Quick Start Guide

Quick Start Guide - Installing a cluster with Ambari (with local VMs)

This document shows how to quickly set up a cluster using Ambari on your local machine using virtual machines.
This utilizes VirtualBox and Vagrant so you will need to install both.
Note that the steps were tested on MacOS 10.8.4 / 10.8.5.

Setup

Install VirtualBox from: https://www.virtualbox.org/wiki/Downloads
Install Vagrant from: http://downloads.vagrantup.com

After you have installed VirtualBox and Vagrant on your computer, check out the “ambari-vagrant” repo on github:

```
git clone https://github.com/u39kun/ambari-vagrant.git
```

Edit your **/etc/hosts** on your computer so that you will be able to resolve hostnames for the VMs:

```
sudo -s 'cat ambari-vagrant/append-to-etc-hosts.txt >> /etc/hosts'
```

Copy the private key to your home directory (or some place convenient for you) so that it’s easily accessible for uploading via Ambari Web:

```
vagrant
```

The above command shows the command usage and also creates a private key as ~/.vagrant.d/insecure_private_key. This key will be used in the following steps.

Starting VMs

First, change directory to ambari-vagrant:

```
cd ambari-vagrant
```

You will see subdirectories for different OS’s. “cd” into the OS that you want to test. **centos6.8** is recommended as this is quicker to launch than other OS's.
Now you can start VMs with the following command:

```
cd centos6.8
cp ~/.vagrant.d/insecure_private_key .
./up.sh <# of VMs to launch>
```

For example, **up.sh 3** starts 3 VMs. 3 seems to be a good number with 16GB of RAM without taxing the system too much.
With the default **Vagrantfile**, you can specify up to 10 (if your computer can handle it; you can even add more).
VMs will have the FQDN <os-code>[01-10].ambari.apache.org, where <os-code> is c59 (CentOS 5.9), c68 (CentOS 6.8), etc.
E.g., c5901.ambari.apache.org, c6801.ambari.apache.org, etc.
VMs will have the IP address 192.168.<os-subnet>.1[01-10], where <os-subnet> is 59 for CentOS 5.9, 68 for CentOS 6.8, etc.
E.g., 192.168.59.101, 192.168.64.101, etc.
Note that **up.sh 3** command is equivalent to doing something like: vagrant up c680{1..3}

Testing Ambari

If it is your first time running a vagrant command, run:

```
vagrant init
```

Log into the VM:

```
vagrant ssh c6801
```

Note that this logs you in as user **vagrant**. Once you are logged in, you can run:

```
sudo su -
```

to make yourself root.

To install Ambari, you can build it yourself from source (see Ambari Development), or you can use published binaries.

As this is a Quick Start Guide to get you going quickly, ready-made, publicly available binaries are referenced in the steps below.
Note that these binaries were built and made publicly available via Hortonworks, a commercial vendor for Hadoop. This is for your convenience. Note that using the binaries shown here would make HDP, Hortonworks' distribution, available to be installed via Apache Ambari. The instructions here should still work (only the repo URLs need to be changed) if you have Ambari binaries from any other vendor/organization/individuals (the instructions here can be updated if anyone wanted to expand this to include such ready-made, publicly accessible binaries from any source - such contributions are welcome). This would also work if you had built the binaries yourself.

```
# CentOS 6 (for CentOS 7, replace centos6 with centos7 in the repo URL)
#
# to test public release 2.5.1
wget -O /etc/yum.repos.d/ambari.repo http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/2.5.1.0
/ambari.repo
yum install ambari-server -y

# Ubuntu 14 (for Ubuntu 16, replace ubuntu14 with ubuntu16 in the repo URL)
# to test public release 2.5.1
wget -O /etc/apt/sources.list.d/ambari.list http://public-repo-1.hortonworks.com/ambari/ubuntu14/2.x/updates/2.
5.1.0/ambari.list
apt-key adv --recv-keys --keyserver keyserver.ubuntu.com B9733A7A07513CAD
apt-get update
apt-get install ambari-server -y

# SUSE 11 (for SUSE 12, replace suse11 with suse12 in the repo URL)
# to test public release 2.5.1
wget -O /etc/zypp/repos.d/ambari.repo http://public-repo-1.hortonworks.com/ambari/suse11/2.x/updates/2.5.1.0
/ambari.repo
zypper install ambari-server -y
```

Ambari offers many installation options (see Ambari User Guides), but to get up and running quickly with default settings, you can run the following to set up and start ambari-server.

```
ambari-server setup -s
ambari-server start
```

For frontend developers only: see Frontend Development section below for extra setup instructions.

Once Ambari Server is started, hit http://c6801.ambari.apache.org:8080 (URL depends on the OS being tested) from your browser on your local computer. Note that Ambari Server can take some time to fully come up and ready to accept connections. Keep hitting the URL until you get the login page.

Once you are at the login page, login with the default username **admin** and password **admin**.
On the Install Options page, use the FQDNs of the VMs. For example:

```
c6801.ambari.apache.org
c6802.ambari.apache.org
c6803.ambari.apache.org
```

Alternatively, you can use a range expression:

```
c68[01-03].ambari.apache.org
```

Specify the the non-root SSH user **vagrant**, and upload **insecure_private_key** file that you copied earlier as the private key.

Follow the onscreen instructions to install your cluster.

When done testing, run **vagrant destroy -f** to purge the VMs.

Basic VM Operations

vagrant up <vm name>
Starts a specific VM. up.sh is a wrapper for this call.
Note: if you don't specify the <vm name> parameter, it will try to start 10 VMs
You can run this if you want to start more VMs after you already called up.sh
For example: vagrant up c6806

vagrant destroy -f
Destroys all VMs launched from the current directory (deletes them from disk as well)
You can optionally specify a specific VM to destroy

vagrant suspend
Suspends (snapshot) all VMs launched from the current directory so that you can resume them later
You can optionally specify a specific VM to suspend

vagrant resume
Resumes all suspended VMs launched from the current directory
You can optionally specify a specific VM to resume

vagrant ssh host
Starts a SSH session to the host. For example: vagrant ssh c6801

vagrant status
Shows which VMs are running, suspended, etc.

Modifying RAM for the VMs

Each VM is allocated 2GB of RAM. These can be changed by editing **Vagrantfile**. To change the RAM allocation, modify the following line:

```
vb.customize ["modifyvm", :id, "--memory", 2048]
```

Taking Snapshots

Vagrant makes it easy to take snapshots of the entire cluster.
First, install the snapshot plugin:

```
vagrant plugin install vagrant-vbox-snapshot --plugin-version=0.0.2
```

This enables the "vagrant snapshot" command. Note that the above installs vesion 0.0.2. if you install the latest plugin version 0.0.3 does not allow taking snapshots of the whole cluster at the same time (you have to specify a VM name).
Run **vagrant snapshot** to see the syntax.
Note that the plugin tries to take a snapshot of all VMs configured in Vagrantfile. If you are always using 3 VMs, for example, you can comment out c68[04-10] in Vagrantfile so that the snapshot commands only operate on c68[01-03].
Note: Upon resuming a snapshot, you may find that time-sensitive services may be down (e.g, HBase RegionServer is down, etc.)

Tip: After starting the VMs but before you do anything on the VMs, run "vagrant snapshot take init". This way, you can go back to the initial state of the VMs by running "vagrant snapshot go init"; this only takes seconds (much faster than starting the VMs up from scratch by using up.sh or "vagrant up"). Another advantage of this is that you can always go back to the initial state without destroying other named snapshots that you created.

Misc

All VMs launched will have a directory called **/vagrant** inside the VM. This maps to the **ambari-vagrant/<os>** directory on your local computer. You can use this shared directory mapping to push files, etc.

If you want to test OS's other than what's currently in the **ambari-vagrant** repo, please see http://www.vagrantbox.es/ for all the readily available OS images you can test. Note that Ambari currently works on RHEL 5/6, CentOS 5/6, Oracle Linux 5/6, SUSE 11, and SLES 11. Ubuntu support is work in progress.

Kerberos Support

Ambari supports adding Kerberos security to an existing Ambari-installed cluster. First setup any one host as the KDC as follows:

Install the Kerberos server on the chosen host. e.g. for Centos/RedHat

```
yum install krb5-server krb5-libs krb5-auth-dialog rng-tools -y
```

Create the Kerberos database.

```
rngd -r /dev/urandom -o /dev/random
/usr/sbin/kdb5_util create -s
```

Update `/etc/krb5.conf` on the KDC host. e.g. if your realm is `EXAMPLE.COM` and kdc host is `c6801.ambari.apache.org`

```
[realms]
EXAMPLE.COM = {
admin_server = c6801.ambari.apache.org
kdc = c6801.ambari.apache.org
}
```

Restart Kerberos services. e.g. for Centos/RedHat

```
/etc/rc.d/init.d/krb5kdc restart
/etc/rc.d/init.d/kadmin restart
```

Create a KDC admin principal `admin/admin@EXAMPLE.COM` using a password.

```
$ sudo kadmin.local
kadmin.local:  add_principal admin/admin@EXAMPLE.COM
WARNING: no policy specified for admin/admin@EXAMPLE.COM; defaulting to no policy
Enter password for principal "admin/admin@EXAMPLE.COM":
Re-enter password for principal "admin/admin@EXAMPLE.COM":
Principal "admin/admin@EXAMPLE.COM" created.
```

Remember the password for this principal. The Ambari Kerberos Wizard will request it later. Distribute the updated `/etc/krb5.conf` file to remaining hosts in the cluster.

Navigate to *Ambari Dashboard —> Admin —> Kerberos* to launch the Kerberos Wizard and follow the wizard steps. If you run into errors, the Ambari server logs can be found at `/var/log/ambari-server/ambari-server.log`.

Pre-Configured Development Environment

Simply edit **Vagrantfile** to launch a VM with all the tools necessary to build Ambari from source.

```
cd ambari-vagrant/centos6.8
vi Vagrantfile  <- uncomment the line with "dev-bootstrap.sh"
vagrant up c6801
```

To build from source, follow the instructions in Ambari Development.

Frontend Development

You can use this set up to develop and test out Ambari Web frontend code against a real Ambari Server on a multi-node environment.

You need to first fork the apache/ambari repository if you haven't already. Read the How to Contribute guide for instructions on how to fork.

On the host machine:

```
cd ambari-vagrant/centos6.8
# Replace the [forked-repository-url] with your fork's clone url
git clone [forked-repository-url] ambari
cd ambari/ambari-web
npm install
brunch w
```

On c6801 (where Ambari Server is installed):

```
cd /usr/lib/ambari-server
mv web web-orig
ln -s /vagrant/ambari/ambari-web/public web
ambari-server restart
```

With this setup, whenever you change the content of ambari-web files (under ambari-vagrant/ambari/) on the host machine, brunch will pick up changes in the background and update ambari-vagrant/ambari/ambari-web/public. Because of the symbolic link, the changes are automatically picked up by Ambari Server. All you have to do is hit refresh on the browser to see the frontend code changes reflected.

Not seeing code changes as expected? If you have run the maven command to build Ambari previously, you will see files called app.js.gz and vendor.js.gz under the **public** folder. You need to delete these files for the frontend code changes to be effective, as the app.js.gz and vendor.js.gz files take precedence over app.js and vendor.js, respectively.