local machine and it must also contain the trusted (CA) certificate for any clients you want to accept.

You can optionally import individual certificates for each client rather than importing the *signer certificate*.

```
fscmidzone> keytool -list -v -keystore keystore.FSC_fscmidzone.jks
-storepass keystore.FSC_fscmidzone.password
    Keystore type: jks
    Keystore provider: SUN
    Your keystore contains 2 entries

    Alias name: fscmidzone.yourcompany.com
    Creation date: Jan 23, 2008
    Entry type: keyEntry
    Certificate chain length: 2
    Certificate[1]:
    Owner: CN=fscmidzone.yourcompany.com, OU=QA, O=YOUR  Corp, L=Plano, ST=Texas, C=US
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number:
485099dcc36d1ea9d773ba153022a951
Valid from: Thu Jan 10 16:44:38 CST 2008 until: Thu Mar 27 13:20:25 CDT 2008 Certificate
fingerprints:
    MD5:  86:7E:16:59:99:E6:6F:B6:27:9B:92:19:E7:65:EB:A2
    SHA1: 6A:D1:64:7A:0A:E1:CB:62:D3:EF:91:BF:E9:A0:CE:AF:A3:3D:E4:1E
    Certificate[2]:
    Owner: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number: 1
Valid from: Wed Jul 31 19:00:00 CDT 1996 until: Thu Dec 31 17:59:59 CST 2020 Certificate
fingerprints:
    MD5:  06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
    SHA1: 62:7F:8D:78:27:65:63:99:D2:7D:7F:90:44:C9:FE:B3:F3:3E:FA:9A

    ********************************************
    ********************************************

    Alias name: thawte premium server ca
    Creation date: Feb 20, 2008
    Entry type: trustedCertEntry
    Owner: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number: 1
Valid from: Wed Jul 31 19:00:00 CDT 1996 until: Thu Dec 31 17:59:59 CST 2020 Certificate
fingerprints:
    MD5:  06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
    SHA1: 62:7F:8D:78:27:65:63:99:D2:7D:7F:90:44:C9:FE:B3:F3:3E:FA:9A
    ********************************************
********************************************
```

$FSC_HOME/fsc.FSC_fscreszone.properties

The property file used to configure the keystore.

```
# fsc.FSC_fscreszone.properties
    com.teamcenter.fms.servercache.keystore.file=${FMS_HOME}/keystore.FSC_fscreszone.jks
    com.teamcenter.fms.servercache.keystore.password=keystore.FSC_fscreszone.password

com.teamcenter.fms.servercache.keystore.ssl.certificate.password=keystore.FSC_fscreszone.pass
word
    # these are not needed for 1-way SSL
    javax.net.ssl.keyStore=${FMS_HOME}/keystore.FSC_fscreszone.jks
    javax.net.ssl.keyStorePassword=keystore.FSC_fscreszone.password
    javax.net.ssl.trustStore=${FMS_HOME}/keystore.FSC_fscreszone.jks
javax.net.ssl.trustStorePassword=keystore.FSC_fscreszone.password
```

$FSC_HOME/keystore.FSC_fscreszone

The keystore for this FSC. The keystore must contain the private key and certificate for the local machine, and it must also contain the trusted (CA) certificate for any clients you want to accept.

You can optionally import individual certificates for each client rather than importing the *signer certificate*.

```
fscreszone> keytool -list -v -keystore keystore.FSC_fscreszone.jks
 -storepass keystore.FSC_fscreszone.password

    Keystore type: jks
    Keystore provider: SUN
    Your keystore contains 2 entries

    Alias name: fscreszone.yourcompany.com
    Creation date: Jan 23, 2008
    Entry type: keyEntry
    Certificate chain length: 2
    Certificate[1]:
    Owner: CN=fscreszone.yourcompany.com, OU=QA, O=YOUR  Corp, L=Plano, ST=Texas, C=US
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number:
485099dcc36d1ea9d773ba153022a951
Valid from: Thu Jan 10 16:44:38 CST 2008 until: Thu Mar 27 13:20:25 CDT 2008 Certificate
fingerprints:
    MD5:  86:7E:16:59:99:E6:6F:B6:27:9B:92:19:E7:65:EB:A2
    SHA1: 6A:D1:64:7A:0A:E1:CB:62:D3:EF:91:BF:E9:A0:CE:AF:A3:3D:E4:1E
    Certificate[2]:
    Owner: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
 O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
 O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number: 1
Valid from: Wed Jul 31 19:00:00 CDT 1996 until: Thu Dec 31 17:59:59 CST 2020 Certificate
fingerprints:
    MD5:  06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
    SHA1: 62:7F:8D:78:27:65:63:99:D2:7D:7F:90:44:C9:FE:B3:F3:3E:FA:9A

    ****************************************
    ****************************************

    Alias name: thawte premium server ca
    Creation date: Feb 20, 2008
    Entry type: trustedCertEntry
    Owner: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
 O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA
    Issuer: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA,
OU=Certification Services
Division,
 O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA Serial number: 1
Valid from: Wed Jul 31 19:00:00 CDT 1996 until: Thu Dec 31 17:59:59 CST 2020 Certificate
fingerprints:
    MD5:  06:9F:69:79:16:66:90:02:1B:8C:8C:A2:C3:07:6F:3A
    SHA1: 62:7F:8D:78:27:65:63:99:D2:7D:7F:90:44:C9:FE:B3:F3:3E:FA:9A
    ****************************************
****************************************
```

Configuring Kerberos authentication on the web tier

Configure JBoss (WildFly) ISAPI with IIS for Security Services login service

You must install the Tomcat ISAPI Redirector version 1.2.31 or later and configure the Windows registry for the redirector. You must also create the **workers.properties** and **uriworkermap.properties** files for the redirector.

For additional information about the settings in these files, see the Tomcat documentation available at https://tomcat.apache.org.

1. Create a directory where you want to install the Tomcat ISAPI Redirector on the Windows Server host, for example:

   D:\jboss_iis

2. Create the a directory structure on the Windows Server host for the new web site:

   jboss_iis

   This is the top level web site directory. Its name can be anything but it is recommended that you use an easily identified name such as **jboss_iis**.

   \bin

   This is the ISAPI redirector install directory. It contains the redirector dll file and its registry file.

   \conf

   Contains the ISAPI redirector configuration files.

   \log

   Contains the ISAPI redirector log files.

   \wwwroot

   This is the physical location of the web site.

3. Download the ISAPI Redirector from https://tomcat.apache.org and save it in the ISAPI redirector install (**bin**) directory.

   - Download the latest version of the 32-bit redirector (**isapi_redirector-***version***.dll**) file, not the 64-bit redirector.

- Only the **isapi_redirector .dll** file is required.

Rename the downloaded file to **isapi_redirect.dll**.

4. Configure Windows registry settings on the Windows Server host.

   a. In the ISAPI redirector install **bin** directory, create a file with a **.reg** extension. The name of this file is discretionary (**isapi_redirector.reg** is recommended).

   b. Create an **isapi_redirect.reg** windows registry file with the following contents:

```
Windows Registry Editor Version 5.00
[HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Apache Software Foundation\
    Jakarta Isapi Redirector\1.0]
"extension_uri"="/jakarta/isapi_redirect.dll"
"log_file"="d:\\iis75-jboss71\\log\\jk_iis.log"
"log_level"="debug"
"worker_file"="d:\\iis75-jboss71\\conf\\workers.properties"
"worker_mount_file"="d:\\iis75-jboss71\\conf\\uriworkermap.properties"
”uri_select”=”unparsed”
```

   It is recommended that you use **debug** for the **log_level** entry when you initially configure the redirector to get all messages. You can change this after you have tested your installation and determined that it is working properly. The following table provides a brief description of these entries:

Name	Description
extension_uri	Represents the IIS virtual directory including the ISAPI Redirector file.
log_file	Defines the name and location of the ISAPI Redirector log file.
log_level	Defines the level of debug messages written to the ISAPI Redirector log file. Valid values are **debug**, **info**, **error**, and **emerg**.
worker_file	Defines the location of the ISAPI redirector **worker.properties** file. You create this file later.
worker_mount_file	Defines the location of the ISAPI redirector **uriworkermap.properties** file. You create this file later.
uri_select	Determines how the forwarded URI is handled. Unparsed indicates the original request URI is forwarded. Siemens Digital Industries Software recommends this option.

Name	Description
	Rewriting the URI and forwarding the rewritten URI does not work correctly.

c. In the ISAPI Redirector installation directory, right-click the **isapi_redirector.reg** file and choose **Merge**.

d. After receiving a confirmation message from Windows, check the ISAPI Redirector settings using the Microsoft Registry Editor program (**regedit.exe**) to ensure the registry settings are correct. For information about using the Microsoft Registry Editor, see the Microsoft Windows online help.

5. Create a text file with contents similar to the following:

```
# Define node1 (one node required for H_SE)
worker.list=node1
worker.node1.port=8009
worker.node1.host=host-name1
worker.node1.type=ajp13
```

The default port is **8009**. If you could not use the default value and you changed the AJP port number in JBoss (WildFly) configuration when you configured the Tomcat ISAPI Redirector, use that value. The port is set (and can be modified) in the *JBoss_home***server\default\deply\jbossweb.sar\server.xml** file.

The *host-name* value is the host where you run JBoss (WildFly).

6. Add the AJP 1.3 connector as the child resource of the **jboss:domain:web** subsystem.

a. Expand the **configuration** directory:

jboss-as-7.1.0.Final\standalone\configuration

b. Open the **standalone.xml** file.

c. Note that each connector references a particular socket binding. Add or modify the following **Connector** element:

```
…
 <subsystem xmlns="urn:jboss:domain:web:1.1" default-virtual-server="default-
host" native="false">
        <connector name="http" protocol="HTTP/1.1" scheme="http" socket-
binding="http"/>
        <connector name="ajp13" protocol="AJP/1.3" scheme="http" socket-binding="ajp"/>
        <virtual-server name="default-host" enable-welcome-root="true">
            <alias name="localhost"/>
```

```
            <alias name="example.com"/>
        </virtual-server>
    </subsystem>
    …

<socket-binding-group name="standard-sockets" default-interface="public"
port-offset="${jboss.socket.binding.port-offset:0}">
    …
    <socket-binding name="ajp" port="8009"/>
    <socket-binding name="http" port="8080"/>
     …
```

IIS forwards requests to JBoss (WildFly) using the AJP 1.3 protocol on this port. This must be set to allow access to the remote user name (**getRemoteUser** method).

This configuration supports basic H-SE redirection (with IIS 7.*x* or 8.*x* and JBoss 7.*x*), *without* authentication. This configuration does not support Tomcat authentication. If Windows authentication is enabled in IIS 7.*x* or 8.*x*, you cannot use JBoss 7.1 for the Security Services login service.

7. Save the file as **workers.properties** in the **conf** directory. This must match the path you defined for it in the registry file.

8. Create a text file with contents similar to the following:

```
# Send all /tc requests to node1
/tc/*=node1
```

Replace *tc* with the name of your Teamcenter Security Services Login Service web application. This configures the redirector to forward all requests with the **/tc/*** signature to **node1**.

9. Save the file as **uriworkermap.properties**. Save this file in the **conf** directory.

10. To open the IIS Manager, choose **Start→Administrative Tools→Internet Information Services (IIS) Manager**.

11. In the navigation tree, expand your host name entry until you see **Sites**.

12. Create a new web site with the home folder set to the directory you created in step 1:

    a. Right-click **Sites** and choose **Add a Web Site**.

    b. In the **Add Web Site** dialog box, type a name for you new web site in the **Site Name** box, for example **jboss-iis**.

    c. Click the browse button next to the **Physical path** box.

    d. In the **Browse for Folder** dialog box, browse to the **wwwroot** directory you created in step 1 and click **OK**.

e. In the **Port** box, type a unique port number (for example, **8128**) and click **OK**.

13. Configure the web site authentication:

    a. In the navigation tree, select your web site name and double-click **Authentication** under the **IIS** section.

    b. In the **Authentication** pane, select **Disabled** for **Anonymous Authentication**.

    c. Select **Enabled** for **Windows Authentication**. This is the 401 negotiate setting.

    d. Under **Actions** in the right pane, click **Providers** and ensure **Negotiate** and **NTLM** are in the **Enabled Providers** box. If they are not, select them from the **Available Providers** list and click **Add**. These settings configure IIS to attempt to authenticate using Kerberos and fall back to NTLM if Kerberos authentication is unsuccessful.

    e. Under **Actions** in the right pane, click **Advanced Settings** and ensure **Enable Kernel-mode authentication** is selected.

14. Configure the web site ISAPI filters:

    a. In the navigation tree, click your web site name and double-click **ISAPI Filters** in the **IIS** section.

    b. In the right pane, click **Add** under **Actions**.

    c. In the **Add ISAPI Filter** dialog box, type **jkfilter** in the **Filter name** box, browse to the **isapi_redirect.dll** file in the **Executable** box, and click **OK**.

15. Create a virtual directory for your web site:

    a. In the navigation tree, right-click your web site name and choose **Add Virtual Directory**.

    b. In the **Add Virtual Directory** dialog box, set **Alias** to **jakarta**. (The alias value can be anything but it must match the first value in the **extension_uri** entry in the **isapi_redirect_reg** file.)

    c. Browse to the **d:\jboss_iis\bin** directory in the **Physical path** box and click **OK**.

16. Configure a handler mapping:

    a. In the navigation tree, click your web site name and double-click **Handler Mappings** in the **IIS** section.

    b. In the right pane, double-click **ISAPI-dll** under **Actions**.

c. In the **Edit Module Mapping** dialog box, type ***** in the **Request path** box (remove any existing entry) and browse to the **isapi_redirector.dll** file in the **Executable** box.

d. Click **Request Restrictions** and click the **Verbs** tab in the **Request Restriction dialog** box and ensure the *All verbs* option is selected.

e. Click the **Access** tab, ensure the **Execute** option is selected, and click **OK**.

17. If your redirector is a 32-bit dll, enable 32-bit applications:

a. In the navigation pane, select **Application Pools**.

b. Select your web site name and click **Advanced Settings** under **Edit Application Pool** in the right pane.

c. In the **Advanced Settings** dialog box, select **True** for **Enable 32-Bit Applications**.

18. In the right pane, click **Restart** under **Manage Web Site**.

B. Troubleshooting four-tier architecture deployment

Identify the problem you encountered in your four-tier rich client architecture and perform the solution described.

Problem	Solution
Cleaning FIFO entries in **/tmp/tctp** disables server manager, MUX, and TcServer processes.	On Linux hosts, if the server manager is running when the **/tmp** directory is cleaned up by deleting its entries, Teamcenter Transfer Protocol (TCTP) is disabled. Running TcServers cannot accept new requests. The server manager no longer accepts server ready health notifications, so new servers are not published, and new user sessions will get a "no servers available" error. In some customer environments and some operating systems, including Redhat Linux, the **/tmp** directory may be automatically cleaned up periodically at a time other than boot time, particularly files that have not been used recently. Also, the **/tmp** directory may be mapped to memory, and need to be cleaned up often. See the **tmpwatch** command, which is often run as a cron job. To configure the location of the TCTP FIFO entries to a directory not monitored by **tmpwatch**, set the **TC_PIPE_NAME_PREFIX** environment variable to the location of the FIFO entries, to avoid locations that are automatically cleaned.
Out-of-memory error during a call to **getAttrMappingsForDatasetType**	If you use WebSphere and this occurs when launching NX from the rich client, you must **modify the JVM arguments** in WebSphere to increase memory allocation.
Error messages about the server manager pool ID	These messages indicate that the pool ID is in use by another server manager in the cluster. Either place the server managers in different clusters or configure a distinct pool ID.
Configuration is correct, but run-time errors occur	Determine from logs whether users are frequently losing a server due to the server timing out and are then having a new server assigned. Server startup can consume a great amount of CPU. Consider increasing timeout values and/or the pool size.
CFI_error displays when running AIE export in batch mode	When you run AIE Export in batch mode, Teamcenter displays a CFI error. This error occurs because **jt.exe** (Microsoft Task Scheduler) file is missing from the **%WINDOWS%** directory. To resolve this problem, download the Microsoft Task Scheduler from the Microsoft Developer Network:

Problem	Solution
	https://msdn.microsoft.com
During peak activity, the web tier encounters errors obtaining JCA connections.	The Teamcenter web application is using all available connections in the connection pool. To avoid this, increase the number of available connections by increasing the **Max_Capacity** context parameter value in the web application WAR file. To set your web application maximum connection pool size: 1. Launch the Web Application Manager (**insweb**) for Windows or Linux. 2. Select the web application name and click **Modify**. 3. In the **Modify Web Application** dialog box, click **Modify Context Parameters**. 4. In the **Modify Context Parameters** dialog box, locate **Max_Capacity**, double-click the **Value** column, and type a larger number. 5. Click **OK** and click **Generate Deployable File**. 6. In the **Generate Deployable File** dialog box, click **OK**. The Web Application Manager displays the status of the installation in the **Progress** dialog box. When the installation is complete, click **OK** to close the **Progress** dialog box. 7. Click **OK** to close the **Modify Web Application** dialog box. 8. Redeploy the WAR file in your application server.
Chinese characters are displayed as square blocks in the Teamcenter rich client.	If you use a nonnative language operating system version of Windows, you must install and enable the Multilingual User Interface (MUI) pack to ensure the language font is displayed properly. 1. Download and install the MUI pack for Windows from Microsoft. 2. Open the **Regional and Language Options** dialog box in the Windows Control Panel. 3. In the **Languages** tab, set the required language for the menus and dialogs.

Problem	Solution
	4. In the **Advanced** tab and the **Regional Options** tab, set the required language.

Teamcenter web application fails to deploy on JBoss (WildFly) with the following error message:
 	
`Did not receive a response to the deployment operation within the allowed timeout period [60 seconds]. Check the server configuration file and the server logs to find more about the status of the deployment.`
	The Teamcenter web application takes longer than the default 60 seconds the JBoss (WildFly) deployment scanner allows for deployments. Add the **deployment-timeout** attribute to the **deployment-scanner** element and set the value to at least **600** seconds before attempting to deploy the web application.
	
	`<subsystem xmlns="urn:jboss:domain:deployment-scanner:1.1">`
	`  <deployment-scanner path="deployments"`
	`  relative-to="jboss.server.base.dir" s`
	`  scan-interval="5000"`
	`  deployment-timeout="600"/>`
	`</subsystem>`

Long running service request that crosses firewalls or proxy servers results in closed connections.	If a user is performing a time-consuming action such as running a large BOM expansion, the server may not respond for 15 minutes or more. When this happens across a firewall, or other proxies, the firewall might automatically close the perceived idle connection. This results in a closed connection in the client application and loss of data. To avoid exceeding these idle connection time limits, enable TCP keepalive functionality in the operating system (OS) of at least one of the machines on the client or server side of the each of the HTTP connections between the client applications and the Teamcenter server. For example: • If a client machine connects to web tier machine, enable TCP keepalive in the OS of the machine where the web tier server runs. This supports both the HTTP connection between client applications and the web tier, and the HTTP connection between the web tier and the Teamcenter server (Server Manager/MUX). • If you use a reverse proxy server between a client machine and the web tier machine, enable TCP keepalive in the OS of the machine where the reverse proxy runs. If your network configuration requires you to *not* enable TCP keepalive on the TCP endpoint (such as a proxy server), you must enable keepalive in the OS on each *client* machine. On Windows machines, enable TCP keepalive by setting the appropriate Windows registry keys. On Linux machines, set TCP

Problem	Solution
	keepalive using kernel parameters. See your operating system documentation for information on how to enable TCP keepalive. Note: TCP keepalive is enabled in Teamcenter client and web tier software by default, and only requires TCP keepalive in the OS of affected hosts to be enabled. Alternatively, if you do not want to enable TCP keepalive, you can increase the timeout setting in the firewall to allow requests to complete.

C. Tuning WebSphere JVM memory consumption

If your Teamcenter application requires more memory than what is currently allocated in WebSphere, out-of-memory errors can occur. For example, if you use the NX Integration and attempt to launch NX from the rich client, Teamcenter may report an out-of-memory error during a call to **getAttrMappingsForDatasetType**.

If errors like this occur, you must modify the JVM arguments in WebSphere to increase memory allocation. For information about how to modify JVM arguments, see the IBM support article titled *Setting generic JVM arguments in WebSphere Application Server* at the following site:

http://www-01.ibm.com

Before you tune JVM arguments, use memory profiling tools to analyze your memory issues and determine which tuning options you need to use. The following table provides some suggestions, but these may not be suitable in all cases.

JVM options for tuning the WebSphere Application Server memory usage

JVM option	Description	Typical default value	Suggested value
-Xms	Controls the initial size of the Java heap. Properly tuning this parameter reduces the overhead of garbage collection, improving server response time and throughput. For some applications, the default setting for this option may be too low, resulting in a high number of minor garbage collections.	50 MB	512 MB
-Xmx	Controls the maximum size of the Java heap. In general, increasing the minimum/maximum heap size can improve startup, reduce the number of garbage collection occurrences, and increase the throughput until the heap no longer resides in physical memory. After the heap begins swapping to disk, Java performance suffers drastically. Therefore, The heap sizes should be set to values such that the maximum amount of memory the VM uses does not exceed the amount of available physical RAM.	256 MB	1024 MB
-XX:PermSize	Sets the section of the heap reserved for the permanent generation of the reflective data for the JVM. This setting should be increased to optimize the	Client: 32 MB	128 MB

JVM option	Description	Typical default value	Suggested value
	performance of applications that dynamically load and unload many classes. **PermSize** memory consumption is in addition to the **-Xmx** value set by the user on the JVM options. Setting this to a value of 128 MB eliminates the overhead of increasing this part of the heap.	Server: 64 MB	
-XX:MaxPermSize	Allows for the JVM to be able to increase the **PermSize** setting to the amount specified. Initially, when a VM is loaded, the **MaxPermSize** is the default value, but the VM does not actually use that amount until it is needed. If you set *both* **PermSize** and **MaxPermSize** to 256 MB, the overall heap increases by 256 MB in addition to the **-Xmx** setting. If an application needs to load or reload a large number of classes, the following error may result: `messageOutOfMemoryError: PermGen space` Typically, this means that the JVM started with an insufficient maximum value for permanent generation.	N/A	256 MB