Red Hat JBoss Enterprise Application Platform 7.3

Amazon Web サービスにおける JBoss EAP のディプロイ

高可用性の設定など、Amazon Web Services で Red Hat JBoss Enterprise Application Platform を使用する手順

Last Updated: 2023-08-23

Red Hat JBoss Enterprise Application Platform 7.3 Amazon Web サービスにおける JBoss EAP のディプロイ

高可用性の設定など、Amazon Web Services で Red Hat JBoss Enterprise Application Platform を使用する手順

法律上の通知

概要

本書では、Amazon EC2 に Red Hat JBoss Enterprise Application Platform をデプロイする方法について説明します。

目次

第1章 AMAZON EC2 について 3
Red Hat Cloud Access 製品ページ 3
Red Hat Cloud Access の機能 3
サポートされる Amazon EC2 インスタンスタイプ 4
サポート対象の Red Hat AMI 4

第2章 AMAZON EC2 コンソールでの JBOSS EAP インスタンスの起動 5

第3章 クラスター化されていない JBOSS EAP インスタンスの起動 6
要件 6
クラスター化されていない JBoss EAP インスタンスの起動 6

第4章 クラスター化していない管理対象ドメインの起動 8
4.1. インスタンスをドメインコントローラーとして提供するための起動 8
4.2. 1 以上のインスタンスを起動し、ホストコントローラーとして提供 9

第5章 クラスター化された JBOSS EAP の起動 11
5.1. クラスター化された JBOSS EAP AMI の起動 (MOD_CLUSTER および VPC なし) 11
5.2. クラスター化された JBOSS EAP AMI の起動 (MOD_CLUSTER および VPC を使用) 13

第6章 トラブルシューティング 16
6.1. AMAZON EC2 のトラブルシューティングについて 16
6.2. 診断情報 16

付録A リファレンス資料 17
A.1. RED HAT CLOUD ACCESS プログラム用の AMAZON EC2 AMI 17
A.2. 設定ファイルおよびデプロイメントの例 17
A.3. システムパス 17
A.4. スクリプトを使用した AMAZON EC2 での JBOSS EAP の起動 18
A.5. JBOSS EAP サブシステムを CLOUD PLATFORM で機能するように設定する 19
A.6. クラスター化された JBOSS EAP インスタンスのユーザーデータの例 20

第1章 AMAZON EC2 について

Amazon.com が運営サービスである Amazon Elastic Compute Cloud (Amazon EC2) は、カスタマイズ可能な仮想コンピューティング環境をお客様に提供します。このサービスでは、Amazon Machine Image (AMI) を起動して仮想マシンまたはインスタンスを作成できます。ユーザーは、インスタンスに必要なソフトウェアをインストールできます。料金は、使用する容量に応じて課金されます。Amazon EC2 は柔軟性が高く、デプロイされたアプリケーションを迅速にスケーリングできるように設計されています。

詳細は Amazon Web Services の web サイトを参照してください。

Amazon Machine Images について

Amazon Machine Image (AMI) は EC2 仮想マシンインスタンスのテンプレートです。ユーザーは、インスタンスを作成するための適切な AMI を選択して EC2 インスタンスを作成します。AMI の主要コンポーネントは、インストール済みのオペレーティングシステムやその他のソフトウェアを含む読み取り専用ファイルシステムです。AMI ごとに、さまざまなユースケース用にインストールされるソフトウェアが異なります。Amazon EC2 には、Amazon Web Services およびサードパーティーが提供する多くの AMI が含まれます。また、ユーザーは独自のカスタム AMI を作成することもできます。

Red Hat Cloud Access 製品ページ

Red Hat Cloud Access は、Amazon EC2 や Microsoft Azure などの Red Hat 認定クラウドプロバイダーで JBoss EAP をサポートする Red Hat サブスクリプション機能です。Red Hat Cloud Access では、従来のサーバーとパブリッククラウドベースのリソース間で、シンプルかつコスト効果の高い方法でサブスクリプションを移動することができます。

詳細は、Red Hat Cloud Access on the Customer Portal を参照してください。

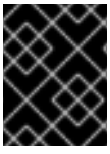
Red Hat Cloud Access の機能

Red Hat Cloud Access プログラムのメンバーシップでは、Red Hat によって作成されるサポート対象のプライベート Amazon Machine Images (AMI) へのアクセスを利用できます。

Red Hat AMI には、以下のソフトウェアが事前にインストールされており、Red Hat によって完全にサポートされています。

- Red Hat Enterprise Linux
- JBoss EAP
- Red Hat Update Infrastructure を使用した RPM による製品の更新

Red Hat AMI はそれぞれ開始点に S ぎません。アプリケーションの要件をさらに設定する必要があります。



重要

Red Hat Cloud Access は、スタンドアロンインスタンスまたは管理対象ドメインのいずれかで **full-ha** プロファイルに対応していません。

注記

Red Hat JBoss Operations Network のインストールに関する詳細は、**Installation Guide** を参照してください。

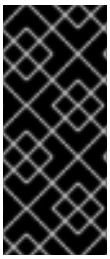
Red Hat JBoss Operations Network の設定に関する詳細は、**Configuring JBoss ON Servers, Agents, and Storage Nodes** を参照してください。

サポートされる Amazon EC2 インスタンスタイプ

Red Hat Cloud Access は、以下の Amazon EC2 インスタンスタイプをサポートします。各インスタンスの詳細は、Amazon Elastic Compute Cloud User Guide for Linux Instances を参照してください。

表1.1 サポートされる Amazon EC2 インスタンスタイプ

インスタンスタイプ	説明
標準のインスタンス	標準インスタンスは、メモリーと CPU の比率のバランスが取れた汎用環境です。JBoss EAP の処理が可能な最小のインスタンスタイプは **t2.small** と **m3.medium** です。
High Memory Instance	High Memory Instance には、標準インスタンスよりも多くのメモリーが割り当てられています。High Memory Instance は、データベースやメモリーキャッシングアプリケーションなどの高スループットアプリケーションに適しています。利用可能な対応のインスタンスタイプの最小タイプは **r3.large** です。
High CPU Instance	High CPU Instance にはメモリーよりも多くの CPU リソースが割り当てられています。このインスタンスは比較的低いスループットですが、CPU 集約型アプリケーションに適しています。利用可能な最小インスタンスタイプは **c3.large** および **c4.large** です。



重要

インスタンスタイプ Micro (**t2.micro**) および Nano (**t2.nano**) は JBoss EAP のデプロイメントには適していません。JBoss EAP 7.3 AMI は、少なくとも 10 GB のボリュームを必要とするスナップショットから構築されます。これは、インスタンスの作成時に EC2 コンソールで設定できます。割り当てたボリュームが小さすぎると、インスタンスの作成は失敗します。

サポート対象の Red Hat AMI

サポートされる Red Hat AMI は、その名前で識別できます。JBoss EAP AMI は、以下の構文で名前が付けられます。

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

- **Version** は、AMI にインストールされている JBoss EAP のバージョン番号です。例 **6.3**
- **osVersion** は、AMI にインストールされている Red Hat Enterprise Linux のバージョン番号です。例: **6.2**
- **arch** は、AMI のアーキテクチャーです。これは **x86_64** または **i386** です。
- **creationdate** は、AMI が YYYYMMDD の形式で作成された日付です。例: **20160315**

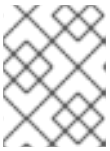
AMI 名の例: **RHEL-7.3_HVM_GA-JBEAP-7.1.0_-20170703-x86_64-1-Access2-GP2**

第2章 AMAZON EC2 コンソールでの JBOSS EAP インスタンスの起動

EC2 コンソールを使用して、Amazon EC2 で JBoss EAP インスタンスを起動できます。

AWS コマンドラインインターフェイスを使用してインスタンスを起動することもできます。AWS CLI の詳細は AWS CLI を参照してください。

1. Amazon EC2 コンソール を開きます。
2. Amazon EC2 コンソールから、**AMI** をクリックします。
3. **Private images** で **jbeap** AMI を検索し、AMI を選択します。例: **RHEL-7.3_HVM_GA-JBEAP-7.1.0_-20170703-x86_64-1-Access2-GP2**
4. インスタンスタイプを選択します。対応の Amazon EC2 インスタンスタイプの詳細は、Supported Amazon EC2 Instance Types を参照してください。
5. **Configure Instance Details** のセクションで、インスタンス設定を設定します。
6. **Advanced Details** セクションの **User data** ボックスで、インスタンスの起動時に JBoss EAP を実行するサンプルスクリプトを貼り付けできます。サンプルスクリプトの詳細は、Launching JBoss EAP on Amazon EC2 Using Script を参照してください。



注記

必要に応じて、ストレージを指定し、インスタンスにタグを付け、セキュリティーグループの詳細を設定できます。

7. **Review and Launch** をクリックします。これにより、インスタンスの起動の確認ページに移動します。
8. **Launch** をクリックしてキーペアを選択し、インスタンスを起動します。

注記

キーペアを選択していない場合は、インスタンスを起動する前にキーペアを指定する必要があります。

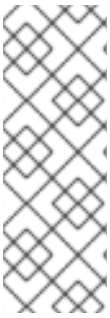
第3章 クラスター化されていない JBOSS EAP インスタンスの起動

本章では、Red Hat Amazon Machine Image (AMI) で JBoss EAP のクラスター化されていないインスタンスを起動する手順を説明します。

要件

- 適切な Red Hat AMI。詳細は、サポート対象の Red Hat AMI を参照してください。
- 少なくともポート 22、8080、および 9990 での受信要求を許可する事前設定されたセキュリティーグループ。

クラスター化されていない JBoss EAP インスタンスの起動



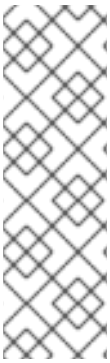
注記

ssh で **ec2-user** ユーザーとして EC2 インスタンスに接続できます。管理者権限が必要な場合は、後で **root** ユーザーに変更できます。以下に例を示します。

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

- Red Hat AMI インスタンスを起動します。
  JBoss EAP のクラスター化されていないインスタンスが Red Hat AMI で設定され、起動されている。
- JBoss EAP を設定するには、サービスに引数を直接渡すことができます。この方法で処理できない引数もあります。サービス設定ファイルの場所は次のとおりです。
  - RHEL 6: **/etc/sysconfig/eap7-standalone**
  - RHEL 7: **/etc/opt/rh/eap7/wildfly/eap7-standalone.conf**

  システムパスの詳細は、システムパス を参照してください。



注記

- 複雑な設定の場合は、 JBoss EAP **bin** directory: **/opt/rh/eap7/root/usr/share/wildfly/bin/** の **standalone.conf** ファイルを使用するか、JBoss EAP サービスを起動し、CLI を使用してサーバーを設定できます。このスクリプトは **bin** ディレクトリーにあります。次に、設定を再読み込みします。
- セキュリティー修正および機能強化を適用するには、**yum -y update** を定期的に実行する必要があります。

- RHEL 6 で JBoss EAP を起動するには、以下のコマンドを実行します。

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

  RHEL 7 で JBoss EAP を起動するには、以下のコマンドを実行します。

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

- RHEL 6 で JBoss EAP を停止するには、以下のコマンドを実行します。

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

RHEL 7 で JBoss EAP を停止するには、以下のコマンドを実行します。

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

注記

Systemctl コマンドは、RHEL 7 にのみ関連します。

注記

JBoss EAP を別の IP アドレスにバインドする場合は、RHEL 7 の **/etc/opt/rh/eap7/wildfly/eap7-standalone.conf** ファイルに以下の行を追加します。内部 IP アドレスは EC2 によりパブリック IP アドレスに変換されます。

```
WILDFLY_BIND=$YOUR_PRIVATE_IP_ADDRESS
```

第4章 クラスター化していない管理対象ドメインの起動

4.1. インスタンスをドメインコントローラーとして提供するための起動

このトピックでは、Red Hat Amazon Machine Image (AMI) でクラスター化されていない JBoss EAP 管理対象ドメインを起動する手順を説明します。

前提条件

- 適切な Red Hat AMI。詳細は、サポート対象の Red Hat AMI を参照してください。

クラスター化されていない JBoss EAP インスタンスの起動

注記

ssh で **ec2-user** ユーザーとして EC2 インスタンスに接続できます。管理者権限が必要な場合は、後で **root** ユーザーに変更できます。以下に例を示します。

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

- Red Hat AMI インスタンスを起動します。
  JBoss EAP のクラスター化されていないインスタンスが Red Hat AMI で設定され、起動されている。
- JBoss EAP を設定するには、サービスに引数を直接渡すことができます。この方法で処理できない引数もあります。サービス設定ファイルの場所は次のとおりです。
  - RHEL 6: **/etc/sysconfig/eap7-domain**
  - RHEL 7: **/etc/opt/rh/eap7/wildfly/eap7-domain.conf**

システムパスの詳細は、システムパス を参照してください。

Amazon EC2 の JBoss EAP サブシステムの詳細は、JBoss EAP サブシステムを Cloud Platform で機能するように設定する を参照してください。

注記

- 複雑な設定の場合は、JBoss EAP **bin** directory: **/opt/rh/eap7/root/usr/share/wildfly/bin/** の **domain.conf** ファイルを使用するか、JBoss EAP サービスを起動し、管理 CLI を使用してサーバーを設定できます。このスクリプトは **bin** ディレクトリーにあります。次に、設定を再読み込みします。
- セキュリティー修正および機能強化を適用するには、**yum -y update** を定期的に実行する必要があります。

- RHEL 6 で JBoss EAP を起動するには、以下のコマンドを実行します。

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

RHEL 7 で JBoss EAP を起動するには、以下のコマンドを実行します。

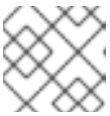
```
$ systemctl start eap7-domain
```

- RHEL 6 で JBoss EAP を停止するには、以下のコマンドを実行します。

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

RHEL 7 で JBoss EAP を停止するには、以下のコマンドを実行します。

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



注記

Systemctl コマンドは、RHEL 7 にのみ関連します。

注記

JBoss EAP を別の IP アドレスにバインドする場合は、RHEL 7 の **/etc/opt/rh/eap7/wildfly/eap7-domain.conf** ファイルに以下の行を追加します。内部 IP アドレスは EC2 によりパブリック IP アドレスに変換されます。

```
WILDFLY_BIND=$YOUR_PRIVATE_IP_ADDRESS
```

4.2.1 以上のインスタンスを起動し、ホストコントローラーとして提供

このトピックでは、Red Hat AMI でクラスター化されていないホストコントローラーとして機能するために JBoss EAP のインスタンスを起動する手順を説明します。

クラスター化されていないドメインコントローラーを設定して起動します。インスタンスをドメインコントローラーとして提供するための起動 を参照してください。

システムパスの詳細は、システムパス を参照してください。

Amazon EC2 の JBoss EAP サブシステムの詳細は、JBoss EAP サブシステムを Cloud Platform で機能するように設定する を参照してください。

ドメインコントローラーインスタンスの場合
Amazon EC2 で実行している管理対象ドメインでは、静的ドメインコントローラーの検出の他に、ホストコントローラーは Amazon Simple Storage (Amazon S3) システムを使用してドメインコントローラーを動的に検出できます。特に、ホストコントローラーとドメインコントローラーは、Amazon S3 バケットにアクセスするために必要な情報で設定できます。

この設定を使用すると、ドメインコントローラーが起動したときに、その通信情報がバケットの S3 ファイルに書き込まれます。ホストコントローラーがドメインコントローラーとの通信を試行するたびに、**S3** ファイルからドメインコントローラーの通信情報を取得します。

たとえば、Amazon EC2 インスタンスの IP アドレスが停止および起動すると変更するのが一般的です。このシナリオでは、ドメインコントローラーの通信情報が変更された場合に、ホストコントローラーを再設定する必要はありません。ホストコントローラーは、**S3** ファイルからドメインコントローラーの新しい通信情報を取得できます。

ユーザーデータのサンプルスクリプトの詳細は、Example User Data for Clustered JBoss EAP Instances を参照してください。

手動によるドメインコントローラーの検出設定は、以下のプロパティーを使用して指定します。

- **access-key**: Amazon AWS ユーザーアカウントのアクセスキー。
- **secret-access-key**: Amazon AWS ユーザーアカウントのシークレットアクセスキー。
- **location**: 使用する Amazon S3 バケット。

1. **domain-ec2.xml** ファイルを **/opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs** から JBoss EAP 設定ディレクトリーにコピーします。
2. 以下の変数を適切なサービス設定ファイルに設定します。

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

3. S3 ドメインコントローラー検出設定を **domain-ec2.xml** ファイルに追加します。

```
<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>
```

第5章 クラスター化された JBOSS EAP の起動

5.1. クラスター化された JBOSS EAP AMI の起動 (MOD_CLUSTER および VPC なし)

このトピックでは、mod_cluster および VPC なしでクラスター化された JBoss EAP AMI を起動する手順について説明します。

注記

- イメージで提供される設定スクリプトサンプルを使用できます。

システムパスの詳細は、システムパス を参照してください。

Amazon EC2 の JBoss EAP サブシステムの詳細は、JBoss EAP サブシステムを Cloud Platform で機能するように設定する を参照してください。

クラスター化された JBoss EAP AMI をスタンドアロンサーバーインスタンスで開始するには、事前設定された S3_PING JGroups スタックが含まれる **/opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/standalone-ec2-ha.xml** ファイルのサンプルを使用できます。詳細は、**Reliable group communication with JGroups** ドキュメントの S3_PING を参照してください。この **standalone-ec2-ha.xml** プロファイルファイルは、**/opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/** から JBoss EAP 設定ディレクトリー **/opt/rh/eap7/root/usr/share/wildfly/standalone/configuration/** にコピーされる必要があります。その後、以下の行を JBoss EAP サービス設定ファイルに追加する必要があります。

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

undertow サブシステムのスタンドアロンサーバーインスタンスごとに、固有の **instance-id** を設定する必要があります。**instance-id** の値は、**standalone-ec2-ha.xml** ファイルを編集して、あるいは管理 CLO を使用して手動で設定できます。たとえば、以下のように管理 CLI を使用して **instance-id** を設定できます。

```
/subsystem=undertow:write-attribute(name=instance-id,value={${jboss.jvmRoute}})
```

Jboss.jvmRoute の値は **JAVA_OPTS** 変数を使用して **standalone.conf** に指定できます。

EC2 設定ファイルの **jgroups** サブシステムでは、クラスターメンバーを検出するために **S3_PING** 固有のプロパティーが必要です。S3、シークレットアクセスキー、および検出に使用する S3 バケットにアクセスキーを指定する必要があります。これらのプロパティーは Java オプションとして指定するか、編集して、あるいは CLI を使用して XML ファイルに直接追加することができます。

検出用に S3 バケットを作成する必要があります。詳細は、Amazon Simple Storage Service Documentation を参照してください。必要なパーミッションの設定が必要になることもあります。JGroups スタックは、他のノードとの通信に使用される IP アドレスにバインドする必要があります。これは、S3 Java オプションとともに **/opt/rh/eap7/root/usr/share/wildfly/bin/standalone.conf** ファイルに Java オプションを追加することで行うことができます。たとえば、プライベート IP アドレスが **10.10.10.10** の場合は、以下の行を **standalone.conf** ファイルに追加します。

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

サンプルアプリケーション: **/opt/rh/eap7/root/usr/share/java/eap7-jboss-ec2-eap-samples/cluster-demo.war** をデプロイして、**/opt/rh/eap7/root/usr/share/wildfly/standalone/log/server.log** でログを確認し、JBoss EAP サーバーがクラスターを作成していることを確認できます。

ドメインコントローラーインスタンスの場合

1. **domain-ec2.xml** ファイルを **/opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs** から JBoss EAP 設定ディレクトリーにコピーします。
2. 以下の変数を適切なサービス設定ファイルに設定します。

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

3. S3 ドメインコントローラー検出設定を **host-master.xml** ファイルに追加します。

```
<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. ユーザーを設定し、ユーザーのシークレット値をホストコントローラーインスタンスに追加します。詳細は、**設定ガイド** の 2 台のマシンで管理対象ドメインを作成 を参照してください。

ホストコントローラーインスタンスの場合

1. 以下の変数を適切なサービス設定ファイルに設定します。

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

2. S3 ドメインコントローラー検出設定を **host-slave.xml** ファイルに追加します。

```
<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>
```

注記

S3 ドメインコントローラーの検出に関する情報は、1 以上のインスタンスを起動し、ホストコントローラーとして提供 を参照してください。

警告

24 ビットより小さいネットワークマスクを持つサブネットで JBoss EAP クラスターを実行するか、複数のサブネットにまたがると、各クラスターメンバーの一意のサーバーピア ID の取得が複雑になります。

重要

Amazon EC2 の自動スケーリング機能は、JBoss EAP クラスターノードで使用できます。ただし、デプロイする前に必ずテストしてください。必要なノード数に特定のワークロードがスケーリングし、使用する予定のインスタンスタイプで必要とされるパフォーマンスを満たすようにする必要があります。インスタンスタイプが受信する EC2 クラウドリソースのシェアは、それぞれによって異なります。

さらに、インスタンスのローカリティーおよび現在のネットワーク/ ストレージ/ ホストマシン/RDS 使用率は、クラスターのパフォーマンスに影響を与える可能性があります。予想される実際の負荷をテストし、予期しない状況を考慮するようにしてください。

警告

Amazon EC2 **スケールダウン** アクションは、正常にシャットダウンする可能性なしにノードを終了します。また、一部のトランザクションが中断する可能性があるため、他のクラスターノードおよびロードバランサーには、フェイルオーバーの時間が必要です。これは、アプリケーションユーザーエクスペリエンスに影響を与える可能性があります。

処理されるセッションが完了するまで mod_cluster 管理インターフェイスからサーバーを無効にしてアプリケーションクラスターを手動でスケールダウンするか、インスタンスまたは Red Hat JBoss Operations Network への SSH アクセスを使用して JBoss EAP インスタンスを正常にシャットダウンして、アすることが推奨されます。

スケールダウンの手順をテストしても、ユーザーエクスペリエンスに悪影響が及ぶことはありません。特定のワークロード、ロードバランサー、および設定には、追加の対策が必要になる場合があります。

5.2. クラスター化された JBOSS EAP AMI の起動 (MOD_CLUSTER および VPC を使用)

このトピックでは、Apache HTTP サーバーインスタンスを起動して **mod_cluster** プロキシーとして機能し、Virtual Private Cloud (VPC) の NAT インスタンスとして機能する手順について説明します。

システムパスの詳細は、システムパス を参照してください。

Amazon EC2 の JBoss EAP サブシステムの詳細は、JBoss EAP サブシステムを Cloud Platform で機能するように設定する を参照してください。

注記

- イメージで提供される設定スクリプトサンプルを使用できます。

Amazon Virtual Private Cloud (Amazon VPC) は、プライベートネットワークで AWS リソースのセットを分離できるようにする Amazon Web Services (AWS) の機能です。このプライベートネットワークのトポロジーおよび設定は、ニーズに合わせてカスタマイズできます。

Amazon VPC の詳細は、Amazon Virtual Private Cloud を参照してください。

注記

VPC 内で **mod_cluster** ロードバランサーを使用してクラスターを起動すると、JBoss EAP サーバーはパブリックにアクセスできなくなります。**mod_cluster** ロードバランサーは、インターネットに接続されるエンドポイントのみにすることができます。

ドメインコントローラーインスタンスの設定は、Launch an Instance to Serve as a Domain Controller を参照してください。

ホストコントローラーインスタンスの設定は、1 以上のインスタンスを起動し、ホストコントローラーとして提供 を参照してください。

S3 ドメインコントローラーの検出に関する情報は、1 以上のインスタンスを起動し、ホストコントローラーとして提供 を参照してください。

5.2.1. VPC および mod_cluster を使用してクラスター化された AMI を起動する

注記

VPC の設定は任意です。詳細は、Amazon VPC ユーザーガイドの Detecting Your Supported Platforms and Whether You Have a Default VPC セクションを参照してください。

1. **Jbcs-httpd24-mod_cluster-native** パッケージとそのすべての依存関係をインストールします。**mod_cluster** 設定ファイルは **/opt/rh/jbcs-httpd24/root/etc/httpd/conf.d/mod_cluster.conf** にインストールされています。

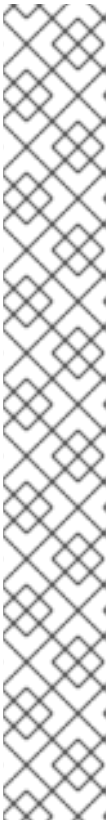
Red Hat JBoss Core Services Apache HTTP Server のインストールの詳細は、**Apache HTTP Server Installation Guide** を参照してください。

1. **mod_cluster** のアドバタイズの無効化以下を **/opt/rh/jbcs-httpd24/root/etc/httpd/conf.d/mod_cluster.conf** 設定ファイルの **VirtualHost** に追加します。

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

2. **SELinux** のポートを許可します。必要に応じて **iptables** を設定します。ポートは、**semanage port -a -t http_port_t -p tcp $PORT_NR** コマンドを使用して SELinux で許可できます。

3. **mod_cluster** がリッスンするアドレスで **mod_cluster** プロキシーを検索するよう JBoss EAP を設定します。



注記

/opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/standalone-ec2-ha.xml 設定ファイルのサンプルが提供されます。**modcluster** サブシステムで **proxies** の一覧を設定する必要があります。

以下の方法のいずれかを使用して **proxies** のリストを定義できます。

- **mod-cluster-proxy1** という **outbound-socket-binding** を適切なホストとポートで定義します。

```
<outbound-socket-binding name="mod-cluster-proxy1">
    <remote-destination host="${jboss.modcluster.proxy1.host}"
port="${jboss.modcluster.proxy1.port}"/>
</outbound-socket-binding>
```

- **modcluster** の **proxies** 属性を適切なホストとポートで **mod-cluster-proxy1** に設定します。

```
/socket-binding-group=standard-sockets/remote-destination-outbound-
socket-binding=mod-cluster-proxy1:add(host=
{${jboss.modcluster.proxy1.host}}, port={${jboss.modcluster.proxy1.port}})
```

第6章 トラブルシューティング

6.1. AMAZON EC2 のトラブルシューティングについて

EC2 は、インスタンスごとに Alarm Status を提供します。これは、致命的なインスタンスの誤動作を示します。ただし、このような警告がなくても、インスタンスが正しく起動し、サービスが正しく実行されていることが保証されるわけではありません。Amazon CloudWatch をカスタムメトリクス機能とともに使用すると、インスタンスサービスの健全性を監視することができますが、エンタープライズ管理ソリューションを使用することが推奨されます。

6.2. 診断情報

JBoss Operations Network、Amazon CloudWatch、または手動の検査によって問題が検出された場合、診断情報の一般的なソースは以下のようになります。

- **/var/log** には、マシンの起動、JBoss EAP、httpd およびその他のサービスから収集されたすべてのログも含まれます。

JBoss EAP ログファイルは **/opt/rh/eap7/root/usr/share/wildfly/** にあります。

これらのファイルへのアクセスは、SSH セッションのみで可能です。

Amazon EC2 を使用して SSD セッションを設定して確立する詳細は、Getting Started with Amazon EC2 Linux Instances を参照してください。

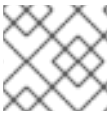
付録A リファレンス資料

A.1. RED HAT CLOUD ACCESS プログラム用の AMAZON EC2 AMI

AMI は、Red Hat Enterprise Linux イメージでの JBoss EAP + JDK の基本的な RPM インストールで、Amazon EC2 の設定例がある可能性があります。高度なスクリプトは利用できなくなりましたが、通常の bash スクリプトは使用できます。

プラットフォーム/JDK の組み合わせの AMI:

- RHEL 6 + Open JDK 8 (1 イメージ)
- RHEL 7 + Open JDK 8 (1 イメージ)



注記

どちらのプラットフォームも 64 ビットアーキテクチャーである必要があります。

AMI のメンテナンス

EC2 に z リリース (パッチ) を適用するには、**yum update** を定期的に実行する必要があります。Y リリース (マイナーリリース) の新しい AMI は、Red Hat から提供されます。

シナリオ 1 (サポートあり)

1. EC2 にサインアップします。
2. Red Hat Cloud Access にサインアップします。
3. 利用可能な AMI の一覧から Red Hat AMI を選択します。
4. (オプション) ユーザースクリプトまたは **ssh** で JBoss EAP 設定をカスタマイズします。
5. メンテナンス:y リリースの新しい AMI z リリース用の **yum update**

A.2. 設定ファイルおよびデプロイメントの例

以下のパッケージでは、RHEL 7 AMI バージョンのデプロイメント例が追加されます。

```
$ 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
```

JBoss EAP 設定ファイルのサンプルには、EC2 全体でクラスターを作成するのに使用できる **S3_PING** プロトコル用にセットアップされた JGroups スタックが含まれます。設定ファイルのサンプルの正確な場所は、System Paths を参照してください。

マルチキャストは EC2 で無効になっているため、両方の設定ファイルには、アドバタイズの代わりにプロキシー mod_cluster 検出を使用するための **modcluster** サブシステムが含まれます。

A.3. システムパス

サービス設定ファイル:

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

JBoss EAP Home:

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

JBoss EAP 設定の場所:

スタンドアロンインスタンス

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

管理対象ドメイン

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

JBoss EAP の設定場所の例:

- /opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/standalone-ec2-ha.xml
- /opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/standalone-ec2-full-ha.xml

A.4. スクリプトを使用した AMAZON EC2 での JBOSS EAP の起動

以下のサンプルスクリプトを使用して、Amazon EC2 で JBoss EAP インスタンスを起動すると、パブリック IP アドレスにバインドされる JBoss EAP を開始できます。

```
#!/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. JBOSS EAP サブシステムを CLOUD PLATFORM で機能するように設定する

Amazon EC2 や Microsoft Azure などのクラウドプラットフォームで適切に動作するように、一部の JBoss EAP サブシステムを設定する必要があります。これは、JBoss EAP サーバーは通常、クラウドプラットフォームからのみ確認可能な **10.x.x.x** などのクラウド仮想マシンのプライベート IP アドレスにバインドされているために必要です。特定のサブシステムでは、このアドレスをサーバーのパブリック IP アドレスにマッピングする必要もあります。これは、クラウド外から見えるものです。

A.5.1. Web サービス

クライアントが **Service.create(SELECTURL, serviceName)** を使用して Web サービスリクエストを行うと、ユーザーはサーバーのパブリック IP アドレスに接続しますが、その後、**webservices** サブシステムのサーバー設定ファイルに定義されたアドレスにリダイレクトされます。デフォルトでは、このアドレスは **${jboss.bind.address:127.0.0.1}** です。つまり、クラウドプラットフォームでは呼び出し元はサーバーのプライベート IP アドレスにリダイレクトされ、要求を解決できません。サーバーのパブリック IP アドレスは、以下のコマンドを使用して **ogg-host** 要素で設定する必要があります。

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

A.5.2. Messaging

クラウドプラットフォームでメッセージングを使用する場合は、クライアントが使用する接続ファクトリーには、サーバーのパブリック IP アドレスを指すコネクターが必要です。

そのため、**完全な** プロファイルを実行している JBoss EAP サーバーに対して新しいコネクターとソケットバインディングを作成する必要があります。

1. 参照される **http-public** ソケットバインディングは **socket-binding-group** 内に作成する必要があります。

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

2. **messaging** サブシステムに新しい **http-connector** 要素を作成します。

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

3. **connection-factory** に **connectors** を設定します。これは、クライアントによって使用されます。たとえば、デフォルトの接続としての **RemoteConnectionFactory** の設定は以下のようになります。

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

A.5.3. 高可用性のリモーティング設定

クラウドプラットフォーム上のクラスター化された EJB で JBoss EAP HA 機能を使用している場合は、EJB クライアントがクラスタービューの更新を受け取るために **remoting** サブシステムに追加の設定が必要になります。

これは **remoting** サブシステムのソケットバインディングに **client-mappings** を設定して行います。

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

A.6. クラスター化された JBOSS EAP インスタンスのユーザーデータの例

以下の例は、いくつかの異なるサーバー設定用に設定されたユーザーデータを示しています。

例: RHEL6/7 でのスタンドアロンモードのファイル

```
#!/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 JBoss 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 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 JBoss EAP to use the ec2-ha profile
cp /opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/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 JBoss EAP, note that RHEL 7 does not wait for JBoss EAP to start before returning from the
service start. In some cases, there could be a delay of more than 90 seconds.

$START_COMMAND
```

例: クラスター化されたドメインコントローラーを起動するファイル

```
#!/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 access key id>
SECRET_ACCESS_KEY=<your secret access key>
S3_PING_BUCKET=<your s3 ping bucket>

#### No further modifications should be needed below to run this example ####
# Set the location of JBoss EAP
JBOSS_HOME=/opt/rh/eap7/root/usr/share/wildfly
CONF_FILE=/opt/rh/eap7/root/usr/share/wildfly/docs/examples/configs/domain-ec2.xml

# 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 JBoss EAP to use the domain-ec2.xml and host-master.xml configuration files
cp ${CONF_FILE} $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=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
chgrp jboss $JBOSS_HOME/domain/log/audit.log
chgrp jboss $JBOSS_HOME/domain/log/host-controller.log
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
chown jboss $JBOSS_HOME/domain/log/audit.log
chown jboss $JBOSS_HOME/domain/log/host-controller.log

# 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 JBoss EAP, do not forget that RHEL 7 does not wait for JBoss EAP 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=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'
```

例: クラスター化されたドメインコントローラーを起動するファイル

```
#!/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 access key id>
SECRET_ACCESS_KEY=<your secret access key>
S3_PING_BUCKET=<your s3 ping bucket>

#### 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 JBoss EAP
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

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:9990}\"/>
+               <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

EAP_HOST_NAME=`$JBOSS_HOME/bin/jboss-cli.sh --commands="embed-host-controller --host-
config=host-slave.xml, :read-resource" | grep \"host\" | cut -d\" -f4`
$JBOSS_HOME/bin/jboss-cli.sh --commands="embed-host-controller --host-config=host-slave.xml,
/host=$EAP_HOST_NAME/core-service=management/security-realm=ManagementRealm/server-
identity=secret:write-attribute(name=value,value=c2VjcmV0X1Bhc3N3MHJk)"

sed -i 's/<host xmlns="urn:jboss:domain:8.0">/<host xmlns="urn:jboss:domain:8.0" name="admin">/'
host-slave.xml
sed -i 's/other-server-group/main-server-group/' host-slave.xml

cd -

# 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
chgrp jboss $JBOSS_HOME/domain/log/audit.log
chgrp jboss $JBOSS_HOME/domain/log/host-controller.log
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
chown jboss $JBOSS_HOME/domain/log/audit.log
chown jboss $JBOSS_HOME/domain/log/host-controller.log
```

```
# Start JBoss EAP, do not forget that RHEL 7 does not wait for JBoss EAP 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 2023-01-28 12:00:09 +1000