Apache Ambari

Introduction

The Apache Ambari project is aimed at making Hadoop management simpler by developing software for provisioning, managing, and monitoring Apache Hadoop clusters. Ambari provides an intuitive, easy-to-use Hadoop management web UI backed by its RESTful APIs.

Ambari enables System Administrators to:

- Provision a Hadoop Cluster

Ambari provides a step-by-step wizard for installing Hadoop services across any number of hosts. Ambari handles configuration of Hadoop services for the cluster.

- Manage a Hadoop Cluster

Ambari provides central management for starting, stopping, and reconfiguring Hadoop services across the entire cluster.

- Monitor a Hadoop Cluster

Ambari provides a dashboard for monitoring health and status of the Hadoop cluster. Ambari leverages Ambari Metrics System (https://issues.apache.org/jira/browse/AMBARI-5707) for metrics collection. Ambari leverages Ambari Alert Framework (https://issues.apache.org/jira/browse/AMBARI-6354) for system alerting and will notify you when your attention is needed (e.g., a node goes down, remaining disk space is low, etc).

Ambari enables Application Developers and System Integrators to:

Easily integrate Hadoop provisioning, management, and monitoring capabilities to their own applications with the Ambari REST APIs (https://github.com/apache/ambari/blob/trunk/ambari-server/docs/api/v1/index.md).

Installation

They are two ways to install ambari, you can download the source and build it by yourself, you can also use the hortonworks distribution yum repo.

You can find out how to build ambari from source here: Build from source

Before installation

set up ssh access

The host which has cloudera manager installed need ssh access and sudo right to install required

services on each node (master, slave, etc)

```
# 1. Create user and group on all hosts
groupadd -g 42030 hadoop

useradd hadoop --uid 100068 --home /home/hadoop --create-home --gid hadoop

# 2. Generate ssh key pair on the cloudera manager host
sudo su hadoop
ssh-keygen -t rsa

# 3. copy the public key into all other nodes
mkdir -p ~/.ssh
vim ~/.ssh/authorized_keys
chmod 0600 ~/.ssh/authorized_keys

# 4. edit the /etc/ssh/sshd_config
AllowUsers hadoop
```

Note that the user name, uid, gid can be changed as you want, you don't need to use the same as in the example.

configure network

Configure hostname of each node in the cluster to ensure that all members can communicate with each other.

```
# 0. get current hostname
hostnamectl status

# 1. set the hostname to a unique name (fqdn)
sudo hostnamectl set-hostname cloudera01.pengfei.org

# 2. Edit /etc/hosts for all nodes to be aware of all other nodes

1.1.1.1  lin01.pengfei.org  lin01
2.2.2.2  lin02.pengfei.org  lin02
3.3.3.3  lin03.pengfei.org  lin03
4.4.4.4  lin04.pengfei.org  lin04

# 3. Edit /etc/sysconfig/network with the fqdn of the host
HOSTNAME=lin01.pengfei.org

# 4. Verify that each host consistently identifies to the network
# a. Run uname -a and check that the hostname matches the output of the
hostname command
# b. Run /sbin/ifconfig and note the value of inet addr in the eth0 (or
bond0) entry, for example:
```

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.6.16  netmask 255.255.255.0  broadcast 172.17.6.255
        inet6 fe80::f816:3eff:fe1f:f6ff  prefixlen 64  scopeid 0x20<link>
        ether fa:16:3e:1f:f6:ff  txqueuelen 1000  (Ethernet)
# c. Run host -v -t A $(hostname) and verify that the output matches the
hostname command.
#    The IP address should be the same as reported by ifconfig for eth0 (or
bond0): for example

[pliu@cclindwcloudera01 ~]$ host -v -t A cclindwcloudera01.in2p3.fr
Trying "cclindwcloudera01.in2p3.fr"
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 49519
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;cclindwcloudera01.in2p3.fr.    IN    A

;; ANSWER SECTION:
cclindwcloudera01.in2p3.fr. 0   IN    A    172.17.6.16

Received 60 bytes from 172.17.6.250#53 in 1 ms
```

Note Important:

- The canonical name of each host in /etc/hosts must be the FQDN (for example myhost-1.example.com), not the unqualified hostname (for example myhost-1). The canonical name is the first entry after the IP address.
- Do not use aliases, either in /etc/hosts or in configuring DNS.
- Unqualified hostnames (short names) must be unique. For example, you cannot have both host01.example.com and host01.standby.example.com managed by the same Cloudera Manager Server.
- FQDN can not contain _ (e.g. host_01.example.com is not allowed)

disabling firewall

To disable the firewall on each host in your cluster, perform the following steps on each host.

```
# if you are using iptables
# Save the existing iptables rule set.
iptables-save > /root/firewall.rules
# stop service from starting at reboot
sudo chkconfig iptables off
# stop service
sudo service iptables stop

# if you are using firewall-cmd
systemctl stop firewalld
```

disabling selinux

```
# check the selinux state
getenforce

# if the output is enforcing, you need to change to permissive or disabled
vim /etc/selinux/config

SELINUX=disabled
# The above config take effect only after system restart
# disable immediately
setenforce 0
```

Install from yum repo

There is an official doc.
https://docs.cloudera.com/HDPDocuments/Ambari-2.7.4.0/bk_ambari-installation/content/install-ambari-server-rhel7.html

In this tutorial, we use the ambari which are pre-compiled by hortonworks. The current version (10/10/2018) is 2.7.4.0

```
# You can also follow the horton works doc to do the installation
https://docs.cloudera.com/HDPDocuments/Ambari-2.7.4.0/bk_ambari-installation
/content/download_the_ambari_repo_lnx7.html

#Login to your host as root

#Download repo to local host
wget -nv
http://public-repo-1.hortonworks.com/ambari/centos7/2.x/updates/2.7.4.0/amba
ri.repo -O /etc/yum.repos.d/ambari.repo

# Important !!!
# Do not modify the ambari.repo file name. This file is expected to be
available on the Ambari Server host during Agent registration.

# check repo list
yum repolist

# Install ambari server
yum install ambari-server

#configure and start server
ambari-server setup

#During the setup step, you need to choose:
```

```
# 1. which user will run the ambari-server daemon(default is root)
# 2. jdk
# 3. database
# 4. init database
Database admin user (postgres):
Database name (ambari):
Postgres schema (ambari):
Username (ambari):
Enter Database Password (ambari):

# if all above is done correctly, you should see the following line
#Ambari Server 'setup' completed successfully.
ambari-server start
```

Create hadoop cluster with ambari

When you create cluster with ambari, you need to the following steps:

Step 1. Enter all node (ip/fqdn) to ambari. So ambari can install ambari agent on them. All these node must have an account which allows ambari server to do ssh on them. These account must have sudo right on these node.

```
# For example, I create a user hadoop on ambari and cluster nodes
# In ambari server, we have the ssh private key, in cluster nodes we have
the public key stored in .ssh/authorized_keys

# If the hostnames of these nodes do not exit in the dns, you need to add ip
and hostnames in the /etc/hosts of ambari server and cluster nodes.

# To set hostname without restart, use the following command

# sethostname
hostnamectl set-hostname myserer.pengfei.org

#show the hostname
hostname -f
```

Step 2 : choose which service you want to run on the cluster.

Base on the resource of each node, ambari will choose which one is mater node or worker node. In the master node, it will install all the master service such as hdfs namenode, journale node, snamenode(standby), hive server, spark master, etc.

In the worker node will install datanode, spark-slaves, etc.

To see how to setup cpu, memory, java heap memory.

Check this page

https://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.5.3/bk_command-line-installation/content/determine-hdp-memory-config.html

Step 3 : Configure each service We need to set up configuration for all services. Almost all of them requires a database.

For example, the following command shows how to setup database for druid, hive

```
# login to psql, create user, database, and grant privilege for hive
create user hive with password 'changeMe';
CREATE DATABASE metastore OWNER hive ENCODING 'UTF8';
```

Ambari will install the chosen service automatically.

The following list shows main services which ambari will install on the cluster.

- Apache Accumulo : is a highly scalable sorted, distributed key-value store based on Google's Bigtable. It is a system built on top of Apache Hadoop, Apache ZooKeeper, and Apache Thrift. Written in Java, Accumulo has cell-level access labels and server-side programming mechanisms. According to DB-Engines ranking, Accumulo is the third most popular NoSQL wide column store behind Apache Cassandra and Hbase and the 61st most popular database engine of any type as of 2018.
- Oozie : is a workflow scheduler system to manage Apache Hadoop jobs. Oozie Workflow jobs are Directed Acyclical Graphs (DAGs) of actions. Oozie Coordinator jobs are recurrent Oozie Workflow jobs triggered by time (frequency) and data availability. Oozie is integrated with the rest of the Hadoop stack supporting several types of Hadoop jobs out of the box (such as Java map-reduce, Streaming map-reduce, Pig, Hive, Sqoop and Distcp) as well as system specific jobs (such as Java programs and shell scripts). Oozie is a scalable, reliable and extensible system.
- Druid is primarily used to store, query, and analyze large event streams. Examples of event streams include user generated data such as click streams, application generated data such as performance metrics, and machine generated data such as network flows and server metrics. Druid is optimized for sub-second queries to slice-and-dice, drill down, search, filter, and aggregate this data. Druid is commonly used to power interactive applications where performance, concurrency, and uptime are important.
- Apache Knox Gateway is an Application Gateway for interacting with the REST APIs and UIs of Apache Hadoop deployments. The Knox Gateway provides a single access point for all REST and HTTP interactions with Apache Hadoop clusters.
- Apache Atlas is designed to exchange metadata with other tools and processes within and outside of the Hadoop stack, thereby enabling platform-agnostic governance controls that effectively address compliance requirements
- SmartSense (product hortonworks). It can diagnostic cluster, give recommendation of cluster configuration, which can boost cluster performance

Cluster security

hdfs

To upload/download data into/from hdfs, you can use

- Ambari web interface / file views
- hdfs cli (hdfs dfs …)

During upload or download, if you encounter errors " Unauthorized connection for super-user: root from IP", you need to do the following config changes.

```
Do the following step :
# 1. In Ambari Web, browse to Services > HDFS > Configs.
# ### Under the Advanced tab, navigate to the Custom core-site section.
# ### Click Add Property… to add the following custom properties:
hadoop.proxyuser.root.groups=*
hadoop.proxyuser.root.hosts=*

# 2. Notice the ambari-server daemon account name root is part of the
property name. Be sure to modify
# this property name for the account name you are running the ambari-server
as. For example, if you were
# running ambari-server daemon under an account name of ambariusr, you would
use the following properties instead:
hadoop.proxyuser.ambariusr.groups=*
hadoop.proxyuser.ambariusr.hosts=*

# 3.Similarly, if you have configured Ambari Server for Kerberos, be sure to
modify this property name for
# the primary Kerberos principal user. For example, if ambari-server is
setup for Kerberos using principal
# ambari-server@EXAMPLE.COM, you would use the following properties instead:
hadoop.proxyuser.ambari-server.groups=*
hadoop.proxyuser.ambari-server.hosts=*

Save the configuration change and restart the required components as
indicated by Ambari.
```

Hdfs use linux like file acl, user/group/other, when ambari web ui upload data, the user of ambari must have correct acl to write in the dir.

For example. /user/hdfs hdfs:hdfs rwxr-r-, as admin user of ambari, admin can't create new dir or upload data in /user/hdfs.

/user/admin admin:hdfs rwxr-r-, as admin user of ambari, admin can create new dir or upload data in /user/admin

Start/Stop ambari server/agent

```
1. Stop ambari-server
```

```
> ambari-server stop (if you see permission denied, su to the right user)
2. Stop ambari-agent service on all nodes
> ambari-agent stop
3. Start ambari-agent service on all nodes
> ambari-agent start
4. Start ambari-server server
> ambari-server start

Ps. they also support ambari-server/agent restart
```

Default installation path for services nodes and worker nodes

```
# The file path of all services currently used in the datalake, which are
all synbolick links
/usr/hdp/current

# Examples of these symbolic links
lrwxrwxrwx 1 root root   27 Oct  2 14:35 hadoop-client ->
/usr/hdp/3.1.4.0-315/hadoop
lrwxrwxrwx 1 root root   32 Oct  7 14:44 hadoop-hdfs-client ->
/usr/hdp/3.1.4.0-315/hadoop-hdfs
lrwxrwxrwx 1 root root   32 Oct  2 14:35 hadoop-hdfs-datanode ->
/usr/hdp/3.1.4.0-315/hadoop-hdfs
lrwxrwxrwx 1 root root   32 Oct  2 14:35 hadoop-hdfs-journalnode ->
/usr/hdp/3.1.4.0-315/hadoop-hdfs
lrwxrwxrwx 1 root root   32 Oct  2 15:15 hadoop-hdfs-namenode ->
/usr/hdp/3.1.4.0-315/hadoop-hdfs
lrwxrwxrwx 1 root root   32 Oct  2 14:35 hadoop-hdfs-nfs3 ->
/usr/hdp/3.1.4.0-315/hadoop-hdfs


# The file path of all installed service, which are grouped with hdp
versions, you can many hdp versions installed.
/usr/hdp/3.1.4.0-315
```

Manage user account

You can add new users via the Ambari web GUI. Click on your login which is on the top right of the window. You should see a list of choice which contains **Manage Ambari**, click on it. You should see a new page, which contains a button users(on the left side). Click on it, you should see a page which allows you to add new users or modify passwords.

Common problems

C1.database jdbc connector jar not found

Ambari does not provide any jdbc driver for any database, so you have to download them and upload them to ambari server.

The following example shows how to install a mysql jdbc driver. For others, it's the same procedure, just download the different jdbc driver.

https://dev.mysql.com/downloads/connector/j/

Go to this page download the jar file

mysql-connector-java-8.0.11.tar.gz

```
# Then put the jar file in /usr/share/java/mysql-connector-java.jar

# if you want put jar file somewhere else, you'd better create a symbolic
link here

# for example  ls -l  /usr/share/java/mysql-connector-java.jar
lrwxrwxrwx 1 root root 31 Apr 19  2017 /usr/share/java/mysql-connector-
java.jar -> /home/pliu/Downloads/mysql-connector-java-8.1.10.jar

# Then update the ambari server with the new jar
ambari-server setup --jdbc-db=mysql --jdbc-driver=/usr/share/java/mysql-
connector-java.jar
# you should see a success page, you can check the jar file is in the
ambari-server resources repo.
ls -l /var/lib/ambari-server/resources/mysql-connector-java.jar
-rw-r--r-- 1 root root 819803 Sep 28 19:52 /var/lib/ambari-
server/resources/mysql-connector-java.jar
```

postgresql java connector

For postgresql jdbc connector, you can download here https://jdbc.postgresql.org/download.html

C2. Can't register server node in ambari

```
# if you can't register your server node and with message errors:
# NetUtil.py:96 - EOF occurred in violation of protocol (_ssl.c:579)
https://cclindwclustermanager.in2p3.fr:8440/ca failed

To solve this problem simply configure the Ambari Agent to use TLSv1.2 when
communicating with the Ambari Server by editing each Ambari Agent’s
/etc/ambari-agent/conf/ambari-agent.ini file and adding the following
```

```
configuration property to the security section:

[security]
force_https_protocol=PROTOCOL_TLSv1_2
```

C3. YARN Registry DNS Start failed

```
# When you start your YARN Registry DNS, you see the following errors
 Execution of 'ulimit -c unlimited; export
HADOOP_LIBEXEC_DIR=/usr/hdp/3.0.1.0-187/hadoop/libexec &&
/usr/hdp/3.0.1.0-187/hadoop-yarn/bin/yarn --config
/usr/hdp/3.0.1.0-187/hadoop/conf --daemon start registrydns' returned 1.
mesg: ttyname failed: Inappropriate ioctl for device
ERROR: Cannot set priority of registrydns process 30580

# java.net.BindException: Problem binding to [node.b2be.com:53]
java.net.BindException: Address already in use; For more details see:
http://wiki.apache.org/hadoop/BindException
 at sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Meth
```

This is caused by the default binding port 53 of yarn registry dns has already been used in the server.

You can check this by using

```
# check the port status
netstat -tnlpa | grep 53
```

To resolve this, you need to change the default value in yarn configs.

Check the value in your YARN configs

“hadoop.registry.dns.bind-port”. (registry.dns.bind-port) default value = 53 change it to a port value which is not used.

C4. Ambari files view null pointer exception

The default hdfs file view created inside ambari raises a null-pointer exception when the user opens it.

Cause : The ambari files view requires a Temporary HDFS Directory to run (default path: /user/${user}/files-view/tmp). For example, If you login as “admin” in ambari, when you open the files view, ambari will create a dir files-view/tmp in /user/admin. So if /user/admin does not exit. You will receive this error.

Solution:

```
# go to your namenode server, use hdfs cli to do following:
hdfs dfs -mkdir /user/admin
hdfs dfs -chown admin:hdfs /user/admin
```

If you still have errors, you need to check your files view configs.

```
# Admin-> Manage Ambari -> Views -> Edit files views config

# If you can't modify it, just delete it and create a new
```