****

Tutorial

**BIG DATA** for Developers

Parte 2: Apache Hadoop hands-on

En la primer parte de este tutorial se presentó el framework **Apache Hadoop** junto a las tecnologías *core* que lo conforman: HDFS (Hadoop Distributed File System) y MapReduce.

En la presente entrega se explicará cómo instalar, configurar e iniciar Hadoop en una VM.

Instalar e iniciar Apache Hadoop es sencillo. Sin embargo, dado que el proceso de instalación requiere la edición de múltiples archivos de configuración es conveniente asegurarse de verificar cada paso.

**Requisitos**

Conocimiento básico de:

* GNU+Linux y shell Bash
* Networking (básico)
* ssh / PuTTY

**Credenciales**

Administrador: ***root*** / ***root***

Usuario: ***training*** / ***training***

Comando *sudo* habilitado.

**Herramientas instaladas**

El hands-on se realizará en una VM Virtualbox corriendo CentOS 6.6 64-Bits.

El entorno de desarrollo consta de las siguientes herramientas instaladas:

* Git 1.7.1
* Java JDK 1.8.0\_25-b17
* Maven 3.2.3
* MySQL 5.1.73 (pwd root: ***root***. BBDD asegurada vía script mysql\_secure\_installation)
* Python 2.6.6 (versión predeterminada, invocar mediante ‘python’)
* Python 2.7.8 (invocar mediante ‘python2.7’) / setuptools: 1.4.2 / pip: 1.5.6 / virtualenv: 1.11.6

También, para comodidad operativa se ha instalado:

* man 1.6f
* Emacs 23.1.1
* Vim 7.2.1-411
* nano 2.0.9
* htop 1.0.3
* GNU Screen 4.00.03
* tmux 1.6
* mc 4.7.0.2
* most (paginador) 4.10.2

**Paso 1: preparación**

Comprobamos que la máquina virtual está correctamente instalada:

$ vboxmanage list vms

"Apache Hadoop hands-on" {baa0ea90-7c44-4fd7-a5cb-c41cc9783bc5}

La Iniciamos desde la interfaz gráfica de Virtualbox o desde el shell:

$ vboxheadless -startvm "Apache Hadoop hands-on" -vrde off &

Esperamos algunos segundos a que inicie y nos logueamos con las credenciales **training**/**training**.

Para mayor comodidad al trabajar en la VM es aconsejable usar ***ssh*** o ***PuTTY***, en este caso vía loopback (**127.0.0.1**) en el puerto **2222** como se muestra en la siguiente captura:



**Paso 2: instalar Apache Hadoop**

**2.1. Java**

Una vez logueados en el entorno de testing lo primero que haremos es verificar la versión de Java instalada:

$ java -version  
java version "1.8.0\_25"  
Java(TM) SE Runtime Environment (build 1.8.0\_25-b17)  
Java HotSpot(TM) 64-Bit Server VM (build 25.25-b02, mixed mode)

**2.2. Crear y configurar llaves ssh**

Hadoop utiliza ssh para acceder de forma segura a sus nodos lo cual normalmente requiere que el usuario ingrese una contraseña. Existen varios métodos para agilizar la administración vinculada a llaves ssh y solucionar este inconveniente, entre ellos:

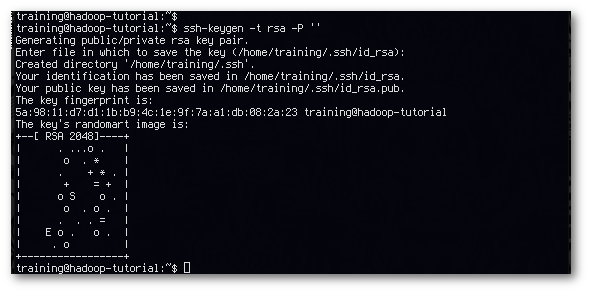


* Crear una llave ssh sin contraseña y distribuirla entre los nodos - METODO INSEGURO.
* Utilizar un llavero ssh a través de un agente gestor como askpass o keychain.
* Configurar el cliente ssh para tal fin (/etc/ssh/ssh\_client, ~/.ssh/config)

|  |
| --- |
| *Emplearemos llaves sin contraseña a los efectos de demostrar el funcionamiento de Hadoop, sin embargo esta no es una práctica aconsejable en entornos de producción.*  *La seguridad de infraestructura está fuera del alcance de este tutorial por lo que no profundizaremos en el tema.* |

Generamos el par SSH: $ ssh-keygen -t rsa -P '' (Verificar que al copiar el comando se utilicen las comillas simples).

Al ejecutar esta instrucción se nos pedirá que ingresemos un nombre de archivo, dejarlo en blanco y pulsar <ENTER>:



Una vez contamos con la llave ssh la agregamos al listado de llaves autorizadas:

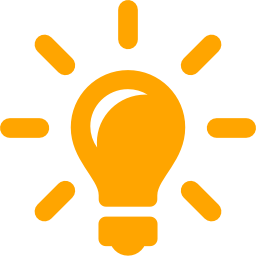
$ cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys

**2.3. Obtener e instalar Apache Hadoop**

Hadoop no se encuentra empaquetado en los repositorios oficiales de CentOS por lo que debemos obtenerlo desde alguno de sus mirrors e instalarlo manualmente:

$ wget <http://apache.dattatec.com/hadoop/common/hadoop-2.6.0/hadoop-2.6.0.tar.gz>





Si se interrumpe la descarga podemos intentar continuarla utilizando el flag **-c** (--continue) de wget:

$ wget -c <http://apache.dattatec.com/hadoop/common/hadoop-2.6.0/hadoop-2.6.0.tar.gz>

Una vez descargado el paquete extraemos Hadoop con el siguiente comando:

$ tar xfz hadoop-2.6.0.tar.gz

El comando extraerá los archivos en el directorio *hadoop-2.6.0*. A los efectos de este tutorial moveremos el directorio extraído a /usr/local/hadoop con el siguiente comando:

$ sudo mv hadoop-2.6.0 /usr/local/hadoop

**Paso 3: configurar Apache Hadoop**

Para completar la puesta a punto de Hadoop es necesario modificar los siguientes archivos:

*~/.bashrc*

*/usr/local/hadoop/etc/hadoop/hadoop-env.sh*

*/usr/local/hadoop/etc/hadoop/core-site.xml*

*/usr/local/hadoop/etc/hadoop/yarn-site.xml*

*/usr/local/hadoop/etc/hadoop/mapred-site.xml.template*

*/usr/local/hadoop/etc/hadoop/hdfs-site.xml*

**3.1. ~/.bashrc**

Para comenzar es necesario saber el valor de la variable JAVA\_HOME ya que la utilizaremos en nuestro archivo de configuración del shell Bash:

$ echo $JAVA\_HOME

/usr/java/jdk1.8.0\_25  
[training@hadoop-training ~]$

En este caso el path completo que nos devuelve el comando es: /usr/java/jdk1.8.0\_25

El próximo paso es abrir ~/.bashrc para edición y agregar:

#HADOOP VARIABLES START

export JAVA\_HOME=/usr/java/jdk1.8.0\_25

export HADOOP\_INSTALL=/usr/local/hadoop

export PATH=$PATH:$HADOOP\_INSTALL/bin

export PATH=$PATH:$HADOOP\_INSTALL/sbin

export HADOOP\_MAPRED\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_HOME=$HADOOP\_INSTALL

export HADOOP\_HDFS\_HOME=$HADOOP\_INSTALL

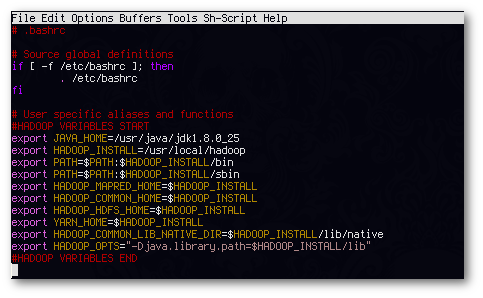
export YARN\_HOME=$HADOOP\_INSTALL

export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_INSTALL/lib/native

export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_INSTALL/lib"

#HADOOP VARIABLES END

Nuestro ~/.bashrc debe verse similar a la siguiente captura:



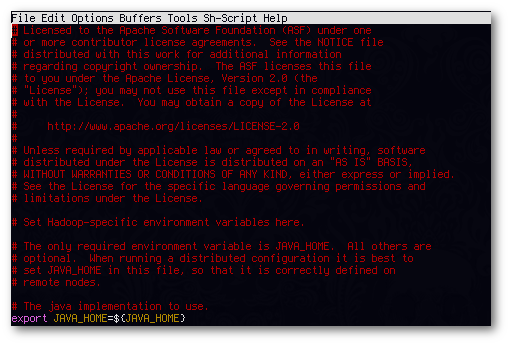
Debido a que ~/.bashrc solo se ejecuta de forma automática únicamente al iniciar una nueva instancia del shell es necesario evaluar el contenido del archivo manualmente para que los cambios sean efectivos a partir de este momento:

$ source ~/.bashrc

**3.2. /usr/local/hadoop/etc/hadoop/hadoop-env.sh**

Este archivo es el responsable de setear las variables de ambiente en el que correrá Hadoop.

Abrir el archivo, localizar la línea que exporta la variable JAVA\_HOME y comprobar que respeta la siguiente sintaxis: export JAVA\_HOME=${JAVA\_HOME}



**3.3. /usr/local/hadoop/etc/hadoop/core-site.xml**

Este archivo contiene las propiedades de configuración que Hadoop habilita al iniciar. El archivo se utiliza para sobreescribir los valores predeterminados por los apropiados para cada caso de ejecución.

Abrir el archivo para edición, buscar la marca (tag): <configuration></configuration>

e insertar la siguiente propiedad:

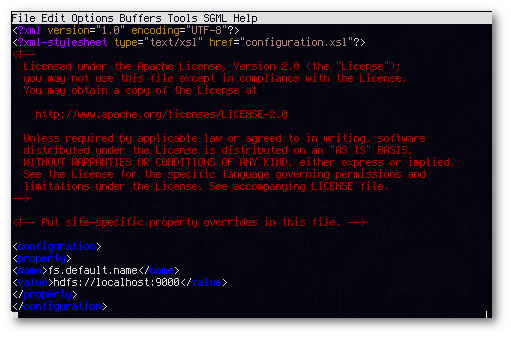
<property>

<name>fs.default.name</name>

<value>hdfs://localhost:9000</value>

</property>

Debería verse así:



**3.4. /usr/local/hadoop/etc/hadoop/yarn-site.xml**

Este archivo contiene las propiedades de configuración que MapReduce utiliza al iniciar. Este archivo puede utilizarse para sobreescribir los valores predeterminados con los que se inicia MapReduce.

Una vez abierto el archivo insertar entre los tags <configuration></configuration> el siguiente contenido:

<property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce\_shuffle</value>

</property>

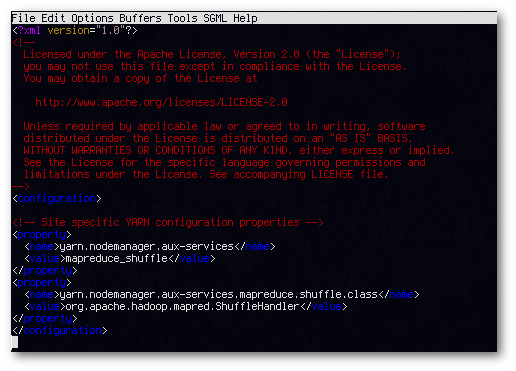
<property>

<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

</property>

Debería verse de esta forma:



**3.5. /usr/local/hadoop/etc/hadoop/mapred-site.xml**

De forma predeterminada el directorio /usr/local/hadoop/etc/hadoop contiene la plantilla /usr/local/hadoop/etc/hadoop/mapred-site.xml.template que debe ser renombrada (o *copiada*) como *mapred-site.xml*. Este archivo especifica qué framework se utilizará para MapReduce:

$ sudo cp /usr/local/hadoop/etc/hadoop/mapred-site.xml.template /usr/local/hadoop/etc/hadoop/mapred-site.xml

Con la nueva copia creada abrimos el archivo para edición y agregamos entre los tags <configuration></configuration>:

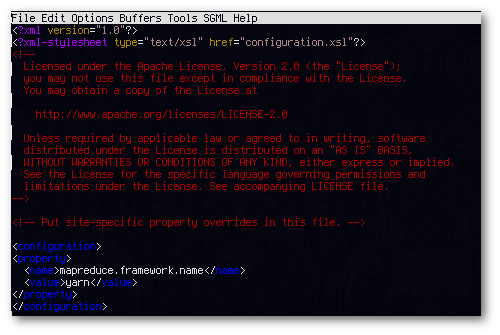
<property>

<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

El resultado debería verse así:



**3.6. /usr/local/hadoop/etc/hadoop/hdfs-site.xml**

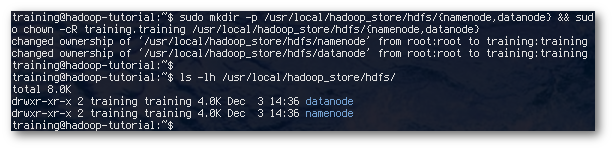
Este archivo debe configurarse para cada host que se utilice en el cluster, su función es la de especificar los directorios que serán usados como ***namenode*** y ***datanode***.

Antes de editar el archivo de configuración necesitamos crear los directorios que contendrán el namenode y el datanode que utiliza HDFS.

Además como vamos a utilizar Hadoop con los permisos de nuestro usuario es necesario cambiar el owner de los directorios que creamos:

$ sudo mkdir -p /usr/local/hadoop\_store/hdfs/{namenode,datanode} && sudo chown -cR training.training /usr/local/hadoop\_store/hdfs/{namenode,datanode}

Los permisos de acceso a los directorios deberían haber quedado seteados como se muestra en esta captura:



A continuación abrimos para edición /usr/local/hadoop/etc/hadoop/hdfs-site.xml e indicamos la ubicación de los directorios recién creados en <configuration></configuration>:

<property>

<name>dfs.replication</name>

<value>1</value>

</property>

<property>

<name>dfs.namenode.name.dir</name>

<value>file:/usr/local/hadoop\_store/hdfs/namenode</value>

</property>

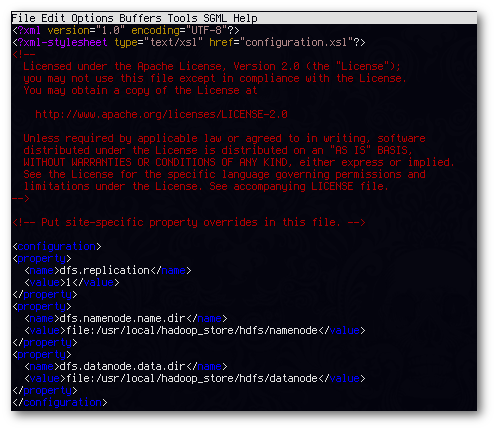
<property>

<name>dfs.datanode.data.dir</name>

<value>file:/usr/local/hadoop\_store/hdfs/datanode</value>

</property>

Finalmente, el archivo correctamente editado debería verse similar al de la siguiente captura:



**Paso 4: formatear el sistema de archivos de Apache Hadoop (HDFS)**

Luego de completar la instalación y configuración de Hadoop descrita en los pasos anteriores necesitamos formatear el sistema de archivos de Hadoop para poder comenzar a usar el framework; para hacerlo utilizamos el siguiente comando:

$ hdfs namenode -format



Atención: el formateo se realiza sólamente la primera vez para inicializar el sistema de archivos, subsecuentes formateos destruirán la información contenida en él.

**Paso 5: iniciar Apache Hadoop**

En este punto y si configuramos correctamente Hadoop ya podemos iniciar el framework:

$ start-dfs.sh

Al ejecutar este comando el sistema nos preguntará si deseamos validar la llave ssh que creamos antes -una por el namenode y otra por el datanode:

*Are you sure you want to continue connecting (yes/no)?*

A lo que responderemos yes. De la misma forma, deberemos validarnos para cada conexión utilizando nuestro password ***training***.

Finalmente ejecutamos:

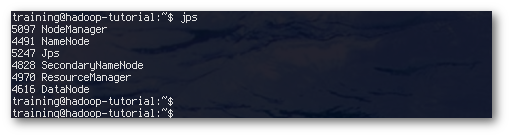
$ start-yarn.sh

Y nos autenticamos cuando el sistema así lo requiera.

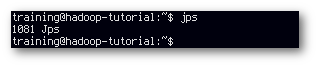
(Más sobre YARN al final del tutorial)

Corriendo los dos comandos arriba mencionados tendremos a Hadoop y listo para trabajar. Podemos verificar que el cluster local recién configurado está funcionando correctamente con el comando jps (Java Virtual Machine Process Status Tool):

$ jps



En el caso que HDFS o MapReduce (YARN) no se hayan iniciado correctamente los procesos relacionados no aparecerán al ejecutar jps:



**Acerca de YARN**

YARN es el acrónimo de "Yet Another Resource Negotiator", la implementación de próxima generación de MapReduce para Apache Hadoop, también conocido como MRv2 (MapReduce v2); la primer versión estable para producción fue liberada con hadoop-0.23.