# Red Hat JBoss Enterprise Application Platform 7.0

## Deploying Red Hat JBoss Enterprise Application Platform on Amazon EC2

For Use with Red Hat JBoss Enterprise Application Platform 7.0

Last Updated: 2018-02-08

# Red Hat JBoss Enterprise Application Platform 7.0 Deploying Red Hat JBoss Enterprise Application Platform on Amazon EC2

For Use with Red Hat JBoss Enterprise Application Platform 7.0

## Legal Notice

## Abstract

This document provides information about deploying Red Hat JBoss Enterprise Application Platform on Amazon EC2.

# Table of Contents

# CHAPTER 1. ABOUT AMAZON EC2

Amazon Elastic Compute Cloud (Amazon EC2) is a service operated by amazon.com that provides customers with a customizable virtual computing environment. An Amazon Machine Image (AMI) can be booted using the service to create a virtual machine or instance. Users can install the software they require on an instance and are charged according to the capacity used. Amazon EC2 is designed to be flexible and allow users to quickly scale their deployed applications.

You can read more about it at the Amazon EC2 website, http://aws.amazon.com/ec2/.

### About Amazon Machine Instances (AMIs)

An Amazon Machine Image (AMI) is a template for a EC2 virtual machine instance. Users create EC2 instances by selecting an appropriate AMI to create the instance from. The primary component of an AMI is a read-only filesystem that contains an installed operating system as well as other software. Each AMI has different software installed for different use cases. Amazon EC2 includes many AMIs to choose from provided by both amazon.com and third parties. Users can also create their own custom AMIs.

## About Red Hat Cloud Access

Red Hat Cloud Access is a Red Hat subscription feature that provides support for JBoss EAP on Red Hat certified cloud infrastructure providers, such as Amazon EC2 and Microsoft Azure. Red Hat Cloud Access allows you to move your subscriptions between traditional servers and public cloud-based resources in a simple and cost-effective manner.

You can find more information about Red Hat Cloud Access on the Customer Portal.

## Red Hat Cloud Access Features

Membership in the Red Hat Cloud Access program provides access to supported private Amazon Machine Images (AMIs) created by Red Hat.

The Red Hat AMIs have the following software pre-installed and fully supported by Red Hat:

- Red Hat Enterprise Linux
- JBoss EAP
- Product updates with RPMs using Red Hat Update Infrastructure

Each of the Red Hat AMIs are only a starting point, requiring further configuration to the requirements of your application.

**IMPORTANT**

Red Hat Cloud Access does not currently provide support for the `full-ha` profile, in either standalone instances or a managed domain.

**NOTE**

For information about installing Red Hat JBoss Operations Network, see the Red Hat JBoss Operations Network *Installation Guide*.

For information about configuring Red Hat JBoss Operations Network, see *Configuring JBoss ON Servers, Agents, and Storage Nodes*.

## Supported Amazon EC2 Instance Types

Red Hat Cloud Access supports the following Amazon EC2 instance types.

**NOTE**

For more details about each instance, refer Amazon Elastic Compute Cloud User Guide for Linux Instances.

**Table 1.1. Supported Amazon EC2 Instance Types**

| Instance Type | Description |
|---|---|
| Standard Instance | Standard Instances are general purpose environments with a balanced memory-to-CPU ratio. Smallest instance types available, which are capable of handling JBoss EAP, are **`t2.small`** and **`m3.medium`**. |
| High Memory Instance | High Memory Instances have more memory allocated to them than Standard Instances. High Memory Instances are suited for high throughput applications such as databases or memory caching applications. Smallest available and supported instance type is **`r3.large`**. |
| High CPU Instance | High CPU Instances have more CPU resources allocated than memory and are suited for relatively low throughput but CPU intensive applications. Smallest available and supported instance types are **`c3.large`** and **`c4.large`**. |

**IMPORTANT**

The instance types Micro (**`t2.micro`**) and Nano (**`t2.nano`**) are not suitable for deployment of JBoss EAP. JBoss EAP 7.0 AMIs are built from a snapshot, which requires a volume of at least 10 GB. This can be set in EC2 console when creating the instance. If the volume assigned is too small, the instance creation will fail.

## Supported Red Hat AMIs

The supported Red Hat AMIs can be identified by their AMI name. JBoss EAP AMIs are named using the following syntax:

```
RHEL-osversion-_HVM_GA-JBEAP-version-creationdate-arch-1-Access2-GP2
```

- **`version`** is the version number of JBoss EAP installed in the AMI. Example **`6.3`**.
- **`osversion`** is the version number of Red Hat Enterprise Linux installed in the AMI. Example **`6.2`**.
- **`arch`** is the architecture of the AMI. This will be **`x86_64`** or **`i386`**.
- **`creationdate`** is the date that the AMI was created in the format of YYYYMMDD. Example **`20160315`**.

Example AMI name: **`RHEL-7.2_HVM_GA-JBEAP-7.0.0-20160516-x86_64-1-Access2-GP2`**.

# CHAPTER 2. LAUNCHING A JBOSS EAP INSTANCE ON AMAZON EC2 CONSOLE

You can launch a JBoss EAP instance on Amazon EC2 using the EC2 console.

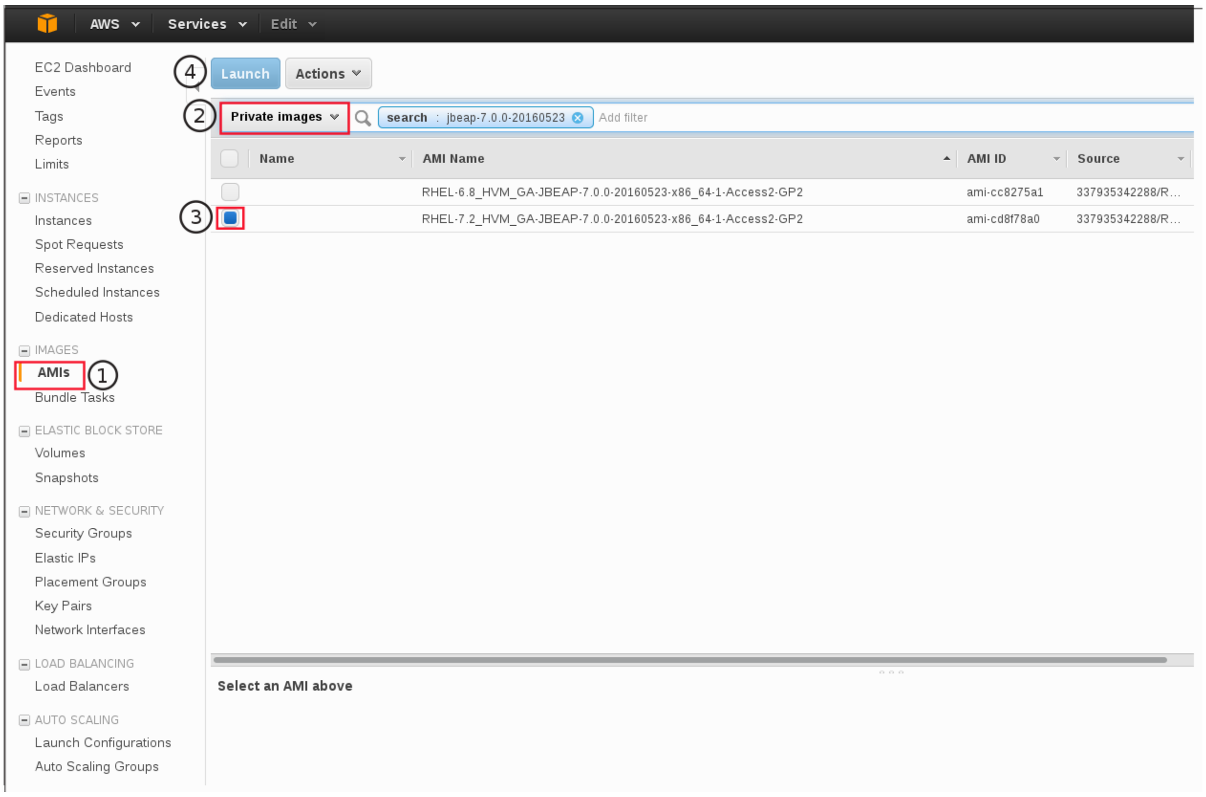
**NOTE**

You can also launch an instance using the AWS Command Line Interface. For more information about AWS CLI, see AWS CLI.

1. Open the Amazon EC2 console.

2. From the Amazon EC2 console, click **AMIs**.

**Figure 2.1. Amazon EC2 Console**



| Name | AMI Name | AMI ID | Source |
|---|---|---|---|
| | RHEL-6.8_HVM_GA-JBEAP-7.0.0-20160523-x86_64-1-Access2-GP2 | ami-cc8275a1 | 337935342288/R... |
| | RHEL-7.2_HVM_GA-JBEAP-7.0.0-20160523-x86_64-1-Access2-GP2 | ami-cd8f78a0 | 337935342288/R... |

3. Search for **`jbeap`** AMI in **Private images** and select the AMI. For example, **`RHEL-7.2_HVM_GA-JBEAP-7.0.0-20160516-x86_64-1-Access2-GP2`**.

4. Choose an instance type. For more information on supported Amazon EC2 instance types, see Supported Amazon EC2 Instance Types.

5. In the **Configure Instance Details** section, configure the instance settings.

6. In the **Advanced Details** section, **User data** box, you can paste the sample script to run JBoss EAP when the instance is launched. For information about the sample script, see Launching JBoss EAP on Amazon EC2 Using Script.

**NOTE**

If you have some specific requirements, you can specify the storage, tag the instance, and configure the security group details.

7. Click **Review and Launch**. This takes you directly to the Review Instance Launch page.

8. Click **Launch** to choose a key pair and launch the instance.

**NOTE**

If you have not selected a key pair, you need to specify a key pair before you launch an instance.

# CHAPTER 3. LAUNCH A NON-CLUSTERED JBOSS EAP INSTANCE

This topic lists the steps to launch a non-clustered instance of JBoss EAP on a Red Hat AMI (Amazon Machine Image).

## Pre-requisites

- A suitable Red Hat AMI. Refer to Supported Red Hat AMIs.
- A pre-configured Security Group that allows incoming requests on at least ports 22, 8080, and 9990.

## Launch a Non-Clustered JBoss EAP Instance

**NOTE**

You can connect to an EC2 instance through **ssh** as a **ec2-user** user. If you need administrative privileges, you can change to root later For example,

```
$ ssh -l ec2-user ${INSTANCE_PUBLIC_IP}
...
$ sudo su -
```

- Launch the Red Hat AMI instance.
  A non-clustered instance of JBoss EAP has been configured, and launched on a Red Hat AMI.
- To configure JBoss EAP, you can pass arguments to the service directly. Some arguments may not be handled in this way. The location of the service configuration files is:
  - RHEL 6: **/etc/sysconfig/eap7-standalone**
  - RHEL 7: **/etc/opt/rh/eap7/wildfly/eap7-standalone.conf**

  **NOTE**

  - For system path details, see System Paths.
  - For complex configuration, you can either use the **standalone.conf** file in the JBoss EAP **bin** directory: **/opt/rh/eap7/root/usr/share/wildfly/bin/**, or you can start the JBoss EAP service and configure the server using CLI. The script can be found in the **bin** directory. Then, reload the configuration.
  - **yum -y update** should be run regularly, to apply security fixes and enhancements.

- To start JBoss EAP on RHEL 6, run the following command:

  ```
  $ service eap7-standalone start
  ```

  To start JBoss EAP on RHEL 7, run the following command:

```
$ systemctl start eap7-standalone
```

- To stop JBoss EAP, run the following command:

  ```
  $ service eap7-standalone stop
  ```

  Or

  ```
  $ systemctl stop eap7-standalone
  ```

**NOTE**

**`systemctl`** command is relevant to only RHEL 7.

**NOTE**

If you want to bind JBoss EAP to a different IP address, add the following line in the **`/etc/opt/rh/eap7/wildfly/eap7-standalone.conf`** file on RHEL 7. The internal IP address is translated into a public IP address by EC2.

```
WILDFLY_BIND=$YOUR_PRIVATE_IP_ADDRESS
```

# CHAPTER 4. LAUNCHING NON-CLUSTERED MANAGED DOMAIN

## 4.1. LAUNCH AN INSTANCE TO SERVE AS A DOMAIN CONTROLLER

This topic lists the steps to launch a non-clustered JBoss EAP managed domain on a Red Hat AMI (Amazon Machine Image).

### Pre-requisite

- A suitable Red Hat AMI. Refer to Supported Red Hat AMIs.

### Launch a Non-Clustered JBoss EAP Instance

**NOTE**

You can connect to an EC2 instance through **ssh** as the **ec2-user** user. If you need administrative privileges, you can change to **root** later. For example,

```
$ ssh -l ec2-user ${INSTANCE_PUBLIC_IP}
...
$ sudo su -
```

- Launch the Red Hat AMI instance.
  A non-clustered instance of JBoss EAP has been configured, and launched on a Red Hat AMI.
- To configure JBoss EAP, you can pass arguments to the service directly. Some arguments may not be handled in this way. The location of the service configuration files is:
  - RHEL 6: **/etc/sysconfig/eap7-domain**
  - RHEL 7: **/etc/opt/rh/eap7/wildfly/eap7-domain.conf**

  **NOTE**

  - For system path details, see System Paths.
  - For information about configuring JBoss EAP subsystems for Amazon EC2, see Configuring JBoss EAP Subsystems to Work on Cloud Platforms.
  - For complex configuration, you can either use the **domain.conf** file in the JBoss EAP **bin** directory: **/opt/rh/eap7/root/usr/share/wildfly/bin/**, or you can start the JBoss EAP service and configure the server using the management CLI. The script can be found in the **bin** directory. Then, reload the configuration.
  - **yum -y update** should be run regularly, to apply security fixes and enhancements.
- To start JBoss EAP on RHEL 6, run the following command:

  ```
  $ service eap7-domain start
  ```

To start JBoss EAP on RHEL 7, run the following command:

```
$ systemctl start eap7-domain
```

- To stop JBoss EAP, run the following command:

```
$ service eap7-domain stop
```

Or

```
$ systemctl stop eap7-domain
```

**NOTE**

**`systemctl`** command is relevant to only RHEL 7.

**NOTE**

If you want to bind JBoss EAP to a different IP address, add the following line in the **`/etc/opt/rh/eap7/wildfly/eap7-domain.conf`** file on RHEL 7. The internal IP address is translated into a public IP address by EC2.

```
WILDFLY_BIND=$YOUR_PRIVATE_IP_ADDRESS
```

## 4.2. LAUNCH ONE OR MORE INSTANCES TO SERVE AS HOST CONTROLLERS

This topic lists the steps to launch one or more instances of JBoss EAP to serve as non-clustered host controllers on a Red Hat AMI.

Configure and launch the non-clustered domain controller. Refer to Launch an Instance to Serve as a Domain Controller.

**NOTE**

- For system path details, see System Paths.
- For information about configuring JBoss EAP subsystems for Amazon EC2, see Configuring JBoss EAP Subsystems to Work on Cloud Platforms.

### For Domain Controller Instance

For a managed domain running on Amazon EC2, in addition to static domain controller discovery, host controllers can dynamically discover a domain controller using the Amazon S3 storage system. In particular, host controllers and the domain controller can be configured with information needed to access an Amazon S3 bucket.

Using this configuration, when a domain controller is started, it writes its contact information to an S3 file in the bucket. Whenever a host controller attempts to contact the domain controller, it gets the domain controller’s contact information from the S3 file.

This means that if the domain controller’s contact information changes, for example, it is common for an EC2 instance’s IP address to change when it is stopped and started, the host controllers do not need to be reconfigured. The host controllers are able to get the domain controller's new contact information from the S3 file.

**NOTE**

To know more about sample script for user data, see Example User Data for Clustered JBoss EAP Instances

The manual domain controller discovery configuration is specified using the following properties:

- **`access-key`**: The Amazon AWS user account access key.
- **`secret-access-key`**: The Amazon AWS user account secret access key.
- **`location`**: The Amazon S3 bucket to be used.

1. Copy the **`domain-ec2.xml`** file from **`/etc/opt/rh/eap7/jboss-ec2-eap/domain`** to the JBoss EAP configuration directory.
2. Set the following variables in the appropriate service configuration file:

```
WILDFLY_SERVER_CONFIG=domain-ec2.xml
WILDFLY_HOST_CONFIG=host-master.xml
```

3. Add S3 domain controller discovery configuration to the **`domain-ec2.xml`** file:

```
<local>
    <discovery-options>
        <discovery-option name="s3-discovery"
module="org.jboss.as.host-controller"
code="org.jboss.as.host.controller.discovery.S3Discovery">
            <property name="access-key" value="S3_ACCESS_KEY"/>
            <property name="secret-access-key"
value="S3_SECRET_ACCESS_KEY"/>
            <property name="location" value="S3_BUCKET_NAME"/>
        </discovery-option>
    </discovery-options>
</local>
```

# CHAPTER 5. LAUNCHING CLUSTERED JBOSS EAP

## 5.1. LAUNCH CLUSTERED JBOSS EAP AMIS (WITHOUT MOD_CLUSTER AND VPC)

This topic lists the steps to launch clustered JBoss EAP AMIs without mod_cluster and VPC.

**NOTE**

- You can use the example configuration scripts that are provided with the image. For system path details, see System Paths.
- For information about configuring JBoss EAP subsystems for Amazon EC2, see Configuring JBoss EAP Subsystems to Work on Cloud Platforms.

To start clustered JBoss EAP AMI on standalone server instance, you can use the example **`/etc/opt/rh/eap7/jboss-ec2-eap/standalone/standalone-ec2-ha.xml`** that contains a preconfigured S3_PING JGroups stack. For more information, see S3_PING. This **`standalone-ec2-ha.xml`** profile file must be copied from **`/etc/opt/rh/eap7/jboss-ec2-eap/standalone/`** to the JBoss EAP configuration directory **`/opt/rh/eap7/root/usr/share/wildfly/standalone/configuration/`**. Then, you have to add the following line to the JBoss EAP service configuration file:

```
WILDFLY_SERVER_CONFIG=standalone-ec2-ha.xml
```

A unique **`instance-id`** needs to be set for each standalone server instance in the **`undertow`** subsystem. The example **`standalone-ec2-ha.xml`** configuration sets this using **`instance-id="${jboss.jvmRoute}"`**. A value for **`jboss.jvmRoute`** can be specified in **`standalone.conf`** using the **`JAVA_OPTS`** variable. Alternatively, a value for the **`instance-id`** can be set manually by editing **`standalone-ec2-ha.xml`** or using the CLI.

The **`jgroups`** subsystem in the EC2 configuration file requires some **`S3_PING`** specific properties to discover cluster members. You must specify access key to S3, secret access key, and the S3 bucket to use for discovery. These properties can either be specified as Java options or put directly into the XML file by editing it or using CLI.

The S3 bucket for discovery needs to be created, for more information, see Amazon Simple Storage Service Documentation. You may also have to configure the required permissions. The JGroups stack needs to be bound to an IP address, which is used to communicate with other nodes. This can be done by adding Java options, along with S3 Java options to the **`/opt/rh/eap7/root/usr/share/wildfly/bin/standalone.conf`** file. For example, if the private IP address was **`10.10.10.10`**, then you would add the following line to the **`standalone.conf`** file:

```
JAVA_OPTS="$JAVA_OPTS -Djboss.bind.address.private=10.10.10.10"
```

You can deploy a sample application: **`/opt/rh/eap7/root/usr/share/java/eap7-jboss-ec2-eap-samples/cluster-demo.war`** and observe the logs in **`/opt/rh/eap7/root/usr/share/wildfly/standalone/log/server.log`** to see that the JBoss EAP servers have created a cluster.

### For Domain Controller Instance

1. Copy the **`domain-ec2.xml`** file from **`/etc/opt/rh/eap7/jboss-ec2-eap/domain`** to the JBoss EAP configuration directory.

2. Set the following variables in the appropriate service configuration file:

```
WILDFLY_SERVER_CONFIG=domain-ec2.xml
WILDFLY_HOST_CONFIG=host-master.xml
```

3. Add S3 domain controller discovery configuration to the **`host-master.xml`** file:

```
<local>
    <discovery-options>
        <discovery-option name="s3-discovery"
module="org.jboss.as.host-controller"
code="org.jboss.as.host.controller.discovery.S3Discovery">
            <property name="access-key" value="S3_ACCESS_KEY"/>
            <property name="secret-access-key"
value="S3_SECRET_ACCESS_KEY"/>
            <property name="location" value="S3_BUCKET_NAME"/>
        </discovery-option>
    </discovery-options>
</local>
```

4. Configure users and add the secret values for users to the host controller instances. For more information, see Create a Managed Domain on Two Machines in the JBoss EAP *Configuration Guide*.

### For Host Controller Instance

1. Set the following variable in the appropriate service configuration file:

```
WILDFLY_HOST_CONFIG=host-slave.xml
```

2. Add S3 domain controller discovery configuration to the **`host-slave.xml`** file:

```
<remote security-realm="ManagementRealm">
    <discovery-options>
        <discovery-option name="s3-discovery"
module="org.jboss.as.host-controller"
code="org.jboss.as.host.controller.discovery.S3Discovery">
            <property name="access-key" value="S3_ACCESS_KEY"/>
            <property name="secret-access-key"
value="S3_SECRET_ACCESS_KEY"/>
            <property name="location" value="S3_BUCKET_NAME"/>
        </discovery-option>
    </discovery-options>
</remote>
```

**NOTE**

For information about S3 domain controller discovery, see Launch One or More Instances to Serve as Host Controllers.

**WARNING**

Running a JBoss EAP cluster in a subnet with network mask smaller than 24-bits or spanning multiple subnets complicates acquiring a unique server peer ID for each cluster member.

**IMPORTANT**

The auto-scaling Amazon EC2 feature can be used with JBoss EAP cluster nodes. However, ensure it is tested before deployment. You should ensure that your particular workloads scale to the required number of nodes, and that the performance meets your needs for the instance type you are planning to use, different instance types receive a different share of the EC2 cloud resources.

Furthermore, instance locality and current network/storage/host machine/RDS utilization may affect cluster performance. Test with your expected real-life loads and try to account for unexpected conditions.

**WARNING**

The Amazon EC2 *scale-down* action terminates the nodes without any chance to gracefully shut down and as some transactions might be interrupted, other cluster nodes and load balancers need time to fail over. This is likely to impact your application users' experience.

It is recommended that you either scale down the application cluster manually by disabling the server from the mod_cluster management interface until processed sessions are completed, or shut down the JBoss EAP instance gracefully using SSH access to the instance or Red Hat JBoss Operations Network.

Test your procedure for scaling down does not lead to adverse effects on your users' experience. Additional measures might be required for particular workloads, load balancers, and setups.

## 5.2. LAUNCH CLUSTERED JBOSS EAP AMIS (WITH MOD_CLUSTER AND VPC)

This topic lists the steps to launch an Apache HTTP server instance to serve as a **`mod_cluster`** proxy and a NAT instance for the Virtual Private Cloud (VPC).

**NOTE**

- You can use the example configuration scripts that are provided with the image. For system path details, see System Paths.
- For information about configuring JBoss EAP subsystems for Amazon EC2, see Configuring JBoss EAP Subsystems to Work on Cloud Platforms.

An Amazon Virtual Private Cloud (Amazon VPC) is a feature of Amazon Web Services (AWS) that allows you to isolate a set of AWS resources in a private network. The topology and configuration of this private network can be customized to your needs.

For more information about Amazon Virtual Private Cloud website, see Amazon Virtual Private Cloud (VPC).

**NOTE**

If you start a cluster with a **`mod_cluster`** load balancer inside a VPC, the JBoss EAP servers are inaccessible to public. The **`mod_cluster`** load balancer can be the only endpoint that is connected to the Internet.

For setting up domain controller instance, see Launch an Instance to Serve as a Domain Controller.

For setting up host controller instance, see Launch One or More Instances to Serve as Host Controllers.

**NOTE**

For information about S3 domain controller discovery, see Launch One or More Instances to Serve as Host Controllers.

### 5.2.1. To launch clustered AMIs with VPC and mod_cluster

**NOTE**

Configuring the VPC is optional. For more information, see http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/default-vpc.html#detecting-platform.

1. Install **`jbcs-httpd24-mod_cluster-native`** package and all of its dependencies. The **`mod_cluster`** configuration file is installed in **`/opt/rh/jbcs-httpd24/root/etc/httpd/conf.d/mod_cluster.conf`**.

   **NOTE**

   For more information about installation of Red Hat JBoss Core Services Apache HTTP Server, see the *Apache HTTP Server Installation Guide*.

2. Disable advertising for **`mod_cluster`**. Add the following to **`VirtualHost`** in the **`/opt/rh/jbcs-httpd24/root/etc/httpd/conf.d/mod_cluster.conf`** configuration file.

```
ServerAdvertise Off
EnableMCPMReceive
# AdvertiseFrequency # comment out AdvertiseFrequency if present
```

3. Allow ports in **`SELinux`**. If required, configure the **`iptables`**. Ports can be allowed in SELinux by using the **`semanage port -a -t http_port_t -p tcp $PORT_NR`** command.

4. Configure JBoss EAP to look for **`mod_cluster`** proxy on the address that **`mod_cluster`** listens on.

**NOTE**

The **`/etc/opt/rh/eap7/jboss-ec2-eap/standalone/standalone-mod_cluster-ec2-ha.xml`** sample configuration file has been provided. You need to configure the **`mod-cluster-binding`** and **`outbound-socket-binding`**. Insert the **`mod_cluster`** address and port.

# CHAPTER 6. TROUBLESHOOTING

## 6.1. ABOUT TROUBLESHOOTING AMAZON EC2

EC2 provides an Alarm Status for each instance, indicating severe instance malfunction but the absence of such an alarm is no guarantee that the instance has started correctly and services are running properly. It is possible to use Amazon CloudWatch with its custom metric functionality to monitor instance services' health but use of an enterprise management solution is recommended.

## 6.2. DIAGNOSTIC INFORMATION

In case of a problem being detected by the JBoss Operations Network, Amazon CloudWatch or manual inspection, common sources of diagnostic information are:

- **`/var/log`** also contains all the logs collected from machine startup, JBoss EAP, httpd and most other services.

JBoss EAP log files can be found in **`/opt/rh/eap7/root/usr/share/wildfly/`**.

Access to these files is only available using an SSH session.

**NOTE**

For more information about how to configure and establish an SSH session with an Amazon EC2 instance, see Amazon EC Getting Started Guide.

# APPENDIX A. REFERENCE MATERIAL

## A.1. AMAZON EC2 AMIS FOR RED HAT CLOUD ACCESS PROGRAM

AMIs are a basic RPM install of JBoss EAP + JDK in the Red Hat Enterprise Linux image, with potentially an Amazon EC2 example configuration. Advanced scripting is no longer available, however regular bash scripts can be used.

### AMIs for Platform/JDK Combinations:

- RHEL 6 + Open JDK 8 (1 image)
- RHEL 7 + Open JDK 8 (1 image)

**NOTE**

Both platforms should be of 64-bit architecture.

### Maintenance of AMIs

**`yum update`** should be run regularly, to apply z releases (patches) on EC2. New AMIs for the y releases (major releases) will be provided by Red Hat.

### Scenario 1 (Supported)

1. Sign up for EC2.
2. Sign up for Red Hat Cloud Accces.
3. Select the Red Hat AMI from the list of available AMIs.
4. (Optional) Customize JBoss EAP configuration using user scripts or **`ssh`**.
5. Maintenance: **`yum update`** for z releases, new AMI for the y releases.

## A.2. EXAMPLE CONFIGURATION FILES AND DEPLOYMENTS

There are two packages that add example configuration files and deployments respectively (RHEL 7 AMI version):

```
$ rpm -ql eap7-jboss-ec2-eap
/etc/opt/rh/eap7/jboss-ec2-eap/domain/domain-ec2.xml
/etc/opt/rh/eap7/jboss-ec2-eap/standalone/standalone-ec2-ha.xml
/etc/opt/rh/eap7/jboss-ec2-eap/standalone/standalone-mod_cluster-ec2-
ha.xml
```

```
$ rpm -ql eap7-jboss-ec2-eap-samples
/opt/rh/eap7/root/usr/share/java/eap7-jboss-ec2-eap-samples/cluster-
demo.war
/opt/rh/eap7/root/usr/share/java/eap7-jboss-ec2-eap-samples/hello.war
/opt/rh/eap7/root/usr/share/java/eap7-jboss-ec2-eap-samples/jboss-as-
helloworld-mdb-7.0.0.ER5-redhat-1.war
```

The example configuration files contain a JGroups stack set up for S3_PING protocol that can be used for creating clusters across EC2. The mod_cluster example configuration also configures the **`modcluster`** subsystem to use proxy mod_cluster discovery instead of advertising, since multicast is disabled on EC2.

## A.3. SYSTEM PATHS

### Service Configuration Files:

- RHEL 6: /etc/sysconfig/*
- RHEL 7: /etc/opt/rh/eap7/wildfly/*

### JBoss EAP Home:

- /opt/rh/eap7/root/usr/share/wildfly/

### JBoss EAP Configuration Locations:

#### Standalone instance

- /opt/rh/eap7/root/usr/share/wildfly/standalone/configuration
- /opt/rh/eap7/root/usr/share/wildfly/bin/standalone.conf

#### Managed domain

- /opt/rh/eap7/root/usr/share/wildfly/bin/domain.conf
- /opt/rh/eap7/root/usr/share/wildfly/domain/configuration

## A.4. LAUNCHING JBOSS EAP ON AMAZON EC2 USING A SCRIPT

The following sample script can be used to start JBoss EAP bound to a public IP address when you launch a JBoss EAP instance on Amazon EC2.

```
#!/bin/bash

# platform dependent variables
if [[ "`cat /etc/redhat-release`" = *"release 7"* ]]; then
    SERVICE_CONF_FILE=/etc/opt/rh/eap7/wildfly/eap7-standalone.conf
    START_COMMAND="systemctl start eap7-standalone"
else
    SERVICE_CONF_FILE=/etc/sysconfig/eap7-standalone
    START_COMMAND="service eap7-standalone start"
fi

# set up addresses
INTERNAL_IP_ADDRESS=`ip addr show | grep eth0 -A 2 | head -n 3 | tail -n 1
| awk '{ print $2 }' | sed "s-/24--g" | cut -d'/' -f1`
echo "JAVA_OPTS=\"$JAVA_OPTS -Djboss.bind.address=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.private=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.management=$INTERNAL_IP_ADDRESS\"" >>
/opt/rh/eap7/root/usr/share/wildfly/bin/standalone.conf
```

```
# start EAP
$START_COMMAND
```

## A.5. CONFIGURING JBOSS EAP SUBSYSTEMS TO WORK ON CLOUD PLATFORMS

Some JBoss EAP subsystems must be configured to work properly on cloud platforms, such as Amazon EC2 and Microsoft Azure. This is required because a JBoss EAP server is usually bound to a cloud virtual machine's private IP address, for example: **`10.x.x.x`**, which is only visible from within the cloud platform. For certain subsystems, this address must also mapped to a server's public IP address, which is visible from outside the cloud.

### A.5.1. Web Services

When a client makes a web service request using **`Service.create(wsdlURL, serviceName);`**, the user connects to the server public IP address, but is subsequently redirected to an address defined in the server configuration files in the **`webservices`** subsystem. By default, this address is **`${jboss.bind.address:127.0.0.1}`**, which means that on a cloud platform, the caller will be redirected to the server's private IP address and will be unable to resolve the request. The server's public IP address has to be configured in the **`wsdl-host`** element, using the following command:

```
/subsystem=webservices:write-attribute(name=wsdl-
host,value=PUBLIC_IP_ADDRESS)
```

### A.5.2. Messaging

When using messaging on a cloud platform, the connection factory that the client uses must have a connector pointing to the server's public IP address.

For this reason a new connector and socket binding must be created for JBoss EAP servers running a **`full`** profile.

1. The referenced **`http-public`** socket binding must be created within the **`socket-binding-group`**:

   ```
   /socket-binding-group=standard-sockets/remote-destination-outbound-
   socket-binding=http-
   public:add(host=PUBLIC_IP_ADDRESS,port=${jboss.http.port:8080})
   ```

2. Create the new **`http-connector`** element in the **`messaging`** subsystem:

   ```
   /subsystem=messaging-activemq/server=default/http-connector=http-
   public-connector:add(endpoint=http-acceptor, socket-binding=http-
   public)
   ```

3. Set the **`connectors`** in the **`connection-factory`**, which will be used by clients. For example, configuration of **`RemoteConnectionFactory`** as the default connection will be:

   ```
   /subsystem=messaging-activemq/server=default/connection-
   factory=RemoteConnectionFactory:write-attribute(name=connectors,
   value=["http-public-connector"]
   ```

### A.5.3. Remoting Configuration for High Availability

If you are using JBoss EAP HA features with clustered EJBs on a cloud platform, some extra configuration for the **`remoting`** subsystem is required to ensure EJB clients can receive cluster view updates.

This is done by configuring **`client-mappings`** for the **`remoting`** subsystem socket binding:

```
/socket-binding-group=standard-sockets/socket-binding=http:write-
attribute(name=client-mappings,value=[{ "destination-address" =>
"PUBLIC_IP_ADDRESS", "destination-port" => "8080" }])
```

## A.6. EXAMPLE USER DATA FOR CLUSTERED JBOSS EAP INSTANCES

The following examples show user data configured for several different server configurations.

**Example: File for Standalone Mode on RHEL6/7**

```
#!/usr/bin/env bash

# This is a sample script for the user data field for EC2, which
demonstrates how to launch a standalone instance using the ec2-ha profile
# This file is for RHEL 6/7, standalone mode only
### This script makes use of the following four Bash variables for
clustering setup,
### be sure to add in your own values for these variables here when
copy/pasting this
### script into the EC2 user data field

ACCESS_KEY_ID=<your AWS access key>
SECRET_ACCESS_KEY=<your AWS secret access key>
S3_PING_BUCKET=<your bucket name>
NODE_NAME=<your node name>

#### No further modifications should be needed below to run this example
####
# Set the location of {ProductShortName}
JBOSS_HOME=/opt/rh/eap7/root/usr/share/wildfly

# Set the internal IP address of this EC2 instance which is mapped to a
public address
INTERNAL_IP_ADDRESS=`ip addr show | grep eth0 -A 2 | head -n 3 | tail -n 1
| awk '{ print $2 }' | sed "s-/24--g" | cut -d'/' -f1`

# Set the location of the standalone.conf file and set the command used to
start EAP in standalone mode
if [[ "`cat /etc/redhat-release`" = *"release 7"* ]]; then
    SERVICE_CONF_FILE=/etc/opt/rh/eap7/wildfly/eap7-standalone.conf
    START_COMMAND="systemctl start eap7-standalone"
else
    SERVICE_CONF_FILE=/etc/sysconfig/eap7-standalone
    START_COMMAND="service eap7-standalone start"
fi

# Configure {ProductShortName} to use the ec2-ha profile
```

```
cp /etc/opt/rh/eap7/jboss-ec2-eap/standalone/standalone-ec2-ha.xml
$JBOSS_HOME/standalone/configuration/standalone-ec2-ha.xml
echo "WILDFLY_SERVER_CONFIG=standalone-ec2-ha.xml" >> $SERVICE_CONF_FILE
echo "WILDFLY_BIND=$INTERNAL_IP_ADDRESS" >> $SERVICE_CONF_FILE
echo "JAVA_OPTS=\"\$JAVA_OPTS -
Djboss.jgroups.s3_ping.access_key='$ACCESS_KEY_ID' -
Djboss.jgroups.s3_ping.secret_access_key='$SECRET_ACCESS_KEY' -
Djboss.jgroups.s3_ping.bucket='$S3_PING_BUCKET' -
Djboss.jvmRoute=$NODE_NAME\"" >> $JBOSS_HOME/bin/standalone.conf
echo "JAVA_OPTS=\"\$JAVA_OPTS -Djboss.bind.address=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.private=$INTERNAL_IP_ADDRESS\"" >>
$JBOSS_HOME/bin/standalone.conf

# Deploy the sample application from the local filesystem
cp /opt/rh/eap7/root/usr/share/java/eap7-jboss-ec2-eap-samples/cluster-
demo.war $JBOSS_HOME/standalone/deployments/

# Start {ProductShortName}, note that RHEL 7 does not wait for
{ProductShortName} to start before returning from the service start. In
some cases, there could be a delay of more than 90 seconds.

$START_COMMAND
```

**Example: File for Starting a Clustered Domain Instance (Domain Controller)**

```
#!/usr/bin/env bash

# This is a sample script for the user data field for EC2, which
demonstrates how to launch a domain controller with clustering enabled
# This file is for RHEL 6/7, domain controller, domain mode only
### This script makes use of the following Bash variables for clustering
and domain
### controller discovery setup, be sure to add in your own values for
these variables here
### when copy/pasting this script into the EC2 user data field

ACCESS_KEY_ID=<your AWS access key>
SECRET_ACCESS_KEY=<your AWS secret access key>
S3_PING_BUCKET=<your bucket name>

#### No further modifications should be needed below to run this example
####
# Set the location of {ProductShortName}
JBOSS_HOME=/opt/rh/eap7/root/usr/share/wildfly

# Set the internal IP address of this EC2 instance which is mapped to a
public address
INTERNAL_IP_ADDRESS=`ip addr show | grep eth0 -A 2 | head -n 3 | tail -n 1
| awk '{ print $2 }' | sed "s-/24--g" | cut -d'/' -f1`

# Set the location of the domain.conf file and set the command used to
start EAP in domain mode
if [[ "`cat /etc/redhat-release`" = *"release 7"* ]]; then
    SERVICE_CONF_FILE=/etc/opt/rh/eap7/wildfly/eap7-domain.conf
    START_COMMAND="systemctl start eap7-domain"
else
```

```
    SERVICE_CONF_FILE=/etc/sysconfig/eap7-domain
    START_COMMAND="service eap7-domain start"
fi

# Configure {ProductShortName} to use the domain-ec2.xml and host-
master.xml configuration files
cp /etc/opt/rh/eap7/jboss-ec2-eap/domain/domain-ec2.xml
$JBOSS_HOME/domain/configuration/domain-ec2.xml

echo "WILDFLY_SERVER_CONFIG=domain-ec2.xml" >> $SERVICE_CONF_FILE
echo "WILDFLY_HOST_CONFIG=host-master.xml" >> $SERVICE_CONF_FILE
echo "WILDFLY_BIND=$INTERNAL_IP_ADDRESS" >> $SERVICE_CONF_FILE
echo "JAVA_OPTS=\"\$JAVA_OPTS -
Djboss.jgroups.s3_ping.access_key='$ACCESS_KEY_ID' -
Djboss.jgroups.s3_ping.secret_access_key='$SECRET_ACCESS_KEY' -
Djboss.jgroups.s3_ping.bucket='$S3_PING_BUCKET'\"" >>
$JBOSS_HOME/bin/domain.conf

echo "JAVA_OPTS=\"\$JAVA_OPTS -Djboss.bind.address=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.private=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.management=$INTERNAL_IP_ADDRESS\"" >>
$JBOSS_HOME/bin/domain.conf

echo 'HOST_CONTROLLER_JAVA_OPTS="$HOST_CONTROLLER_JAVA_OPTS $JAVA_OPTS"'
>> $JBOSS_HOME/bin/domain.conf

# Add a management user with the following credentials:
# User name: admin
# Password: secret_Passw0rd
$JBOSS_HOME/bin/add-user.sh -u admin -p secret_Passw0rd -e -g Management

# Update the main-server-group in domain-ec2.xml to use the ec2-ha profile
$JBOSS_HOME/bin/jboss-cli.sh --commands="embed-host-controller --domain-
config=domain-ec2.xml, /server-group=main-server-group:write-
attribute(name=profile, value=ec2-ha)"

# Need to modify permissions since this script is executed as the root
user
chgrp jboss $JBOSS_HOME/domain/configuration/domain_xml_history/
chgrp jboss $JBOSS_HOME/domain/configuration/host_xml_history/
chgrp jboss $JBOSS_HOME/domain/configuration/domain-ec2.xml
chown jboss $JBOSS_HOME/domain/configuration/domain_xml_history/
chown jboss $JBOSS_HOME/domain/configuration/host_xml_history/
chown jboss $JBOSS_HOME/domain/configuration/domain-ec2.xml

# Configure S3 domain controller discovery
yum install patch -y
cd $JBOSS_HOME/domain/configuration
echo "--- host-master.xml 2016-03-18 17:34:26.000000000 -0400
+++ host-master2.xml 2016-04-11 08:28:02.771000191 -0400
@@ -54,7 +54,15 @@
         </management-interfaces>
     </management>
     <domain-controller>
-        <local/>
+<local>
```

```
+    <discovery-options>
+        <discovery-option name=\"s3-discovery\"
module=\"org.jboss.as.host-controller\"
code=\"org.jboss.as.host.controller.discovery.S3Discovery\">
+            <property name=\"access-key\" value=\"$ACCESS_KEY_ID\"/>
+            <property name=\"secret-access-key\"
value=\"$SECRET_ACCESS_KEY\"/>
+            <property name=\"location\" value=\"$S3_PING_BUCKET\"/>
+        </discovery-option>
+    </discovery-options>
+</local>
     </domain-controller>
     <interfaces>
         <interface name=\"management\">
" | patch host-master.xml

cd -

# Start {ProductShortName}, do not forget that RHEL 7 does not wait for
{ProductShortName} to start before returning from the service start. In
some cases, there could be a delay of more than 90 seconds.

$START_COMMAND
sleep 20
# Set up EC2 HA socket bindings for main server group
$JBOSS_HOME/bin/jboss-cli.sh -c --controller=$INTERNAL_IP_ADDRESS:9990 --
timeout=120000 --command='/server-group=main-server-group:write-
attribute(name=socket-binding-group,value=ec2-ha-sockets)'

# Deploy the sample application from the local filesystem to the main-
server-group
$JBOSS_HOME/bin/jboss-cli.sh -c --controller=$INTERNAL_IP_ADDRESS:9990 --
timeout=120000 --command='deploy /opt/rh/eap7/root/usr/share/java/eap7-
jboss-ec2-eap-samples/cluster-demo.war --server-groups=main-server-group'
```

**Example: File for Starting a Clustered Domain Instance (Host Controller)**

```
#!/usr/bin/env bash

# This is a sample script for the user data field for EC2, which
demonstrates how to launch a host controller with clustering enabled
# This file is for RHEL 6/7, host controller, domain mode only
### This script makes use of the following Bash variables for clustering
and domain
### controller discovery setup, be sure to add in your own values for
these variables here
### when copy/pasting this script into the EC2 user data field

ACCESS_KEY_ID=<your AWS access key>
SECRET_ACCESS_KEY=<your AWS secret access key>
S3_PING_BUCKET=<your bucket name>

#### No further modifications should be needed below to run this example
####
# Set the location of EAP
JBOSS_HOME=/opt/rh/eap7/root/usr/share/wildfly
```

```
# Set the internal IP address of this EC2 instance which is mapped to a
public address
INTERNAL_IP_ADDRESS=`ip addr show | grep eth0 -A 2 | head -n 3 | tail -n 1
| awk '{ print $2 }' | sed "s-/24--g" | cut -d'/' -f1`

# Set the location of the domain.conf file and set the command used to
start EAP in domain mode
if [[ "`cat /etc/redhat-release`" = *"release 7"* ]]; then
    SERVICE_CONF_FILE=/etc/opt/rh/eap7/wildfly/eap7-domain.conf
    START_COMMAND="systemctl start eap7-domain"
else
    SERVICE_CONF_FILE=/etc/sysconfig/eap7-domain
    START_COMMAND="service eap7-domain start"
fi

# Configure variables needed by {ProductShortName}
echo "WILDFLY_BIND=$INTERNAL_IP_ADDRESS" >> $SERVICE_CONF_FILE
echo "WILDFLY_HOST_CONFIG=host-slave.xml" >> $SERVICE_CONF_FILE
echo "JAVA_OPTS=\"\$JAVA_OPTS -
Djboss.jgroups.s3_ping.access_key='$ACCESS_KEY_ID' -
Djboss.jgroups.s3_ping.secret_access_key='$SECRET_ACCESS_KEY' -
Djboss.jgroups.s3_ping.bucket='$S3_PING_BUCKET'\"" >>
$JBOSS_HOME/bin/domain.conf
echo "JAVA_OPTS=\"\$JAVA_OPTS -Djboss.bind.address=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.private=$INTERNAL_IP_ADDRESS -
Djboss.bind.address.management=$INTERNAL_IP_ADDRESS\"" >>
$JBOSS_HOME/bin/domain.conf
echo 'HOST_CONTROLLER_JAVA_OPTS="$HOST_CONTROLLER_JAVA_OPTS $JAVA_OPTS"'
>> $JBOSS_HOME/bin/domain.conf

# Configure S3 domain controller discovery
yum install patch -y
cd $JBOSS_HOME/domain/configuration

#sed -i 's/<!--.*-->//g' host.xml # remove nasty '!' signs which break
bash
#sed -i '/^[ ]*$/d' host.xml # remove nasty lines with ' ' whitespaces
which break the patch
#sed -i 's/name="master"/name="admin"/' host.xml # rename host controller
to admin - needed for proper authentication

echo "--- host-slave.xml.orig 2016-06-07 09:55:27.183390617 +0200
+++ host-slave.xml 2016-06-07 09:56:52.540170784 +0200
@@ -57,7 +57,11 @@
     <domain-controller>
         <remote security-realm=\"ManagementRealm\">
             <discovery-options>
-                <static-discovery name=\"primary\"
protocol=\"\${jboss.domain.master.protocol:remote}\"
host=\"\${jboss.domain.master.address}\"
port=\"\${jboss.domain.master.port:9999}\"/>
+                <discovery-option name=\"s3-discovery\"
module=\"org.jboss.as.host-controller\"
code=\"org.jboss.as.host.controller.discovery.S3Discovery\">
+                    <property name=\"access-key\"
```

```
value=\"$ACCESS_KEY_ID\"/>
+                     <property name=\"secret-access-key\"
value=\"$SECRET_ACCESS_KEY\"/>
+                     <property name=\"location\"
value=\"$S3_PING_BUCKET\"/>
+                 </discovery-option>
             </discovery-options>
         </remote>
     </domain-controller>
" | patch host-slave.xml

sed -i 's/<!--.*-->//g' host-slave.xml # remove nasty '!' signs which
break bash
sed -i '/^[ ]*$/d' host-slave.xml # remove nasty lines with ' '
whitespaces which break the patch

echo "--- host-slave.xml.orig 2016-06-07 10:21:03.111440684 +0200
+++ host-slave.xml 2016-06-07 10:20:20.774045727 +0200
@@ -7,7 +7,7 @@
         <security-realms>
             <security-realm name=\"ManagementRealm\">
                 <server-identities>
-                     <secret value=\"c2xhdmVfdXNlcl9wYXNzd29yZA==\"/>
+                     <secret value=\"c2VjcmV0X1Bhc3N3MHJk\" />
                 </server-identities>
                 <authentication>
                     <local default-user=\"\$local\" skip-group-
loading=\"true\"/>
" | patch host-slave.xml

sed -i 's/<host xmlns="urn:jboss:domain:4.1">/<host
xmlns="urn:jboss:domain:4.1" name="admin">/' host-slave.xml
sed -i 's/other-server-group/main-server-group/' host-slave.xml

cd -

# Start {ProductShortName}, do not forget that RHEL 7 does not wait for
{ProductShortName} to start before returning from the service start. In
some cases, there could be a delay of more than 90 seconds.
$START_COMMAND
```

*Revised on 2018-02-08 10:16:09 EST*