

# Generating JBoss 4 Server Instances

## Table of contents

1 Audience.....	3
2 Purpose.....	3
3 Prerequisites.....	3
4 JBoss Server Instance Generation.....	3
4.1 High Level Overview.....	3
4.2 Overview.....	5
4.3 Specifying generated server(s) name(s).....	6
4.4 Specifying non-standard port values.....	7
4.5 Specifying server name in XML Properties file.....	8
4.6 Specifying application family name in XML Properties file.....	8
4.7 Specifying server-specific JVM arguments.....	8
4.8 Enabling HTTP and HTTPS Connectors.....	9
4.9 Optional Services: setting additional properties.....	9
4.10 Specify files to exclude from template JBoss server.....	10
4.11 Specifying JMS-specific files to copy.....	10
4.12 Specifying third-party files for embedded Tomcat server's common/lib.....	10
4.13 Specifying third-party files for server library.....	11
4.14 Specifying streek (UCB local library) files for server library.....	11
4.15 Specifying third-party files for JBoss /client directory.....	12
4.16 Specifying basic JBoss services.....	12
4.17 Specifying Application Policies (JBoss server's login-config.xml).....	14
4.18 Specifying Data Sources.....	15
4.19 JGSS Properties (kerberos keytab file references).....	16
4.20 Keystore Properties.....	16

4.21 Setting JBossMail properties.....	17
4.22 Setting JMX Console properties.....	17
4.23 Generating JBoss server instances.....	18
4.24 Running a JBoss server instance.....	18
4.25 Determining ports on which JBoss servers are listening.....	19
4.26 Testing JBoss server instances.....	19

## 1. Audience

Those who wish to generate [JBoss](#) (v 4.0.x) server instances from pre-configured components and user-definable properties.

## 2. Purpose

To easily, via Ant build scripts, generate [JBoss](#) 4.0.x server instances appropriate to deployment requirements.

## 3. Prerequisites

- Projects checked out from UC Berkeley's CVS repository:
  - [JBoss](#) (a **clean** copy of the appropriate version, e.g. jboss-4.0.3)
  - `servant` (up-to-date copy)
  - `maven-repo-j5` (up-to-date copy)
  - `berkeley-maven-repo-j5` (up-to-date copy)
  - `jca-access` (up-to-date copy). If you do not have read-rights on the repository where this project resides, someone who does have those rights will need to help you generate the servers (but you can do the preparatory work described in this document without assistance).
- [Apache-Ant](#) (included in the base [Eclipse](#) IDE used by Streek developers; as well as in the `ist-jxde` project).
- The environment variable `JBOSS_HOME` populated with the filesystem path of the [JBoss](#) installation's root directory.

## 4. JBoss Server Instance Generation

### 4.1. High Level Overview

Ant scripts drive the generation of JBoss servers, which are configured based on specifications in properties files. Much of this document describes how these properties are to be set. A detailed overview of the properties themselves is [here](#).

The process for generating servers, in high-level overview, is as follows:

- For the servers to be generated, specify the names, base port-assignments, and port-assignment-method in one of the following two ways:
  - If the generated servers are to use ports incremented from a base *set* of ports:
    - Specify the names of the servers to be created (in `JBOSS_HOME/server`) in a file

whose name is passed to Ant as `server.names.properties` as described [here](#)

- Specify the base set of ports (multiple servers will be incremented from these levels) in a file whose name is passed to Ant as `initial.bindings.properties`, as described [here](#) (note the defaults may well work for you, depending on what ports are available on your target server host!).
- If each generated server is to use a port incremented from a base *value* (e.g., ports to be used are in a range such as 10000-10019):
  - Specify the names and base port value of the servers to be created (in `JBOSS_HOME/server`) in a file whose name is passed to Ant as `server.names.properties` as described [here](#)
- Assure that an XML properties file exists for each of the servers you wish to generate, as described in the [Overview](#). The name of each such property is: `${your-server-name}-properties.xml`.
- Be certain prerequisites are met (cf. [Prerequisites](#)), above, particularly that you have the `jca-access` project checked out in your workspace (`CVSHOME`).
- Run the default target (`build-servers`) in `servant/jboss4/build.xml`, passing in appropriate parameters (`-Dparameter.name=parameter.value`).  
Appropriate parameters include:
  - deploy.mode**  
Always specify (as `test`, `qa`, or `prod`).
  - server.names.properties**  
Specify unless the contents of `servant/jboss4/conf/default-server-set.properties` will suffice.
  - initial.bindings.properties**  
Only necessary if incrementing from a *set* of port values (as opposed to a base *value*). No need to specify for ports incremented from a set unless the ports generated with `servant/jboss4/binding-manager/default-bindings.properties` conflict with those already in use on the machine where the JBoss servers will be run. Developers building servers to match those that will be deployed to `integration-test`, `qa`, or production machines will probably want to use the `integration-test` server's bindings properties.
  - clean.generated.servers**  
delete servers listed in `server.names.properties` and `service-bindings.xml` prior to generating new servers
  - localhost.access.only**  
generate servers that can be accessed from the host machine only - i.e., servers only listen for requests from localhost (127.0.0.1)

## 4.2. Overview

The procedure described in this document involves setting properties in an Ant [XML Property](#) file, and running an Ant target. The properties files and build.xml file can be found in the `jboss` directory of the `servant` project (cf. [Prerequisites](#), above)

The properties to be set and the target to be run are described in this HowTo document.

In overview, for each generated-server specified in the configuration file described (cf. [here](#)), the server generation script:

- configures a server's port-settings based either on:
  - a standard set of base values (auto-incremented if multiple servers are generated), or on a non-standard set of base values if so-specified (cf. [here](#))
  - a base value specified alongside the generated server's name in the configuration file described [here](#)
- copies server components from the `server/default` directory under `JBOSS_HOME`
- copies third-party library files into embedded Tomcat server's `common/lib` directory (cf. [here](#))
- copies third-party library files into server's `/lib` directory (cf. [here](#))
- copies Streek (UCB local library files) into server's `/lib` directory (cf. [here](#))
- copies third-party client files into the JBoss `/client` directory (cf. [here](#))
- enables HTTP and HTTPS connectors in the JBoss server if a web server is **not** to be deployed to handle http and https requests (cf. [here](#))
- configures server's basic services, based on `server-props.services.files` property (cf. [here](#))
- configures server's datasources, based on `server-props.datasource.files` property (cf. [here](#))
- configures server's application policies (security), based on `server-props.policy.files` property (cf. [here](#))
- configures keytab file references for Kerberos authentication of the server via JGSS (cf. [here](#))
- renames and secures jmx-console application (cf. [here](#))
- copies a specified keystore (generally containing a UCB-signed X.509 certificate used to authenticate the JBoss-Tomcat SSL/TLS connector as localhost) (cf. [here](#))
- configures server's optional services (jms, axis, etc.), based on appropriate property settings (cf. [here](#))

Not every configurable property is listed in the overview above; but all configurable properties are described in this document. Also, additional (non-configurable) tasks are performed by the Ant script that generates servers, such as setting appropriate UNIX

permissions on shell scripts; non-configurable tasks are not described in this document.

**Note:**

All XML Property nodes described in this document, unless explicitly noted, refer to nodes in `conf/streak-reference-properties.xml` or another server-specific instance of an XML document conforming to the DTD `conf/dtd/servant-properties.dtd`.

### 4.3. Specifying generated server(s) name(s)

Property	Value	Notes
<code>use-base-value-port-incrementing</code>	the value of this property may be anything (e.g., YES, XXX, or WHATEVER)	may be set (to any value) or not set (commented out); normally set in <code>conf/default-server-set.properties</code> (or other <code>*-server-set.properties</code> if one other than the default is being used), but may be set as a parameter passed to Ant; cf. explanation of this property's significance, below.
<code>server-names-list</code>	comma-delimited list of one or more server names to be generated	expected in <code>conf/default-server-set.properties</code> unless specified via passed parameter <code>server.names.properties</code>

Any number of servers may be generated. The names of servers to be generated are specified in a property named `server-names-list`, located in the file `conf/default-server-set.properties`. An alternate file in which the `server-names-list` is specified may be passed as a parameter to the top-level `servant/jboss` build script (`build.xml`), as follows:

```
-Dserver.names.properties=conf/your-server-set-file-name.properties
```

The value of `server-names-list` is dependent on whether or not `use-base-value-port-incrementing` is set (to any value) or not-set. This is described in detail in comments to `conf/default-server-set.properties`, and is summarized below. [The very short summary: if `use-base-value-port-incrementing` is set, port-specification occurs via the token(s) in `server-names-list` as described in the bulleted list below; otherwise, cf. [here](#).]

- if `use-base-value-port-incrementing` is NOT set
  - the value of `server-names-list` is a comma delimited list of server names

- each server name is one token
- there should be no spaces in the value of this property
- example: `server1,server2,server3,server4`
- if `use-base-value-port-incrementing` is set (to any value)
  - the value of `server-names-list` is a comma delimited list of server names and base port values
  - each server name is delimited from its base port value by a colon (:)
  - the ports for a server will be incremented from the base-port value; e.g., if the generated server uses 12 ports, and the server name:port token is `my-server:10000`, the generated server will use ports 10000-10011
  - there should be no spaces in the value of this property
  - example:  
`server1:10000,server2:10020,server3:10040,server4:10060`

For each named server in `server-names-list`, an XML properties file must exist in the `/conf` directory, and must be named `{server-name}-properties.xml` (e.g., `streak-reference-properties.xml`). The XML properties files must conform to the DTD `conf/dtd/servant-properties.dtd`.

When choosing names for servers, be sure that they contain no spaces (e.g., no spaces whatsoever should occur in the comma-delimited list `servers-names-list`), nor may they contain characters that are disallowed in filenames on the filesystem where servers will be deployed.

Note that each generated server's port bindings will be specified in `JBOSS_HOME/server/service-bindings.xml`.

#### Warning:

##### Generate all servers for the JBoss instance simultaneously.

As of March 2005, the facility for generating servers in multiple batches requires careful attention. It is a cleaner, less error-prone process to generate all servers to be used in an instance of JBoss at the same time (i.e., by the same `servant` build). Generate all servers simultaneously prevents port collisions (overlapping assignment of port bindings among the generated servers); should such collisions occur, regeneration of all servers or manual port assignment in the configuration file `JBOSS_HOME/server/service-bindings.xml` will be necessary.

A currently-workable, if awkward, method to generate additional servers is to execute a subsequent run of the `servant` build after determination of the highest already-used port assignments for each parameter in the first set of generated servers. From these data, a new base-bindings set can be specified in a file cloned from the `servant` project's `jboss4/binding-manager/default-bindings.properties`, where the values in the cloned file are set higher than the highest values used by already-generated servers. Usage of an alternate base-bindings set is described [here](#). Use of this method is **not advised**.

## 4.4. Specifying non-standard port values

A standard set of port bindings are specified in

`servant/jboss4/binding-manager/default-bindings.properties`.

This set of bindings is used for the first named server in the list contained in the property named `server-names-list` (cf. [here](#) for more info); the values of bound ports are incremented for any additional servers generated. The increment values are set in the `configure-bindings` target of `servant/jboss4/build-servers.xml`.

Should need arise, an alternate set of port values may be specified. The non-standard values should be written to a file in the same directory as `default-bindings.properties` (e.g., `servant/jboss4/binding-manager/my-bindings.properties`). To enable use of this alternate set of values, pass the property `initial.bindings.properties` as a parameter to the top-level `servant/jboss` build script (`build.xml`), as follows:

`-Dinitial.bindings.properties=binding-manager/your-bindings-file-name.properties`

#### 4.5. Specifying server name in XML Properties file

XML Property Node	Value Attribute	Notes
<code>server-props/name</code>	name of the server described in this XML Property file	expected to match the server-name prefix of the XML Property file <code>name,{server-name}-properties.xml</code>

#### 4.6. Specifying application family name in XML Properties file

XML Property Node	Value Attribute	Notes
<code>server-props/app-family</code>	name of the family of applications to be deployed in the server	this token is used as a prefix to server-specific web application names for which a unique-to-server context-root is necessary (e.g., the <code>jmx-console webapp</code> )

#### 4.7. Specifying server-specific JVM arguments

XML Property Node	Value Attribute	Notes
<code>server-props/jvmargs</code>	arguments to be passed to the JVM in which this JBoss server will run	will be appended to the <code>JAVA_OPTS</code> environment variable when this JBoss server is started

## 4.8. Enabling HTTP and HTTPS Connectors

XML Property Node	Value Attribute	Notes
server-props/jboss-web-c	Valid values for this property are:  <b>jboss-web-config-template</b> disable HTTP and HTTPS connectors <b>jboss-web-config-template</b> enable HTTP, HTTPS, and AJP13 connectors	jboss-web-config-template-ajp13-only is correct value if a web server will be deployed to handle HTTP and HTTPS requests

This property governs whether HTTP and HTTPS connectors are enabled or not in generated servers.

The HTTP and HTTPS connectors should **not** be enabled if a web server is to be deployed in front of JBoss to handle HTTP and HTTPS requests.

### Note:

This property is overridden if the `localhost.access.only` property is set. In this case, the template `jboss-web-config-template-ajp13-disabled.xml` is utilized, and the value of this property in the server config file is ignored. (Cf. [note on Ant runtime parameters](#), above, for more info.)

## 4.9. Optional Services: setting additional properties

Property	Value	Notes
server-props/enable.axis	any value - if node exists, Axis will be enabled	<a href="#">Axis</a> is an implementation of SOAP ("Simple Object Access Protocol"). If this service is to be disabled, the XML Property node must be omitted or commented out.
server-props/enable.jms	any value - if node exists, JMS will be enabled	<a href="#">JMS</a> is Java Message Services. If this service is to be disabled, the XML Property node must be omitted or commented out.
server-props/enable.mail	any value - if node exists, JBossMail will be enabled	If <a href="#">JbossMail</a> is enabled, the <code>server-props/mail</code> node must also be included in the configuration file (cf. <a href="#">here</a> ). If this service is to be disabled,

		the XML Property node must be omitted or commented out.
--	--	---

Additional services that may be enabled by appropriately setting properties in the `conf/streak-reference-properties.xml` file include [JBossMail](#), [Java Message Service](#) (JMS), and [Apache-Axis](#) (web services).

#### 4.10. Specify files to exclude from template JBoss server

XML Property Node	Value Attribute	Notes
<code>server-props/default.ser</code>	pattern of files to be excluded when copying files from the template JBoss server to the server(s) being generated	pattern format conforms to <a href="#">Ant patterns</a>

A typical value for this node is: `**/CVS/**, **/data/**, **/log/**, **/tmp/**, **/work/**, **/jms/jbossmq-destinations-service.xml**, **/mail-service.xml`

If JMS is not enabled (cf. [here](#)), add to the above: `deploy/jms/**`

#### 4.11. Specifying JMS-specific files to copy

XML Property Node	Value Attribute	Notes
<code>server-props/jms.files</code>	pattern of files to be copied from the <code>servant/jboss</code> tree to <code>\${jboss.home}/server/\${s</code>	pattern format conforms to <a href="#">Ant patterns</a>

Note that `mail-queue-service.xml` must be in this list if JBossMail is enabled (cf. [here](#)).

#### 4.12. Specifying third-party files for embedded Tomcat server's common/lib

XML Property Node	Value Attribute	Notes
<code>server-props/lib.third.p</code>	A comma-delimited list of third-party JAR files to be copied from a local copy of the <code>maven-repo-j5</code> project (or obtained from the <a href="#">Maven repository</a> at <a href="#">ibiblio.org</a> ) to <code>/deploy/jbossweb-tomcat5</code> in the generated server.	The format of specified files must conform to the <code>maven-repo-j5</code> project's directory structure, including the desired version-number, e.g., <code>commons-beanutils/jars/commons-beanu</code> . If no files are to be copied,

		this property should be commented-out in conf/streak-reference-properties.xml.
--	--	--

Third-party software deployed to this directory is utilized by applications deployed to the embedded Tomcat server.

Third-party jar files commonly specified here for use by Streak applications include:

**commons-beanutils-\*.jar**

required when deploying an application that utilizes [Struts](#) (e.g., commons-beanutils/jars/commons-beanutils-1.7.0.jar).

**commons-digester-\*.jar**

required when deploying an application that utilizes [Struts](#) (e.g., commons-digester/jars/commons-digester-1.7.jar).

### 4.13. Specifying third-party files for server library

XML Property Node	Value Attribute	Notes
server-props/lib.third.p	A comma-delimited list of third-party JAR files to be copied from a local copy of the maven-repo-j5 project (or obtained from the <a href="#">Maven repository</a> at <a href="#">ibiblio.org</a> ) to JBOSS_HOME/<generated-se	The format of specified files must conform to the maven-repo-j5 project's directory structure, including the desired version-number, e.g., castor/jars/castor-0.9.4.3.jar . If no files are to be copied, this property should be commented-out in conf/streak-reference-properties.xml.

Third-party software deployed to this directory is utilized by applications deployed to the generated server.

Third-party jar files commonly specified here for use by Streak applications include:

**castor\*.jar**

[Castor](#) is used to bind Java objects to XML documents and relational tables (e.g., castor/jars/castor-0.9.4.3.jar).

**ojdbc14.jar**

[Oracle](#) JDBC drivers for Java version 1.4.x (e.g., ojdbc14/jars/ojdbc14-10.2.0.1.0.jar)

### 4.14. Specifying streak (UCB local library) files for server library

XML Property Node	Value Attribute	Notes
server-props/lib.streek.	A comma-delimited list of streek JAR files to be copied from a local copy of the berkeley-maven-repo-j5 project to JBOSS_HOME/<generated-se	The format of specified files must conform to the berkeley-maven-repo-j5 project's directory structure, including the desired version-number, e.g., streek-libs/jars/streek-dao-5.0.0.jar. If no files are to be copied, this property should be commented-out in conf/streek-reference-properties.xml.

#### 4.15. Specifying third-party files for JBoss /client directory

XML Property Node	Value Attribute	Notes
server-props/client.thir	A comma-delimited list of files to be copied from a local copy of the maven-repo-j5 project (or obtained from the <a href="#">Maven repository</a> at <a href="#">ibiblio.org</a> ) to JBOSS_HOME/client.	The format of specified files must conform to the maven-repo-j5 project's directory structure, including the desired version-number, e.g., junit/jars/junit-3.8.1.jar. If no files are to be copied, this property should be commented-out in conf/streek-reference-properties.xml.

Third-party software deployed to this directory is utilized by clients (such as monitors and application-testing classes) of applications running in a generated JBoss server.

Third-party client code commonly used in Streek applications includes:

**junit\*.jar**

[JUnit](#) is unit-testing software (e.g., junit/jars/junit-3.8.1.jar).

#### 4.16. Specifying basic JBoss services

XML Property Node	Value Attribute	Notes
server-props/services.fi	A comma-delimited list of files containing XML fragments (expected to be located in jboss4/server-jboss-serv in the servant project).	Specified file fragments are assembled to make up the configuration file JBOSS_HOME/server/{server-name}/conf

**Note:**

Several entity files used with JBoss 3.2.x are deprecated in JBoss 4.0.x, as they are no longer deployed via jboss-service.xml. Instead these are deployed as hot-swappable components in the JBoss server's /deploy directory. These entity files include those pertaining to: Ear Deployer, EJB Deployer, and BeanShell Deployer.

Minimal or required filenames in this list are given below in *emphasized* text (note epilogue.ent, at the end of the list). Omit these elements only if you are certain of what you're doing. The descriptions below may be augmented in the \*.ent files themselves.

**prologue.ent**

jboss-service.xml prolog

**jsr77-manager.ent**

J2EE management mappings (e.g., SAR, EAR, WAR, EJB Deployers; JNDI; JMS; etc.)

**service-binding-manager.ent**

Binding service manager for port/host mapping

**webservice-classloader.ent**

dynamic class loading for RMI access to the server EJBs

**log4j.ent**

[log4j](#) (apache logging) initialization

**jndi.ent**

[JNDI](#) service

**security-manager.ent**

security configuration, including [JAAS](#) security manager

**xmbean-persistence.ent**

[XMBeanAttributePersistenceService](#)

**threadpool-service.ent**

[Basic ThreadPool Service](#)

**jdbc-type-mapping-metadata.ent**

[JDBC Type Mapping Metadata](#)

**bean-cache-monitor.ent**

JMX monitoring of the bean cache. This file must be edited (per instructions in its comments) to enable monitoring.

**entity-lock-monitor.ent**

JMX monitoring of entity bean locking. This file must be edited (per instructions in its comments) to enable monitoring.

**jmx-invokers.ent**

Invokers to the JMX node

**rmi-classloader.ent**

[RMI](#) service

**transaction-managers.ent**  
 transaction management  
**deployment-scanner.ent**  
 hot deployment & undeployment of archives  
**epilogue.ent**  
 jboss-service.xml epilog

#### 4.17. Specifying Application Policies (JBoss server's login-config.xml)

XML Property Node	Value Attribute	Notes
server-props/policy.file	A comma-delimited list of files containing XML fragments	Files referred to in the list are expected to be located in jboss4/security-domain/ in the servant project. Specified file fragments are assembled to make up the configuration file JBOSS_HOME/server/\${server-name}/con...

Minimal or required filenames in this list are given below in *emphasized* text (note epilogue.ent, at the end of the list). Omit these minimal or required elements only if you are certain of what you're doing! Note that application-specific policies need (should) not be included unless the application in-question is to be deployed in the server(s) being generated. The descriptions below may be augmented in the \*.ent files themselves

Application-specific database realm entities are generally not included as separate files (as of February 2005); instead, these nodes are incorporated in the application- & deployment-specific entity file. Similarly, application-specific jgss service nodes are incorporated in the application- & deployment-specific entity file.

**prologue.ent**  
 login-config.xml prolog  
**client-login.ent**  
 policy used by clients within the application server VM  
**default-policy.ent**  
 used by any security domain that does not have an application-policy entry with a matching name  
**hsqldbrealm.ent**  
 policy for Hypersonics data source  
**jmx-console-*{deploy.mode}*.ent**  
 policy for secured jmx-console application  
**jggs-service-*{deploy.mode}*.ent**

**Depracated** Kerberos authentication of the server via jgss, using a server-specific keytab file. Application-specific jgss policies replace this policy; they are normally included in {application-name}-{deploy-mode}.ent

**{application-name}-{deploy-mode}.ent**

application policy for the application named {application-name} deployed in {deploy-mode}, which may include a login stack, a database realm, and/or a jgss-service node

**{application-name}-dbrealm.ent**

**Depracated** application policy for a database realm for the application named {application-name} deployed in {deploy-mode}; as of Feb 2005, practice is to include a database realm application policy node in the {application-name}-{deploy-mode}.ent file

**calnet-krb-test.ent**

application policy for courseweb, including CalNet authentication and courseweb-specific roles policies

**jbossmq.ent**

policy for JMS message queues (omit if not used; may require editing if needed)

**jxdev-dbrealm.ent**

policy for JCA connector

**epilogue.ent**

login-config.xml epilog

#### 4.18. Specifying Data Sources

XML Property Node	Value Attribute	Notes
server-props/datasource.	A comma-delimited list of datasource configuration files (named *-ds.xml).	Files referred to in the list are expected to be located in jboss4/jca/ in the servant project. Specified files are copied into the generated-server's deploy/ directory.

Generally, the only data sources needed in a [JBoss](#) server are those required by applications that will run in that server - see caveat re: Hypersonic database, below.

The local, Hypersonic database (datasource configuration file: hsqldb-ds.xml) is used as a default data source, and must be included in the list of datasources, as the generated server will utilize it for internal processes; omission of hsqldb-ds.xml from this list will cause other services to fail.

#### 4.19. JGSS Properties (kerberos keytab file references)

XML Property Node	Value Attribute	Notes
server-props/jgss/principi	Name of the principal to which the keytab (for deploy.mode = test) corresponds	generally of the form j2ee/<dns-name-of-server-host>
server-props/jgss/principi	Name of the principal to which the keytab (for deploy.mode = qa) corresponds	generally of the form j2ee/<dns-name-of-server-host>
server-props/jgss/principi	Name of the principal to which the keytab (for deploy.mode = prod) corresponds	generally of the form j2ee/<dns-name-of-server-host>
server-props/jgss/keytab	Name of the keytab deployed on the server (for deploy.mode = test)	generally of the form /usr/local/jboss/krb5/\${app-family}-\${server.name}
server-props/jgss/keytab	Name of the keytab deployed on the server (for deploy.mode = qa)	generally of the form /usr/local/jboss/krb5/\${app-family}-\${server.name}
server-props/jgss/keytab	Name of the keytab deployed on the server (for deploy.mode = prod)	generally of the form /usr/local/jboss/krb5/\${app-family}-\${server.name}

The properties described in this section need only be set if policies that utilize [JGSS](#) handshaking are included in the generated server(s) (cf. [Application Policies](#) section in this document).

If Kerberos authentication of the server via [JGSS](#) is to be implemented, a keytab for the server host must exist on the filesystem. The principal name corresponding to this keytab and its location in a secure directory on the server host's filesystem are specified, for each deployment mode, via properties described above.

#### 4.20. Keystore Properties

XML Property Node	Value Attribute	Notes
server-props/keystore-params/ke	The name of the keystore in <code>servant/keystore/</code> to be copied in to the generated server.	This should be set to the name of a keystore containing a UCB-generated certificate that authenticates localhost, generally <code>localhost.jboss.keystore</code> .

server-props/keystore-params/ke	The target name of the copied keystore file, generally <code>jboss.keystore</code> .	
server-props/keystore-params/ke	A password for the keystore file.	The same password as is used for the certificate contained in the keystore.

In order to enable SSL-encrypted access to applications, a signed certificate must be available to JBoss-Tomcat. The properties described in this section are employed in copying a keystore containing an X.509 certificate issued by UCB's certificate authority. The certificate is issued in the name `localhost` because in deployments other than a developer-desktop, a web server (Apache httpd) is expected to be deployed in front of the JBoss application server, in which case httpd handles authentication to the requesting browser. Developers are expected to run local JBoss servers in a mode that allows only requests that originate from localhost (cf. [Running a JBoss server instance](#) in this document; or [Running JBoss in Eclipse on a Developer's Workstation](#) on this site).

Note that prior to 27 Sept 2007, a self-signed, generated keystore was used instead. This solution no longer works when modern browsers access JBoss directly (via https).

#### 4.21. Setting JBossMail properties

XML Property Node	Value Attribute	Notes
server-props/mail/user	The e-mail address to receive notices from the jboss server.	For example, <code>myname</code> if the desired e-mail address is <code>myname@mailserver.berkeley.edu</code> .
server-props/mail/pop3.host	The pop3 host of the e-mail address that will receive notices from the jboss server.	For example, <code>mailserver.berkeley.edu</code> if the desired e-mail address is <code>myname@mailserver.berkeley.edu</code> .
server-props/mail/smtp.host	The name of the machine on which JBoss will be running.	For example, <code>localhost</code> .
server-props/mail/from	The e-mail address that should appear in the From: line of e-mail sent by JBoss	For example, <code>jboss@myhost.berkeley.edu</code> .

These properties should be set if `server-props/enable.mail` is included in the XML Properties file (cf. [here](#)).

#### 4.22. Setting JMX Console properties

XML Property Node	Value Attribute	Notes
<code>jmx.console.security/datasource/</code>	Datasource name for the secured <code>jmx-console</code> login module's queries (for <code>deploy.mode = test</code> ).	For example, <code>MyAppTestDS</code> .
<code>jmx.console.security/datasource/</code>	Datasource name for the secured <code>jmx-console</code> login module's queries (for <code>deploy.mode = qa</code> ).	For example, <code>MyAppQADS</code> .
<code>jmx.console.security/datasource/</code>	Datasource name for the secured <code>jmx-console</code> login module's queries (for <code>deploy.mode = prod</code> ).	For example, <code>MyAppProdDS</code> .
<code>jmx.console.security/sql</code>	The query executed by the <code>JdbcRolesLoginModule</code> in the <code>jmx-console</code> application-policy.	For example, <code>select 'MyRoleName' from dual where [...]</code> .
<code>jmx.console.security/streak.depe</code>	A comma-delimited list of streak JAR files to be copied from a local copy of the <code>berkeley-maven-repo-j5</code> project to the console's <code>WEB-INF/lib</code> directory. Servant also copies this set of JAR files into the analogous Axis console directory if <a href="#">Axis is enabled</a> .	The format of specified files must conform to the <code>berkeley-maven-repo-j5</code> project's directory structure, including the desired version-number, e.g., <code>streak-libs/jars/streak-security-5.0</code> .

### 4.23. Generating JBoss server instances

After setting environment variables and configuring the `conf/streak-reference-properties.xml` file appropriately - as described above - run the Ant target `build-servers` from the `build.xml` in the `jboss` directory of the servant project. The target `build-servers` is the default target in this build script.

After a successful build, run and test the generated server(s).

### 4.24. Running a JBoss server instance

To run a JBoss server, you may either:

- run the appropriate shell script in `JBOSS_HOME/bin` (e.g., `run-myservername.sh`)

- [preferred]
- type your OS's equivalent of the following in a console window (or include in a shell script or .bat file):  
JBOSS\_HOME/bin/run -c <your-server-name> [-b 127.0.0.1]
- or, to run the server and send console output to a file, type your OS's equivalent of the following in a console window (or include in a shell script or .bat file):  
JBOSS\_HOME/bin/run -c <your-server-name> [-b 127.0.0.1] >  
%JBOSS\_HOME%/server/<your-server-name>/log/console.log

Note that OS-appropriate reference to the environment variable JBOSS\_HOME is necessary if the examples given above are to work properly.

The "[b 127.0.0.1]" is an [optional] switch that, if included, causes JBoss to listen ONLY for requests coming from localhost. The shell scripts in JBOSS\_HOME/bin include this switch if the localhost.access.only property is set when the servers are built (cf. [note on Ant runtime parameters](#), above, for more info).

#### 4.25. Determining ports on which JBoss servers are listening

To determine which port(s) the server(s) are listening on, examine the configuration file JBOSS\_HOME/server/service-bindings.xml. The port bindings for each generated server are assigned in this file. Ports on which JBoss are listening are also emitted in the console log.

#### 4.26. Testing JBoss server instances

For each server you wish to test, start the server and be sure you know which port the server is using to listen for HTTP requests (if Apache is running in front of your JBoss instance, you'll want to make sure that mod\_jk2 is properly set up as well!).

To test, point a browser at the server, e.g.:

http://localhost:<your-port-number> ... if you see the JBoss application server home page in your browser, your server is running and you are ready to examine whether it is configured as you intended (via the JMX console, etc.).