JBoss Enterprise Application Platform 5

Getting Started on Amazon EC2

for JBoss Enterprise Application Platform 5 and JBoss Enterprise Web Server 5

Edition 5.2.0

Last Updated: 2017-10-13

JBoss Enterprise Application Platform 5 Getting Started on Amazon EC2

for JBoss Enterprise Application Platform 5 and JBoss Enterprise Web Server 5
Edition 5.2.0

Aleksandar Kostadinov

John Doyle

Edited by

Eva Kopalova

Petr Penicka

Russell Dickenson

Scott Mumford

Abstract

This Getting Started Guide documents information needed to get JBoss Enterprise Application Platform 5 and JBoss Enterprise Web Server 5 running on Amazon's Elastic Computing (EC2) platform.

Table of Contents

CHAPTER 1. INTRODUCTION

JBoss Cloud Access is a Red Hat subscription feature which allows you to easily move your JBoss instances between traditional on-premise servers and JBoss-certified cloud servers. You have the freedom to choose the best computing resources according to your needs, while keeping your existing business and support models.

This industry-leading feature of Red Hat's JBoss subscriptions makes usage of public cloud resources simpler, more cost-effective, and more reliable than ever before.

Migrating to a cloud-based infrastructure with JBoss **Cloud Access** is a simple, gradual process that uses the infrastructure investments you have already made.

Cloud Access allows you to expand your infrastructure as required or move to an external cloud when your criteria are met. This enables you to evolve your operational processes at your own pace.

Cloud computing is one of the most fundamental shifts in Information Technology to happen in decades. Existing business models must evolve to take advantage of new cloud business and operational opportunities. To this end, Red Hat **Cloud Access** offers enterprise-level software, competitive subscriptions, and comprehensive support, built into business and operational models that were designed specifically for the cloud.

CHAPTER 2. OVERVIEW

2.1. WHAT IS PROVIDED?

Membership in the JBoss **Cloud Access** program enables you to access private Amazon Machine Images (AMIs) created by Red Hat. These AMIs have your choice of **JBoss Enterprise Application Platform 5.1.x** or **JBoss Enterprise Web Server 1.0.x** pre-installed and are fully supported by Red Hat.

For ongoing management and monitoring of the platform, a **JBoss Operations Network** (JON) 3.0.x agent is pre-installed. Product updates are managed using RPMs, enabling you to update via **Red Hat Update Infrastructure**.

Each of the Red Hat AMIs are only a starting point, requiring further configuration to the requirements of your application. This guide assumes knowledge of how to configure the JBoss products, Red Hat Enterprise Linux and Amazon EC2. Refer to the Red Hat Documentation library and the Amazon *EC2 Getting Started Guide*.

2.2. DEPLOYMENT PROCESS

Follow this procedure to deploy your application:

Procedure 2.1. Deployment

1. Identify a suitable Red Hat AMI.
2. Configure instance and deploy application.
3. Confirm that the platform is working as expected.
4. Establish monitoring with JBoss Operations Network (JON).

This guide is structured to match this process, with the aim of getting your application deployed quickly and easily. It is not possible to document every deployment scenario so this guide provides example configurations, which should be modified to suit your environment.

CHAPTER 3. IDENTIFY A SUITABLE RED HAT AMI

3.1. SUPPORTED INSTANCE TYPES

JBoss Cloud Access supports a subset of the instance types available on Amazon EC2, full details of which can be found in Amazon's User Guide for Amazon Elastic Compute Cloud . Choose from the following instance types according to your requirements:

Supported Instance Types

Standard Instance

Have memory-to-CPU ratios suitable for most general-purpose applications.

High Memory Instance

Have proportionally more memory resources and are well suited for high throughput applications, such as database and memory caching applications.

High CPU Instance

Have proportionally more CPU resources than memory (RAM) and are well suited for compute-intensive applications.

The instance type "Micro (`t1.micro`)" is **not** suitable for deployment of JBoss Enterprise Application Platform.

3.2. FINDING THE RED HAT AMIS

Red Hat's AMIs are identified by their AMI Name which is formatted as: *OperatingSystem-OperatingSystemVersion-JBossProduct-JBossProductVersion-Architecture-CreationDate*.

OperatingSystem

RHEL (Red Hat Enterprise Linux)

OperatingSystemVersion

6.2

JBossProduct

JBEAP (JBoss Enterprise Application Platform) or JBEWS (JBoss Enterprise Web Server)

JBossProductVersion

5.2.0

Architecture

x86_64 or i386

When searching for the required AMI, you can search with a general criterion ("*JBEAP*", for example), or a specific one ("*RHEL-6.2-JBEAP-5.1.2-x86_64*").

Continue to the appropriate deployment chapter, depending on your desired configuration.

CHAPTER 4. LAUNCHING A STANDALONE JBOSS ENTERPRISE WEB SERVER INSTANCE

In this example a sample web application is deployed on Apache Tomcat 6 server. An Apache HTTPD server is used as a reverse proxy.

4.1. PREREQUISITES

- Suitable Red Hat AMI identified in Chapter 3, *Identify a suitable Red Hat AMI*;
- A pre-configured Security Group which allows incoming requests on ports **22** and **80**.

4.2. LAUNCHER INSTANCE

Launch the Red Hat AMI identified in Chapter 3, *Identify a suitable Red Hat AMI*, adding the following lines to the "**User Data:**" field.

```
      # deploy the hello.war sample application to the Apache Tomcat 6
server which is part of the JBoss Enterprise Web Server offering
      cp /usr/share/java/jboss-ec2-ews-samples/hello-1.0.war
/usr/share/tomcat6/webapps/hello.war

      # Configure HTTPd to act as a proxy to the Apache Tomcat 6 server
      cat >> /etc/httpd/conf.d/proxy_ajp.conf <<"EOF"
      <Location /hello>
      ProxyPass         http://localhost:8080/hello
      ProxyPassReverse  http://localhost:8080/hello
      </Location>
      ProxyPreserveHost On
      EOF

      # Start the Apache Tomcat 6 server
      service tomcat6 start

      # Instruct SELinux to allow HTTPd to serve as a proxy relay
      setsebool -P httpd_can_network_relay 1

      # Start the HTTPd server
      service httpd start
```

4.3. TEST JBOSS ENTERPRISE WEB SERVER

Procedure 4.1. Test JBoss EWS

1. In the instance's details pane, note the instance's "**Public DNS**".
2. In a browser, navigate to **http://*<public-DNS>*/hello**
3. Confirm that the text "**Hello World!**" appears, otherwise refer to Appendix B, *Troubleshooting*.

IMPORTANT

In JBoss Enterprise Web Server deployed on EC2, the contents of the **`User Data`** field are treated as a script.

If the first line is "**`#!`**" then it is run as any executable file, otherwise its contents are executed/sourced into the Red Hat Enterprise Linux **init** shell.

If this is a production instance, add the following text as the first line in the **`User Data`** field to ensure that security updates are applied on boot:

```
yum -y update
```

CHAPTER 5. LAUNCHING A STANDALONE JBOSS ENTERPRISE APPLICATION PLATFORM INSTANCE

In this example a sample web application is deployed to a *standalone* JBoss Enterprise Application Platform instance.

For an example using clustering and Amazon's Relational Database Service (RDS), refer to Chapter 6, *Launching clustered JBoss Enterprise Application Platform instances*.

5.1. PREREQUISITES

- Suitable Red Hat AMI identified in Chapter 3, *Identify a suitable Red Hat AMI*;
- A pre-configured Security Group which allows incoming requests on at least ports **`22`** and **`8080`**.

5.2. LAUNCH INSTANCE

Launch the Red Hat AMI identified in Chapter 3, *Identify a suitable Red Hat AMI*, adding the following lines to the "**`User Data:`**" field, inserting your JBoss admin password instead of the placeholder.

```
JBOSSAS_ADMIN_PASSWORD=<your password for opening admin console>
JBOSS_IP=0.0.0.0 #listen on all IPs and interfaces
cat> $USER_SCRIPT << "EOF"
## Deploy your application from an Internet URL
# wget https://<your_secure_storage_hostname>/<path>/<app_name>.war -O
/var/lib/jbossas/server/$JBOSSCONF/deploy/

## deploy sample application from the local filesystem
cp /usr/share/java/jboss-ec2-eap-samples/hello-1.0.war
/var/lib/jbossas/server/$JBOSSCONF/deploy/hello.war
EOF
```

This example below deploys a sample application already on the Red Hat AMI. To deploy your own application instead, comment the line which deploys the sample application and uncomment the line which deploys your application, modifying the source URL as required.

If this is to be a production instance, add the following command under **`USER_SCRIPT`** in the "**`User Data:`**" field, to ensure that security updates are applied on boot:

Also ensure that **`yum -y update`** is run regularly to apply security fixes and enhancements.

```
yum -y update
```

IMPORTANT

There are a number of parameters which can be used in the User Data field to customize the configuration and deployment of JBoss Enterprise Application Platform. Refer to Appendix A, *User Script Parameters* for details.

5.3. TEST JBOSS ENTERPRISE APPLICATION SERVER

If the sample application has been deployed, as per the example, continue testing according to the steps below. However if your own application has been deployed then follow your own testing methodology.

1. In the instance details pane, note the instance's "**`Public DNS`**".
2. In a browser, navigate to **`http://`*`<public-DNS>`*`:8080/hello`**.
3. Confirm that the text "**`Hello World!`**" appears, otherwise refer to the Troubleshooting section below.
4. In a browser, navigate to **`http://`*`<public-DNS>`*`:8080`**.
5. Confirm that the JBoss Enterprise Application Platform home page appears, including a hyperlink to the Admin Console.
6. Click on the "**`Admin Console`**" hyperlink and log in with the username "**` admin`**" and the password you entered in the "**`User Data`**" field.
7. Logout of the JBoss Enterprise Application Platform Admin Console.

CHAPTER 6. LAUNCHING CLUSTERED JBOSS ENTERPRISE APPLICATION PLATFORM INSTANCES

In this example clustered JBoss Enterprise Application Platform instances are deployed with a simple JMS sender and receiver application.

To support the application, a JBoss Enterprise Web Server instance is deployed as a proxy and a Relational Database Service instance is deployed to provide the required MySQL DBMS.

6.1. LAUNCH CLUSTERED ENTERPRISE APPLICATION PLATFORM

Procedure 6.1. Launch clustered EAP

1. Create a Relational Database Service instance.
2. Create a Virtual Private Cloud (VPC).
3. Launch a JBoss Enterprise Web Server instance to serve as a mod_cluster proxy and a NAT instance for the VPC.
4. Configure VPC routing tables for the JBoss Enterprise Application Platform (EAP) nodes.
5. Launch JBoss Enterprise Application Platform nodes.
6. Verify the JBoss Enterprise Application Platform cluster.

Detailed instructions on each of these steps are contained in the following sections.

6.2. CREATE A RELATIONAL DATABASE SERVICE DATABASE INSTANCE

Relational Database Service (RDS) is a convenient service provided by Amazon to easily deploy and manage an RDBMS with failover, back-up and restore capabilities. Currently several versions of Oracle and MySQL database servers are available. This example uses the MySQL database server but JBoss Enterprise Application Platform also supports Oracle Database.

1. Go to the RDS tab in the AWS console.
2. Subscribe to the service if needed.
3. Click on "**`Launch DB instance`**".
4. Click on **`MySQL`**
   a. Select a version (for example **`5.5.12`**).
   b. Select **`small instance`**.
   c. Ensure ***`Multi-AZ Deployment`*** and ***`Auto upgrade`*** are **`off`**.
   d. Set ***`Storage`*** to 5GB.
   e. Choose the database administrator's username and password.

f. On the next screen choose a database name to be created with the instance.

g. On the next screen you can disable back-up and maintenance.

h. Confirm your settings.

The database should initialize and be ready for use after a few minutes.

WARNING

The above settings are sufficient for this example scenario but for a production environment it is highly recommended that you consider the failover and back-up features.

It is good practice to create separate user/password pairs for each application accessing the database. Tune other configuration options according to your application's requirements.

6.3. CREATE A VIRTUAL PRIVATE CLOUD (VPC)

1. Go to the VPC tab in the AWS console.
2. Subscribe to the service if needed.
3. Click on "**`Create new VPC`**".
4. Choose a VPC with one public and one private subnet.
   a. Set the public subnet to be **`10.0.0.0/24`**.
   b. Set the private subnet to be **`10.0.1.0/24`**.
5. Go to **`Elastic IPs`**.
6. Create an elastic IP for use by the EWS mod_cluster proxy/NAT instance.
7. Go to **`Security groups`** and create a security group to allow all traffic in and out.
8. Go to Network ACLs
   a. Create an ACL to allow all traffic in and out.
   b. Create an ACL to allow all traffic out and traffic in on only TCP ports **`22`**, **`8009`**, **`8080`**, **`8443`** and **`16163`**.

NOTE

VPC is recommended for a JBoss Enterprise Application Platform cluster setup as it greatly simplifies secure communication between cluster nodes, a JON Server and the mod_cluster proxy. Without a VPC all these communication channels need to be encrypted and authenticated. Please refer to JBoss Enterprise Application Platform for detailed instructions on configuring SSL.

Also note that we are using a database external to the VPC in this example. Your security policies may require connection to the database to be encrypted. Please refer to Amazon's *RDS FAQ* for details about encrypting the database connections.

6.4. LAUNCH A JBOSS ENTERPRISE WEB SERVER INSTANCE TO SERVE AS A MOD_CLUSTER PROXY AND A NAT INSTANCE FOR THE VPC

1. Create an elastic IP for this instance.
2. Select an AMI.
3. Go to `Security Group` and allow all traffic (use Red Hat Enterprise Linux's built-in firewall capabilities to restrict access if desired).
4. Choose "`running`" in the public subnet of the VPC.
5. Choose a static IP (e.g. `10.0.0.4`).
6. Put the following in the `User Data:` field:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
echo 0 > /proc/sys/net/ipv4/conf/all/rp_filter
echo 0 > /proc/sys/net/ipv4/conf/eth0/rp_filter

iptables -I INPUT 4 -s 10.0.1.0/24 -p tcp --dport 7654 -j ACCEPT
iptables -I INPUT 4 -p tcp --dport 80 -j ACCEPT

iptables -I FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -I FORWARD -s 10.0.1.0/24 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth0 ! -s 10.0.0.4 -j MASQUERADE

# balancer module incompatible with mod_cluster
sed -i -e 's/LoadModule proxy_balancer_module/#\0/'
/etc/httpd/conf/httpd.conf

cat > /etc/httpd/conf.d/mod_cluster.conf << "EOF"
#LoadModule proxy_module modules/mod_proxy.so
#LoadModule proxy_ajp_module modules/mod_proxy_ajp.so
LoadModule slotmem_module modules/mod_slotmem.so
LoadModule manager_module modules/mod_manager.so
LoadModule proxy_cluster_module modules/mod_proxy_cluster.so
LoadModule advertise_module modules/mod_advertise.so

Listen 7654

# workaround JBPAPP-4557
```

```
MemManagerFile /var/cache/mod_proxy/manager

<VirtualHost *:7654>
   <Location /mod_cluster-manager>
      SetHandler mod_cluster-manager
      Order deny,allow
      Deny from all
      Allow from 127.0.0.1
   </Location>

   <Location />
      Order deny,allow
      Deny from all
      Allow from 10.
      Allow from 127.0.0.1
   </Location>

   KeepAliveTimeout 60
   MaxKeepAliveRequests 0
   ManagerBalancerName mycluster
   ServerAdvertise Off
   EnableMCPMReceive On
</VirtualHost>
EOF

echo "`hostname | sed -e 's/ip-//' -e 'y/-/./'`        `hostname`"
>> /etc/hosts

semanage port -a -t http_port_t -p tcp 7654 #add port in the apache
port list for the below to work
setsebool -P httpd_can_network_relay 1 #for mod_proxy_cluster to
work
chcon -t httpd_config_t -u system_u
/etc/httpd/conf.d/mod_cluster.conf
service httpd start
```

7. Disable the Amazon EC2 cloud source/destination checking for this instance so it can act as a router.

   1. Right-click on the running EWS instance and choose "**`Change Source/Dest check`**".

   2. Click on **`Yes, Disable`**.

8. Assign the elastic IP to this instance.

6.5. STARTING INSTANCES IN A CLUSTER

6.5.1. EC2 clustering profiles

The JBoss Enterprise Application Platform AMIs provided by Red Hat feature two additional server profiles called cluster_ec2 and mod_cluster-ec2.

They both feature the ability to form a cluster inside the EC2 environment where multicast is not available. This is done by configuring **JGroups** to use only TCP unicast for cluster communication and ***`S3_PING`*** as the discovery protocol.

Additionally mod_cluster-ec2 is pre-configured to easily register with mod_cluster proxies.

The following sections describe the additional setup required for ***`S3_PING`***, clustering and **mod_cluster**.

6.5.2. VPC private subnet default route

Since JBoss Enterprise Application Platform cluster nodes will be run in the private subnet of the VPC, but cluster nodes require Internet access for S3 connectivity, a default route needs to be set to go through the NAT instance.

1. Navigate EWS instance in the Amazon AWS console.
2. Go to **VPC** → **route tables** and click on the routing table used by the private subnet.
3. In the field for a new route enter **`0.0.0.0/0`**.
4. Click on "**`select a target`**" and choose "**`Enter Instance ID`**".
5. Choose the ID of the running EWS instance.

6.5.3. IAM setup

The ***`S3_PING`*** protocol, as suggested by its name, uses an S3 bucket to discover other cluster members.

The **JGroups** versions 2.6.x require Amazon AWS account access and secret keys to authenticate against the S3 service. It is a security risk to enter your main account credentials in the user-data field, store them online or in an AMI.

To circumvent this, a separate account can be created using the Amazon IAM feature which would be only granted access to a single S3 bucket:

1. Go to the IAM tab in the AWS console.
2. Click on users and then "**`Create New Users`**".
3. Choose a name (*jbosscluster*, for example) and ensure the "**`Generate an access key for each User`**" option is checked.
4. Click "**`Download credentials`**" and save them in a secure location.
5. Close the window and click on the newly created user.
6. In the summary tab you will see **User ARN** (*arn:aws:iam::05555555555:user/jbosscluster**, for example). Make a note of this because it's required to set up the S3 bucket.

6.5.4. S3 bucket setup

1. Open the S3 tab in the AWS console.
2. Click on "**`Create Bucket`**".
3. Choose a name (*clusterbucket123* for example) and click on **`Create`**.

Note that bucket names are unique across the entire S3 so you will not be able to use the chosen name again.

4. Right click over the new bucket and choose **`Properties`**.

5. In the permissions tab click on "**`Add bucket policy`**".

6. If you click on "**`New policy`**" the policy creation wizard will open.

7. For ease of completion the following can be pasted into the policy, but be sure to replace **`arn:aws:iam::05555555555:user/jbosscluster*`** with the value noted in the previous procedure:

```
{
    "Version": "2008-10-17",
    "Id": "Policy1312228794320",
    "Statement": [
        {
            "Sid": "Stmt1312228781799",
            "Effect": "Allow",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::055555555555:user/jbosscluster"
                ]
            },
            "Action": [
                "s3:ListBucketVersions",
                "s3:GetObjectVersion",
                "s3:ListBucket",
                "s3:PutBucketVersioning",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion",
                "s3:GetObject",
                "s3:ListBucketMultipartUploads",
                "s3:ListMultipartUploadParts",
                "s3:PutObject",
                "s3:GetBucketVersioning"
            ],
            "Resource": [
                "arn:aws:s3:::clusterbucket123/*",
                "arn:aws:s3:::clusterbucket123"
            ]
        }
    ]
}
```

6.5.5. Launching the JBoss Enterprise Application Platform AMIs

All necessary preparation for running a cluster are now complete. The final task is to put the right configuration in the **`User Data`** field.

1. Select an AMI.

2. Choose desired number of instances (this will be the cluster size).

3. Choose VPC and instance type.

4. Go to Security Group and allow all traffic from the JBoss Enterprise Application Platform cluster subnet (set other restrictions as desired).

5. Put the following into the `User Data` field:

```
MOD_CLUSTER_PROXY_LIST=10.0.0.4:7654

## clustering setup
JBOSS_JGROUPS_S3_PING_SECRET_ACCESS_KEY=<your secret key>
JBOSS_JGROUPS_S3_PING_ACCESS_KEY=<your access key>
JBOSS_JGROUPS_S3_PING_BUCKET=clusterbucket123

JBOSS_CLUSTER_ID=S3

## database credentials configuration
JAVA_OPTS="$JAVA_OPTS -Ddb.host=instancename.something.rds.amazonaws.com -
Ddb.database=mydatabase -Ddb.user=<user> -Ddb.passwd=<pass>"

PORTS_ALLOWED="1024:65535"
JBOSSAS_ADMIN_PASSWORD=<your password for opening admin console>
JBOSS_IP=`hostname` #listen on public/private EC2 IP address

cat> $USER_SCRIPT << "EOF"
cp /usr/share/java/jboss-ec2-eap-samples/jmssender-1.0.war
$JBOSS_DEPLOY_DIR/jmssender.war
cp /usr/share/java/jboss-ec2-eap-samples/mdbtest-1.1.jar
$JBOSS_DEPLOY_DIR/
yum -y install mysql-connector-java
yum -y install mysql-connector-java
cp -v /usr/share/java/mysql-connector-java-*.jar
/var/lib/jbossas/server/$JBOSSCONF/lib/
## DefaultDS configuration
cd /tmp
rm -f /var/lib/jbossas/server/$JBOSSCONF/deploy/hsqldb-ds.xml
/var/lib/jbossas/server/$JBOSSCONF/deploy/messaging/hsqldb-persistence-
service.xml
sed -e 's#\("Clustered">\)false\(</attribute>\)#\1true\2#' -e 's#\
("FailoverOnNodeLeave">\)false\(</attribute>\)#\1true\2#'
/usr/share/doc/jbossas-<version>/examples/jms/mysql-persistence-
service.xml > /var/lib/jbossas/server/$JBOSSCONF/deploy/messaging/mysql-
persistence-service.xml
sed -i -e 's#<fk-constraint>false</fk-constraint>#<fk-constraint>true</fk-
constraint>#' /var/lib/jbossas/server/$JBOSSCONF/conf/standardjbosscmp-
jdbc.xml
cat > /var/lib/jbossas/server/$JBOSSCONF/deploy/mysql-ds.xml << "EODS"
<?xml version="1.0" encoding="UTF-8"?>
<datasources>
   <local-tx-datasource>
      <jndi-name>DefaultDS</jndi-name>
      <connection-url>jdbc:mysql://${db.host}:3306/${db.database}
</connection-url>
      <driver-class>com.mysql.jdbc.Driver</driver-class>
      <user-name>${db.user}</user-name>
      <password>${db.passwd}</password>
      <metadata>
         <type-mapping>mySQL</type-mapping>
      </metadata>
      <transaction-isolation>TRANSACTION_READ_COMMITTED</transaction-
isolation>
   </local-tx-datasource>
</datasources>
EODS

cat > /var/lib/jbossas/server/$JBOSSCONF/deploy/sample-destinations-
service.xml << "EODS"
<server>
  <mbean code="org.jboss.jms.server.destination.TopicService"

name="jboss.messaging.destination:service=Topic,name=jms/SampleTopic"
      xmbean-dd="xmdesc/Topic-xmbean.xml">
      <depends optional-attribute-
name="ServerPeer">jboss.messaging:service=ServerPeer</depends>
      <depends>jboss.messaging:service=PostOffice</depends>
      <attribute name="JNDIName">SampleTopic</attribute>
      <attribute name="Clustered">true</attribute>
   </mbean>
</server>
EODS

## this will workaround the problem that in a VPC, instance hostname is
not resolvable
grep -q 10.0.1.15 /etc/hosts || \
  for (( i=1 ; i<255 ; i++ )); do
     echo -e "10.0.1.$i\tip-10-0-1-$i" ;
  done >> /etc/hosts

EOF
```

Running JBoss Enterprise Application Platform 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.

Please refer to the ***CLUSTER_ID*** variable in Appendix A, *User Script Parameters* for information on how to make such a configuration work reliably.

The auto-scaling Amazon EC2 feature can be used with JBoss Enterprise Application Platform cluster nodes. However make sure you test **before** deployment. You should ensure that your particular workloads scale to the desired number of nodes and that the performance meets your needs with the 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 can affect performance of a cluster. 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) will need time to fail over. This is likely to impact your application users' experience.

It is recommended that you scale down your application cluster manually by disabling the server from the mod_cluster management interface until processed sessions are completed or shutting down the JBoss Enterprise Application Platform instance gracefully (SSH access to the instance or JON can be used).

Test that your chosen 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.

6.6. VERIFYING EVERYTHING IS CONNECTED AND RUNNING

1. Open **`http://`*`<elastic IP of EWS>`*** in browser.
2. Open **`http://`*`<elastic IP of EWS>`*`/jmssender`** in browser.
   - Check that all cluster nodes are logging a message per every **jmssender** request (no matter which node **jmssender** is opened from).
3. Connect to the JBoss Enterprise Web Server instance:

   ```
   ssh -L7654:localhost:7654 <elastic IP of EWS>
   ```
4. Open http://localhost:7654/mod_cluster-manager in a browser.

If these are to be production instances, add the following text to the contents of the User Data field, to ensure that security updates are applied on boot:

```
yum -y update
```

CHAPTER 7. ESTABLISH MONITORING WITH JBOSS OPERATIONS NETWORK (JON)

With your business application deployed to a correctly-configured AMI instance, the next step is to establish monitoring of the platform with JBoss Operations Network (JON).

The JON server is commonly located inside a corporate network, so it's necessary to establish a secure connection between the server and each of its agents. Establishing a VPN between the two points is the most common solution but this complicates the required networking configuration.

This chapter provides network configuration guidelines for enabling communication between the JON agent and JON server. For more extensive information on JBoss Operations Network's configuration, management and usage refer to the Red Hat documentation.

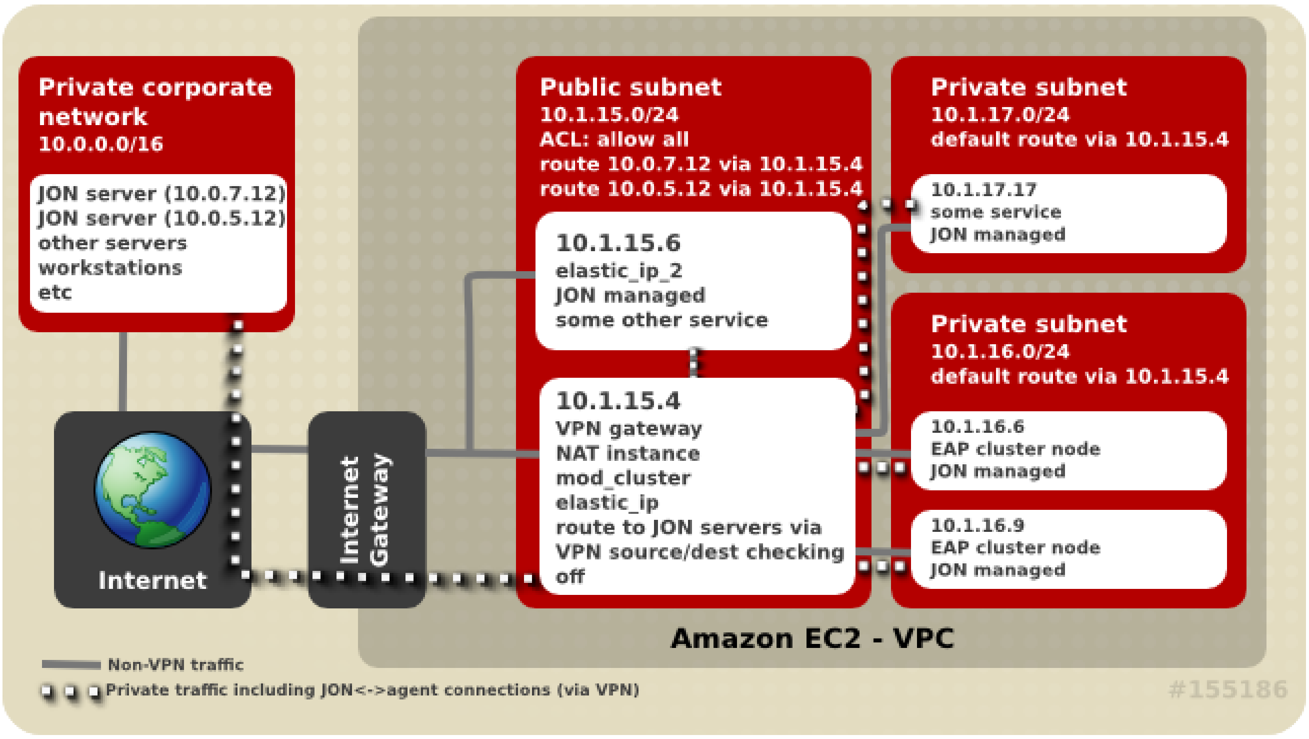

Figure 7.1. Network connectivity between JON Server and its agents

7.1. CONNECTIVITY REQUIREMENTS

Registering a JON agent with its servers requires **two-way** communication between agent and servers. The JON Agent needs access to port **7080** (or **7443** in case SSL is used) on **all** JON servers, and each JON server must be able to access each of the connected agents on a **unique** IP:TCP port pair (agent port is usually **16163**).

If there are multiple, clustered JON servers, make sure each agent can communicate with **all** servers in the JON cluster via the IP/hostname pairs as configured through the JON server administration console. The JON server used by the agent to register may not be the server it tries to use after initialization.

7.2. NETWORK ADDRESS TRANSLATION (NAT)

A corporate VPN gateway acting in routed mode greatly simplifies network configuration. However, if the corporate VPN gateway is acting in NAT mode, the JON server does not have direct visibility of agents. Port forwarding needs to be configured so that, for **each** agent, one port on the gateway is forwarded to the JON agent's address or port on the managed machine. The JON agent also needs to be configured to tell the server the forwarded port number and IP address (see *rhq.communications.connector.** description in agent-configuration.xml for more information).

7.3. DNS

JON servers and JON agents need to be able to resolve each others' hostnames but DNS resolution is complicated in a VPN configuration. Connected servers can use the Amazon EC2 DNS servers, the corporate network's DNS servers or use a split DNS configuration where the corporate DNS servers are used for resolving names in particular domains and the Amazon EC2 DNS servers are used for resolving all other names.

7.4. ROUTING IN EC2

All EC2 servers have, by default, a "source/destination checking" routing feature activated. This feature drops any packets to the server which have a destination different from the machine's IP address. If the VPN solution selected for connecting agents to the JON Server includes a router, this feature needs to be turned off for the server(s) acting as routers/VPN gateways. This configuration setting can be accessed via the Amazon AWS console by right-clicking on the instance. Disabled source/destination checking is also required in a Virtual Private Cloud (VPC).

Some VPN configurations, by default, route traffic intended for the Internet through the corporate VPN. Avoid this for EC2 instances because it's generally much slower and less efficient.

While the use of a proper addressing schema is not a concern specific to JON, poor schemas can affect it. Amazon EC2 assigns IP addresses from the `10.0.0.0/8` network. Instances usually have a public IP address also but only network traffic on the internal IP address within the same availability zone is free. To avoid using the `10.0.0.0/8` network in private addressing, there are a few things to consider:

- When creating a VPC, avoid allocating addresses already in use in the private network to avoid connectivity problems;
- If an instance needs access to availability zone local resources, make sure EC2 private addresses are used and traffic is **not** routed through the VPN;
- If an EC2 instance will access a small subset of corporate private network addresses (for example only JON servers), only these addresses should be routed through the VPN for increased security and a lower chance of EC2/private network address space collisions.

7.5. TERMINATING AND RESTARTING INSTANCES

In a cloud environment it is very easy to terminate a machine instance and, if required, launch a new instance identical to the initial one.

There is, however, a potential problem if a new instance tries to register with JON servers using the same agent name as a previously running agent. If this happens the JON server will not allow an agent to reconnect with a missing or non-matching identification token.

TO avoid this, ensure that terminated agents are removed from the JON inventory before trying to connect an agent with the same name or specify the correct identification token when starting new agent.

Another problem is when an agent machine is assigned a new VPN IP address (i.e. machine is restarted or VPN connection is terminated). Refer to the *Configuring JON Servers and Agents Guide*document (available at http://docs.redhat.com/docs/en-US/JBoss_Operations_Network/index.html) for instructions on how to change the agent's IP address.

If this does happen it is best to bind the JON agent's life cycle to the VPN connection's life cycle. When the connection drops, stop the agent. When the connection is up again, update ***JON_AGENT_ADDR*** in **/etc/sysconfig/jon-agent-ec2** to reflect the new IP address and restart the agent.

IMPORTANT

If there is a high number of instances launched and/or terminated it can become impractical to add and remove them manually from the JON inventory. JON's scripting capabilities can be used for automate these steps. Refer to the JON documentation for further information.

7.6. CONFIGURE EAP AND EWS INSTANCES TO REGISTER WITH JON

For JBoss Enterprise Application Platform, add this to the User Data field:

```
JON_SERVER_ADDR=jon2.it.example.com
## if instance not already configured to resolve its hostname
JON_AGENT_ADDR=`ip addr show dev eth0 primary to 0/0 | sed -n 's#.*inet \
([0-9.]\+\)/.*#\1#p'`
PORTS_ALLOWED=16163
# insert other JON options when necessary, see Appendix I
```

For JBoss Enterprise Web Server add this to the User Data field:

```
cat > /etc/sysconfig/jon-agent-ec2 << "EOF"
   iptables -D INPUT -p tcp --dport 16163 -j ACCEPT
   iptables -I INPUT -p tcp --dport 16163 -j ACCEPT
   JON_SERVER_ADDR=jon2.it.example.com
   ## if instance not already configured to resolve its hostname
   JON_AGENT_ADDR=`ip addr show dev eth0 primary to 0/0 | sed -n 's#.*inet
\([0-9.]\+\)/.*#\1#p'`
   # insert other JON options when necessary, see Appendix I
EOF

## start the agent
service jon-agent-ec2 start
```

APPENDIX A. USER SCRIPT PARAMETERS

Deploying and configuring JBoss Enterprise Application Platform using the User Data field provides for flexibility and ease of management of multiple configurations. The flexibility comes from the use of various parameters which can be used to configure the platform or in the deployment of your custom application.

A.1. PERMANENT CONFIGURATION PARAMETERS

The following parameters can be used to influence the configuration and operation of JBoss Enterprise Application Platform. Their contents are written to /etc/sysconfig/jbossas and /etc/sysconfig/jon-agent-ec2.

Name	Description	Default
JBOSS_JGROUPS_S3_PING_ACCESS_KEY	Amazon AWS user account access key for S3_PING discovery if clustering is used.	N/A
JBOSS_JGROUPS_S3_PING_SECRET_ACCESS_KEY	Amazon AWS user account secret access key.	N/A
JBOSS_JGROUPS_S3_PING_BUCKET	Amazon S3 bucket to be used for S3_PING discovery.	N/A
JBOSS_CLUSTER_ID	ID of cluster member nodes. Only used for clustering. Accepted values are (in order): • a valid cluster ID number in the range 0-1023 • a network interface name who's IP last octet is used as the value • "S3" as a value would coordinate ID usage through the S3 bucket used for jgroups' S3_PING It's recommended to use the last octet of IP (i.e. the default) when all cluster nodes are located in the same 24 or more bits subnet (for example in a VPC subnet).	Last octet of eth0's IP address.
MOD_CLUSTER_PROXY_LIST	Comma-delimited list of IPs/hostnames of mod_cluster proxies if mod_cluster is to be used.	N/A

Name	Description	Default
PORTS_ALLOWED	List of incoming ports to be allowed by firewall in addition to the default ones.	N/A
JBOSSAS_ADMIN_PASSWORD	JBoss EAP password for "admin" user.	N/A
JON_SERVER_ADDR	JON server hostname or IP with which to register. This is only used for registration, after that agent may communicate with other servers in the JON cluster.	N/A
JON_SERVER_PORT	Port used by the agent to communicate with the server.	7080
JON_AGENT_NAME	Name of JON agent, must be unique.	Instance's ID
JON_AGENT_PORT	Port that the agent listens on.	16163
JON_AGENT_ADDR	IP address to which the JON agent is to be bound. This is used when the server has more than one public address, e.g. VPN.	JON agent chooses the IP of local hostname by default.
JON_AGENT_OPTS	Additional JON agent system properties which can be used for configuring SSL, NAT and other advanced settings.	N/A
JBOSSCONF	Name of JBoss EAP profile to start. If S3 config is present, then cluster-ec2 profile is used. If MOD_CLUSTER_PROXY_LIST is specified, the mod_cluster-ec2 profile is selected me. If neither of these options are used then the 'default' profile is used.	**`default`**, **`cluster-ec2`** or **`mod_cluster-ec2`** depending on the other parameters.
JAVA_OPTS	Custom values to be added to the variable before JBoss Enterprise Application Platform starts.	JAVA_OPTS is built from values of other parameters.
JBOSS_IP	IP address to which server is to be bound.	127.0.0.1

A.2. CUSTOM SCRIPT

The following parameters can be used in the user customization section of the **`User Data:`** field.

JBOSS_DEPLOY_DIR	Deploy directory of the active profile (e.g. /usr/lib/jbossas/server/cluster-ec2/conf)
JBOSS_CONFIG_DIR	Config directory of the active profile (e.g. /usr/lib/jbossas/server/cluster-ec2/conf)
JBOSSCONF	Name of EAP active profile
USER_SCRIPT	Path to the custom configuration script, which is available prior to sourcing user-data configuration

APPENDIX B. TROUBLESHOOTING

EC2 does not provide any method out of the box to indicate an instance has started correctly and services are running properly. Use of an external system for monitoring and management is recommended so that you can be proactive. JBoss Operations Network (JON) can automatically discover, monitor and manage many services on an EC2 instance with the JON agent installed, including JBoss Enterprise Application Platform and its services, Tomcat, Httpd, PostgreSQL, etc. Since there's no difference between an EC-hosted or locally-hosted instance of JBoss Enterprise Application Platform or JBoss Enterprise Web Server, established JON monitoring of both types of deployments is identical.

B.1. DIAGNOSTIC INFORMATION

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

- `/var/log/jboss_user-data.out` is the output of the jboss-ec2-(eap|ews) init script and user custom configuration script;
- `/var/cache/jboss-ec2-eap/` contains the actual user data and custom script used at instance start-up (for an EAP instance);
- `/var/cache/jboss-ec2-ews/`contains the actual user data used at instance start-up (for an EWS instance);
- `/var/log` also contains all the logs collected from machine start up, EAP, Tomcat, httpd and most other services.

Access to these files is only available via an SSH session. Refer to the Amazon EC Getting Started Guide for details on how to configure and establish an SSH session with an EC2 instance.

B.2. JBOSS ENTERPRISE WEB SERVER

Issue

When trying to access the URL http://<public-DNS>:8080/hello, the browser reports a 404 error status instead of the "Hello world" application.

Action

1. Confirm the instance's state is "running" and if not, start it.
2. Check the Security Group which applies to the instance and confirm that access via ports 22 and 80 are enabled. If not, either modify the applicable Security Group or create a new one which permits access via these ports.
3. Check the application WAR file is inside the server's `deploy` directory and check server log for any errors.
4. Confirm that JBoss Enterprise Web Server is running by navigating to the URL http://<public-DNS>, checking that the "Red Hat Enterprise Web Server Test Page" is loaded.

APPENDIX C. REVISION HISTORY

Revision 5.2.0-100.400	**2013-10-30**	**Rüdiger Landmann**

Rebuild with publican 4.0.0

Revision 5.2.0-100	**Wed 23 Jan 2013**	**Russell Dickenson**

Incorporated changes for JBoss Enterprise Application Platform 5.2.0 GA. For information about documentation changes to this guide, refer to *Release Notes 5.2.0*.

Revision 5.1.2-201	**Tue July 3 2012**	**Russell Dickenson**

First edition.