

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

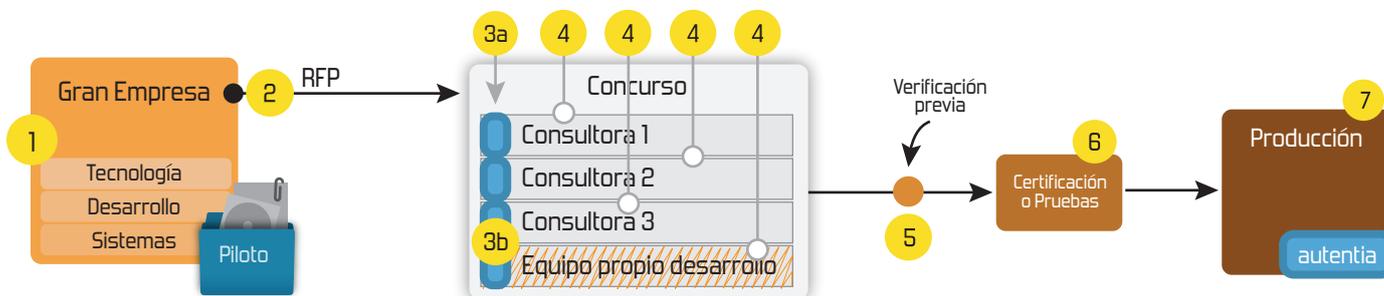
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Control de autenticación y
 acceso (Spring Security)
 UDDI

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Web Services
 Rest Services
 Social SSO
 SSO (Cas)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)

AdictosAlTrabajo

Temporada Completa de Terrakas
terrakas.com



autentia
Soporte a desarrollo informático
Hosting patrocinado por **enredados**

Entra en Adictos a través de

E-mail

Contraseña

Entrar

Deseo registrarme
Olvídate mi contraseña

[Inicio](#) [Quiénes somos](#) [Formación](#) [Comparador de salarios](#) [Nuestros libros](#) [Más](#)

» Estás en: [Inicio](#) [Tutoriales](#) [Primeros pasos con Hadoop: instalación y configuración en Linux](#)



Juan Alonso Ramos

Consultor tecnológico de desarrollo de proyectos informáticos.

Ingeniero en Informática, especialidad en Ingeniería del Software

Puedes encontrarme en [Autentia](#): Ofrecemos de servicios soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE

[Ver todos los tutoriales del autor](#)

Fecha de publicación del tutorial: 2014-02-25

Tutorial visitado 356 veces [Descargar en PDF](#)

Primeros pasos con Hadoop: instalación y configuración en Linux

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. Instalación de Hadoop.
- 3.1. Configuración de SSH.
- 4. Arquitectura HDFS.
- 4.1 Configuración HDFS.
- 5. Conclusiones.

1. Introducción.

Para poner en contexto lo que es Apache Hadoop podemos decir que se trata de un framework opensource escrito en Java utilizado principalmente para ejecutar aplicaciones distribuidas bajo un cluster de máquinas 'commodity'.

Dispone de un sistema de archivos propio: el HDFS (Hadoop Distributed File System). Se trata de un sistema de archivos distribuido en cada nodo del cluster. Utiliza un tamaño de bloque de entre 64Mb y 128Mb y está pensado para trabajar con grandes ficheros de datos. Está basado en el Google File System (GFS) creado en 2003 ya que en ese momento Google comprobó que los sistemas utilizados hasta entonces para tratar con toda la información de que disponían no eran suficientes. Necesitaban un sistema de ficheros distribuido, escalable, tolerante a fallos, con un intensivo acceso a datos y alta concurrencia.

En 2006 Doug Cutting crea Hadoop en un sistema de procesar datos a nivel web. En 2008 se gradúa como proyecto independiente de Apache Software Foundation.

Se basa en el paradigma MapReduce utilizado para paralelizar procesos en dos fases. La fase de mapeo donde se realiza el 'escaneo' o recogida de los datos de entrada produciendo una lista de pares (clave, valor). Estos pares son agrupados por clave y pasados a la función reduce que se encarga de procesarlos y generar un resultado agrupado de los mismos.

Hay varias formas de utilizar Hadoop. Dependiendo de nuestras necesidades podemos optar a varios tipos de instalación o modos de funcionamiento:

- Un único nodo en local (single node), utilizado por ejemplo para hacer pruebas de concepto corriendo Hadoop en una misma máquina
- Un cluster pseudo-distribuido para simular un cluster de varios nodos pero corriendo en una misma máquina es decir en la misma Java VM.
- Montar un cluster entre distintas máquinas (multi node) totalmente distribuido que sería el modo que utilizaríamos para montar un sistema Big Data en producción.

En este tutorial vamos a mostrar la forma de instalar Apache Hadoop en Linux de forma pseudo-distribuida.

2. Entorno.

Catálogo de servicios Autentia



Síguenos a través de:



Últimas Noticias

- » [IX Autentia Cycling Day \(ACTUALIZADO\)](#)
- » [Mi semana de desk-surfing en Otagami](#)
- » [Enamórate de un geek](#)
- » [Buscamos quien nos ayude en Autentia con WordPress](#)
- » [XXII Charla Autentia - PhoneGap/Cordova ¡qué bueno que viniste!](#)

[Histórico de noticias](#)

Últimos Tutoriales

- » [Como configurar CloudFlare en nuestra web](#)
- » [Crea todo un entorno de máquinas virtuales con un solo comando, gracias a Vagrant](#)
- » [Depurar Tomcat en remoto.](#)

El tutorial se ha realizado con el siguiente entorno:

- Ubuntu 12.04 64 bits
- Oracle Java SDK 1.6.0_27
- Apache Hadoop 2.2.0

3. Instalación de Hadoop.

Vamos a partir de que en la máquina ya está instalada una JDK de Java, preferiblemente la 1.6. Lo primero que haremos, obviamente será descargar Apache Hadoop de la página oficial. [Descargar versión 2.2.0](#)

A continuación muestro los pasos para descomprimir el archivo y copiarlo a un directorio de nuestro equipo.

```
1 | sudo tar xzf hadoop-2.2.0.tar.gz
2 | mv hadoop-2.2.0 /usr/local/
3 | mv /usr/local/hadoop-2.2.0 /usr/local/hadoop
```

Es una buena práctica crear un usuario y un grupo específico para configurar y administrar hadoop. Pongo los pasos necesarios para crear el usuario, asignarle la password, añadir el usuario al fichero sudoers, etc.

```
1 | useradd -d /home/hadoop -m hadoop
2 | passwd hadoop
3 | usermod -a -G sudo hadoop
4 | usermod -s /bin/bash hadoop
```

Nos logamos en la máquina con el usuario hadoop `su hadoop`. A continuación añadimos las variables de entorno para hadoop en el `$HOME/.bashrc`.

```
1 | export HADOOP_HOME=/usr/local/hadoop
2 | export PATH=$PATH:$HADOOP_HOME/bin
3 | export PATH=$PATH:$HADOOP_HOME/sbin
4 | export HADOOP_MAPRED_HOME=${HADOOP_HOME}
5 | export HADOOP_COMMON_HOME=${HADOOP_HOME}
6 | export HADOOP_HDFS_HOME=${HADOOP_HOME}
7 | export YARN_HOME=${HADOOP_HOME}
```

Una vez editado el `.bashrc` hacemos un `source .bashrc` para cargar la nueva configuración. Para comprobar que todo ha ido bien escribimos en la consola `hadoop version` y nos debería devolver la versión de Hadoop con la que estamos trabajando.

3.1 Configuración de SSH.

Ahora vamos a configurar el ssh para que Hadoop pueda autenticarse con una clave pública y acceder a los nodos del cluster y sobre la máquina local para el usuario que creamos anteriormente. Generaremos una nueva clave pública añadiéndola al directorio de claves del usuario. No debemos poner password ya que hadoop necesita poder conectarse al cluster directamente, imaginad si cada vez que intenta acceder a un nodo del cluster tuvieramos que meter contraseña!

```
1 | sudo apt-get install ssh
2 | ssh-keygen -t rsa -f ~/.ssh/id_rsa
3 | cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

Le damos permisos:

```
1 | sudo chmod go-w $HOME/.ssh
2 | sudo chmod 600 $HOME/.ssh/authorized_keys
3 | sudo chown `whoami` $HOME/.ssh/authorized_keys
```

Para comprobar que se ha instalado correctamente puedes ejecutar el comando `ssh localhost` y si se conecta es que todo ha ido bien. Si no fuera así revisa los pasos anteriores. Salimos de la sesión ssh con `exit`.

Para dejar lista la configuración de Ubuntu debemos deshabilitar IPv6 ya que como dice la [documentación oficial](#), Hadoop no lo admite para gestionar correctamente el cluster. Ubuntu trae por defecto esta configuración por lo que debemos desactivarla. Para ello añadimos las siguientes líneas al fichero `/etc/sysctl.conf`.

```
1 | net.ipv6.conf.all.disable_ipv6 = 1
2 | net.ipv6.conf.default.disable_ipv6 = 1
3 | net.ipv6.conf.lo.disable_ipv6 = 1
```

Para que coja los cambios debemos reiniciar la máquina. Una vez reiniciada podemos comprobar que ha cogido bien la nueva configuración con el comando `cat /proc/sys/net/ipv6/conf/all/disable_ipv6`.

4. Arquitectura HDFS.

HDFS es una implementación del sistema de ficheros descrito en el paper de Google (GFS). Sigue una filosofía "Write once, read more" en concreto para albergar grandes ficheros y principalmente pensado para programas batch donde los datos no son en tiempo real.

En HDFS se trabaja con una cantidad mínima de información llamada **bloque** que normalmente estará comprendida entre 64-128 Mb debido a que se trabaja con ficheros muy grandes (Gigabytes, Petabytes...). El objetivo de HDFS es dividir el fichero en bloques de tamaño fijo y distribuirlo en los distintos nodos del cluster. Debido a que se produce mucho tráfico por la red para distribuir la información con el coste que esto conlleva se opta por un tamaño de bloque alto. Recuerdo que HDFS es tolerante a fallos y caídas de nodos del cluster por lo que se replica la información en varios nodos (por defecto 3 veces).

La gestión del cluster la realiza el **NameNode**. Es el nodo maestro encargado de gestionar los metadatos de los ficheros y los directorios, coordina los bloques que se envían a cada datanode monitorizando su estado para asegurar que todos los

» Firewall de alta disponibilidad con balanceo de carga (Clúster)

» Máquina de estados con Apache SCXML

Últimos Tutoriales del Autor

» Trabajando con Mule ESB

» Crear un proyecto de Mule ESB con Mule Studio

» Crear un proyecto de Mule ESB con Maven

» Primeros pasos con Mule ESB

» Ejecución de tareas asíncronas y planificadas con Spring.

Categorías del Tutorial

Big Data

Últimas ofertas de empleo

2011-09-08
Comercial - Ventas - MADRID.

2011-09-03
Comercial - Ventas - VALENCIA.

2011-08-19
Comercial - Compras - ALICANTE.

2011-07-12
Otras Sin catalogar - MADRID.

2011-07-06
Otras Sin catalogar - LUGO.

trabajos se completan correctamente, redirigiendo bloques a otros datanodos si alguno estuviera caído. La información de los metadatos se almacena en memoria RAM para que el acceso sea más rápido. El NameNode es vital en el cluster por lo que se suele montar en una máquina mucho más potente que para los datanodos con mayor capacidad de proceso y sobre todo mucha memoria RAM.

Los **DataNodes** son los nodos esclavos encargados del almacenamiento de los bloques realizando las operaciones de lectura y escritura. Estos informan al NameNode de los bloques almacenados.

El **SecondaryNameNode** es un servicio auxiliar y opcional en el cluster. Es un error pensar que se trata de un segundo Namenode por si el primero fallara. Puede usarse como backup de los metadatos.

Otro demonio existente en HDFS es el **JobTracker** encargado de gestionar los jobs encargados de las tareas MapReduce.

Por último el **TaskTracker** es un demonio encargado de ejecutar una determinada tarea en cada nodo. Las tareas son aplicaciones Java MapReduce.

4.1 Configuración HDFS.

Con la instalación que hemos realizado hasta ahora tendríamos un nodo de hadoop instalado en la máquina. Si queremos configurar hadoop en un modo pseudo-distribuido debemos modificar algunos ficheros de configuración. Nos situamos en el directorio `/user/local/hadoop/etc/hadoop` y editamos el fichero **core-site.xml**

```

1 <?xml version="1.0"?>
2 <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3
4 <configuration>
5   <property>
6     <name>fs.default.name</name>
7     <value>hdfs://localhost:8020</value>
8     <description>Nombre del filesystem por defecto.</description>
9   </property>
10 </configuration>

```

Esto configura el directorio hdfs por defecto en localhost.

A continuación configuramos algunas propiedades del sistema de ficheros en el fichero **hdfs-site.xml**. Como estamos configurando un cluster en una única máquina no tiene sentido indicar un factor de replicación mayor a 1.

```

1 <?xml version="1.0"?>
2 <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3
4 <configuration>
5   <property>
6     <name>dfs.namenode.name.dir</name>
7     <value>file:/home/hadoop/workspace/dfs/name</value>
8     <description>Path del filesystem donde el namenode almacenará los metadatos.</des<
9   </property>
10
11   <property>
12     <name>dfs.datanode.data.dir</name>
13     <value>file:/home/hadoop/workspace/dfs/data</value>
14     <description>Path del filesystem donde el datanode almacenará los bloques.</descr<
15   </property>
16
17   <property>
18     <name>dfs.replication</name>
19     <value>1</value>
20     <description>Factor de replicación. Lo ponemos a 1 porque sólo tenemos 1 máquin<
21   </property>
22 </configuration>

```

Creamos los directorios `/home/hadoop/workspace/dfs/name` y `/home/hadoop/workspace/dfs/data`

```

1 mkdir /home/hadoop/workspace/dfs/name
2 mkdir /home/hadoop/workspace/dfs/data

```

Editamos ahora el fichero **hadoop-env.sh** para indicar el directorio `JAVA_HOME` de nuestra máquina.

```

1 export JAVA_HOME=/usr/lib/jvm/java-6-oracle

```

Otra propiedad que vamos a configurar será la que configura los directorios donde se realiza el MapReduce, editamos el **mapred-site.xml**. También configuraremos el MapReduce de nuestro Hadoop al nuevo framework MapReduce disponible a partir de la versión 2 de Hadoop llamado Yarn (Yet Another Resource Negotiator). Es un framework MapReduce mejorado capaz de realizar más trabajos y un sinfín de cosas más que veremos en otros tutoriales. De momento lo dejaremos configurado. Como nuestro cluster es pseudo-distribuido únicamente habrá una tarea map y una reduce.

Por defecto viene como `mapred-site.xml.template` por lo que hacemos una copia y renombramos a `mapred-site.xml`

```

1 <?xml version="1.0"?>
2 <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3
4 <configuration>
5   <property>
6     <name>mapreduce.framework.name</name>
7     <value>yarn</value>
8   </property>
9
10  <property>
11    <name>mapred.system.dir</name>
12    <value>file:/home/hadoop/workspace/mapred/system</value>
13    <final>true</final>
14  </property>
15

```

```

16 <property>
17 <name>mapred.local.dir</name>
18 <value>file:/home/hadoop/workspace/mapred/local</value>
19 <final>true</final>
20 </property>
21 </configuration>
    
```

Creamos los directorios:

```

1 mkdir /home/hadoop/workspace/mapred/system
2 mkdir /home/hadoop/workspace/mapred/local
    
```

Por último configuramos un par de propiedades en el **yarn-site.xml**. Habilitamos la fase de Shuffle para que se pueda hacer entre las fases de Map y Reduce ya que YARN por defecto no lo incluye.

```

1 <configuration>
2 <property>
3 <name>yarn.nodemanager.aux-services</name>
4 <value>mapreduce_shuffle</value>
5 </property>
6 <property>
7 <name>yarn.nodemanager.aux-services.mapreduce_shuffle.class</name>
8 <value>org.apache.hadoop.mapred.ShuffleHandler</value>
9 </property>
10 </configuration>
    
```

Una vez instalado Hadoop vamos a comentar algunos comandos que se utilizan normalmente a la hora de trabajar con un cluster Hadoop. Lo primero será formatear el sistema de ficheros HDFS mediante el comando:

```

1 hadoop namenode -format
    
```

Lo siguiente será arrancar el cluster, es decir los diferentes demonios (Namenode, Datanode...) encargados de la ejecución de las tareas MapReduce y de la gestión del sistema de ficheros HDFS. Para hacer esto ejecutamos el script **start-all.sh** situado en **/usr/local/hadoop/sbin/**

```

1 ./start-all.sh
    
```

Si ejecutamos el comando **Jps** que nos muestra los procesos Java corriendo en la máquina debería salir algo como esto:

```

1 hadoop@juan:/usr/local/hadoop/sbin$ jps
2 11461 DataNode
3 11706 SecondaryNameNode
4 12875 NodeManager
5 12675 ResourceManager
6 11268 NameNode
7 13151 Jps
    
```

Una vez arrancado podemos acceder al interfaz WebUI en **http://localhost:8088/cluster/nodes** donde podemos realizar el seguimiento de los jobs que vayamos ejecutando. Es también muy útil para ver los logs, el histórico de jobs ejecutados, la configuración de MapReduce, etc.



Nodes of the cluster

Logged in as: drwho

- Cluster
- About
- Nodes
- Applications
- NEW
- NEW_SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- REMOVING
- FINISHING
- FINISHED
- FAILED
- KILLED
- Scheduler
- Tools

Cluster Metrics												
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	Active Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes
0	0	0	0	0	0 B	8 GB	0 B	1	0	0	0	0

Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Mem Used	Mem Avail
/default-rack	RUNNING	localhost:38490	localhost:8042	24-Feb-2014 00:17:38		0	0 B	8 GB

Showing 1 to 1 of 1 entries

Otra interfaz muy útil para ver la información del NameNode es accesible en **http://localhost:50070/dfshealth.jsp**. Aquí podemos ver la información de nuestro NameNode y también podemos acceder de forma visual al filesystem hdfs.

NameNode 'localhost:8020' (active)

Started:	Mon Feb 24 00:07:16 CET 2014
Version:	2.2.0, 1529768
Compiled:	2013-10-07T06:28Z by hortonmu from branch-2.2.0
Cluster ID:	CID-a15ec203-6ebd-4be2-902f-69d36ace1e2d
Block Pool ID:	BP-456311633-127.0.0.1-1393189076486

[Browse the filesystem](#)
[NameNode Logs](#)

Cluster Summary

Security is OFF

38 files and directories, 20 blocks = 58 total.

Heap Memory used 33.95 MB is 48% of Committed Heap Memory 70.53 MB. Max Heap Memory is 966.69 MB.

Non Heap Memory used 32.61 MB is 81% of Committed Non Heap Memory 40.16 MB. Max Non Heap Memory is 118 MB.

Configured Capacity	:	20.54 GB			
DFS Used	:	660 KB			
Non DFS Used	:	6.26 GB			
DFS Remaining	:	14.28 GB			
DFS Used%	:	0.00%			
DFS Remaining%	:	69.52%			
Block Pool Used	:	660 KB			
Block Pool Used%	:	0.00%			
DataNodes usages	:	Min %	Median %	Max %	stdev %
		0.00%	0.00%	0.00%	0.00%
Live Nodes	:	1 (Decommissioned: 0)			
Dead Nodes	:	0 (Decommissioned: 0)			
Decommissioning Nodes	:	0			
Number of Under-Replicated Blocks	:	0			

5. Conclusiones.

En este tutorial he querido hacer una primera introducción a la arquitectura de Hadoop, su instalación, configuración y arranque. En sucesivos tutoriales iremos entrando en detalle en los aspectos más destacados de la arquitectura y veremos las distintas partes en que se compone.

Espero que te haya sido de ayuda.

Un saludo.

Juan

A continuación puedes evaluarlo:

[Regístrate para evaluarlo](#)

Por favor, vota +1 o compártelo si te pareció interesante

Share |  

Anímate y coméntanos lo que pienses sobre este TUTORIAL:

» [Regístrate](#) y accede a esta y otras ventajas «



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

IMPULSA Impulsores Comunidad ¿Ayuda?

35
clicks

2 personas han traído clicks a esta página

  + + + + + +

powered by [karmacracy](#)

Copyright 2003-2014 © All Rights Reserved | [Texto legal y condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) | [Contacto](#)

