Red Hat JBoss Enterprise Application Platform 7.0

Using the Red Hat JBoss Enterprise Application Platform Docker Image

For Use with Red Hat JBoss Enterprise Application Platform 7.0

Last Updated: 2018-02-08

Red Hat JBoss Enterprise Application Platform 7.0 Using the Red Hat JBoss Enterprise Application Platform Docker Image

For Use with Red Hat JBoss Enterprise Application Platform 7.0

Legal Notice

Abstract

This document provides a practical guide for running Red Hat JBoss Enterprise Application Platform 7.0 inside a container.

Table of Contents

CHAPTER 1. OVERVIEW

IMPORTANT

Before getting started using JBoss EAP with containers, it is important to understand the basic concept behind Linux Containers as well as have Docker installed and configured.

Running JBoss EAP in a container lets you quickly and repeatably set up and tear-down instances of JBoss EAP in an isolated environment. Those containers are also extremely portable, tend to have a much smaller footprint than traditional virtual machines, and are extendable.

IMPORTANT

This guide covers the JBoss EAP 7 base image pulled from the Red Hat Docker Registry using Docker running on a Red Hat Enterprise Linux Server 7 host, specifically **`RHEL 7.2 x86_64`** running Docker version **`1.8.2-el7`** or later. The Red Hat Docker Registry is configured and available by default when running Docker on Red Hat Enterprise Linux Server 7. Users attempting to pull images from the Red Hat Docker Registry from hosts other than Red Hat Enterprise Linux Server 7 must also include the Red Hat Docker Registry address when issuing the **`pull`** command: **`registry.access.redhat.com`**. More details on using the **`pull`** command with different registries can be found in the Getting images from remote Docker registries section of Red Hat Enterprise Linux Atomic Host 7 Getting Started with Containers.

WARNING

The JBoss EAP base image for containers, **`registry.access.redhat.com/jboss-eap-7-tech-preview/eap70`**, distributed through the Red Hat Docker Registry is Technology Preview and is intended for development use only. It is *NOT* supported for use in production. This release is targeted at users and developers for experimentation purposes. This release is *NOT* targeting users who are running JBoss EAP containers in production. The Red Hat Docker Registry also requires that hosts have the appropriate entitlements to access this image. For more details on checking and configuring those entitlements, see Required Entitlements.

Limitations

The scope of this document is to describe how to use the JBoss EAP base image for containers in a development capacity. As a result, features such as high availability, clustering, load balancing, managed domains and Kubernetes are not covered. If you would like use any of those topics with JBoss EAP images you need to use the OpenShift Container Platform JBoss EAP images with OpenShift Container Platform. For more details, refer to the OpenShift Container Platform documentation.

1.1. REQUIRED ENTITLEMENTS

In order to pull the JBoss EAP base image for containers, **`registry.access.redhat.com/jboss-eap-7-tech-preview/eap70`**, from the Red Hat Docker Registry, you must have at least one Red Hat Middleware entitlement associated with your subscription.

To view your current subscriptions:

```
$ subscription-manager list --consumed
```

If you do not have the appropriate entitlements, you can check if any of them are available to you

```
$ subscription-manager list --available --all
```

and attach to the appropriate subscription:

```
$ subscription-manager attach --pool=POOL_ID
```

Once you have attached to a subscription with the appropriate entitlements, a key and a certificate for those entitlements are created in **`/etc/pki/entitlement`** for each subscription you have attached. The key files are named **`SUBSCRIPTION_SERIAL-key.pem`** and the certificate files are named **`SUBSCRIPTION_SERIAL.pem`**. For example, if you had the following subscription:

```
$ subscription-manager list --consumed
+-------------------------------------------+
   Consumed Subscriptions
+-------------------------------------------+
Subscription Name:  ....
....
Serial:              1122334455667788990
....
```

You would see the following files:

```
$ ls /etc/pki/entitlement
1122334455667788990-key.pem 1122334455667788990.pem
```

To use your entitlements when pulling from the Red Hat Docker Registry, you need to ensure the corresponding keys and certificates are added to **`/etc/docker/certs.d/`**. This is accomplished using **`rhsmcertd-worker`** to refresh the local certificate store, which is part of the RHSM plugin: **`subscription-manager-plugin-container`**.

To install the RHSM plugin **`subscription-manager-plugin-container`**:

```
$ sudo yum install subscription-manager-plugin-container
```

IMPORTANT

You must have the **`rhel-7-server-optional-rpms`** repository enabled to install **`subscription-manager-plugin-container`**. Instructions for enabling this repository as well as all other required repositories are covered in the Get Started with Docker Formatted Container Images section of *Red Hat Enterprise Linux Atomic Host 7 Getting Started with Containers*.

To run **`rhsmcertd-worker`** to refresh the local certificate store:

```
$ sudo /usr/libexec/rhsmcertd-worker
```

IMPORTANT

`rhsmcertd-worker` must be run as the superuser, otherwise it may fail to work without a warning.

You also need to restart Docker:

```
$ systemctl restart docker
```

To verify that the certificates and keys were added to **`/etc/docker/certs.d/`**, ensure the following files are now present in **`/etc/docker/certs.d/`**:

```
$ ls -l /etc/docker/certs.d/ | grep access.redhat.com
drwxr-xr-x. 2 root root   67 Jun 01 10:30 access.redhat.com
drwxr-xr-x. 2 root root   67 Jun 01 10:30 registry.access.redhat.com
```

To verify you can access the JBoss EAP images in the Red Hat Docker Registry, a search for the image using **`docker search`** should return at least one result from the Red Hat Docker Registry:

```
$ docker search registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
INDEX        NAME
DESCRIPTION                                     STARS     OFFICIAL
AUTOMATED
redhat.com   registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
...   0
```

CHAPTER 2. CONTAINER QUICK START

If you are familiar with the concepts behind Linux Containers and have Docker installed and configured, here is everything you need to pull the JBoss EAP 7 image from the Red Hat Docker Registry and start a container with a JBoss EAP instance running:

Pull and Start a Base JBoss EAP Instance

```
$ docker pull registry.access.redhat.com/jboss-eap-7-tech-preview/eap70

$ docker run -p 8080:8080 registry.access.redhat.com/jboss-eap-7-tech-
preview/eap70
```

NOTE

By default, the JBoss EAP image ships with only the public interface bound to **`0.0.0.0`** (**`-b 0.0.0.0`**). See the Extending the Image section for more details on binding to additional interfaces and publishing those ports.

Example Dockerfile for Binding the Management Interface

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
EXPOSE 9990
CMD ["/opt/eap/bin/standalone.sh", "-b", "0.0.0.0", "-bmanagement",
"0.0.0.0"]
```

Example Dockerfile for Creating an Admin User

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
RUN $JBOSS_HOME/bin/add-user.sh adminUser StrongPassword#1 --silent
```

Example Dockerfile for Executing a CLI Batch Script

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
COPY my-commands.txt $JBOSS_HOME
USER root
RUN chown jboss:jboss $JBOSS_HOME/my-commands.txt
USER jboss
RUN $JBOSS_HOME/bin/jboss-cli.sh --commands=embed-server,run-batch\ --
file=$JBOSS_HOME/my-commands.txt,stop-embedded-server
```

Example Dockerfile for Deploying an Application Using the Deployment Scanner

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
COPY app.war $JBOSS_HOME/standalone/deployments/
USER root
RUN chown jboss:jboss $JBOSS_HOME/standalone/deployments/app.war
USER jboss
```

CHAPTER 3. GETTING THE IMAGE

To begin working with JBoss EAP containers, you first need to get the JBoss EAP image from the Red Hat Docker Registry. This is done using the **`docker pull`** command.

Getting the JBoss EAP Image using Pull

```
$ docker pull registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
```

IMPORTANT

To ensure you are pulling the official JBoss EAP image that originates from the Red Hat Docker Registry, you need to include the host name of the registry, followed by a slash, and then image name when using **`docker pull`**, for example: **`docker pull registry.access.redhat.com/jboss-eap-7-tech-preview/eap70`**. If you do not include the host name of the registry, for example: **`docker pull jboss-eap-7-tech-preview/eap70`**, you may be pulling a different image from a different location and not the official JBoss EAP image.

CHAPTER 4. STARTING AND STOPPING THE CONTAINER

Once you have downloaded the JBoss EAP image, you can execute the **`docker run`** command to start up a container based on that image. By default, if you start a JBoss EAP container with no arguments, it will start a standalone JBoss EAP instance with the **`jboss.bind.address`** bound to **`0.0.0.0`**.

Start a Base JBoss EAP Instance

```
$ docker run registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
```

This starts the container and attaches to it. To stop this container, press **`Ctrl+C`** in the terminal where it was started. Alternatively, you may start a container in detached mode using the **`-d`** option.

Start a Detached Base JBoss EAP Instance

```
$ docker run -d registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
```

To see all running containers, use the **`docker ps`** command.

List Running Docker Containers

```
$ docker ps

CONTAINER ID        IMAGE               COMMAND                  CREATED
STATUS              PORTS
NAMES
0e0a93c7455f        my-eap-image        "/opt/eap/bin/standal"   6
minutes ago       Up 6 minutes        0.0.0.0:8080->8080/tcp,
0.0.0.0:9990->9990/tcp   distracted_mietner
```

Once a container is running, you can stop or reattach to it using the **`docker stop CONTAINER_ID`** and **`docker attach CONTAINER_ID`** commands.

Stop a Detached Base JBoss EAP Instance

```
$ docker stop 0e0a93c7455f
```

Publishing Ports

When you start a container, by default it will receive its own IP address on a separate docker network interface. You have the option to have any of the container's ports published to your host's ports so they are accessible using your host's address. This is accomplished using **`-p hostPort:containerPort`**.

Example Port Publishing

```
$ docker run -p 18080:8080 registry.access.redhat.com/jboss-eap-7-tech-
preview/eap70
```

The above example publishes port **`8080`** in the container to port **`18080`** on the host.

IMPORTANT

By default, the JBoss EAP image ships with only the public interface bound to **`0.0.0.0`** (**`-b 0.0.0.0`**). See the Extending the Image section for more details on binding to additional interfaces. You will need to do this *before* you can publish additional ports such as the management interface port.

You can also publish multiple ports at the same time as well as publish to the same port on the host.

Example Port Publishing

```
$ docker run -p 8080:8080 -p 9990:9990 registry.access.redhat.com/jboss-
eap-7-tech-preview/eap70
```

The above example publishes ports **`8080`** and **`9990`** from the container to the same ports on the host.

NOTE

You can also publish all exposed ports from the container to the host using **`-P`**.

IMPORTANT

If you want access any interfaces on the JBoss EAP instance running the container, for example the HTTP interface on **`8080`** or the management port on **`9990`**, you need to publish that port. The addition of Undertow in JBoss EAP 7 added HTTP Upgrade and allows for multiple protocols to be multiplexed over of a single port. JBoss EAP 7 moved nearly all of its protocols to be multiplexed over ports **`8080`** and **`9990`** so access to WebSockets, remoting EJB invocations, JMS, JMX, and AJP are also available when you publish those ports.

CHAPTER 5. EXTENDING THE IMAGE

In addition to running the containers based on the default JBoss EAP image, you can also extend that image and create your own custom images based on that default image.

Example Dockerfile

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
EXPOSE 9990
CMD ["/opt/eap/bin/standalone.sh", "-b", "0.0.0.0", "-bmanagement",
"0.0.0.0"]
```

The above example creates a new image based on the default JBoss EAP image but will now run **`$JBOSS_HOME/bin/standalone.sh -b 0.0.0.0 -bmanagement 0.0.0.0`** if no commands are passed to it.

You can use the **`docker build`** command to build your new image based on the **`Dockerfile`** and then run that image using the **`docker run`** command.

Example Building and Running a New Image from a Dockerfile

```
$ docker build --tag=myeapimage /dir/with/dockerfile/

$ docker run myeapimage
```

NOTE

Using the **`--tag=TAG-NAME`** flag is optional, but it can make it easier to identify your image when using the **`docker run`** command. If you do not use the **`--tag=TAG-NAME`** flag, you will have to use the image's ID when using the **`docker run`** command, which can be found by running the **`docker images`** command.

5.1. ADDING A MANAGEMENT USER

To add a management user, you can use the **`RUN`** command in your **`Dockerfile`** to execute the **`add-user.sh`** script.

Example Dockerfile for Creating a Management User

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
RUN $JBOSS_HOME/bin/add-user.sh adminUser StrongPassword#1 --silent
```

The above example adds a user **`adminUser`** with a password of **`StrongPassword#1`** to the image.

5.2. MODIFYING THE CONFIGURATION

If you want to modify the configuration of your JBoss EAP instance in your container, for example adding a security realm or data source, you can do one of the following:

- Modify the a running container using the management console or management CLI.
- Use the **`RUN`** command in your **`Dockerfile`** to run a command or set of batch commands.

To modify the configuration of a running container, simply access the management console or management CLI of your JBoss EAP instance and make your desired configuration updates. If the appropriate ports are published, you should be able to access the desired management interface using the host machine's address, for example `http://localhost:9990`.

To modify the image using a **`Dockerfile`**, you can use the **`RUN`** command and invoke the **`jboss-cli.sh`**.

Example Dockerfile for Executing a CLI Batch Script

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
COPY my-commands.txt $JBOSS_HOME
USER root
RUN chown jboss:jboss $JBOSS_HOME/my-commands.txt
USER jboss
RUN $JBOSS_HOME/bin/jboss-cli.sh --commands=embed-server,run-batch\ --
file=$JBOSS_HOME/my-commands.txt,stop-embedded-server
```

The above example invokes the **`jboss-cli.sh`** script and uses the **`--commands=`** flag to pass in a set of commands to modify the configuration. The **`embed-server`** and **`stop-embedded-server`** commands starts and stops the server, which allows you to modify the configuration. The **`run-batch\ --file=$JBOSS_HOME/my-commands.txt`** command runs a batched set of commands from the **`$JBOSS_HOME/my-commands.txt`** file. The **`my-commands.txt`** file exists on the host's file system and was added to the container's **`$JBOSS_HOME`** using the **`COPY`** command in the **`Dockerfile`** and is updated to the proper permissions using the **`RUN`** command.

NOTE

In the JBoss EAP container, JBoss EAP runs as the **`jboss`** user but copied files are brought in as **`root`**.

5.3. DEPLOYING APPLICATIONS

To deploy an application to a JBoss EAP instance running inside a container, you can do one of the following:

- Deploy the application to a running container using the management console or management CLI.
- Use the **`COPY`** command in the **`Dockerfile`** to deploy the application with the deployment scanner when the container starts.

To deploy an application a running container, simply access the management console or management CLI of your JBoss EAP instance and deploy your application. If you the appropriate ports published you should be able to access the desired management interface using the host machine's address, for example `http://localhost:9990`.

To use the deployment scanner to deploy an application when the container starts, you can use the **`COPY`** command in the **`Dockerfile`**.

Example Dockerfile for Deploying an Application Using the Deployment Scanner

```
FROM registry.access.redhat.com/jboss-eap-7-tech-preview/eap70
COPY app.war $JBOSS_HOME/standalone/deployments/
```

```
USER root
RUN chown jboss:jboss $JBOSS_HOME/standalone/deployments/app.war
USER jboss
```

The above example copies the **app.war** file to a location which is scanned by the deployment scanner and updates the file to have the appropriate permissions. This will cause **app.war** to be deployed when the JBoss EAP instance is started as the container is started.

NOTE

In the JBoss EAP container, JBoss EAP runs as the **jboss** user but copied files are brought in as **root**.

5.4. PERSISTING CHANGES

In addition to using a Dockerfile to build a new image with a modified configuration, you can also modify a running container and persist those changes. To do so, you need to ensure that the management ports have been published so you can access the management interface. From there, you can make any desired changes as you normally would, for example deploying an application or adding a security domain. To persist those changes, you can use the **docker commit** command.

Persisting a Change

```
$ docker commit -m "commit message" -a "commit author" running-container-
id new-image-name
```

This creates a new image based on the container with the additional changes. It will appear in your list of images, using **docker images**, and you use that image as you would any other, for example **docker run new-image-name**.

Revised on 2018-02-08 10:20:17 EST