Lab Work: Setting Up JBoss EAP in standalone mode

In this lab, you will create two new directories outside of the EAP installation folder as the base directory, run two independent servers with port offset, and use the management console to increase the logging level for one category.

Resources	
Files	/opt/jboss/standalone /opt/jboss/standalone2
app url	http://localhost:19990 http://localhost:9990
Resources	N/D

Results

You should be able to run two instances of EAP with a custom base directory and port offset, as well as use the management console to customize a standalone server.

before you start

Use the following command to download the relevant lab directory and to verify that no EAP instance is running:

```
[student@workstation ~]$ lab configuring-lab setup
```

1. To avoid interfering with the base EAP installation, and to keep a copy backup server configuration, create a new base directory for EAP. Using the current EAP installation owned by the jboss user located at /opt/jboss-eap-7.0, copy the required directories as the jboss user to the lab directory located at /opt/jboss/standalone.

2. Change to the jboss user and start a standalone EAP server using the / directory opt/jboss/standalone as the EAP base directory. Before proceeding, wait for the server to finish starting up.

3. Your organization wants to be able to run two separate EAP servers simultaneously to configure the server differently for separate applications.

   Open a new terminal window and create a new base directory for another server, based on the original EAP installation located at /opt/jboss-eap-7.0. Use the /opt/jboss/standalone2 directory as the base directory for the new EAP instance.

4. Leave the other EAP instance running, open a new terminal, and start a new instance using the command line parameter to set a port offset to 10000 to avoid port conflicts. The new server management console should listen on port 19990 with the offset.

Also, be sure to use a command line argument when starting EAP as the jboss user, to use /opt/jboss/standalone2 as the EAP base directory.

5. **Wait for the second server to finish starting. Next, access the administration console. Update the CONSOLE handler registry level and set it to DEBUG. This can be configured in the logging subsystem handler section. Reboot the server to see the debug information displayed on the console.**

6. **Access the administration console for the first server and notice that the Console handler's logging level is still set to INFO. The first server did not change, as the two servers have different configuration files in each of their respective base directories.**

7. **Stop the EAP instances by pressing Ctrl+C in each terminal window that is running EAP. Use the following command to run the qualification script:**

```
[student@workstation bin]$ lab configuring-lab grade
```

This concludes the lab work.

Solution

In this lab, you will create two new directories outside of the EAP installation folder as the base directory, run two independent servers with port offset, and use the management console to increase the logging level for one category.

Resources	
Files	/opt/jboss/standalone /opt/jboss/standalone2
app url	http://localhost:19990 http://localhost:9990
Resources	N/D

Results

You should be able to run two instances of EAP with a custom base directory and port offset, as well as use the management console to customize a standalone server.

before you start

Use the following command to download the relevant lab directory and to verify that no EAP instance is running:

```
[student@workstation ~]$ lab configuring-lab setup
```

1. **To avoid interfering with the base EAP installation, and to keep a copy backup server configuration, create a new base directory for EAP. Using the current EAP installation owned by the jboss user located at /opt/jboss-eap-7.0, copy the required directories as the jboss user to the lab directory located at / opt/jboss/standalone.**

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/standalone [student@workstation
standalone]$ sudo -u jboss \ cp -r configuration deployments lib /
opt/jboss/standalone
```

2. **Change to the jboss user and start a standalone EAP server using the / directory opt/jboss/standalone as the EAP base directory. Before proceeding, wait for the server to finish starting up.**

2.1.

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/bin
[student@workstation bin]$ sudo -u jboss ./standalone.sh \
-Djboss.server.base.dir=/opt/jboss/standalone/
```

2.2. **Verify that the server is running by accessing the console administration at http://localhost:9990 and verifying that a message similar to the following appears in the terminal output:**

```
17:03:30,497 INFO [org.jboss.as] (Controller Boot Thread) WFLYSRV0025: JBoss EAP 7.0.0.GA (WildFly Core
  2.1.2.Final-redhat-1) started in 3313ms -Started 261 of 509 services (332 services are lazy, passive or on-
  demand)
```

3. Your organization wants to be able to run two separate EAP servers simultaneously to configure the server differently for separate applications.

   Open a new terminal window and create a new base directory for another server, based on the original EAP installation located at /opt/jboss-eap-7.0. Use the / opt/jboss/standalone2 directory as the base directory for the new EAP instance.

```
[student@workstation bin]$ cd /opt/jboss-eap-7.0/standalone [student@workstation
standalone]$ sudo -u jboss \ cp -r configuration deployments
lib /opt/jboss/standalone2
```

4. Leave the other EAP instance running, open a new terminal, and start a new instance using the command line parameter to set a port offset to 10000 to avoid port conflicts. The new server management console should listen on port 19990 with the offset.

   Also, be sure to use a command line argument when starting EAP as the jboss user, to use /opt/jboss/standalone2 as the EAP base directory.

```
[student@workstation standalone]$ sudo -u jboss \ /opt/jboss-
eap-7.0/bin/standalone.sh \ -Djboss.server.base.dir=/
opt/jboss/standalone2/ \ -Djboss.socket.binding.port-
offset=10000
```

5. Wait for the second server to finish starting. Next, access the administration console. Update the CONSOLE handler registry level and set it to DEBUG. This can be configured in the logging subsystem handler section. Reboot the server to see the debug information displayed on the console.

   5.1. Access the administration console at http://localhost:19990.

   5.2. Click Settings at the top of the admin console. Click Subsystems in the first column; then click Records in the second column and finally View.

   5.3. Click HANDLER at the top of the page, and then click Console to access the Console handler configuration page.

   5.4. Under the Attributes header, click the Edit button.

   5.5. Change the INFO Level to DEBUG. Then click Save.

   5.6. Stop the EAP instance by pressing Ctrl+C in the terminal window that run EAP and use the following command, the same one used previously, to start the server again as the jboss user:

```
[student@workstation standalone]$ sudo -u jboss \ /opt/jboss-
eap-7.0/bin/standalone.sh \ -Djboss.server.base.dir=/
opt/jboss/standalone2/ \ -Djboss.socket.binding.port-
offset=10000
```

Verify that the server is running by accessing the administration console at http://localhost:19990.

5.7. Notice the additional debugging information provided by the server in the console that reflects the log level change made in the management console:

```
 ...
10:05:20,503 DEBUG [org.jboss.as.config] (MSC service thread 1-3) Configured system properties:
  [Standalone] =  awt.toolkit
   =
  sun.awt.X11.XToolkit file.encoding = UTF-8
  file.encoding.pkg = sun.io
  file.separator = / java.awt.graphicsenv
  =
  sun.awt.X11GraphicsEnvironment java.awt.headless = true
  java.awt.printerjob =
  sun.print.PSPrinterJob java.class.path = /opt/jboss-eap-7.0/
  jboss-modules.jar java.class.version = 52.0 java.endorsed.dirs = /usr/lib/
  jvm/java-1.8.0-
  openjdk-1.8.0.71-2.b15.el7_2.x86_64/jre/lib/endorsed
java.ext.dirs = /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.71-2.b15.el7_2.x86_64/
  jre/lib/ext:/usr/java/packages/lib/ext java.home = /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.71-2.b15.el7_2.x86_64/
jre java.io.tmpdir = /tmp


 ...
```

6. Access the administration console for the first server and notice that the Console handler's logging level is still set to INFO. The first server did not change, as the two servers have different configuration files in each of their respective base directories.

6.1. Access the administration console at http://localhost:9990.

6.2. Click Settings at the top of the admin console. Click Subsystems in the first column; then click Records in the second column and finally View.

6.3. Click HANDLER at the top of the page, then click Console to access the Console handler configuration page and note the log level.

7. Stop the EAP instances by pressing Ctrl+C in each terminal window that is running EAP. Use the following command to run the qualification script:

```
[student@workstation bin]$ lab configuring-lab grade
```

This concludes the lab work.

Summary

In this chapter, you learned the following:

- Running EAP as a standalone server gives users a way to run a single EAP instance with a single server.
- The default configuration for a standalone server is contained within the standalone.xml configuration file.
- Port offsets can be a simple solution to manage port conflicts when multiple EAP instances are running on the same host.
- The three folders required to use as the EAP base directory are: deployments, lib y configuration.
- Extensions are modules that extend the core capabilities of the server.
- The profile section of standalone.xml has a set of child elements of the subsystem, each with its own custom configuration scheme.
- Direct editing of the standalone.xml file is not recommended. Instead, the Users must use the EAP Command Line Interface (CLI) tool or the management console.
- Runtime properties can be overridden on the command line using the "-D" option when starting the server.
- The multiplexing approach in EAP 7 allows opening multiple connections on the same two ports (9990 and 8080, by default).