

Red Hat Application Stack V.1.1 Release Notes

1.1

ISBN: N/A

Publication date:

Red Hat Application Stack V.1.1 Release Notes

Copyright © 2006-2007 Red Hat, Inc.

1. Introduction	1
1. Overview of This Release	1
2. Installation Prerequisites	1
3. Package-specific Notes	1
2. Configuring the Java Environment	3
1. Install a supported SDK	3
1.1. Installing a 32-bit Sun SDK	3
1.2. Installing a BEA or IBM SDK	5
2. Install the <code>jbossas</code> package	6
2.1. Using <code>up2date</code> / RHN (Preferred Method)	6
2.2. Using <code>up2date</code> / DVD	7
3. Installing Red Hat Application Stack	9
4. Using JBoss AS	13
1. Using the Linux service (production)	13
2. Using <code>run.sh</code> (development)	15
3. Creating copies of <code>JBOSS_HOME</code>	16
4. Using the <code>jbossas-*</code> commands from any directory	16
5. Running multiple instances of JBoss AS	17
5.1. Running multiple instances of JBoss AS using different sets of ports	18
5.2. Running multiple instances of JBoss AS binding to different local IP addresses	19
6. More Information	20
5. Known Issues	21
6. Export Control	23

Introduction

1. Overview of This Release

Red Hat Application Stack is a set of software components that have been tested and certified to work together in an integrated fashion. Red Hat Application Stack is designed for dynamic web applications for both the LAMP and Java platforms.

2. Installation Prerequisites

- Red Hat Application Stack V.1.1 has been tested and certified on Red Hat Enterprise Linux 4 U4 systems. It is not certified for use with earlier versions of Red Hat Enterprise Linux. Please follow the separate instructions for installing Red Hat Enterprise Linux 4 before installing Red Hat Application Stack V.1.1.
- Red Hat Application Stack V.1.1 is available for 32- and 64-bit Intel architectures only (i386 and x86_64).
- Red Hat Application Stack V.1.1 includes JBoss AS 4.0.5, for running web applications in a standard middleware environment. Note that JBoss AS uses an embedded Tomcat and does not require a stand-alone Tomcat. Before installing and running JBoss AS you will require a working installation of Java 1.4 or 1.5. Currently Red Hat Application Stack is certified with the Sun JVM version 1.4.2 update 13 or later, 1.5.0 update 11 or later, and the latest BEA JVM available through RHN. Please refer to [Chapter 2, Configuring the Java Environment](#) for instructions on configuring the Java environment for JBoss AS.
- Red Hat Application Stack V.1.1 also includes JBoss EJB3 1.0.0-RC9, available as a technology preview. You will need a working Java 1.5 installation in order to use this package.

3. Package-specific Notes

In most cases, Red Hat Application Stack contains more recent versions of packages than those that are included in Red Hat Enterprise Linux. Installing Red Hat Application Stack may update/overwrite these packages. Please ensure you review the complete package list before beginning the installation process.

Configuring the Java Environment

JBoss AS relies on an installed JVM, such as those by Sun, or BEA. The BEA JVM is available from the Red Hat Enterprise Linux (AS/ES) Extras channel. The Sun SDK must be downloaded directly from Sun (instructions below).



THIS RELEASE HAS BEEN CERTIFIED WITH THE SUN AND BEA JVMs.

Currently Red Hat Application Stack is certified with the Sun JVM version 1.4.2 update 13 or later, 1.5.0 update 11 or later and the latest BEA JVM available through RHN. This may change in future releases.

1. Install a supported SDK

1.1. Installing a 32-bit Sun SDK

1. Get the Sun JDK 5.0 from:

http://java.sun.com/javase/downloads/index_jdk5.jsp

by choosing the "JDK 5.0 Update N" "Download" button, and then choosing "RPM in self-extracting file" for Linux on the page that displays after pressing the button.



Note

If you don't need/want to use the SysV service scripts you can also install the "Linux self-extracting file" instead.



Important

Do NOT install the "Linux x64" version of the SDK.

If you prefer the older Java 1.4.2 SDK, get it from:

<http://java.sun.com/j2se/1.4.2/download.html> [<http://java.sun.com/j2se/1.4.2/download.html>]

by choosing the "Download J2SE SDK" link and from there the "RPM in self-extracting file"

for Linux.



Note

If you don't want/need to use the SysV service scripts you can also install the "self-extracting file" for Linux instead.

2. Install `java-1.4.2-sun-compat` or `java-1.5.0-sun-compat` (optional)

This step is only necessary if you want to use the SysV service scripts, which we recommend for production servers.

If you just plan on doing `cd $JBOSS_HOME` and `./run.sh`, which should be done just for development, you can skip this step and the next.

Download and install the appropriate `-compat` RPM from JPackage at:

<ftp://jpackage.hmdc.harvard.edu/JPackage/1.7/generic/RPMS.non-free/>

Make sure to match the version of the `-compat` package to the SDK you've installed in the first step. The `-compat` RPM requires that the RPM self-extracting file from Sun be used in the previous step, not the plain (non-RPM) one.

For instance, for a Sun SDK 1.5.0_11 you should get:

<ftp://jpackage.hmdc.harvard.edu/JPackage/1.7/generic/RPMS.non-free/java-1.5.0-sun-compat-1.5.0.11-1jpp.i586>

and for a Sun SDK 1.4.2_13 you should use:

<ftp://jpackage.hmdc.harvard.edu/JPackage/1.7/generic/RPMS.non-free/java-1.4.2-sun-compat-1.4.2.13-1jpp.i586>



Note

If the `-compat` RPM for the latest Sun release is not yet available, contact support or request an update in the mailing list.

For more details see <http://jpackage.org/>

3. Selecting alternatives for `java`, `javac` and `java_sdk_VERSION` (optional, where `VERSION` = 1.4.2 or 1.5.0 depending on what version `java` "java" and "javac" are being set to)

This is only needed if you performed the second step above and you want to use the SysV service script and/or want this installed SDK to be the default `java` and `javac` in the system. This choice can often be overridden by setting the `JAVA_HOME` environment variable.

The `alternatives` system allows different versions of Java, from different sources to co-exist on your system. You should make sure the Sun one is selected so that the service script uses the one you want.

As root, issue the following command:

```
/usr/sbin/alternatives --config java
```

and make sure the Sun one is selected (marked with a '+'), or select it by entering its number as prompted.

Make sure you do the same for `javac` and `java_sdk_VERSION` (where `VERSION` = 1.4.2 or 1.5.0 depending on what version java "java" and "javac" are being set to). We recommend that all point to the same manufacturer and version.

1.2. Installing a BEA or IBM SDK



Note

THIS RELEASE HAS BEEN CERTIFIED WITH THE SUN AND BEA JVMs

Currently Red Hat Application Stack is certified with the Sun JVM version 1.4.2 update 13 or later, 1.5.0 update 11 or later and the latest BEA JVM available through RHN. This may change in future releases.

With RHEL 4 U4 there are 1.4.2 and 1.5 versions of the IBM and BEA (JRockit) SDKs.

We have adopted the `jpackage.org` style for the packaging of our Java™ offerings. This requires that the `jpackage-utils` rpm be installed on your system. A JPackage-style JVM must be installed via `up2date` (see channel details below) and `jpackage-utils` will be automatically brought in as a dependency.

1. Install a Java SDK from RHN

Java SDKs are provided by the Red Hat Enterprise Linux 4 Extras channel for your Linux variant and architecture. The channel names are as follows:

```
rhel-arch-variant-4-extras
```

where:

```
arch = i386 or x86_64 variant = as, or es
```

The `java-1.4.2-ibm`, `java-1.4.2-bea`, `java-1.5.0-ibm` and `java-1.5.0-bea` SDKs are

available from these channels. Make sure you also install the `-devel` subpackages.

There is no BEA JRockit JVM for the `x86_64` architecture at this time, but the BEA i386 JVM will work on `x86_64` and it is available from the `x86_64` channel as well.

2. Selecting alternatives for `java`, `javac` and `java_sdk_VERSION` (optional, where `VERSION` = 1.4.2 or 1.5.0 depending on what version `java java` and `javac` are being set to)

This is only needed if you want to use the SysV service script and/or want this installed SDK to be the default `java` and `javac` in the system. This choice can often be overridden by setting the `JAVA_HOME` environment variable.

The `alternatives` system allows different versions of Java, from different sources to co-exist on your system. You should make sure the desired one is selected so that the service script uses the one you want.

As root, issue the following command:

```
/usr/sbin/alternatives --config java
```

and make sure the desired one is selected (marked with a '+'), or select it by entering its number as prompted.

Make sure you do the same for `javac` and `java_sdk_VERSION` (where `VERSION` = 1.4.2 or 1.5.0 depending on what version `java java` and `javac` are being set to). We recommend that all point to the same manufacturer and version.

2. Install the `jbossas` package

2.1. Using `up2date` / RHN (Preferred Method)

1. `up2date` your RHEL4 installation and install the JVM as described above.
2. Subscribe to the Red Hat Application Stack child channel:

e.g: `rhel-x86_64-as-4-appstk-1` or `rhel-i386-as-4-appstk-1`

3. Install JBoss AS:

- For the Application Server (which includes an embedded Tomcat), run:

```
up2date jbossas
```

Refer also to [Chapter 4, Using JBoss AS](#) below.

2.2. Using `up2date` / DVD

To install using `up2date` / DVD:

1. `up2date` your installation and install the JVM as described above.
2. Mount the Red Hat Application Stack V.1.1 DVD.
3. Add the following line to `/etc/sysconfig/rhn/sources` to use `up2date` directly on the RPMs on the disk:

```
dir RedHat-Application-Stack-V1 path_to_mounted_DVD/RedHat/RPMS
```

4. Install JBoss AS:

- For the Application Server (which includes an embedded Tomcat), run:

```
up2date jbossas
```

5. After the installation is complete, remove the line you added to `/etc/sysconfig/rhn/sources` in Step 3. Failure to do this will result in `up2date` producing warnings whenever you run `up2date` again without mounting the Red Hat Application Stack V.1.1 DVD.

Refer also to [Chapter 4, Using JBoss AS](#) below.

Red Hat Application Stack V.1.1 includes two versions of JBossAS. The package 'jbossas' will function with both 1.4 and 5 JVMs. The package 'jbossas-ejb3' includes a technology preview of EJB3 (1.0.0-RC9) and requires a working Java 5 installation. Only one version may be installed at a time.

If you are upgrading from version 1 of Red Hat Application Stack and want to install the EJB3 package you must uninstall the 'jbossas' package first, and install a Java 5 JVM.



Note

Please note that in order to use EJB3 ('jbossas-ejb3') you will need a working Java 1.5 installation. 'jbossas' can be used with 1.4.2 as before.

Installing Red Hat Application Stack

To install Red Hat Application Stack V.1.1.1:

1. If you have an existing database, back up your data and shut down the database:

If you are using PostgreSQL:

- Back-up your database. For example, run:

```
su -  
rm -rf /tmp/pg.backup  
su - postgres  
/usr/bin/pg_dumpall > /tmp/pg.backup  
exit
```



Note

Note: If you have large objects in the database, you need to use `pg_dump` instead of `pg_dumpall`. See

<http://www.postgresql.org/docs/8.1/static/largeobjects.html> and

<http://www.postgresql.org/docs/8.1/static/app-pg-dumpall.html>

For more information, see <http://www.postgresql.org/docs/8.1/static/install-upgrading.html>

If you are using MySQL:

- Backing-up your database is optional but recommended. Run as root:

```
mysqldump --all-databases -p > /tmp/mysqldumpfile.sql
```

2. Stop the daemons:

-

```
/sbin/service httpd stop
```

- To shut down MySQL:

```
/sbin/service mysqld stop
```

- To shut down PostgreSQL:

```
/sbin/service postgresql stop
```

3. Install the software:

- Option 1. To install from RHN, which is the preferable option to avoid dependency issues, run as root:

```
up2date --exclude=jboss-ejb3 --installall rhel-i386-es-4-appstk-1
```

where *rhel-i386-es-4-appstk-1* is the channel name for Red Hat Application Stack

The channel names are as follows:

```
rhel-arch-variant-4-appstk-1
```

where:

```
arch = i386 or x86_64 variant = as, or es
```



Note

If you wish to use the `jbossas-ejb3` package (needs a working Java 1.5 installation), run:

```
up2date --exclude=jbossas --installall rhel-i386-es-4-appstk-1
```

- Option 2. To install from RPMs on Disk:

Follow the instructions in [Section 2.2, “Using up2date / DVD”](#) to mount the DVD and set up the local RHN channel (steps 1-3). Then as root run:

```
up2date --exclude=jbossas-ejb3 --installall RedHat-Application-Stack-V1
```



Note

If you wish to use the `jbossas-ejb3` package (needs a working Java 1.5 installation), run:

```
up2date --exclude=jbossas --installall RedHat-Application-Stack-V1
```

After the installation is complete, be sure to follow step 5 from [Section 2.2, “Using up2date /DVD”](#) to prevent unnecessary warnings.



Note

You may need to install the following packages to satisfy dependencies before executing the above command, if they are not already installed:

- `apr-devel`
- `apr-util-devel`
- `libc-client`
- `mx`
- `pcre-devel`

4. Restore your database:

- For PostgreSQL:
 - Move (or remove) the database directory:

```
mv /var/lib/postgresql/data /var/lib/postgresql/data.backup
```

- Start the new PostgreSQL service:

```
/sbin/service postgresql start
```

- Restore your data from back-up

```
psql -U username < /tmp/pg.backup
```

- Start the http daemon:

```
/sbin/service httpd start
```

- For MySQL:
 - Update MySQL:

```
mysql_update
```

- Start MySQL:

```
/sbin/service mysqld start
```

- Restore your data:

```
mysql -p < /tmp/mysqldumpfile.sql
```

- Start the http daemon:

```
/sbin/service httpd start
```

The installation is now complete.

Using JBoss AS

1. Using the Linux service (production)

JBoss AS can be started, stopped, and configured to start automatically at boot time either from the command line or using a graphical tool.

You can start and stop the `jbossas` service using the `service` command as root on a console window (as is typical of a network service):

```
service jbossas start
...
service jbossas stop
```

The behavior at boot can be controlled with the `chkconfig` command, as any other Linux service (see `chkconfig` man page).

Alternatively, you can issue the command `system-config-services` to activate the graphical Red Hat Service Configuration Tool, which can also be activated from the main menu. Select:

System Settings > Server Settings > Services



Note

For security purposes, beginning with Red Hat Application Stack V.1.1, authentication for the `jmx-console`, `web-console`, `jmx-invoker` and `http-invoker` is turned on. Additionally, no user accounts are active by default, so as to prevent default user / password based attacks.

- Accounts for the `jmx-console` and the `invokers` can be set up by modifying:

```
$JBOSS_HOME/server/$CONFIG/conf/props/jmx-console-users.properties
```

- Accounts for the `web-console` users can be set up by modifying:

```
$JBOSS_HOME/server/$CONFIG/deploy/management/console-mgr.sar
/web-console.war/WEB-INF/classes/web-console-users.properties
```

Please note that the file name above has been split onto two lines for readability. It should be understood and entered as one continuous line.

Where `$JBOSS_HOME` is the install directory (`/var/lib/jbossas`) and `$CONFIG` is the server configuration you are using.

Please see http://kbase.redhat.com/faq/FAQ_107_9963.shtml for more information

JBoss AS is configured with its internal servlet engine to listen for HTTP traffic on port 8080. Once the `jbossas` service has been started, you can verify that it is running by pointing your web browser to:

```
http://localhost:8080/web-console
```

You can use the Web Console to administer JBoss AS.

The entire JBoss AS suite runs under a new `jboss` system user. It may be necessary to use the `'su -s /bin/bash jboss'` system command to deposit `.ear` / `.war` / `.jar` files under the JBoss AS deployment directory, due to file system permissions. Alternatively, a developer can be listed in the `jboss` user group by the system administrator. The best approach is to use the Web Console (URL above) to deploy the application.



Note

Depositing files for which the user `jboss` has no read access in the deployment directory will cause the server to fail to deploy.

To create additional JBoss AS configurations besides the provided 'default', 'minimal' and 'all', you must create a new directory for your configuration as follows (note the switches given to the `cp` command):

```
export JBOSS_BASE=/var/lib/jbossas
cd $JBOSS_HOME
cp -pL -R server/default server/myownconfig
```

You can then change the configuration and request it to be used by setting the `JBOSSCONF` variable in the file:

```
/etc/sysconfig/jbossas
```

Optionally, you can just set the `JBOSSCONF` variable in `/etc/sysconfig/jbossas` to a non-existent subdirectory (existing parent with write access by the `jboss` user) or an empty directory (with write access by `jboss`) and the init script will create a new configuration directory tree for you based on the current "all" configuration when the service is first started. You can then stop the service, adjust the configuration as desired, and start it again.



Note

Please note that automated updating of configurations created using method above is not supported. If you create custom configurations, you will manually have to port files / changes from a new update to that configuration.

2. Using run.sh (development)

For development, you can activate the JBoss AS server with the familiar `run.sh` command as usual. The `jbossas` RPM installs the `JBOSS_HOME` in

```
/var/lib/jbossas
```

So, you can `cd` to `/var/lib/jbossas/bin` and use `run.sh` from there.

If the `-compat` RPM for the Sun SDK was not installed, you must also set `JAVA_HOME` before starting the server. If it is installed, just make sure the alternatives are pointing to the SDK you want to use -- there is no need to set `JAVA_HOME` in this case, unless you want to use a different SDK than the one pointed to by `alternatives`.



Tip

`JAVA_HOME` can be set system-wide in `/etc/profile`. But beware, some users may not want to have `JAVA_HOME` set to this same SDK. Also, some may prefer to use the `alternatives`-selected one instead. These users may be affected by this global setting of `JAVA_HOME`.

With the Sun RPM, the SDK is usually located in `/usr/java/...` (it may vary between their releases). If the `-compat` RPM is installed but you still need to set `JAVA_HOME` for some other reason, the `alternatives`-selected SDK is in `/usr/lib/jvm/java`, so a `export JAVA_HOME=/usr/lib/jvm/java` will do.



IMPORTANT

The entire JBoss AS suite runs under a new `jboss` system user. It may be necessary to use the `su -s /bin/bash jboss` system command to run the server, change configuration, deploy etc., due to file system permissions. Alternatively, a developer can be listed in the `jboss` user group by the system administrator.

Yet another possibility is to create a personal copy of `JBOSS_HOME` (see [Creating copies of JBOSS_HOME](#) below).

To create additional JBoss AS configurations besides the provided 'default', 'minimal' and 'all', you must create a new directory for your configuration as follows (note the switches given to the `cp` command):

```
JBOSS_BASE=/var/lib/jbossas
cd $JBOSS_HOME
cp -pL -R server/default server/myownconfig
```

You can then change the configuration and request it to be used by specifying it in the `-c` flag to `run.sh`, as usual.

3. Creating copies of `JBOSS_HOME`

The analog of unzipping another copy of JBoss AS in a different directory is to make a copy of the RPM-installed `JBOSS_HOME`, located at `/var/lib/jbossas`.

You probably want this copy to have your own userid and group, so use:

```
cp -L -R /var/lib/jbossas my_new_jboss_home_directory
```

If you want to retain the `jboss` user and group ownership, use instead:

```
cp -pL -R /var/lib/jbossas my_new_jboss_home_directory
```

To run multiple servers simultaneously you will need to adjust the configuration to avoid port conflicts, as described in the JBoss AS documentation. Refer to [Running multiple instances of JBoss AS](#) for more information.



Note

Please note that automated updating of copies created using methods above is not supported. If you create custom copies, you will manually have to port files / changes from a new update to that copy.

4. Using the `jbossas-*` commands from any directory

Most of the commands found in `$JBOSS_HOME/bin` have equivalents in `/usr/bin` with names like `jbossas-command`, where `command` is the original JBoss AS command name. So, for instance, there is a `jbossas-run` command that is available from any directory.

To use these commands, however, you must set two environment variables:

```
export JBOSS_HOME=/var/lib/jbossas
export JAVA_HOME=/usr/lib/jvm/java
```

where the value you give for `JAVA_HOME` is the location of your installed Java VM.

5. Running multiple instances of JBoss AS



Note

Please note that automated updating of instances created using methods below is not supported. If you create custom instances, you will manually have to port files / changes from a new update to that instance.

The description here is restricted to running multiple instances of the JBoss AS Linux service. Running multiple instances with `run.sh` is already described elsewhere in the JBoss AS documentation.

You can administer multiple SysV services that run JBoss AS by creating additional service scripts. The process basically consists of:

- creating a link with the new service name to the original `jbossas` init script in `/etc/init.d`
- copying the `jbossas` file in `/etc/sysconfig` to one with the new service name (the init script and the configuration file names must match)
- making the desired configuration changes making sure:
 - port conflicts are avoided
 - a different log file is specified



Note

The original init script log files are created under `/var/log/jbossas` in directories that correspond to the configuration used (like 'default') with the usual `server.log` name. For instance:

```
/var/log/jbossas/default/server.log
```

In general, each log file is created by the SysV script as

```
/var/log/service-name/${JBOSSCONF}/server.log
```

so no file conflicts shall exist.

For more flexibility in the configuration of the different servers, you may want to create full copies of `JBOSS_HOME`. This is described in [Creating copies of JBOSS_HOME](#) above -- make sure you use the `cp` command's `-p` switch to preserve the `jboss` user and group ownership. Then, in `/etc/sysconfig/service-name`, uncomment and update the value of the `JBOSS_HOME` variable.

Optionally, you can just uncomment and change the value of `JBOSS_HOME` as described above so as to specify an empty or non-existent directory and the script will automatically create a copy of the RPM-installed `/var/lib/jbossas` (original `JBOSS_HOME`) for you when the service is first started. The current "minimal", "default" and "all" configurations will be copied. Make sure the parent directory, in case of a new directory, or the directory itself, if already existent, have write permissions for the user `jboss`.



Note

Starting the server this first time may take a little longer due to the file copying.

There are 2 ways to run additional instances of JBoss AS and avoid port conflicts:

- By using different sets of ports
- By binding to different local IP addresses

5.1. Running multiple instances of JBoss AS using different sets of ports

Using different sets of ports

The `BindingManager` in JBoss AS can be used to dynamically override service configurations (e.g. port numbers). `Jbossas` has some pre-configured overrides, namely `ports-01`, `ports-02` and `ports-03`. The configuration file can be found at `/var/lib/jbossas/docs/examples/binding-manager/sample-bindings.xml`. These sets of overrides can be used when multiple instances of JBoss AS must be started. Please follow these steps when multiple instance of JBoss AS need to be started:

1. Create a new file `/etc/init.d/service-name` which is a symlink to `/etc/init.d/jbossas`, e.g:

```
ln -s /etc/init.d/jbossas /etc/init.d/jbossas-ports-01
```

2. Create a new config file for *service-name*, `/etc/sysconfig/service-name`. The contents of this file should be similar to `/etc/sysconfig/jbossas`, except with instance specific values uncommented and updated, e.g:

```
cp /etc/sysconfig/jbossas /etc/sysconfig/jbossas-ports-01. Uncomment and update JBOSSCONF to ports-01 in /etc/sysconfig/jbossas-ports-01.
```

3. Then add *service-name* for management by `chkconfig`, e.g:

```
chkconfig --add jbossas-ports-01
```

4. To start/stop the service, simply: `service service-name start/stop`, e.g:

```
service jbossas-ports-01 start
```

5.2. Running multiple instances of JBoss AS binding to different local IP addresses

Each JBoss AS instance is bound to a particular IP address.

1. Create a new file `/etc/init.d/service-name` which is a symlink to `/etc/init.d/jbossas`, e.g:

```
ln -s /etc/init.d/jbossas /etc/init.d/jbossas-instance2
```

2. Create a new config file for *service-name*, `/etc/sysconfig/service-name` with instance specific values uncommented and updated , e.g:

```
cp /etc/sysconfig/jbossas /etc/sysconfig/jbossas-instance2. Uncomment and update JBOSSCONF to instance2. Create a new variable JBOSS_IP and set it to another IP address in /etc/sysconfig/jbossas-instance2
```

3. Then add *service-name* for management by `chkconfig`, e.g:

```
chkconfig --add jbossas-instance2
```

4. To start/stop the service, simply: `service service-name start/stop`, e.g:

```
service jbossas-instance2 start
```

6. More Information

More general documentation for JBoss AS is available under
`/usr/share/doc/jbossas-4.0.5.`

Known Issues

1. Known Bugs

Despite all our efforts, as with any complex piece of software, some issues remain unresolved at the time of this release. These will eventually be fixed in future releases or updates.

The following bugs, identified by number, have been filed in Bugzilla. The Bugzilla's URL is <http://bugzilla.redhat.com/>

List of bug numbers:

Generic: 200992, 206026, 220605, 220606, 220607,220608, 220611, 220613, 220616, 223081, 223085

With a Sun 1.4.2 SDK (32-bit): 200992, 203668

With a Sun 1.5.0 SDK (32-bit): 200992, 203647, 203654, 203668, 203722, 203724, 203807, 219743

With an IBM 1.4.2 SDK (32-bit): 200992, 203647, 203668, 203682, 204425, 204426, 204427, 204428, 205256

With an IBM 1.5.0 SDK (32-bit): 203682

With BEA 1.4.2 SDK: 200992

With BEA 1.5.0 SDK: 200992, 209481

2. Additional Notes

- Removing the `-compat` RPM

There is an unconfirmed report that if the Sun `compat rpm` is installed and alternatives are set to it, when that `compat RPM` is uninstalled the alternatives don't fallback to existing JVMs. They should revert to `--auto` and point to whatever the higher priority Java remains. To be on the safe side, please inspect the alternative settings for `java`, `javac` and `java_sdk_VERSION` after uninstalling the `-compat` RPM.

For more information, see also the Package Listing for Red Hat Application Stack V.1.1

Export Control

As required by U.S. law, you represent and warrant that you:

- a. Understand that certain parts of the software are subject to export controls under the U.S. Commerce Department's Export Administration Regulations (EAR)
- b. Are not located in a prohibited destination country under the EAR or U.S. sanctions regulations (currently Cuba, Iran, Iraq, Libya, North Korea, Sudan, and Syria)
- c. Will not export, re-export, or transfer the software to any prohibited destination, entity, or individual without the necessary export license(s) or authorization(s) from the U.S. Government
- d. Will not use or transfer the software for use in any sensitive nuclear, chemical or biological weapons, or missile technology end-uses unless authorized by the U.S. Government by regulation or specific license
- e. Understand and agree that if you are in the United States and export or transfer the Software to eligible end users, you will, as required by EAR Section 740.17(e), submit semi-annual reports to the Commerce Department's Bureau of Industry & Security (BIS) that include the name and address (including country) of each transferee
- f. Understand that countries other than the United States may restrict the import, use, or export of encryption products and that it shall be solely responsible for compliance with any such import, use, or export restrictions.

