Guided Exercise: Deploy Applications with Management Tools

In this lab work, you will implement two applications using the management tools provided by EAP 7.

Resources	
Files:	**/home/student/JB248/labs/deploy-tools**
App URL:	**http://localhost:18080/example http://localhost:18080/version**

Results

You should be able to deploy an application using the management console and using the CLI.

Before you begin

Before you begin the guided exercise, run the following command to verify that EAP is installed to /opt/jboss-eap-7.0 and that no EAP instances are running, as well as to copy the files for the exercise :

```
[student@workstation ~]$ lab deploy-tools setup
```

1. Deploy using the management console.

1.1. Open a terminal window from the workstation virtual machine (Applications > Favorites > Terminal) and run the following commands to start the EAP server using /home/student/JB248/labs/standalone-instance as the base directory:

```
[student@workstation standalone]$ cd /opt/jboss-eap-7.0/bin [student@workstation
bin]$ ./standalone.sh \ -Djboss.server.base.dir=/home/
student/JB248/labs/standalone-instance/
```

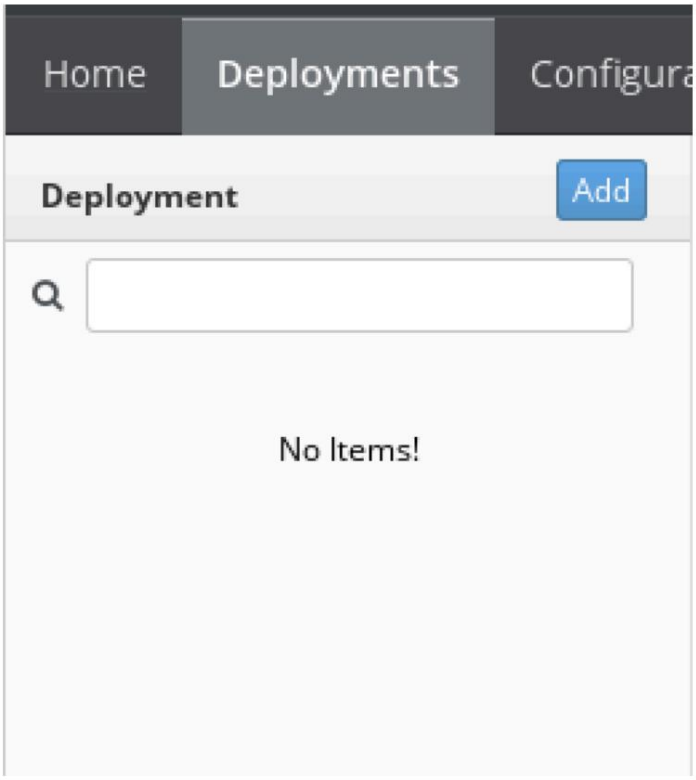
use

Remember that the standalone.xml file in the base directory was configured with the port-offset set to 10000. This means that the management console will be available on port 19990 and deployed applications will be available on port 18080.

1.2. Navigate to http://localhost:19990 to access the administration console page. Use the following credentials to access it:

- **username: jbossadm**
- **password: JBoss@RedHat123**

1.3. Click Deployments in the navigation menu bar. Since you haven't done any deployment yet, the list of deployments will be empty.



1.4. Click Add to add a new implementation.

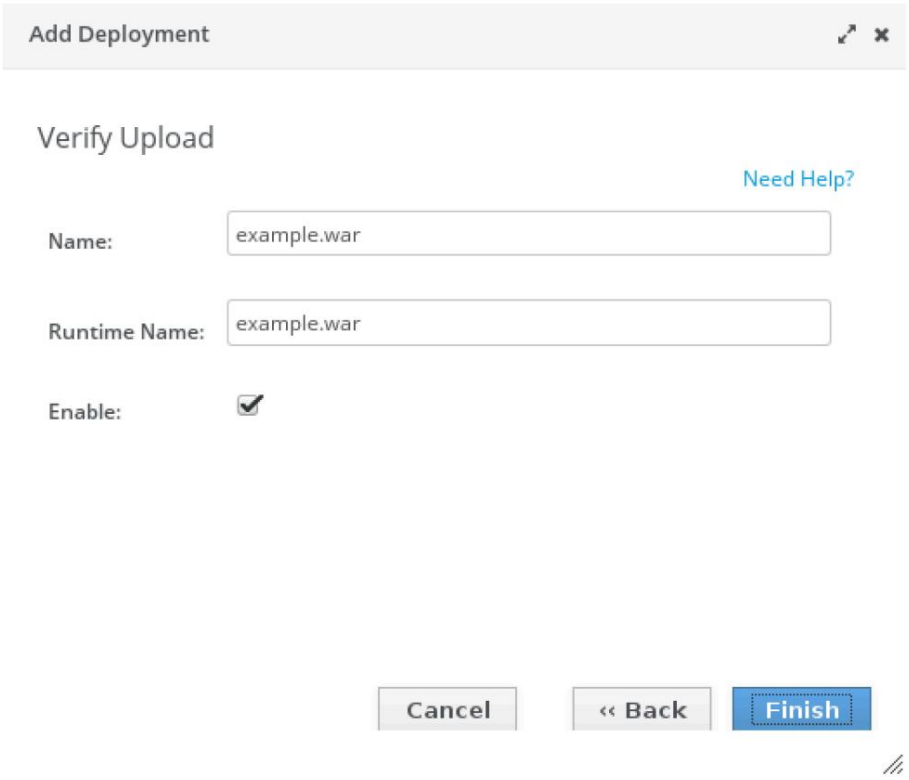
1.5. Select the Upload a new implementation option, and click Next.

use

Another option is available in the administration console to deploy an application. The Create an unmanaged deployment option deploys an application using a configured path that indicates a folder.

1.6. Click the Browse button and select the example.war file in your /home/student/JB248/labs/deploy-tools folder. Click Next to go to step 3 of the wizard.

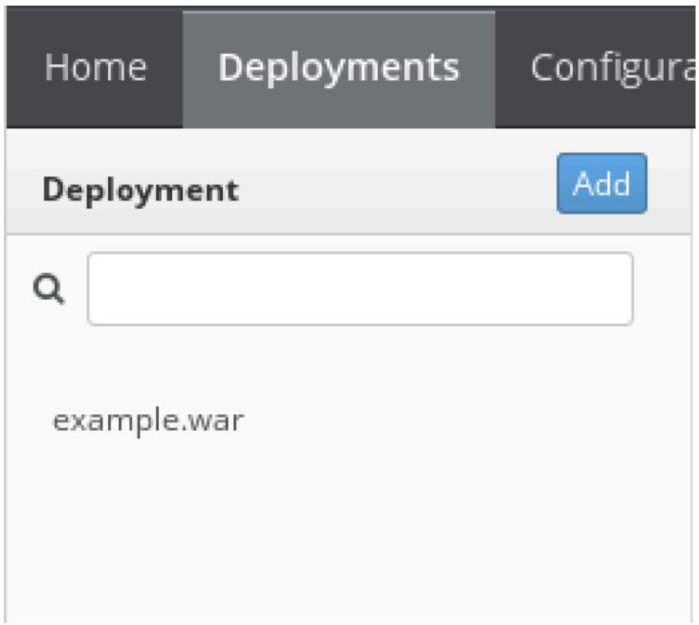
1.7. In step 3, you can use the default values or you can change the name and the runtime name of a deployment. The name identifies the implementation, and the runtime name specifies the context of the application. A runtime name of myapp means that the application is available at http:// server-address:port/myapp.



use

Loaded deployment is enabled by default, and will automatically deploy the app. Cleaning it will not deploy the app, and a manual deploy request must be executed.

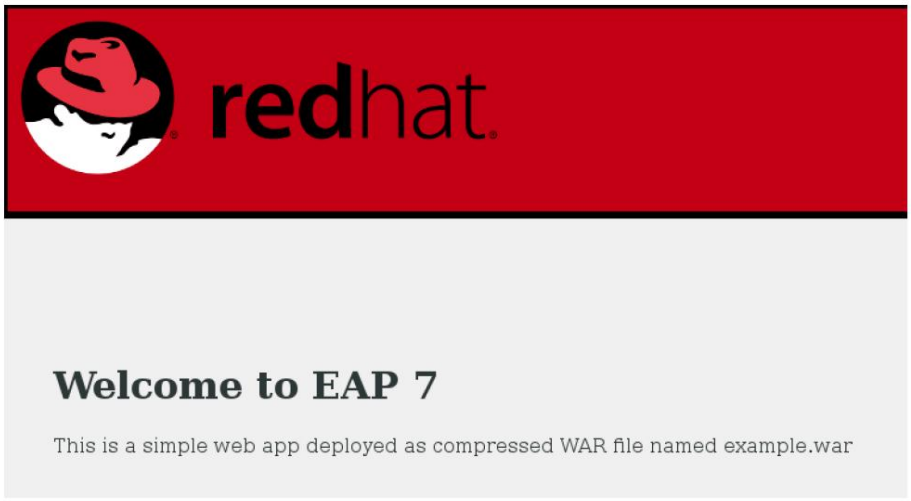
1.8. Click Finish to complete the wizard. You should now see example.war in the list of implementations.



1.9. Look at the terminal window of the running EAP instance. must see a result similar to the following:

```
INFO [org.jboss.weld.deployer] (MSC service thread 1-2) WFLYWELD0009: Starting weld service for deployment
  example.war
 ...OUTPUT OMMITED...
INFO [org.jboss.as.server] (XNIO-1 task-6) WFLYSRV0010: Deployed
  "example.war" (runtime-name : "example.war")
```

1.10. Go to http://localhost:18080/example. You should see the sample app, which displays a simple web page.



1.11. Open the file /home/student/JB248/labs/standalone-instance/ configuration/ standalone.xml to see its contents. must see one

<deployments> section at the end of this file containing your example.war implementation:

```
<deployments>
    <deployment name="example.war" runtime-name="example.war">
        <content sha1="0a07b224819ce516b231b1afba0eadc45b272298"/>
    </deployment>
</deployments>
```

1.12. Navigate to the /home/student/JB248/labs/standalone-instance/content/data folder. This folder contains child folders that are responsible for persisting all applications that were deployed using the management tools provided by EAP 7. You can identify each deployment using the sha1 code provided in the standalone.xml file. The first level of the directory contains a folder named using the first and second characters of sha1. This folder contains another folder named using the rest of the sha1 code, and finally a file named content will be the application.

2. Deactivate the deployment using the management console.

2.1. You can deactivate a deployment to undeploy the application without removing it from the server. This can be useful if you want to undeploy the app with a chance to redeploy it later without reloading the app. Return to the Administration Console Deployments page.

2.2. Click on the example.war application. A combo box should appear. Click the down arrow and select the Disable option to undeploy the standalone server application. Click Confirm when the confirmation window appears.

23. Click on the example.war application and notice that the application is no longer enabled.

2.4. Look at the terminal window of the running EAP instance. must see a result similar to the following:

```
INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 63)
  WFLYUT0022: Unregistered web context: /example
...OUTPUT OMMITED...
INFO [org.jboss.as.server] (XNIO-1 task-8) WFLYSRV0009: Undeployed "example.war" (runtime-
  name: "example.war")
```

2.5. Try reloading the URL http://localhost:18080/example in your browser. You should get a 404 error.

3. Deploy applications by using the CLI.

3.1. The CLI can implement an application. Open a new terminal window and start the CLI by running the jboss-cli.sh script in the EAP bin folder.

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/bin [student@workstation
bin]$ ./jboss-cli.sh --connect --controller=localhost:19990
```

3.2. Applications can be deployed using the deploy command, sending the location of the file to be deployed:

```
[standalone@localhost:19990 /] deploy \ /home/student/
JB248/labs/deploy-tools/version.war
```

3.3. Look at the terminal window of the running EAP instance. must see a result similar to the following:

```
INFO [org.jboss.weld.deployer] (MSC service thread 1-2) WFLYWELD0009: Starting weld service for deployment
  version.war
 ...OUTPUT OMMITED...
INFO [org.jboss.as.server] (XNIO-1 task-6) WFLYSRV0010: Deployed
  "version.war" (runtime-name : "version.war")
```

3.4. Go to http://localhost:18080/version. You must see the application of the version, which displays a simple web page.

4. Deactivate the deployment using the CLI.

4.1. It is also possible to disable an application using the CLI tool. Please list the available applications:

```
[standalone@localhost:19990 /] cd /deployment
[standalone@localhost:19990 deployment] ls
```

You should see the following output:

```
example.war version.war
```

4.2. Undeploy with the undeploy operation:

```
[standalone@localhost:19990 deployment] ./version.war:undeploy
```

4.3. Look at the terminal window of the running EAP instance. must see a result similar to the following:

```
INFO [org.wildfly.extension.undertow] (ServerService Thread Pool -- 63)
  WFLYUT0022: Unregistered web context: /version
 ...OUTPUT OMMITED...
INFO [org.jboss.as.server] (XNIO-1 task-8) WFLYSRV0009: Undeployed "version.war" (runtime-
  name: "version.war")
```

4.4. Try reloading the URL http://localhost:18080/version in your browser. You should get a 404 error.

4.5. Use the redeploy operation to reactivate the deployment:

```
[standalone@localhost:19990 deployment] ./version.war:redeploy
```

4.6. Try reloading the URL http://localhost:18080/version in your browser. You should see the version app.

5. Perform cleaning.

5.1. Remove the example.war application.

```
[standalone@localhost:19990 /] /deployment=example.war:remove
```

5.2. Remove the version.war application.

```
[standalone@localhost:19990 /] /deployment=version.war:remove
```

5.3. Exit the CLI tool.

```
[standalone@localhost:19990 /] exit
```

5.4. Stop the EAP instance by pressing Ctrl+C in the terminal window that is running EAP.

This concludes the guided exercise.

Lab Work: Configuring Scripts and Deploying Applications

In this lab work, you will configure the standalone server and deploy the bookstore application using the CLI tool.

Resources	
Files:	/home/student/JB248/labs/cli-lab
App URL:	http://localhost:8081 http://localhost:8081/ bookstore

Results

You must be able to configure the standalone server and deploy an application using the CLI tool.

before you start

Before beginning the guided exercise, run the following command to verify that EAP was installed to /opt/jboss-eap-7.0 and that no EAP instances are running, as well as to copy the files for the exercise:

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

The /opt/jboss/standalone2 server was deployed as a second server in the Configuring JBoss EAP lab in standalone mode. For this lab, the first server (configured in the /opt/jboss/standalone dir) will be taken down for maintenance, and an application will be deployed on the second server (configured in /opt/jboss/standalone2).

The system administrator will install a new service in the operating system, which requires port 8080. (The service is not related to Java.) To avoid port conflicts and due to network port limitations, applications on the second server must be available on port 8081. Also, the server log level was very verbose and the sysadmin was asking you to lower the verbose level to minimize registration overhead.

1. **Start a standalone EAP server using the /opt/jboss/standalone2 directory as the EAP base directory. This server must be started using the jboss user.**
   Before proceeding, wait for the server to finish starting up.

   Verify that the server is running by accessing the administration console at http://localhost:9990.

2. **By default, applications deployed in EAP 7 are available on port 8080. However, the sysadmin uses port 8080 for another service, and it conflicts with the EAP port responsible for providing access to web applications.**
   In this step, change only the web application port to be 8081.
   This port is defined in a socket-binding called http. This socket binding is available in a socket-binding-group named standard-

sockets. Using the CLI tool, find out the attribute responsible for defining the default port and change its value to 8081.

3. Change the logging level

The running standalone server is configured to display debugging messages in the CONSOLE. The log output was huge and the sysadmin asked you to decrease the number of logs. Change the level of this console handler to INFO using the CLI tool:

4. Deploy the bookstore application.

The bookstore app is an eCommerce app that sells books.
Deploy the application using the CLI tool. The application is available in the /tmp folder.

5. Rating

5.1. Exit the CLI tool:

```
[standalone@localhost:9990 deployment] exit
```

5.2. Run the following command to grade your work in this lab paper:

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

5.3. Stop the EAP instance by pressing Ctrl+C in the terminal window that is running EAP.

This concludes the lab work.

Solution

In this lab work, you will configure the standalone server and deploy the bookstore application using the CLI tool.

Resources	
Files:	/home/student/JB248/labs/cli-lab
App URL:	http://localhost:8081 http://localhost:8081/ bookstore

Results

You must be able to configure the standalone server and deploy an application using the CLI tool.

Before you begin

Before you begin the guided exercise, run the following command to verify that EAP is installed to /opt/jboss-eap-7.0 and that no EAP instances are running, as well as to copy the files for the exercise :

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

The /opt/jboss/standalone2 server was deployed as a second server in the Configuring JBoss EAP lab in standalone mode. For this lab, the first server (configured in the /opt/jboss/standalone dir) will be taken down for maintenance, and an application will be deployed on the second server (configured in /opt/jboss/standalone2).

The system administrator will install a new service in the operating system, which requires port 8080. (The service is not related to Java.) To avoid port conflicts and due to network port limitations, applications on the second server must be available on port 8081. Also, the server log level was very verbose and the sysadmin was asking you to lower the verbose level to minimize registration overhead.

1. Start a standalone EAP server using the /opt/jboss/standalone2 directory as the EAP base directory. This server must be started using the jboss user.
Before proceeding, wait for the server to finish starting up.

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/bin
[student@workstation bin]$ sudo -u jboss ./standalone.sh \
-Djboss.server.base.dir=/opt/jboss/standalone2/
```

Verify that the server is running by accessing the administration console at http://localhost:9990.

Open a web browser and navigate to the management console to see if it started.

2. By default, applications deployed in EAP 7 are available on port 8080. However, the sysadmin uses port 8080 for another service, and it conflicts with the EAP port responsible for providing access to web applications.
In this step, change only the web application port to be 8081.

This port is defined in a socket-binding called http. This socket binding is available in a socket-binding-group named standard sockets. Using the CLI tool, find out the attribute responsible for defining the default port and change its value to 8081.

2.1. Open a new terminal and connect to the CLI tool using localhost on port 9990.

```
[student@workstation ~]$ cd /opt/jboss-eap-7.0/bin
[student@workstation bin]$ sudo -u jboss ./jboss-cli.sh -c -- controller=localhost:9990
```

2.2. Go to the socket-binding-group named standard-sockets:

```
[standalone@localhost:9990 /] cd /socket-binding-group=standard-sockets
```

23. Go to the socket-binding named http:

```
[standalone@localhost:9990 socket-binding-group=standard-sockets] cd \ socket-binding=http
```

2.4. Change the port attribute to 8081:

```
[standalone@localhost:9990 socket-binding=http] :write-attribute\ (name=port,value=8081)
```

The following result is expected:

```
{
        "outcome" => "success",
        "response-headers" =>
                { "operation-requires-reload" => true, "process-
                state" => "reload-required"
        }
}
```

2.5. Read the http socket association resource properties:

```
[standalone@localhost:9990 socket-binding=http] :read-resource
```

The following result is expected:

```
{
        "outcome" => "success", "result"
        => {
                "client-mappings" => undefined, "fixed-port"
                => false, "interface" =>
                undefined, "multicast-address" =>
                undefined, "multicast-port" => undefined,
                "name" => "http", "port" => expression "$
                {jboss.http.port:8081}"

        }
```

```
}
```

2.6. The standalone server must be reloaded for it to start listening on the port 8081. Reload the server:

```
[standalone@localhost:9990 socket-binding=http] cd /
[standalone@localhost:9990 /] :reload
```

2.7. Open a web browser and point to http://localhost:8081 to test the new puerto.

3. Change the logging level

The running standalone server is configured to display debugging messages in the CONSOLE. The log output was huge and the sysadmin asked you to decrease the number of logs. Change the level of this console handler to INFO using the CLI tool:

3.1. Navigate to the console-handler named CONSOLE in the logging subsystem:

```
[standalone@localhost:9990 /] cd /subsystem=logging/console-handler=CONSOLE
```

3.2. Change the level to INFO:

```
[standalone@localhost:9990 console-handler=CONSOLE] :write-attribute\
(name=level,value=INFO)
```

3.3. Read the properties of the CONSOLE console handler resource:

```
[standalone@localhost:9990 console-handler=CONSOLE] :read-resource
```

The following result is expected:

```
{
    "outcome" => "success",
    "result" =>
        { "autoflush" => true,
        "enabled" => true,
        "encoding" => undefined, "filter"
        => undefined, "filter-spec" =>
        undefined, "formatter" =>
        "%d{HH:mm:ss,SSS} %-5p [%c] (%t) %s%e%n", "level" => "INFO", "name"
        => "CONSOLE", "named-
        formatter" => "COLOR-
        PATTERN", "target" => "System.out"

    }
}
```

4. Deploy the bookstore application.

The bookstore app is an eCommerce app that sells books.
Deploy the application using the CLI tool. The application is available in the /tmp folder.

4.1. Deploy the application using the deploy command:

```
[standalone@localhost:9990 console-handler=CONSOLE] cd /
[standalone@localhost:9990 /] deploy \ /tmp/
bookstore.war
```

4.2. Look at the terminal window of the running EAP instance. must see a result similar to the following:

```
INFO [org.jboss.as.server.deployment] (MSC service thread 1-3) WFLYSRV0027: Starting deployment of
  "bookstore.war" (runtime-name: "bookstore.war")
 ...OUTPUT OMMITED...
INFO [org.jboss.as.server] (XNIO-2 task-6) WFLYSRV0010: Deployed "bookstore.war" (runtime-
  name : "bookstore.war")
```

4.3. Open a web browser and navigate to http://localhost:8081/bookstore. Has to see the bookstore app.

5. Rating

5.1. Exit the CLI tool:

```
[standalone@localhost:9990 deployment] exit
```

5.2. Run the following command to grade your work in this lab paper:

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

5.3. Stop the EAP instance by pressing Ctrl+C in the terminal window that is running EAP.

This concludes the lab work.

Summary

In this chapter, you learned the following:

- There are three ways to configure EAP servers:
  - ÿ Using the management console
  - ÿ Using the CLI
  - ÿ Using XML files
- The CLI provides a new feature to incorporate an EAP server within the process of the CLI.
- EAP 7 still uses the DMR syntax.
- The CLI is started by running the jboss-cli.sh script in the bin folder of the EAP.
- Operations are a low-level way of managing the EAP server.
- The commands use a simple syntax and most of the commands are translated into operation requests.
- There are three ways to deploy an application: ÿ The management console
  - ÿ The CLI
  - ÿ The file system implementer
- The deployment process must specify a name and a runtime name, and if this is enabled or no.
- Using the file system deployer, the deployment process is managed using the marker files.