Flink on Apache BigTop - Project Proposal

Vision and Goals Of The Project:

Bigtop is an Apache Foundation project for comprehensive packaging, testing, and configuration of the leading open source big data components. Bigtop supports a large pallet of components/projects like Hadoop, HBase and Spark. It is aimed at creating an open source and community driven Big Data distribution based on Apache Hadoop and is hence also known as "Fedora of Hadoop".

- The primary goal of Bigtop is to build a community around the packaging, deployment and interoperability testing of Hadoop-related projects. This includes testing at various levels developed by a community with a focus on the system as a whole.
- Build, packaging and integration test code that depends upon official releases of the Apache Hadoop-related projects (example : HDFS, MapReduce, HBase, Hive, Pig, ZooKeeper, etc...) is developed and released.

The goal of this project to provide Apache Flink as a component in the Apache Bigtop stack. A patch will be integrated into the existing development upon successful completion.

1. Scope and Features Of The Project:

Flink on Apache Bigtop

- Automates the java packaging and deployment of Apache Flink onto a cluster.
    - This will be achieved by using Apache's Bigtop framework.
- This is achieved through
    - Packaging of Apache Flink in rpm and deb.
    - Virtualization using vagrant and docker for easy testing of bigtop deployment and smoke tests.
    - Deployment on the cluster.
    - Integration of the test code that depends upon official release of Apache Flink.

1. Solution Concept

Currently, the latest Apache Bigtop v1.0 provides the following Big Data components, including but not limited to Hadoop, that can be easily deployed across the cluster of nodes:

```
:all-components
bigtop 1.0.0 stack includes the following components
        bigtop_groovy            2.3.8
        bigtop_jsvc             1.0.15
        bigtop_tomcat           6.0.36
        bigtop_utils             1.0.0
        crunch                  0.10.0
        datafu                   1.0.0
        flume                    1.5.2
        giraph                   1.1.0
        hadoop                   2.6.0
        hbase                   0.98.12
        hive                     1.0.0
        hue                      3.8.1
        ignite_hadoop            1.2.0
        kafka                   0.8.1.1
        mahout                  0.10.0
        oozie                    4.0.1
        phoenix                  4.3.1
        pig                     0.14.0
        solr                     4.9.0
        spark                    1.3.1
        sqoop                    1.4.5
        sqoop2                  1.99.4
        tachyon                  0.6.0
        tez                      0.6.0
        zookeeper                3.4.6
```

Our task is to provide Apache Flink as a packaged component that can be deployed as a part of the Hadoop cluster using Apache Bigtop. The provided solution would be a patch on top of the existing Apache Bigtop implementation.

Global Architectural Structure Of the Project and a Walkthrough:

There are four major steps in the process of deploying the Hadoop packages via Bigtop:

1) Packaging of the Hadoop ecosystem into rpm or debian
2) Integration testing of the framework
3) Virtualization using vagrant or docker based hadoop provisioning
4) Deployment over the cluster via puppet recipes for the included Hadoop components.

Below are the main components that aid in creating and deploying Hadoop packages across a distributed cluster using Apache Bigtop:

- Maven

- Gradle
- Puppet
- Vagrant and VirtualBox

1) Packaging the components using Gradle and Maven: One of the longer term goals is to have Bigtop packages serve as as reference to eventually get Hadoop introduced into most Linux distributions. Currently, this is problematic owing to the lack of maven3 native packaging for many major Linux distributions. However, Bigtop makefiles and other artifacts, create an alternative build root, which then uses native rpm or .deb packaging tools to create packages.Gradle is used to test the above built rpm and deb packages and identify the errors. Finally the smoke tests are added for the built packages.

2) Launching Virtual Machines using Vagrant and VirtualBox: Vagrant uses a preconfigured box image of the operating system such as CentOS or the Debian and powers up a virtual machine using the Virtualbox or other provisioning providers like AWS and Docker (refer Fig 1). Vagrant manages the lifecycle of the launched VM and allows us to create or destroy the instances. Vagrant allows to SSH into the VM and provides a terminal based access into the running virtual system. Also the Vagrant home directory is synced with the virtual machine which allows for networking and sharing resources between the host and the guest OS.

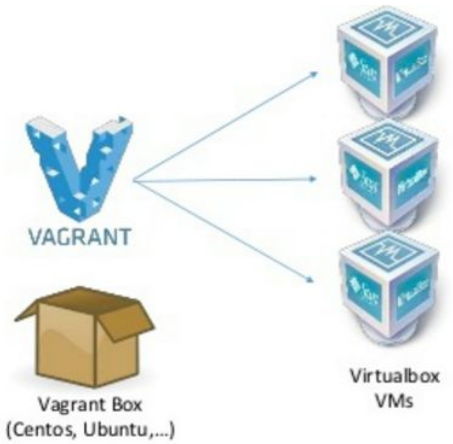

Fig 1

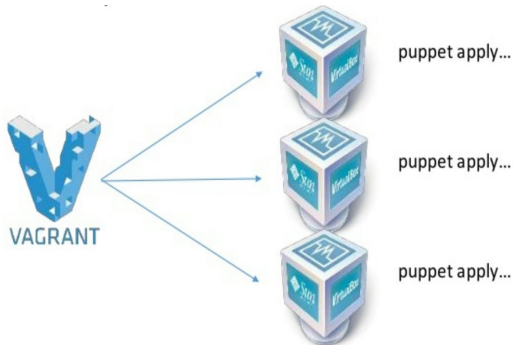

Fig 2

During the initialisation of Vagrant a vagrantfile is created which serves as the configuration file and tells Vagrant with what properties the virtual image is to be launched such as nodes, processors, memory etc. All the above parameters and the components to be loaded into the boxed image can be specified in another file called

"vagrantconfig.yaml" file which is easily reconfigurable and aids the users to set the following properties. The contents of this file are finally fetched into the vagrantfile as parameters, which then sets up the properties.

```
memory_size: 4096
number_cpus: 1
box: "puppetlabs/centos-7.0-64-nocm"
repo: "http://bigtop-repos.s3.amazonaws.com/releases/1.0.0/centos/7/x86_64"
num_instances: 1
distro: centos
components: [hadoop, yarn]
enable_local_repo: false
run_smoke_tests: false
smoke_test_components: [mapreduce, pig]
jdk: "java-1.7.0-openjdk-devel.x86_64"
```

These properties are utilised by vagrant and Puppet to determine what all components are to be installed into the individual nodes, in this case Hadoop.

3) Using Puppet as a provisioner on each of the cluster nodes: The vagrantfile writes a shell script out on each node that runs locally and then uses "puppet-apply" to install components (refer Fig 2).
The puppet classes for Bigtop deployment setup includes the following tasks:

1) service installation
2) pointing slaves to masters
3) starting the services

The puppet-apply is a standalone puppet execution tool that is used to apply individual manifests

```
puppet apply --modulepath=<path_to_bigtop> -e "include
bigtop_toolchain::puppet-modules"
```

4) Going into the the node terminals: Once the puppet and vagrant recipes are set to work, it is assured that the nodes in the cluster are installed with the same components throughout as required by the user (refer Fig 3).

By issuing a single command "vagrant up", a user can spin or re-spin the entire cluster from scratch and start using the worker nodes by just "vagrant ssh" command.

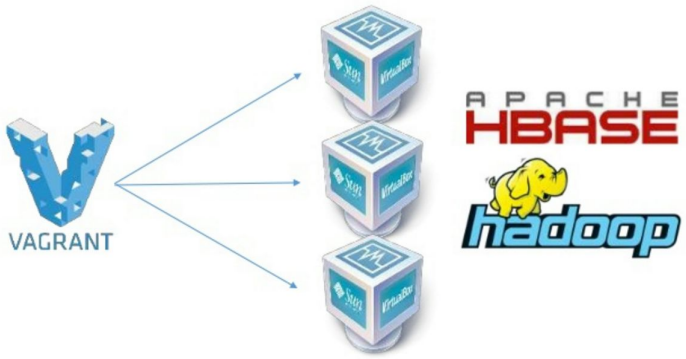

Fig 3

Release Planning

Release #1 (due by Week 2):
Build the rpm and debian packages.

Release #3 (due by Week 6):
Setup A DEBIAN and CentOS build server for bigtop on the cloud. It should be able to successfully run "spark-deb" and write packages to output/ dir.

Release #4 (due by Week 10):
Deliver vagrant and puppet recipes to package Apache Flink.

Release #5 (due by Week 13):
Test the packaged Flink by deploying it as a component on hadoop cluster on VM or on the services like docker and AWS.