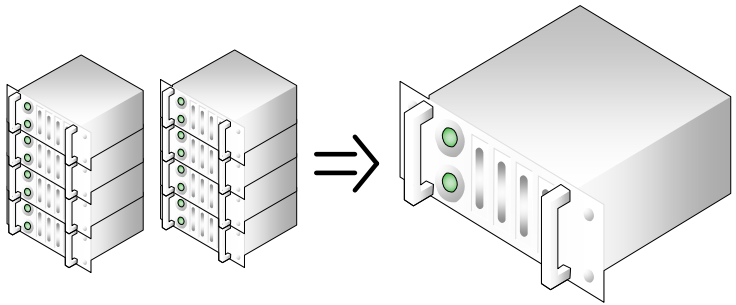
Procesamiento de datos masivos

Jesús Morán

viu | Universidad Internacional de Valencia

29.11.21

- Varios servidores que trabajan como si fuese uno
  - Rendimiento: tareas con alta memoria, CPU, etc
  - Disponibilidad: tolerancia a fallos
  - Eficiencia: gestión de recursos y tareas
  - Escalado: vertical vs horizontal

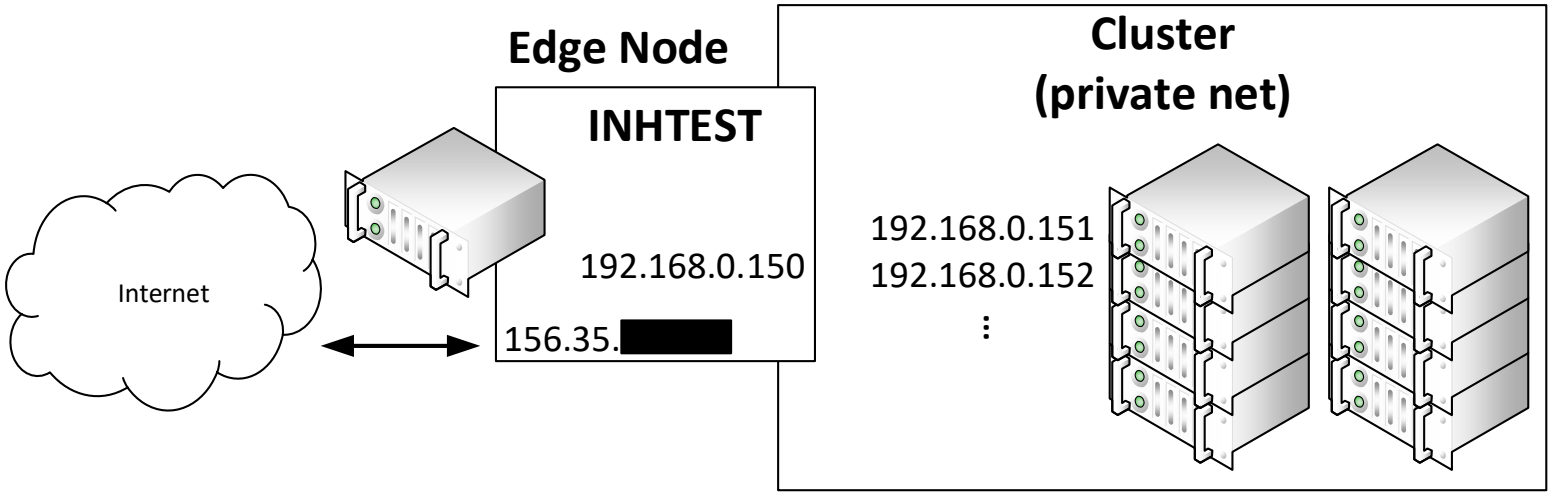


- Servidores y racks
- Red y conexiones
- Sistemas de archivos distribuidos
- Gestión de usuarios
- Permisos y seguridad
- Configuraciones y políticas de acceso
- Monitorización
- Fallos de servidores
- Recursos en caliente
- Interoperabilidad de servicios
- Balanceo de carga

- Altas prestaciones (HPC)
  - Requieren el análisis de muchos nodos
  - Paradigmas distribuidos
- Alta disponibilidad
  - Replicaciones
  - Sin puntos de fallo únicos
  - Servicio disponible (si falla se restaura)
- Balanceo de carga
  - Varios nodos corren en mismo servicio y se balancea el acceso

- Análisis y cálculo:
  - Bases de datos: SQL, NoSQL, newSQL
  - Procesamiento Big Data
- Renderizado
- Desencriptación
- Simulaciones:
  - Mecánica
  - Química
  - …

Cluster a desplegar



- Operativo: CentOS (o Almalinux)
- Asignar IP (ej. DHCP)
- Instalar Java
  - Variable de entorno $JAVA_HOME
  - Enlace simbólico:

```
alternatives --install /usr/bin/java java
/usr/...
```

- Crear usuario hadmin y grupo hadoop
- Descargar Hadoop en /usr/local (ej. wget)
- Asignar a hadmin y hadoop como propietarios
- Crear carpeta de archivos temporales en /hadoopTemp

- Nombre de hosts y red
- Abrir puertos [maestro = INHTEST1]
  - NameNode: 9000 y 50070 (sólo maestro)
  - Secondary NameNode: 50090 (sólo maestro)
  - DataNode: 50020, 50010 y 50075
  - ResourceManager: 8088, 8031, 8032 y 8030 (sólo maestro)
  - NodeManager: 8042, 8040 y 10200
  - HistoryServer: 19888 y 10020
  - Shuffle: 13562
  - Hadoop ephemeral ports: 50200:51200
  - Otros puertos de Spark, Flink, etc. (generalmente sólo maestro)

- Instalamos ssh
- Creamos una clave pública/privada

```
ssh-keygen -t rsa -P "" -f ~/.ssh/id_rsa
```

- La ponemos como clave autorizada:

```
cat ~/.ssh/id_rsa.pub >>
~/.ssh/authorized_keys
```

- Las copiamos a otros nodos:

```
ssh-copy-id –i ~/.ssh/id_rsa.pub
hadmin@INHTEST2
```

- Crear variables de entorno de Hadoop:
  - HADOOP_HOME=/usr/local/hadoop/hadoop-x.y.0
  - HADOOP_MAPRED_HOME=$HADOOP_HOME
  - HADOOP_COMMON_HOME=$HADOOP_HOME
  - HADOOP_HDFS_HOME=$HADOOP_HOME
  - HADOOP_YARN_HOME=$HADOOP_HOME
  - HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
  - HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
  - export HADOOP_OPTS="-Djava.library.path=/usr/local/hadoop/hadoop-2.2.0/lib"
- Añadir $HADOOP_HOME/bin al path

- Configuramos Hadoop
  - $HADOOP_HOME/etc/hadoop/core-site.xml

```
<configuration>
    <property>
        <name>fs.default.name</name>
        <value>hdfs://INHTEST1:9000</value>
    </property>
    <property>
        <name>hadoop.tmp.dir</name>
        <value>/hadoopTemp</value>
    </property>
</configuration>
```

- Configuramos Hadoop
  - $HADOOP_HOME/etc/hadoop/core-site.xml
  - **$HADOOP_HOME /etc/hadoop/yarn-site.xml**
    - yarn.resourcemanager.hostname: INHTEST1
    - yarn.resourcemanager.address: INHTEST1:8032
    - yarn.resourcemanager.resource-tracker.address: INHTEST1:8031
    - yarn.resourcemanager.scheduler.address: INHTEST1:8030
    - yarn.log.server.url: http://INHTEST1:19888/jobhistory/logs/
    - yarn.nodemanager.aux-services: mapreduce_shuffle
    - yarn.nodemanager.aux-services.mapreduce_shuffle.class: org.apache.hadoop.mapred.ShuffleHandler
    - yarn.log-aggregation-enable: true

- Configuramos Hadoop
  - $HADOOP_HOME/etc/hadoop/core-site.xml
  - $HADOOP_HOME /etc/hadoop/yarn-site.xml
  - **$HADOOP_HOME /etc/hadoop/hdfs-site.xml**
    - dfs.data.dir: file:///usr/local/hadoop/Data/Datanode
    - dfs.name.dir: file:///usr/local/hadoop/Data/Namenode
    - dfs.namenode.secondary.http-address: INHTEST1:50090
    - dfs.namenode.http-address: INHTEST1:50070
    - dfs.namenode.secondary.http-address: INHTEST1:50090
    - dfs.permissions.superusergroup: hadoop
    - dfs.replication: 3
    - dfs.permissions.enabled: true

- Configuramos Hadoop
  - $HADOOP_HOME/etc/hadoop/core-site.xml
  - $HADOOP_HOME /etc/hadoop/yarn-site.xml
  - $HADOOP_HOME /etc/hadoop/hdfs-site.xml
  - **$HADOOP_HOME /etc/hadoop/mapred-site.xml**
    - mapreduce.jobtracker.http.address: INHTEST1:50030
    - mapreduce.jobhistory.address: INHTEST1:10020
    - mapreduce.jobhistory.webapp.address: INHTEST1:19888
    - yarn.app.mapreduce.am.job.client.port-range: 50200-51200
    - mapreduce.framework.name: yarn
    - mapreduce.jobtracker.staging.root.dir: /user

- Indicar los esclavos: $HADOOP_HOME/etc/hadoop/slaves
- Formateamos el sistema de archivos distribuido:

```
hadoop namenode -format
```

  - Formatea los metadatos del namenode ~ formatear sistema de archivos

- Arrancar los servicios:

```
$HADOOP_HOME/sbin/start-all.sh
```

- Crear las carpetas de usuario:

```
hadoop fs -mkdir /user
hadoop fs -mkdir /user/hadmin
```

- Tiene que ser un Edge node
- Instalar:
  - Java
  - Hadoop (configuración de Hadoop)
  - ssh
- Crear usuario
  - Creamos usuario local con grupo hadoop
  - Crear carpeta en sistema de archivos distribuido

```
hadoop fs -mkdir /user/cristian/
```

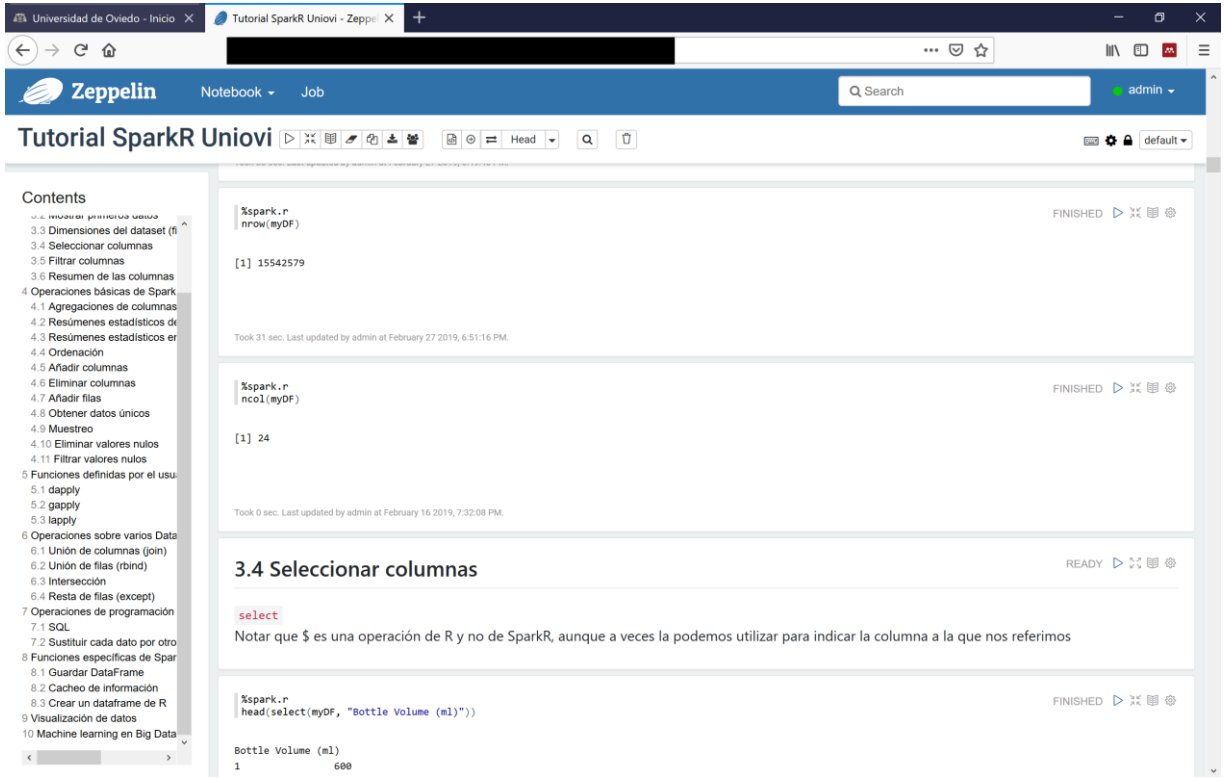
  - Ponerlo como propietario

```
hadoop fs -chown -R cristian:hadoop
/user/cristian/
```

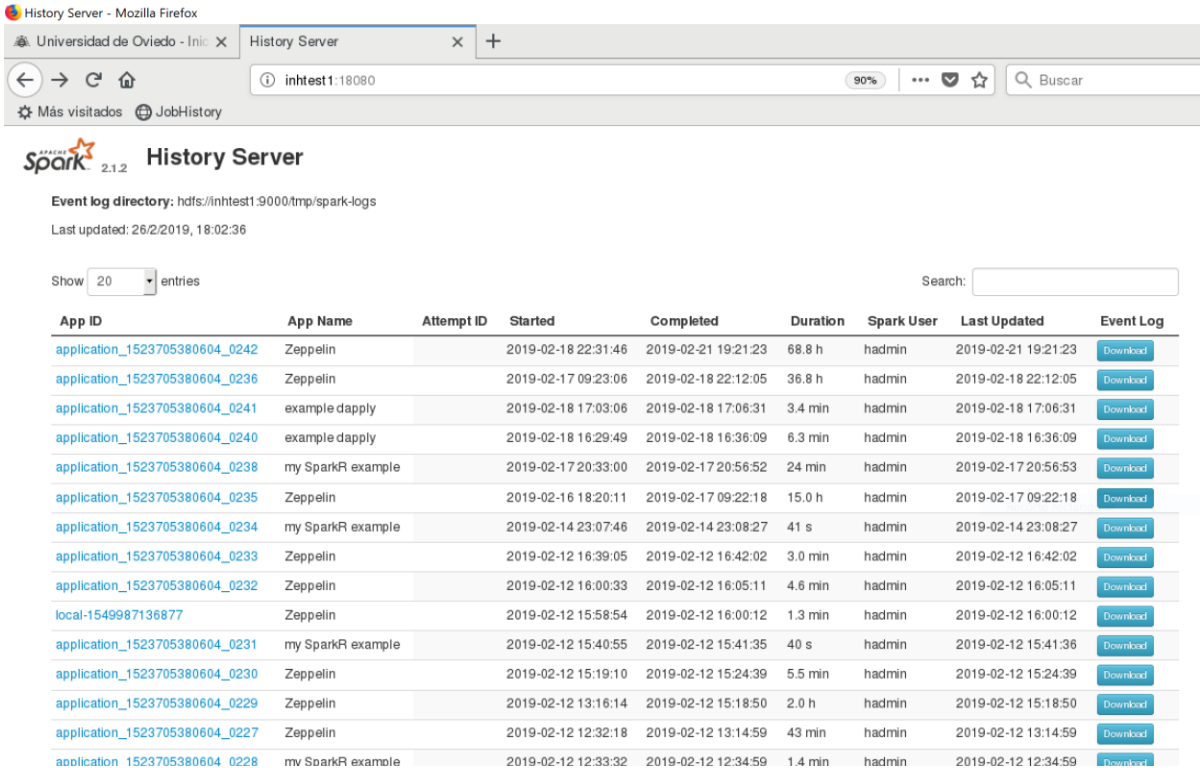
- Sistema de archivos distribuido: HDFS (maestro y esclavos)
- Gestor de recursos: YARN (maestros y esclavos)
- Motores de procesamiento: Hadoop, Spark,…
- Otros:
  - Servidores web del cluster
  - Historial
  - Logs distribuidos

- Ganglia: monitorizar el uso de recursos
- Nagios: alertar de problemas en el cluster
- Kerberos: autenticación integrada con Hadoop
- Squid:
  - Proxy transparente para actualizar los operativos
  - SSL BUMP man-in-the-middle para otras instalaciones
- Otros frameworks/librerías de cálculo:
  - R (para utilizar SparkR)
  - Zeppelin: notebook Big Data
  - …

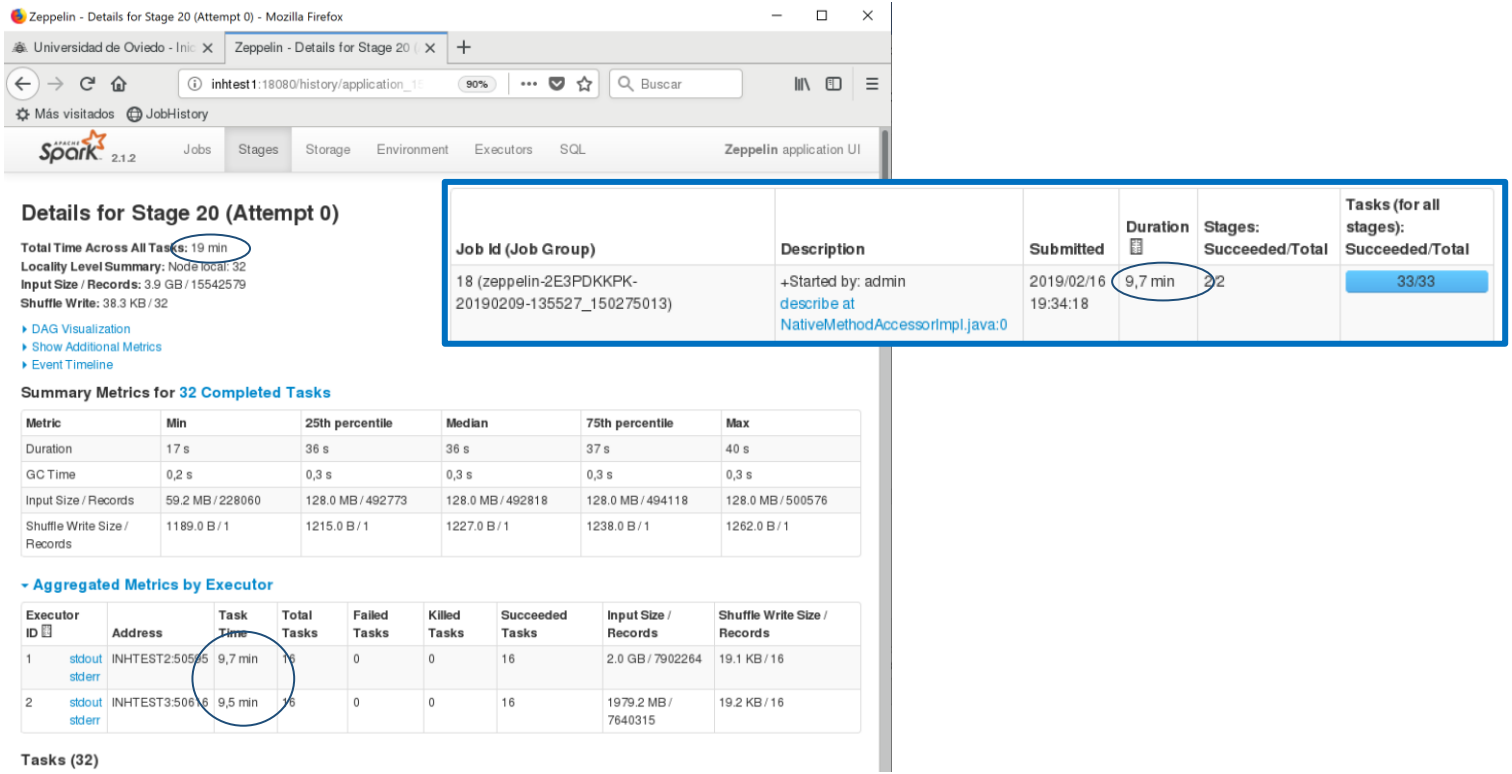
Otros servicios



Otros servicios



Otros servicios



Gracias

viu | Universidad Internacional de Valencia