KEEDIO
Data Stack

DEPLOYMENT OF KDS: Vagrant

INDEX

INTRODUCTION

This is a **Vagrant-based test environment**, designed to test the integration of the different packages of the Keedio Data Stack, which can be used to deploy virtual clusters using either Ambari or by manually configuring the services. A hiera-based configuration file can be used to adapt Keedio-Vagrant to different environments and requirements. By default, it uses the Keedio public repository to install the packages. To limit the external bandwidth requirement, a local mirror of the main Keedio repository *repo.keedio.org* can be created in a VM called buildoop.

The same VM contains the buildoop packaging system and can be used to build new versions of the software components. These new versions can then be deployed to the test VMs using the local repo. The vagrant plugin "vagrant-vbox-snapshot" can be used to snapshot the state of the cluster, and move easily back and forward in time by selecting the right snapshot.

Obtaining Access To KEEDIO STACK

Keedio uses Red Hat satellite to manage the distribution of the Keedio Data Stack. In order to install KDS, you need to request an activation code. You can request it here https://www.keedio.org/demo/ and use it in the setup process as described below.

Alternatively, you can package your own repository using buildoop and you can tell keedio-vagrant to use default yum repositories by setting the variable satellite: false in ./hiera/configuration.yaml.

1. PRELIMINARY STEPS

Install Vagrant and Virtualbox before starting.

Make sure you install the vagrant snapshotting plugin for virtualbox:

```
vagrant plugin install vagrant-vbox-snapshot
```

Install hiera-eyaml ruby gem:

```
gem install hiera-eyaml
```

Download the keedio-vagrant stack:

```
git clone --recursive https://github.com/keedio/
keedio-vagrant.gitcd  keedio-vagrant
```

Setup The Cluster

The following procedure is agnostic to the hypervisor or cloud provider you are going to use.

VERSIONS

This section lists the versions of the different components used in this guide, it should work with other versions, but we strongly recommend **using the following** versions:

- **Keedio Data Stack 1.4**
- **Vagrant 1.9.3**
- **VirtualBox 5.1.20**

2. SETUP SCRIPT

This script helps the user configure basic things on the cluster, you'll need a Keedio Activation key. The script uses some defaults stored at hiera.eyalm:

```
./setup.sh
################################################
Setting up environment for Keedio Stack deploymentThis
will use default encryption keys and passwordsPlease
use setup_secure.sh if you want to change them
################################################
[hiera-eyaml-core] hiera-eyaml (core): 2.0.8Please
provide the activation key:
################################################
The hiera/secure.eyaml file has been createdYou can
check the values and modify it with the command eyaml
edit hiera/secure.eyaml
################################################
```

Setup The Cluster

3. OPTIONAL: SECURE MODE

This script helps you generate new encryption keys, and set non-default password. You can use the following command:

```
./setup_secure.sh

##################################################
Setting up environment for Keedio Stack deployment
This will use default encryption keys and passwords
Please use setup_secure.sh if you want to change them
##################################################
[hiera-eyaml-core] hiera-eyaml (core): 2.0.8
Creating Hiera encryption keys
Are you sure you want to overwrite "./keys/private_key.pkcs7.pem"? (y/N): y
Are you sure you want to overwrite "./keys/public_key.pkcs7.pem"? (y/N): y
[hiera-eyaml-core] Keys created OK
Please provide the activation key:
Please provide the admin password for the backend database:[adminadmin]
Please provide the hue user password for the backend database:[hue]
Please provide the hive user password for the backend database:[hive]
Please provide the oozie user password for the backend database:[oozie]
Please provide the ambari user password for the backend database:[bigdata]
Please provide the Free IPA admin password:[adminadmin]
Please provide the Ambari admin password:[admin]
##################################################
The hiera/secure.eyaml file has been created
You can check the values and modify it with the command

eyaml edit hiera/secure.eyaml
##################################################
Making a backup-copy of the existing ssh keys in ./files/ssh-backup
Generating new ssh keys in ./files/.ssh/
Generating public/private rsa key pair.
./files/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Your identification has been saved in ./files/.ssh/id_rsa.
Your public key has been saved in ./files/.ssh/id_rsa.pub.
The key fingerprint is...
```


KEEDIO-VAGRANT WITH VIRTUALBOX

This section describes the deployment procedure of Keedio Data Stack (KDS) on VirtualBox platform.

We assume VirtualBox is already installed on your system.

KEEDIO-VAGRANT WITH VIRTUALBOX

1. PRELIMINARY STEPS

First, enter the directory in which you cloned the keedio-vagrant project as described in *Preliminary Steps*.
Populate /etc/hosts of your machine with the provided information:

```
sudo cat  append-to-etc-hosts.txt  >> /etc/hosts
```

Prepare Vagrantfile and configuration.yaml

```
cd  ambari1
cp vagrantfiles/Vagrantfile.virtualbox Vagrantfile
cp configurations/standard-user.yaml hiera/configuration.yaml
```

KEEDIO-VAGRANT WITH VIRTUALBOX

2. STARTING THE CLUSTER

You can now start your ambari cluster, you should always start the master machine, and at least one slave (ambari1, ambari2, ambari3...). In this example we start the master and two slaves:

```
vagrant up master ambari1 ambari2
```

This can take several minutes. As soon as it is completed, you should be able to access the Ambari web page: master.ambari.keedio.org:8080. The default credentials are both "admin".
You can suspend the execution of all the VMs with:

```
vagrant suspend
```

And restart with:

```
vagrant resume
```


KEEDIO-VAGRANT ON AMAZON AWS

This section describes the deployment procedure of Keedio Data Stack (KDS) on Amazon AWS platform.

KEEDIO-VAGRANT ON AMAZON AWS

1. PRELIMINARY STEPS

To deploy on AWS you need:

- An account on Amazon Web Services.
- Install aws-cli (AWS CLI INSTALL)
- Install the Vagrant plugin for AWS

2. INSTALLING AWS-CLI

We strongly recommended to follow the installation guide from AWS doc (see the url). If you had pip already installed:

```
pip install --upgrade --user awscli
```

To install pip:

```
curl -O https://bootstrap.pypa.io/get-pip.py
python3 get-pip.py —user
```

You need to add the AWS-CLI executable to your command line path, add the export command to profile script:

```
export PATH=~/.local/bin:$PATH
```


KEEDIO-VAGRANT ON AMAZON AWS

Then, load the profile into your current session:

```
source ~/.bash_profile
```

3. CONFIGURING AWS-CLI

First of all, you need to configure aws-cli:

NOTE: Output format is Json.

```
aws configure
```

To interact with AWS, we'll need to get AWS tokens. Execute the next command and follow keep safe the tokens printed on the screen:

```
aws sts get-session-token
```

4. INSTALL THE AWS VAGRANT PLUGIN

```
vagrant plugin install vagrant-aws
```


KEEDIO-VAGRANT ON AMAZON AWS

5. GETTING THE SECURITY GROUP ID

You can use the default group, it is all open:

```
aws ec2 describe-security-groups --group-names default |grep GroupId
```

6. CREATE KEYPAIR

You also need to create a keypair on AWS, take a look at the next URL to learn how to create it on AWS: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html You can create it from CLI:

```
aws ec2 create-key-pair --key-name MyKeyPair
```

You need to save the output to a file.

7. CREATE THE VAGRANTFILE

```
cp -p vagrantfiles/Vagrantfile.aws Vagrantfile
cp -p configurations/configuration_aws.yaml hiera/configuration.yaml
```


KEEDIO-VAGRANT ON AMAZON AWS

8. EXPORTING THE ENV VARIABLES

In order to use Vagrant, you will need to write some variables on the Vagrant file:

- AWS_INSTANCE_TYPE (default is t2.medium)
- AWS_VPC_SUBNET_ID
- AWS_VPC_SECURITY_GROUP
- AWS_ACCESS_KEY_ID (get on token-session)
- AWS_SECRET_ACCESS_KEY
- AWS_KEYPAIR_NAME (MyKeyPair)
- AWS_SSH_PRIVATE_KEY_PATH (path to your id_rsa)
- AWS_REGION OR AWS_DEFAULT_REGION
- AWS_SESSION_TOKEN

You can also export it like ENV variables:

```
export VARIABLE="value"
```

9. LAUNCHING VAGRANT AWS

1. Edit hiera/default:

```
nameresolution: 'script' ======> nameresolution: 'aws'
```


KEEDIO-VAGRANT ON AMAZON AWS

2. Vagrant up:

```
vagrant up --provider=aws --no-provision
```

3. Exec the aws_hosts.sh script: When vagrant up is finished, you need to launch this script to correctly populate /etc/hosts.

```
bash aws_hosts.sh
```

4. Run rsync:

```
vagrant rsync
```

5. Run the provision:

```
vagrant provision
```


KEEDIO-VAGRANT ON MICROSOFT AZURE

This section describes the deployment procedure of Keedio Data Stack (KDS) on Microsoft Azure platform.

KEEDIO-VAGRANT ON MICROSOFT AZURE

1. PRELIMINARY STEPS

In order to deploy KDS on Microsoft Azure you need:

- At least a trial account on Microsoft Azure.
- Install Azure-cli (https://docs.microsoft.com/en-us/cli/azure/install-azure-cli)
- Install the Vagrant plugin for Azure

2. INSTALLING AZURE CLI

It is strongly recommended to install the Azure CLI using the bash script provided by Microsoft:

```
curl -L https://aka.ms/InstallAzureCli | bash
```

You must restart the shell:

```
exec -l $SHELL
```

3. CONFIGURING AZURE CLI

In order to interact with Azure machines, we'll need to login in our Azure's account. Execute the next command and follow the steps printed on the screen:

```
az login
```


KEEDIO-VAGRANT ON MICROSOFT AZURE

4. GENERATING KEYS FOR AZURE DEPLOYMENTS

The Azure provider seems to only like PEM files with both the public and private keys, and X509 certificates, so let's get us some of that:

```
openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout ~/.ssh/azurevagrant.key -out ~/.ssh/
azurevagrant.key
chmod 600 ~/.ssh/azurevagrant.key
openssl x509 -inform pem -in ~/.ssh/azurevagrant.key -outform der -out ~/.ssh/azurevagrant.cer
```

The .cer file contains our public key, but the .pem file contains both our public and private keys, so we'll need to secure it appropriately. Now we can upload that .cer file as a management certificate in Azure using the browser.

5. INSTALL THE AZURE VAGRANT PLUGIN

```
vagrant plugin install vagrant-azure --plugin-version '2.0.0.pre6'
```

6. EXPORT ENV VARIABLES

In order to use Azure Vagrant you will need to export some variables. First of all, we need to get some info:

```
az ad sp create-for-rbac
```


KEEDIO-VAGRANT ON MICROSOFT AZURE

Keep an eye on the output, you are going to need it.

The list of variables is the following:

- AZ_VM_SIZE=ENV["AZ_VM_SIZE"] || "Standard_D1_v2" (Default value)
- AZ_RESOURCE_GROUP_NAME=ENV["AZ_RESOURCE_GROUP_NAME"] || "KDSRG" (Default value)
- AZ_KEYPAIR_PATH=ENV["AZ_KEYPAIR_PATH"]
- AZ_SUBCRIPTION_ID=ENV["AZ_SUBCRIPTION_ID"]
- AZ_TENANT_ID=ENV["AZ_TENANT_ID"]
- AZ_CLIENT_ID=ENV["AZ_CLIENT_ID"]
- AZ_CLIENT_SECRET=ENV["AZ_CLIENT_SECRET"]
- AZ_LOCATION=ENV["AZ_LOCATION"]

You can export it executing the next command:

```
export VARIABLE="value"
```

7. CREATE THE VAGRANT FILE

```
cp -p vagrantfiles/Vagrantfile.azure Vagrantfile
cp –p configurations/configuration_aws.yaml hiera/configuration.yaml
```

KEEDIO-VAGRANT ON MICROSOFT AZURE

8. LAUNCHING VAGRANT AZURE

1. Edit hiera/default:

```
nameresolution: 'script' ======> nameresolution: 'azure'
```

2. Vagrant up:

```
vagrant up --provider=azure --no-provision --no-parallel
```

3. Exec the az_hosts.sh script: When vagrant up is finished, you need to launch this script for create the /etc/hosts.

```
bash az_hosts.sh
```

4. Run rsync:

```
vagrant rsync
```

5. Run the provision:

```
vagrant provision
```

www.keedio.com info@keedio.com @keedio keedio

Calle Virgilio 25
Edificio Ayessa I, Bajo D
Pozuelo de Alarcón
28223 Madrid