ILLINOIS DATA SCIENCE INITIATIVE

TECHNICAL REPORT

Setting up Ambari on Openstack Nebula

Author:
Quinn Jarrell

March 1, 2017

Setting up Ambari on Openstack Nebula

QUINN JARRELL[1] **AND PROFESSOR ROBERT J. BRUNNER**[2]

[1] *National Center For Supercomputing Applications (NCSA)*
[2] *Laboratory for Computation, Data, and Machine Learning*

Compiled March 1, 2017

https://github.com/lcdm-uiuc

INTRODUCTION

This technical report will guide you through running Ambari on top of an Openstack Nebula environment. We refer to individual virtual machines as nodes. The included scripts used within this report are all included in the scripts directory under the name openstack_setup.py.

SETTING UP THE NETWORK FOR CENTOS 6

Ambari requires each node in the cluster to know every other node's domain name. There are two ways of doing this, either through a Domain Name Server(DNS) or through editing a file called hosts. The hosts file is a slightly easier but more rigid method as it does only works well with a small private network (Less than 10000 hosts). To create the hosts file we generate a mapping like

192.168.0.0 192-168-0-0.local

With dots replaced with dashes, it is an acceptable domain name and the .local is an unused suffix which can be changed to anything. We translate every address within the local subnet. The 192.168.x.x is dedicated to a local area network so it will not affect requests to other IPs. It should be noted that this requires an entirely private network. If you are running on a shared network, you should only map the private IP addresses of the nodes on your cluster and check that they do not change.

Hosts File Example

So our hosts file ends up looking this:

```
127.0.0.1   localhost localhost.localdomain
127.0.0.1   localhost4 localhost4.localdomain4
192.168.100.0 192-168-100-0.local
192.168.100.1 192-168-100-1.local
192.168.100.2 192-168-100-2.local
192.168.100.3 192-168-100-3.local
192.168.100.4 192-168-100-4.local
192.168.100.5 192-168-100-5.local
192.168.100.6 192-168-100-6.local
192.168.100.7 192-168-100-7.local
192.168.100.8 192-168-100-8.local
192.168.100.9 192-168-100-9.local
::1         localhost localhost.localdomain
::1         localhost6 localhost6.localdomain6
```

With lots of more similar lines for the rest of the local subnet.

PREPARING SERVERS

We use the tool Fabric to provision servers. We assume you have installed Fabric on one of the computers in the cluster and configured it.

From here you can run the following lines to have ambari fully installed on a new Centos 6 node

```
fab -f setup-fqdn.py \
install_packages copy_hosts_file fix_fqdn \
disable_huge_pages disable_firewall turn_on_ntp \
disable_selinux import_ambari_repo \
-H $NODE1,$NODE2,$NODE3

fab -f setup-fqdn.py \
setup_ambari_server \
-H $MASTERNODE
```

where $NODEn is the IP or hostname of each node in your cluster and $MASTERNODE is the node you want to run the Ambari web interface on. The $MASTERNODE should also be one of $NODEn as it requires the same resources as the nodes.

FABRIC FUNCTIONS

This section will explain each Fabric function. If you have not run the previous command then you will need to run each command below in order.

Installing Packages

Running the install_packages function installs Wget, ntpdate, ntp, epel-release, and tmux.

```
fab -f setup-fqdn.py \
install_packages \
-H $NODE1,$NODE2,$NODE3
```

For the rest of the report we assume you have installed these packages.

Copying Host File to Remote Nodes

Each node in the cluster requires the same Hosts file in order to match IP to FQDN. The function copy_hosts_file will copy the file `ip_to_host` from the current directory to `/etc/hosts` on each node.

```
fab -f setup-fqdn.py \
copy_hosts_file \
-H $NODE1,$NODE2,$NODE3
```

Fixing FQDN

Ambari requires the Fully Qualified Domain Name (FQDN) to be set for each node in the cluster. A FQDN is a domain name ending in a .suffix. The hosts file set up previously implicitly defines an FQDN for every node in the cluster. To make each node in the cluster aware of it's own FQDN, run the fix_fqdn function with the host names of the computers in your cluster.

```
fab -f setup-fqdn.py \
fix_fqdn \
-H $NODE1,$NODE2,$NODE3
```

where $NODEn is the IP or hostname of each node in your cluster.

Disabling Huge Pages

Transparent huge pages allow applications to allocate very large amounts of memory very quickly. Unfortunately this memory allocation sometimes causes the Linux kernel to move memory around a lot. This reduces Spark's and Hadoop's thoroughput because they request large allocations very often. To disable them, run the `disable_huge_pages` function

```
fab -f setup-fqdn.py \
disable_huge_pages \
-H $NODE1,$NODE2,$NODE3
```

Disabling Firewalls

We turn off the firewall for all nodes within the cluster. We suggest implementing the firewall with Openstack security groups instead. See the Openstack configuration report for details. To turn the firewall off run the disable_firewall function with the host names of the computers in your cluster.

```
fab -f setup-fqdn.py \
disable_firewall \
-H $NODE1,$NODE2,$NODE3
```

Enabling NTP

Hadoop, Spark and many other services provided by Ambari require as accurate time as possible. We install and sync the nodes explicitly during setup through the Network Time Protocol (NTP) service.

```
fab -f setup-fqdn.py \
turn_on_ntp \
-H $NODE1,$NODE2,$NODE3
```

Disabling SELinux

SELinux is enabled by default for CentOS 6. It provides a system of access security for the kernel. Unfortunately many application including Ambari do not work well with SELinux. To disable it run

```
fab -f setup-fqdn.py \
disable_selinux \
-H $NODE1,$NODE2,$NODE3
```

Importing the Ambari Repository

The Hortonworks company provides public Ambari package repositories for the package manager Yum. We use these packages as a base to avoid recompiling Ambari from source. To download the repository, run

```
fab -f setup-fqdn.py \
import_ambari_repo \
-H $NODE1,$NODE2,$NODE3
```

Setting up Ambari Server

Finally we need to set up a server on *one* of the nodes in the cluster. This node should have all of the above functions run on it already. This node will operate the web interface to manage your cluster. It will also most likely be the only node accessible from the outside internet.

```
fab -f setup-fqdn.py \
setup_ambari_server \
-H $MASTERNODE
```

where $MASTERNODE is an IP address or FQDN of one of the nodes in your cluster.