

WebSphere Application Server v8.5 - Liberty Profile compared to Tomcat 7



Make an informed decision when choosing your application serving environment

Contents

- 1 Introduction
 - 3 Quick comparison
 - 7 Feature comparison
 - 12 Administrative tasks
 - 19 Performance
 - 21 Summary
-

Introduction

The goal of this document is to assist you in making an informed decision when choosing between IBM® WebSphere® Application Server Version 8.5 - Liberty Profile and Apache Tomcat 7 as an application serving environment. While both servers provide a specification-compliant servlet container, there are also significant differences that this paper will describe.

What is Liberty?

The Liberty profile is a recently-developed dynamic profile for WebSphere Application Server v8.5. The profile enables rapid development and deployment of web applications in a simpler and lightweight manner. The Liberty profile provides fidelity to the WebSphere Application Server full profile, enabling applications to be developed and tested against the WebSphere Application Server v8.5 - Liberty profile and enabling applications to be deployed in production against the full profile of WebSphere Application Server (including WebSphere Application Server 8.0) without any runtime behavioral differences. You can also deploy applications to a WebSphere Application Server v8.5 - Liberty profile server for production, with performance that is similar to that of the full-profile WebSphere Application Server.

The Liberty architecture is inherently composable; users can configure at a fine-grained level the components that are needed by their applications, so the server is very fast to start and the footprint remains minimal.



Features that can be optionally configured include a number of Java Enterprise services—for example, servlet, JavaServer Pages (JSP) and JavaServer Faces (JSF). You can also optionally configure security services and OSGi application support.

What is Tomcat?

Tomcat is an open-source web application server from the Apache Foundation. Tomcat started as the official reference implementation for Java servlet and JSP technologies. Tomcat consists of three open source components: a servlet container (Catalina), a JSP engine (Jasper) and an HTTP Transport (Coyote). Although Sun Microsystems (now Oracle) currently provides the official Java EE reference implementation (known as GlassFish), Tomcat remains compliant with the servlet 3.0 specifications and with the JSP 2.2 specifications.

Importantly, this server's scope has not changed. There is no support for other services or for application types beyond servlet and JSP.¹ Third-party libraries typically provide the capabilities beyond this scope—capabilities that would allow Tomcat to be compared on a more equal footing to Java EE containers such as WebSphere Application Server. The integration of these third-party libraries is left to the consumer, with compatibility and integration remaining the developer's responsibility. Support for Tomcat itself is provided by the Tomcat community through Internet forums. In addition, the support for Tomcat and its various derivatives can be purchased from a number of third-party vendors such as VMware Spring TCserver, Mule TCat and other vendors.

“Urban myths”

Tomcat is often confused with the Apache HTTP Web Server. Tomcat is a pure Java implementation of a web server and web container which can run Java applications, whereas the Apache HTTP Web Server is an implementation of a web server written in C language.

Some people believe that Tomcat is integrated inside WebSphere Application Server. WebSphere Application Server Community Edition is based upon Apache Geronimo, which does include Tomcat, but the other editions of WebSphere Application Server (including the Liberty profile) have their own web container which is Java EE compliant and which also supports OSGi applications.

When you consider WebSphere Application Server, it is important to understand the differences between the Community Edition and the Commercial Editions:

- WebSphere Application Server Community Edition is an open-source application server based on Apache Geronimo. Community Edition is provided by IBM and is available at no charge for development and production use. IBM offers a support option for a fee.

(Visit: <http://www-01.ibm.com/software/webservers/appserv/community/#>)

- The commercial editions of WebSphere Application Server are not open source. The commercial editions share a common code base with each other but not with WebSphere Application Server Community Edition. Of the commercial WebSphere Application Server editions, WebSphere Application Server for Developers is available for no charge for use on development machines and offers a for-fee support option. The other editions of WebSphere Application Server are licensed with PVU or socket-based pricing.

(Visit: <http://www-01.ibm.com/software/webservers/appserv/was/>)

Scope

This document has been written with the developer's perspective in mind. The document will present a feature comparison between the IBM WebSphere Application Server Liberty profile and Tomcat.

Quick comparison

1. Installing Liberty and Tomcat

Both the WebSphere Application Server Liberty profile (Liberty) and Tomcat are designed to be obtained and installed very easily.

1.1 Obtaining

Liberty and Tomcat can be downloaded from their respective web sites. Liberty is a 46 MB installable jar, and Tomcat is 8 MB. Liberty provides all the functions that are required by most web and OSGi applications.

Tomcat is only a web servlet and JSP container; its footprint will increase² as third-party libraries are added to achieve functional equivalency to that provided by Liberty (see section 3. Feature comparison). Liberty profile makes it possible for you to create and run multiple servers from the same installation directory, with each server sharing the same set of Liberty runtime binaries. To use multiple servers with Tomcat you need multiple copies of the installed binaries—or you must handcraft new scripts for startup and for shutdown, changing \$CATALINA_HOME and pointing to a different server.xml file.³

1.2 Archive installation

1.2.1. Liberty and Tomcat

To install Liberty or Tomcat, just extract the downloaded archive into your chosen directory. Extraction will take only a few seconds. The Tomcat installation includes a default server configuration.⁴ Liberty will create a default server upon first use of the “server” command in the bin directory.

1.3 Managed installation

1.3.2 Liberty

Liberty can be installed using IBM Installation Manager. IBM Installation Manager makes it possible for an administrator to install Liberty, download the Liberty profile and to apply

fixes and upgrades that are distributed by IBM. You can roll back changes if required and you can uninstall Liberty if desired. Liberty can also be installed or uninstalled (and started or stopped) on remote machines using the Job Manager in WebSphere Application Server Network Deployment v8.5. See the Feature Comparison section for a detailed discussion.

1.3.2 Tomcat

There is no direct equivalent to installation manager in Tomcat.⁵ An automated installation could be achieved by creating an installation script using Secure Shell (SSH), thus “handcrafting” a strategy for backup and restore. Some of this functionality can be provided by Mulesoft’s Tcat⁶, Puppet Labs’ Puppet⁷, Opscode’s Chef⁸ and others.

1.4 Upgrade

1.4.1 Liberty

Depending on the installation type, Liberty can be upgraded automatically using IBM Installation Manager or can be upgraded manually by obtaining a newer distribution archive; the existing server configurations can be used without modification with the updated runtime.

1.4.2 Tomcat

Tomcat is only provided as a zip file, so any update must be manual.⁹ Any added third-party libraries must be updated individually, requiring manual checks for compatibility with Tomcat and any other third-party libraries that are used. This action also requires manual handling of any necessary changes that are needed in configuration files.

1.5 Prerequisites

Both Liberty and Tomcat require Java version 6 or later; both Liberty and Tomcat will run with any compliant Java 2 Platform Standard Edition (J2SE) implementation, which creates a wide choice of operating systems.

1.6 Summary table

Actions	Liberty	Tomcat	Notes
Archive extract installation	Yes	Yes	
Managed installation	Yes (1)	No (2)	1.WebSphere Application Server Liberty Profile can be installed with IBM Installation Manager 2.Apache Tomcat can be installed with products such as Mulesoft's Tcat, Puppet Labs' Puppet or Opscode's Chef
Upgrade	Yes (3)	Yes (4)	3.WebSphere Application Server Liberty profile can be automatically upgraded using IBM Installation Manager or can be manually upgraded 4.Care must be taken to prevent breakage of third-party libraries

2. Development environment

Both Liberty and Tomcat can be used in Eclipse JavaEE as servers. Here we compare the installation procedure for each of these approaches.

2.1 Liberty

IBM WebSphere Application Server v8.5 - Liberty Profile Developer Tools for Eclipse is available from the Eclipse Marketplace or from the WebSphere Application Server Developer Community website (wasdev.net). These tools can be installed into the Eclipse integrated development environment for Java EE Developers 3.7.1 (Indigo) and in later versions using the standard Eclipse update function. This plugin provides a lightweight set of tools for developing, assembling and deploying applications to Liberty. When a Liberty profile server definition is first created in Eclipse, the tools' plugin can also download and install the runtime, if required.

The Liberty tools include these capabilities:

1. Extensive integration between tools and runtime.
 Configuration is shown directly in the server's view, the console links that help you to fix problems are revealed, and more.
2. Form-based configuration editor, including support for variables and references.

3. Support for runtime utilities such as package, generating *plugin-cfg.xml*, SSL certificate generation and dump.
4. Shared library support.
5. Support for managing multiple runtimes, servers and shared configuration snippets—(Runtime Explorer).

The rest of the WebSphere Developer Tools for Eclipse include these capabilities:

1. Enhanced JavaEE tools including form-based editors for data-definition and Java Persistence API (JPA), Enterprise navigator, validation and "quick fixes."
2. Web and mobile tools, including a rich page editor (a WYSIWYG HTML and JSP editor), support for Dojo, improved JavaScript tools and debug abilities.
3. Full OSGi developer tools, including blueprint and application editors, along with bundle dependency viewer.
4. WebSphere Application Server programming model support: WebSphere Application Server JPA platform; form-based bindings and extension editors.

2.2 Tomcat

Tomcat can be used from Eclipse by installing Eclipse integrated development environment for Java EE developers. Tomcat is part of the default list of servers available in Eclipse JEE.¹⁰ During the server set-up, you will need to point to an existing Tomcat installation on the machine. If you have not already downloaded the Tomcat runtime, the Eclipse server wizard provides a button to download and install the Tomcat runtime.

2.3 Summary Table

Function	Liberty	Tomcat	Notes
Eclipse JavaEE server	Yes (1) (2)	Yes	1. Plug-in can be obtained from Eclipse Marketplace 2. Enhanced functions with WebSphere Application Server v8.5 - Liberty Profile Developer Tools for Eclipse and IBM Rational® Application Developer Tools tool sets

3. Configuration

Both Liberty and Tomcat are configured using a server.xml file; both environments support inclusion of other xml files in the server.xml, making it possible for the configuration to be constructed in a modular fashion. Both Liberty and Tomcat environments support the use of Eclipse to edit and configure the servers—and once installed, both environments support the typical Eclipse “run on server” operation.¹¹

3.1 Liberty

Liberty configuration is sparse, which means that a full set of configuration defaults are built into the runtime. These defaults have been designed to make the server work well in the development environment. The default server created by the Liberty runtime includes servlet and JSP support, and will contain the configuration for the default HTTP ports to make them easier

to change. Additional features are typically added with a single line in server.xml. The configuration of Liberty through the server.xml file has been designed to be as concise and simple as possible. The Liberty runtime monitors its configuration file(s), and any changes will be applied automatically to the runtime. You have no need to restart the server.

3.2 Tomcat

The Tomcat server.xml is supplied already configured with nearly all of Tomcat’s features enabled, coming up at 142 lines long.¹² Ideally, before deploying applications to Tomcat, users need to review the server.xml. Users must disable features that are not required. Any change to the server.xml requires a restart of the Tomcat server.¹³

3.3 Summary Table

Features	Liberty	Tomcat	Notes
Concise, minimal configuration	Yes	No	
Flexible, modular configuration	Yes	Yes	
Dynamic updates to server.xml	Yes	No (1)	1. Must be restarted after changes are made to server.xml

4. Application deployment

The application deployment experience in Eclipse is similar for both Liberty and Tomcat. WebSphere Application Server (and hence the Liberty profile) and Tomcat offer server specific application programming interfaces (APIs) that go beyond the standard JavaEE offering.¹⁴ Since Liberty is a profile of WebSphere Application Server, applications using WebSphere Application Server APIs on the Liberty profile run the same on the full profile of WebSphere Application Server.

4.1 Web applications

The Eclipse JavaEE perspective provides the developer with a web-application layout view. This creates the necessary default files and makes it possible for the application to be deployed and to be run on the server in an instant.

4.2 Deployment

Liberty and Tomcat appear in the servers view; the servers can be configured, started and stopped very easily. Liberty has a slight advantage in terms of deployment. As has been explained,

4.3 Summary Table

Function	Liberty	Tomcat	Notes
Application development	Yes	Yes	
Easier deployment	Yes (1)	Yes (1)	1. Hot deploy directories are available
Applications in configurations	Yes	Yes	
Dynamic updates to server.xml	Yes	No	

the Liberty server does not need to be restarted after a change has been made to its server.xml. This means that the server can remain running while the application is being developed, modified and redeployed. Outside of Eclipse, the developer can deploy applications to Tomcat by copying the .war file into the BHAGAYAtomcat-installSACHIN/webapps directory.

Using the Liberty runtime, .war, .ear, .eba and .wab application files can be copied into the server's <liberty-install>/usr/servers/<servername>/dropins directory in order to be deployed, or these file types can be specified directly as applications within the server.xml configuration file.

Tomcat can also define applications using its configuration file by adding a sub-context element to *host* in server.xml, but the user must restart the server to pick up this application.¹⁵

Feature Comparison

As already discussed, the Liberty profile is supplied with many JavaEE features, whereas Tomcat is essentially just a servlet container. The following table helps you to compare the various features of the two environments.

1. Basic functionality

Functionality	Liberty	Tomcat	Notes
Servlet 3.0	Yes	Yes	
JSP 2.2	Yes	Yes	
Secure Socket Layer (SSL)	Yes	Yes	
Federal Information Processing Standard (FIPS) 140-2	Yes	Yes	
FIPS 800-331	Yes	No	
Java Database Connectivity (JDBC)	Yes	Yes	
Lightweight Directory Access Protocol (LDAP)	Yes	Yes	
Java Naming and Directory Interface (JNDI)	Yes	Yes	
Java Management Extensions (JMX)	Yes (1)	Yes	1. has local and remote Representational State Transfer (REST) connectors
Shared libraries	Yes	Yes	Different concepts; see next section
Servlet security	Yes	Yes	
JavaServer Faces (JSF)	Yes	No (2)	2. Apache MyFaces
Java Persistence API (JPA)	Yes	No (3)	3. Apache OpenJPA
Java Transaction API (JTA)	Yes	No (4)	4. Apache Geronimo Transaction, JOTM
Web archive (WAR) application	Yes	Yes	
Enterprise archive (EAR)	Yes (5)	No (5a)	5. No Enterprise JavaBeans (EJB) support (5a) Apache OpenEJB
Web Application Bundle (WAB)	Yes	No (6)	6. Only in Apache Geronimo
Enterprise Bundle Archive (EBA)	Yes	No (7)	7. Only in Apache Geronimo
Bean validation	Yes	No (8)	8. Apache Bean Validation
IBM z/OS® support	Yes	No (9)	9. Dovetailed Technologies T:Z
iSeries support	Yes	Yes	

There are also many runtime features in Liberty including JAX-RS, OSGi-JPA, web security functions and a number of IBM z/OS exploitation features.

2. Java virtual machine (JVM) comparisons

Here are some of the differences between JVMs (other than the support factor):

Function	Liberty Profile (IBM J9)	Tomcat (Hotspot Open JDK)	Notes
Faster garbage collection for large heap sizes (>4 GB) -xgcpolicy: balanced	Yes	No	
System-class data sharing for reduced memory usage and faster startup	Yes	Yes (1)	1. Client only
Application-class data sharing for smaller memory usage and faster startup	Yes	No	
JVM restarted when PermGen fills up	Yes	No	
Compressed 64-bit references for faster runtime and smaller memory	Yes	Yes (2)	2. Recent addition
Dump analyzer for hang, crash and memory management	Yes	Yes	IBM's dump analyzer is better
Garbage collection and memory visualizer for monitoring memory usage and performance	Yes	No	
Memory analyzer for troubleshooting memory leaks and excessive heap consumption	Yes	Yes	
Health center for near-real time monitoring of running virtual machines	Yes	No	

3. JDK 1.6 and 1.7 for FIPS compliance

The Federal Information Processing Standard (FIPS) Publication 140-2, FIPS PUB 140-2, is a US government computer-security standard that is used to accredit cryptographic modules. The title is Security Requirements for Cryptographic Modules. Initial publication was on May 25, 2001 and the most recent update was December 3, 2002.¹⁶ There is a more recent FIPS draft publication that provides more-specific guidance for transitions to stronger cryptographic keys and more robust algorithms—Special Publication (SP) 800-131 *A Framework for Designing Cryptographic Key Management Systems*.¹⁷

3.1 Liberty

Liberty can be run with a FIPS-compliant JDK without any impact to its functionality or to any deployed applications. Liberty also supports the enhanced FIP 800-131.

3.2 Tomcat

Tomcat FIPS compliance was developed for version 7.0.23 and later, and was back-ported to version 6.0.36.¹⁸ When using Tomcat in FIPS-compliant mode, you must take care that these minimum versions are used, and that any third-party libraries are compatible with these relatively recent versions of Tomcat.

3.3 Summary Table

Feature	Liberty	Tomcat	Notes
FIPS 140-2	Yes	Yes	
FIPS 800-131	Yes	No	

4. Java Transaction API (JTA)

Java Transaction API (JTA) specifies standard Java interfaces between a transaction manager and the parties that are involved in a distributed transaction system: the resource manager, the application server and the transactional applications.

4.1 Liberty

Support for recoverable “eXtended Architecture” (XA) transactions is provided by the WebSphere Application Server transaction manager.

4.2 Tomcat

There is no native support for JTA in Tomcat.¹⁹ Third-party libraries must be used to provide equivalent functionality.

4.3 Summary Table

Feature	Liberty	Tomcat	Notes
JTA	Yes	No (1)	1. Use Apache Geronimo Transaction, JOTM

5. Java Persistence API (JPA)

The Java Persistence API provides a mapping facility that is object and relational. Java developers use this mapping facility to manage relational data in Java applications.

5.1 Liberty

When you want to enable the Liberty profile to support an application that uses the Java Persistence API (JPA), you add the `jpa-2.0` feature to the `server.xml` file. You also need to define persistence contexts and persistence units, and you must configure access to the entity manager and to the entity-manager factory. The `jpa-2.0` feature provides support for applications that use application-managed and container-managed JPA that is written to the JPA 2.0 specification. Common with the WebSphere Application Server full-profile, support is built on top of Apache OpenJPA, with extensions to support the container-managed programming model.

5.2 Tomcat

JPA is not part of Tomcat; to support this functionality, you would need to use something such as Apache OpenJPA and then integrate it to your Tomcat installation.

5.3 Summary Table

Feature	Liberty	Tomcat	Notes
JPA	Yes	No (1)	1. Use Apache OpenJPA

6. Java Database connectivity (JDBC)

The JDBC application programming interface was designed to keep simple things simple. This means that the JDBC makes everyday Database tasks easy. You can use JDBC to execute common Structured Query Language (SQL) statements, and to perform other objectives that are common to Database applications.

6.1 Liberty

To enable the Liberty profile to support an application that uses JDBC, you add the `jdbc-4.0` feature to the `server.xml` file, along with configuration that tells the server the location of the JDBC driver and specifies properties that the JDBC driver uses to connect to the Database. Robust connection management is provided by WebSphere Application Server (including provision in the Liberty profile).

6.2 Tomcat

The default data-source support in Tomcat is based upon the JDBC Connection Pool `org.apache.tomcat.jdbc.pool`.²⁰ This pool is a replacement for (or an alternative to) the Apache Commons DBCP project, which Tomcat used in the past. The official Tomcat documentation has not been updated to reflect this change.²¹ However, it is possible to use any other connection pool that implements `javax.sql.DataSource`; accomplish this by writing your own custom resource factory. You must modify the web application deployment descriptor

(/WEB-INF/web.xml) to declare the JNDI name under which you will look up the preconfigured data source. Then, modify the server.xml to add a resource factory of type java.sql.DataSource.

6.3 Summary Table

Feature	Liberty	Tomcat	Notes
JDBC	Yes	Yes	

7. LDAP

The Lightweight Directory Access Protocol (LDAP) is an application protocol for accessing and maintaining distributed directory information services over an Internet Protocol (IP) network.

LDAP is often used for authentication.

7.1 Liberty

You can configure a LDAP server with the Liberty profile for authentication. To do this, you add the appSecurity-1.0 server feature to the server.xml file, and specify in the server.xml file the configuration information for connecting to the LDAP server.

7.2 Tomcat

You can configure LDAP on Tomcat using the JNDIRealm²², which is an implementation of the Tomcat Realm interface. This Tomcat Realm interface looks up users in an LDAP directory server that is accessed by a JNDI provider (typically, the standard LDAP provider that is available with the JNDI API classes). The realm supports a variety of approaches to using a directory for authentication.

7.3 Summary Table

Feature	Liberty	Tomcat	Notes
LDAP	Yes	Yes	

See the Administration tasks section of this document for more-detailed information about LDAP.

8. Shared libraries

A shared library is a Java archive (JAR) file, or is a collection of files that can be used by multiple web applications. Both Liberty and Tomcat support “global” shared libraries, as required by the servlet specification.

A shared library can be placed in a particular directory; `<liberty-install-dir>/usr/shared/lib/global` on Liberty and `<tomcat-install-dir>/common/lib` on Tomcat. The library will be visible to all applications that are deployed in the server; all applications will access the same instances of those classes.

While the global shared library function can be useful in some scenarios, it does have two major limitations:

- All applications have visibility to the same static variables in the library classes (so the applications are not isolated)
- It is not possible to provide different versions of the same library for use by different applications.²³

To avoid those limitations in Tomcat, it is necessary to package a distinct copy of the required library within each application that needs the library. Liberty provides additional shared-library options that provide greater flexibility depending on how the shared library is used.

8.1 Liberty

There are three types of shared libraries in Liberty:

1. Global shared libraries (described earlier in this document) that allow all applications to share a common instance of the library
2. Configured common libraries that make it possible for specific applications to share a common instance of the library
3. Configured private libraries that make it possible for specific applications to use private instances of the library

You can configure a collection of classes and JAR files in the `server.xml` as a named shared library. The library can then be added to the classpath of an application through the application's configuration entry, where the use of the library by that application is also defined as "common" or "private." Typically, a shared library is either copied into a directory in the Liberty installation, or the shared library is referred to directly from its own installation path.

Here is an example entry that is added to the server configuration (`server.xml`) file:

```
<library id="AppSharedLibrary">
<fileset dir="${shared.resource.dir}/lib"
includes="*.jar" />
</library>
```

The library can be made accessible from applications using the `classloader` child element of the application elements:

```
<application location="MyFirstApp.war">
<classloader commonLibraryRef=
"AppSharedLibrary" />
</application>
<application location="MySecondApp.war">
<classloader privateLibraryRef=
"AppSharedLibrary" />
</application>
```

MyFirstApp will share its copy of the *AppSharedLibrary* classes with any other applications that specify *MyFirstApp* as a `commonLibraryRef`. *MySecondApp* will have its own private copy of the *AppSharedLibrary* classes. Common libraries are useful in cases in which different versions of a library should be made visible to different applications, but the same instance of each version can be safely shared by the consuming applications, thus limiting the memory use of the library classes.

Private libraries are useful when a shared library instance must be isolated for each application, or if the library must be able to access the application classes.

8.2 Summary Table

Feature	Liberty	Tomcat	Notes
Global shared libraries	Yes	Yes	
Different versions of same library	Yes	No	
Private instances of library classes	Yes	No	

9. OSGi applications

A Web Application Bundle (WAB) is a bundle that contains a web application and that can be deployed in a web container that supports OSGi dynamic-modularity semantics. WABs are defined as part of the OSGi Enterprise Specification. A WAB can most simply be thought of as the OSGi bundle version of a web application archive (WAR) file. A WAB uses the OSGi bundle classpath, rather than the traditional JavaEE ordering of `WEB-INF` or `classes`, `jars` in `WEB-INF` or `lib`. A WAB can contain servlets, JSF, static content or JavaServer Pages (JSPs).

An enterprise bundle archive (EBA) file contains a set of OSGi bundles that are deployed as a single OSGi application, and that are isolated from other OSGi applications. Each OSGi

application runs in its own isolated OSGi framework instance, with its own OSGi service registry. Bundles in one OSGi application cannot see bundles, services or packages that are defined in another OSGi application—unless the bundles, services or packages are shared explicitly by both applications.

An OSGi application can also load packages and can consume OSGi services from a shared bundle space—that is, from the OSGi framework instance that is the parent of all the isolated framework instances of the OSGi applications.

9.1. Liberty

Liberty is built on an OSGi framework and Liberty supports the deployment of OSGi applications. Web Application Bundle (WAB) and enterprise bundle archive (EBA) are supported as OSGi application types.

9.2. Tomcat

Tomcat is not an OSGi container and does not support OSGi applications.²⁴

9.3. Summary Table

Feature	Liberty	Tomcat	Notes
OSGi applications	Yes	No	

Administrative tasks

1. Security considerations

1.1. Liberty

To enable security for your Liberty profile, you utilize the *appSecurity-1.0 server* feature. This feature provides support for user registries, authentication and authorization. The supported user registry types are basic user registry and LDAP user registry. The *appSecurity-1.0* feature enables better security for web applications. Depending upon your security configuration, you might also need to add one or more of the following additional server features such as *ssl-1.0* which supports Secure Socket Layer (SSL) connections using the secure HTTPS listener.

Summary from the WebSphere Application Server Liberty Profile documentation

a. Authentication

a.1. Basic

You can configure a basic user registry in the Liberty profile for authentication.

Passwords in the clear, or base64 encoded.

a.2. LDAP

You can configure a Lightweight Directory Access Protocol (LDAP) server with the Liberty profile for authentication.

The following types of LDAP server are supported:

- IBM Directory Server
- Microsoft Active Directory Server

a.3. JAAS

You can add a custom Java Authentication and Authorization Service (JAAS) login module before or after you add the Liberty profile server login module.

a.4 Single Sign-On (SSO)

With Single Sign-On (SSO) support, web users can authenticate once when they access Liberty profile resources (such as HTML, JavaServer Pages [JSP] files, and servlets), or when they are accessing resources in multiple Liberty profile servers that share the same Lightweight Third Party Authentication (LTPA) keys.

a.5 TAI

You can integrate a third-party security service with the Liberty profile using Trust Association Interceptors (TAI). The TAI can be invoked before or after invocation of single sign-on (SSO). A TAI is used to validate HTTP requests between a third-party security server and a Liberty profile server. The TAI inspects the HTTP requests from the third party security server to see if there are any security attributes. If the validation for a request is successful in the interceptor, the Liberty profile server authorizes the request by checking whether the client user has the required permission to access the resources.

b. Application security

Configure a user registry for authentication on the Liberty profile server by enabling the appSecurity-1.0 server features in the server.xml file. You can configure the authorization table in the following ways: either in the server.xml file under the respective application element, or to maintain WebSphere Application Server full profile compatibility, you can add the authorization table definition to the *ibm-application-bnd.xml* or to the *ibm-application-bnd.xmi* file.

1.2. Tomcat

Tomcat provides multiple interfaces to enable security, but only one of the interfaces is recommended for production use: DataSourceRealm.²⁵ The Tomcat documentation is very specific and warns users about the others. This puts Tomcat at a serious disadvantage when you are considering security.

Summary from the official Tomcat documentation²⁶:

a. Realms

a.1. MemoryRealm

MemoryRealm is a simple demonstration implementation of the Tomcat Realm interface. It is not designed for production use. At startup time, MemoryRealm loads information about all users, and their corresponding roles, from an XML document (*tomcat-user.xml* by default). The MemoryRealm is not intended for production use as any changes to *tomcat-users.xml* require a restart of Tomcat to take effect.

a.2. JDBCRealm

JDBCRealm is an implementation of the Tomcat Realm interface that looks up users in a relational Database accessed via a JDBC driver. The JDBCRealm is not recommended for production use as it is single threaded for all authentication and authorization options. Use the DataSourceRealm instead.

a.3. DataSourceRealm

DataSourceRealm is an implementation of the Tomcat Realm interface that looks up users in a relational Database accessed via a JNDI named JDBC DataSource. The UserDatabaseRealm is not intended for large-scale installations. It is intended for small-scale, relatively static environments.

a.4. UserDatabaseRealm

UserDatabaseRealm is an implementation of the Tomcat Realm interface that uses a JNDI resource to store user information. By default, the JNDI resource is backed by an XML file. It is not designed for large-scale production use. At startup time, the UserDatabaseRealm loads information about all users,

and their corresponding roles, from an XML document (tomcat-users.xml by default). The users, their passwords and their roles may all be editing dynamically, typically via JMX. Changes may be saved and will be reflected in the XML file.

a.5. JAASRealm

JAASRealm is an implementation of the Tomcat 6 Realm interface that authenticates users through the Java Authentication & Authorization Service (JAAS) framework which is now provided as part of the standard Java SE API. The JAASRealm is not widely used and therefore the code is not as mature as the other realms. Additional testing is recommended before using this realm.

a.6. JNDIRealm

JNDIRealm is an implementation of the Tomcat Realm interface that looks up users in an LDAP directory server accessed by a JNDI provider (typically, the standard LDAP provider that is available with the JNDI API classes). The realm supports a variety of approaches to using a directory for authentication. By default, the realms do not implement any form of account lock-out. This means that brute force attacks can be successful. To prevent a brute force attack, the chosen realm should be wrapped in a LockOutRealm.

b. Application Security

You can secure Web Applications by configuring the server.xml as well as the WEBINF/web.xml for the applications you want secured. You can have a mix of secured and nonsecured applications. See http://tomcat.apache.org/tomcat-7.0-doc/config/http.html#SSL_Support for more information.

1.3. Summary Table

Feature	Liberty	Tomcat	Notes
Basic security	Yes	Yes	
LDAP	Yes	Yes	
JAAS	Yes	Yes	
SSO	Yes	No (1)	1. Implement by yourself
TAI	Yes	No	
Application security	Yes	Yes	

2. IBM z/OS integration

For highest availability, strengthened security and optimized local integration with enterprise systems, deploy web applications on IBM z/OS.

2.1. Liberty

On IBM z/OS systems, the Liberty profile provides an operations environment. You can start, stop or modify the Liberty profile using IBM MVSTTM operator commands. Liberty supports Resource Access Control Facility (RACF) user registry, System Authorization Facility (SAF) keyring, and SAF authorization. The SAF registry holds information that is needed to perform security-related functions such as authenticating users and retrieving information about users, groups, or groups associated with users. You activate and configure the SAF registry through the configuration of the Liberty server.xml. You can configure IBM z/OS native transactionality, create dumps for troubleshooting, and connect to IBM DB2[®] on z/OS using the JDBC driver. Application endpoints and methods can be classified for workload management using z/OS native capabilities.

2.2 Tomcat

There is no native support for IBM z/OS systems on Tomcat, but Dovetailed Technologies sells a product called “T:Z,” which is a distribution of Apache Tomcat that is specially packaged for easier installation, configuration and operation on the IBM z/OS® operating system.²⁷

2.3 Summary Table

Feature	Liberty	Tomcat	Notes
IBM z/OS operations	Yes	Yes (1)	1. Using a third-party commercial version of Tomcat
IBM z/OS quality-of-service exploitation	Yes (2)	No	2. z/OS security, transactions and workload management

3. Troubleshooting

Troubleshooting is important when developing and in a production environment. You must have the facilities to find out quickly what is happening on the server.

3.1. Liberty profile

3.1.1. Trace and logging

The product has a unified logging component. The Liberty profile provides base implementations of trace and First Failure Data Capture (FFDC) services, for runtime code and application code, to gather debug information. The trace and FFDC implementations apply an initial configuration during static initialization. You can modify this default initial configuration by specifying properties in the server configuration files (see Configuring the Liberty profile runtime environment) or bootstrap.properties file (see Specifying Liberty profile bootstrap properties).

Messages are written to standard output format and are written to the defined trace destination. OSGi logging output is intercepted and is output through the trace support. There is also interception of java.util.logging output. Each Liberty server produces its own logging, trace and FFDC information in the `<libertyinstall>/usr/servers/<servername>/logs` directory. A web application running on Liberty profile can perform these operations:

- Use system logging API, java.util.logging.
- Use the logging API provided by the Java servlets specification, javax.servlet.ServletContext.log(...)
- Use virtually any logging framework of your choice.

3.1.2. Capturing the status of a Liberty profile server

From the command prompt, you can use the server dump command to capture state information for a Liberty profile server. The command can be applied to either a running or to a stopped server. The server dump command is useful for problem diagnosis of a Liberty profile server because the result file contains server configuration, log information and details of the deployed applications in the work area directory. The output is in a “human-friendly” text format. For a running server, the following information is also included:

- State of each OSGi bundle in the server
- Wiring information for each OSGi bundle in the server
- Component list managed by the Service Component Runtime (SCR)
- Detailed information of each component from SCR
- Thread dump

3.2. Tomcat

3.2.1. Logging

The Tomcat server produces several logging outputs in the `<tomcat-installation>/logs` directory. Additional log files may be produced if third-party libraries have been installed and the libraries use their own logging output. For example, Apache OpenJPA logs its output in a file called `openjpa.Log` in `<tomcat-install>/logs` directory.

A web application running on Apache Tomcat can perform the following functions²⁸:

- Use system logging API, `java.util.logging`.
- Use the logging API provided by the Java servlets specification, `javax.servlet.ServletContext.log(...)`
- Use any logging framework of your choice.

You can configure the logging level in the `server.xml`, setting the property

`org.apache.catalina.level` to one of the levels available in `java.util.logging`. For third-party libraries, you might need to edit more files to configure logging. Each library might have its own syntax. Non-integrated logs make evaluating timelines of application processing difficult, because information must be collated from multiple sources.

3.2.2. Server dump

There is no built-in functionality to do an equivalent server dump (logs, trace files, JVM heapdump). By changing the Tomcat's log level to the finest setting and sending a SIGBREAK signal to the JVM in which the Tomcat server is running, similar information can be gathered manually.

3.3. Summary Table

Feature	Liberty	Tomcat	Notes
Trace and logging	Yes	Yes	
FFDC	Yes	No	
Server dump	Yes	Yes (1)	1. Not built-in, use SIGBREAK, set log levels to finest

4. Monitoring and management

Both Liberty and Tomcat have pre-deployed JMX MBeans to support monitoring and management through jconsole.

4.1. Liberty

4.1.1. JMX MBeans

Liberty has a number of MBeans deployed by default in a server. The local connector is implicitly secured (can only be used by the same operating-system userid that is running the server). The remote (REST) connector is always security-rich using simple configuration in `server.xml`. Users can develop custom Mbeans and deploy them to the Liberty server.

4.1.2. WebSphere Application Server Network Deployment Job Manager

You can administer Liberty profile servers by running jobs in the Job Manager console or the Deployment Manager console. The Job Manager component is part of the WebSphere Application Server Network Deployment edition. It can be used for automating repeatable unzip-install of packaged servers

to a group of host machines, and for managing the lifecycle of these servers thereafter. Through the Job Manager you can define a group of remote hosts and can execute a wide variety of jobs against this group. Jobs are provided for installing, uninstalling, starting and stopping WebSphere Application Server Liberty profile instances.

You can generate an “embedded server” zip file of an existing server installation plus configuration and applications. Generate the embedded server zip file by using the package command in Liberty. This action allows you to deploy multiple instances of the same server configuration throughout host machines. Each host defined to Job Manager can optionally specify host-specific variables which can be captured in the *bootstrap.properties* of each managed WebSphere Application Server Liberty profile instance. The embedded server zip can also be distributed manually (without using the Job Manager) by copying to the host machines, unzipping, then editing the configuration file to update any local values (these values can also be isolated as variables in the *bootstrap.properties* file).

4.2. Tomcat

4.2.1. JMX MBeans

Local and remote connectors can be secured by editing the JAVA_OPTS environment variable. A wide selection of MBeans is deployed by default in Catalina. A list of all the MBeans deployed in Catalina can be found at <http://tomcat.apache.org/tomcat-7.0-doc/funcspecs/mbeannames.html>. You can develop custom MBeans to perform tasks that are beyond the scope of the JMX API. The Tomcat documentation does not have a lot of information on how to develop custom MBeans, so you might need to search the Internet for examples. One such example can be viewed at <http://oss.wxnet.org/mbeans.html>.

4.2.2. Mulesoft Tcat

Mulesoft Tcat is a commercial product that provides administration of Tomcat servers; you must purchase a subscription for production use. Mulesoft Tcat provides additional capabilities for production-deployment centralized application management (including application deployment and roll-back), security, diagnostics and configuration management.²⁹

4.3 Summary Table

Feature	Liberty	Tomcat	Notes
JMX MBeans	Yes	Yes	
Management	Job Manager	Mulesoft Tcat	VMware TCserver is another option for TCat management

5. Scaling

Scaling is an important feature of Liberty and Tomcat, because Liberty and Tomcat can be deployed in a production environment.

5.1. HTTP clustering and load balancing

5.1.1 Liberty

5.1.1.1. Web server plug-in

A web server plug-in is used to forward HTTP requests from a supported web server to one or more application servers. The plug-in takes a request and checks the request against configuration data in the *plugin-cfg.xml* file. The configuration data maps the Uniform Resource Identifier (URI) for the HTTP request to the hostname of an application server. The web server plug-in then uses this information to forward the request to the application server.

5.1.1.1.1. Plugin creation

The web server plug-in *plugin-cfg.xml* file for a Liberty configuration can be generated through the defaultPluginConfig generation MBean.

5.1.1.1.2. Plugin installation

Install the generated merged *plugin-cfg.xml* file on the web server. Typically, you must enable the plug-in within the *httpd.conf* file of the web server by using the LoadModule phrase, and you must specify the location of *plugin-cfg.xml* file using the WebSpherePluginConfig phrase.

5.1.1.1.3. Workload balancing and failover

By merging the plug-in information for multiple Liberty servers, the plug-in provides workload balancing for those servers, maintaining HTTP session affinity. HTTP session state can be persisted to a Database so that if a failure occurs and the plug-in moves traffic to another server, then the session state is available on the other server.

5.1.2. Tomcat

5.1.2.1. HTTP Clustering

You can configure a cluster in Tomcat easily. By default, this configured cluster will enable all-to-all session replication using the DeltaManager (which is provided as part of Tomcat) to replicate session deltas. This means that the session gets replicated to all the other nodes in the cluster. This works well for small clusters, but the Tomcat documentation does not recommend it for larger clusters (a large number of Tomcat nodes).

Also, when you use the DeltaManager, DeltaManager will replicate to all nodes—even nodes that do not have the application deployed. To get around this problem, you will want to use the BackupManager. This manager only replicates the session data to one backup node, and only replicates to nodes that have the application deployed. The downside of the BackupManager is that it is not quite as “battle-tested” as the DeltaManager.³⁰

5.1.2.2. Load balancing

Tomcat servers can be load-balanced by using the Apache HTTP server configured with the mod_proxy module.³¹

5.2. IBM WebSphere eXtreme Scale

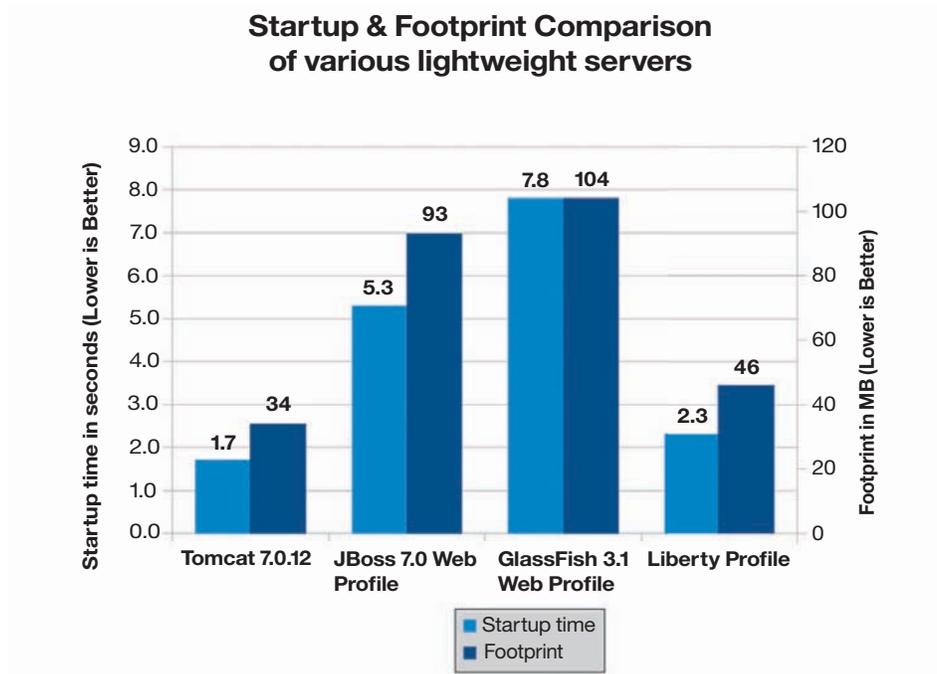
Liberty and Tomcat are supported server configurations in WebSphere eXtreme Scale v8.5. Tomcat is supported in earlier versions also. As part of the general availability of the WebSphere Application Server v8.5 release, WebSphere eXtreme Scale was improved to provide enhanced capabilities for the new Liberty profile.

5.3 Summary Table

Feature	Liberty	Tomcat	Notes
HTTP Clustering	Yes	Yes	
Load Balancing	Yes	Yes	
eXtreme Scale	Yes (1)	Yes	1. Enhanced capabilities; HTTP session persistence

Performance

The problem of a lightweight development environment in WebSphere has been solved. Liberty Profile startup and footprint are about the same level as Tomcat. And Liberty Profile starts up in less than half the time of JBoss web profile.



Note: Tomcat, JBoss and GlassFish were measured with the HotSpot JDK, while Liberty was measured with the IBM JDK.

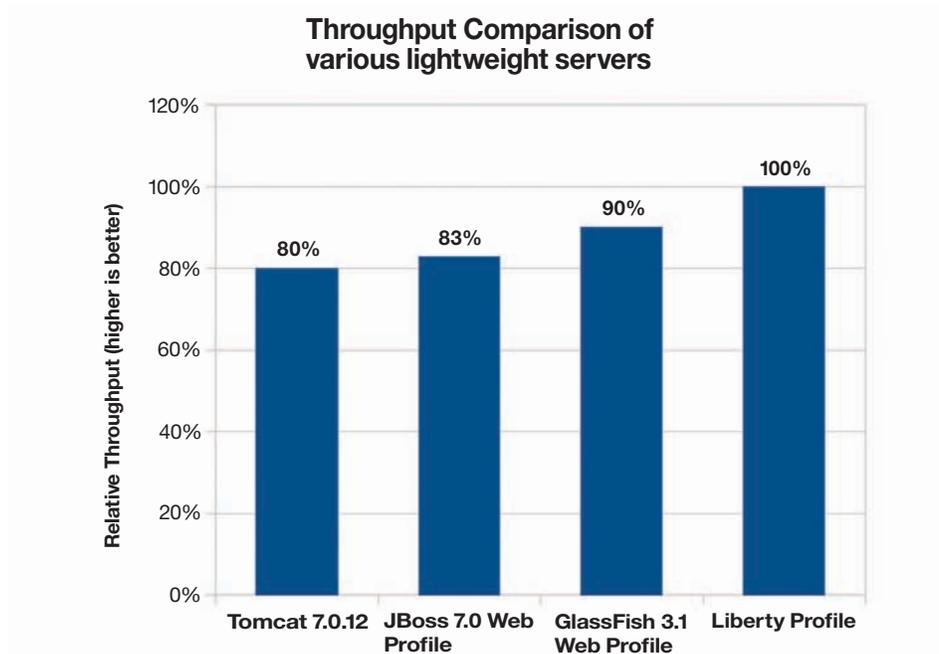
System Info:

Lenovo T60p - 2 x 2.16 GHz Intel Core Duo T2600
2GB RAM, Windows XP 32-bit
Apache Tomcat 7.0.12
JBoss Community Edition 7.0 Web Profile server
GlassFish Server 3.1 Open Source Edition Web Profile
WebSphere Application Server v8.5 - Liberty Profile

(All servers had the TradeLite benchmark application installed.)
Based on IBM internal tests. Results may not be typical and will vary based on actual configuration, applications, and other variables in a production environment.

Liberty is a lightweight server that can service requests with the speed of a full production server.

Liberty Profile provides up to 20 percent better runtime performance than JBoss and 25 percent better than Tomcat.³²



Note: Tomcat, JBoss, and GlassFish were measured with the HotSpot JDK, while Liberty was measured with the IBM JDK.

System Info:

IBM x3550 – 4 x 1.86 GHz Intel Xeon E5320, 8 GB RAM
RedHat Linux 5.3 32-bit
Apache Tomcat 7.0.12
JBoss Community Edition 7.0 Web Profile server
GlassFish Server 3.1 Open Source Edition Web Profile
WebSphere Application Server V8.5 - Liberty Profile

(All servers had the TradeLite benchmark application installed.)
Based on IBM internal tests. Results may not be typical and will vary based on actual configuration, applications, and other variables in a production environment.

Summary

Liberty and Tomcat provide a basic servlet engine, but can also be grown to provide additional Java enterprise services and related services. With Liberty, those services have already been integrated with the Liberty kernel, so the services have a common configuration file, log outputs, and more. With Tomcat, the user performs the integration of third-party libraries.

The Liberty profile default configuration is very small and is simpler—unlike Tomcat, which has most of its functionality enabled by default. Liberty features are enabled by the deployed applications as needed and when needed, keeping the server footprint as small as possible.

Liberty supports various security implementations which have been hardened and comprehensively tested in both test and production environments, whereas Tomcat's various security implementations are provided as-is and have not been thoroughly tested. Hence, Tomcat security implementations are not recommended for production deployments.

Tomcat has a slightly faster restart, although both servers start in less than three seconds. Liberty requires fewer restarts due to its dynamic configuration updates. Memory footprints are comparable and will vary in both cases, depending upon the features that are configured (Liberty) or libraries added (Tomcat). The Liberty shared library support has the potential to make significant footprint reductions when applications need different versions of the same library.

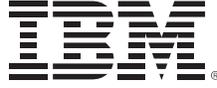
Production throughput for the JSP/JDBC workload is about 30 percent higher for Liberty³³, which could make Liberty a better choice where high-volume performance is important.

For more information

To learn more about the IBM WebSphere Application Server Liberty Profile, please contact your IBM representative or IBM Business Partner, or visit the following website:

ibm.com/software/webservers/appserv/was/ or visit the www.wasdev.net, the community forum for the Liberty profile.

- ¹ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ² From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ³ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ⁴ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ⁵ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ⁶ <http://www.mulesoft.com/understanding-apache-tomcat>
- ⁷ <http://puppetlabs.com/>
- ⁸ <http://www.opscode.com/chef/>
- ⁹ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ¹⁰ <http://www.eclipse.org/webtools/>
- ¹¹ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ¹² From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ¹³ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ¹⁴ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ¹⁵ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ¹⁶ http://en.wikipedia.org/wiki/FIPS_140-2
- ¹⁷ <http://csrc.nist.gov/publications/PubsDrafts.html#800-131>
- ¹⁸ https://issues.apache.org/bugzilla/show_bug.cgi?id=50570
- ¹⁹ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²⁰ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²¹ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²² From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²³ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²⁴ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²⁵ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²⁶ <http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²⁷ <http://www.dovetail.com/products/tomcat.html>
- ²⁸ From Tomcat documentation,
<http://tomcat.apache.org/tomcat-7.0-doc/setup.html>
- ²⁹ <http://www.mulesoft.com/tcat-leading-enterprise-apache-tomcat-application-server>
- ³⁰ <http://tomcat.apache.org/tomcat-7.0-doc/cluster-howto.html>
- ³¹ <http://tomcat.apache.org/tomcat-7.0-doc/balancerhowto.html>
- ³² Based on IBM internal tests. Results may not be typical and will vary based on actual configuration, applications and other variables in a production environment.
- ³³ Based on IBM internal tests. Results may not be typical and will vary based on actual configuration, applications and other variables in a production environment.



© Copyright IBM Corporation 2012

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
October 2012

IBM, the IBM logo, ibm.com, iSeries, MVS, Rational, System z, WebSphere, DB2, and z/OS are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at ibm.com/legal/copytrade.shtml

Intel, Intel logo, Intel Inside, Intel Centrino, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows and Windows NT are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

This document is current as of the initial date of publication and may be changed by IBM at any time. Not all offerings are available in every country in which IBM operates.

The performance data discussed herein is presented as derived under specific operating conditions. Actual results may vary.

It is the user's responsibility to evaluate and verify the operation of any other products or programs with IBM products and programs.

THE INFORMATION IN THIS DOCUMENT IS PROVIDED “AS IS” WITHOUT ANY WARRANTY, EXPRESS OR IMPLIED, INCLUDING WITHOUT ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND ANY WARRANTY OR CONDITION OF NON-INFRINGEMENT. IBM products are warranted according to the terms and conditions of the agreements under which they are provided.

The client is responsible for ensuring compliance with laws and regulations applicable to it. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the client is in compliance with any law or regulation.

Actual available storage capacity may be reported for both uncompressed and compressed data and will vary and may be less than stated.



Please Recycle