Automated Deployment of OpenStreetMap Data in Apache Spark Cluster

Rebecca Appelbaum, Colin McKibben, Thinh Vu

June 1, 2015

1 Overview

The goal of this project is to automate the task of creating clusters of VMs on OpenStack and installing Apache Spark on the clusters. We also looked to import OpenStreetMap data and utilize the speed of Spark to analyze this data. We took a DevOps approach when planning our implementation and utilized cloudmesh and ansible.

2 Assumptions

1. User already has cloudmesh installed

3 Instructions

1. Make a copy of our project library from github. It can be found at

   `https://github.com/futuresystems/465-project-mckibbenc-rebecca-appelbaum-imthinhvu`

   - Example command to copy library:

     ```
     git clone git@github.com:futuresystems/465-mckibbenc-rebecca-appelbaum-imthinhvu.git
     ```

2. Create a file called config in your home ssh directory. Write the following in the config file:

   - `StrictHostKeyChecking=no`
   - This file will disable strict host key checking. The user could chose to apply this to all hosts or you can specify the hosts that you want this to apply to. We chose to incorporate this so that users will not have any manual entry when implementing our cm command.

3. Change directories into our project folder.

   ```
   cd 465-project-mckibenc-rebecca-appelbaum-imthinvu/cloudmesh\
       _spark
   ```

4. Run the following command:

   - `Python setup.py install`

5. Install the ansible by running the ansible shell script. Run the script below. If you are running in a virtualenv, please install Ansible via: pip install ansible.

   - `sudo sh ansible/install-ansible.sh`

6. Export the cm command with the following

   ```
   CM\_SPARK\_DIR=/home/ubuntu/465-project/mckibbenc-rebecca-
       appelbaum-imthinhvu/cloudmesh\_spark
   ```

7. Before running the cm spark command, create an ssh agent. This will allow you to avoid typing in your password multiple times.

   - `eval $(ssh-agent)`
   - `ssh-add`

8. Deploy clusters and have spark installed on each cluster. The command will automatically deploy 3 clusters. However this can be customized by adding –count=N where N is the number of clusters you want deployed. Additional documentation can be found if you write cm spark –help.

   - `cm deploy spark example`

9. You can also select the node that you want as the master by running a start command with the cluster that you want to be the master.
    - `Ex. cm spark start example_1`
10. If you would like to test any of the clusters you can login using your public ip address. It would look something like `"ssh ubuntu@149.158.213.56". If you go cd` you can run scripts on the clusters. Use the below code to see how many nodes and railways there are in Munich Germany according to OpenStreetMaps.
    - `~/spark-1.3.1-bin-hadoop2.6/bin/spark-submit`
    - `--master local[4]`
    - `Osm.py`
11. To destroy the clusters you can run
    - `cm spark destroy example`

4 Troubleshooting

1. If your nodes are not running, there may be an issue with how the /etc/hosts file was generated on each node. To fix, login to each node, and open up the /etc/hosts file to edit as mentioned below:
    - `sudo vi /etc/hosts`
    - Remove text from the top of the file up until it reaches 127.0.0.1 localhost, your file should begin with this line. Then try re-running the "cm spark start" command again to start your Spark cluster.