

Administración y monitorización de sistemas Hadoop.

Caso práctico

El Banco Español de Inversiones, BEI, es uno de los principales bancos nacionales de banca privada de España, con más de 100 años de historia.

Su modelo de negocio se ha centrado en grandes inversores, y el principal valor añadido que les aporta es que dispone de un grupo de expertos de gran reputación mundial que recomienda a los clientes qué inversiones realizar para maximizar la rentabilidad.

Hasta ahora, su negocio estaba muy focalizado en el cliente y en el gran conocimiento interno, y nunca se había planteado una apuesta por una disrupción tecnológica para aportar mayor valor a los clientes por la tecnología en lugar de por el conocimiento de negocio tradicional. Es por ello que a nivel de arquitectura de sistemas, tiene casi todo su negocio en un sistema mainframe junto con una herramienta de CRM.

Sin embargo, recientemente su consejo de administración ha nombrado nuevos consejeros que pretenden dar una revolución al banco, incorporando la tecnología para ayudar a los clientes y a su grupo de expertos a tomar mejores decisiones con modelos predictivos, o dando funcionalidades en cualquier canal, tanto web como móvil u oficinas.

Dentro de esta nueva estrategia que quiere abordar el banco, han decidido invertir en tecnologías Big Data, y por ello han implantado una plataforma Hadoop a la que han volcado todos los datos del mainframe y el CRM. Tienen un equipo de científicos de datos trabajando ya en el clúster, desarrollando modelos predictivos de los mercados o de valoración, así como analistas de negocio que, mediante herramientas de visualización, realizan su trabajo pudiendo cruzar cualquier dato que exista en el banco, algo que antes era imposible, ya que los datos eran poco accesibles para estos analistas.

El equipo de IT, que está liderado por María Robles, debe manejar ahora todo el clúster, ya que son cada vez más los usuarios que acceden a él para lanzar consultas, ver datos o ejecutar modelos.

María y su equipo están viendo qué capacidades les ofrece Hadoop para la administración de la plataforma.



[Colossus Cloud](#) (Dominio público)

Hadoop es una plataforma que requiere un mayor esfuerzo de monitorización o administración, principalmente por estos motivos:

- ✓ Tiene un **amplio ecosistema de componentes**, cada uno desarrollado de forma independiente, con su propio funcionamiento, versión, sus propios ficheros de configuración o métricas de monitorización.
- ✓ Se ejecuta en un **entorno distribuido**, es decir, en un conjunto de servidores que puede llegar a ser numeroso. Cada servidor tiene su sistema operativo, y por supuesto, su propio hardware que puede tener fallos.

- ✔ Hadoop es una plataforma con un nivel de **madurez inferior** a otras soluciones que tienen una base instalada muy grande y la experiencia de un fabricante que se traduce en una documentación muy rica, unas herramientas de administración potentes, etc.

La administración de un sistema Hadoop, por lo tanto, no es algo trivial. Afortunadamente, existen varias herramientas que facilitan esta labor, entre las que destacan:

- ✔ Las consolas de **monitorización de HDFS** (Namenode UI) y **YARN** (ResourceManager UI), que ofrecen una visión simple de lo que ocurre en cada servicio, permitiendo acceder a bajo nivel hasta ver los logs de las aplicaciones o procesos que se están ejecutando.
- ✔ **Apache Ambari y Cloudera Manager**, que son sin lugar a dudas las herramientas más potentes y a su vez más fáciles de utilizar para la instalación y la administración de plataformas Hadoop, ya que ofrecen funcionalidades desde la visualización de cuadros de mando con cualquier métrica de salud del sistema, la creación de alarmas, la modificación de la configuración del sistema o la administración de usuarios.
- ✔ **Ganglia**, que es una herramienta opensource para monitorizar clústeres de servidores. Aunque no es una herramienta específica para Hadoop, puede ser útil para la monitorización de los sistemas.

En esta unidad vamos a conocer estas herramientas:

- ✔ En primer lugar, conoceremos los interfaces web de HDFS y YARN.
- ✔ A continuación, entraremos en detalle en Apache Ambari, así como Cloudera Manager.
- ✔ Por último, conoceremos cómo utilizar Ganglia para monitorizar plataformas Hadoop.



[Ministerio de Educación y Formación Profesional](#) (Dominio público)

Materiales formativos de FP Online propiedad del Ministerio de Educación y Formación Profesional.

[Aviso Legal](#)

1.- Introducción a la administración de Hadoop.

Caso práctico

María Robles, la responsable de IT del Banco Español de Inversiones tiene más de 20 años de experiencia en la administración de todo tipo de sistemas que han utilizado en el banco.

Ahora bien, administrar una plataforma Hadoop es un reto para ella y su equipo, sabe que tienen muchos servidores y muchos servicios en cada servidor, ¿cómo van a poder administrar un sistema tan complejo?



[Csaba Nagy](#) (Dominio público)

Para la administración de una plataforma Hadoop se puede distinguir tres tipos de actividades: la configuración, operación de los servicios y la monitorización.

En este tema nos centraremos en la monitorización, ya que la configuración suele ser más estática, y no requiere grandes intervenciones una vez la plataforma ha sido instalada, y la operación de los servicios son tareas sencillas que consisten en el arranque, parada o reinicio de los mismos. En cualquier caso, ahora veremos cómo se configura o parametriza una plataforma Hadoop, para posteriormente entrar a detalle en la tarea más habitual, la monitorización de la plataforma.

Configuración

La configuración de un clúster Hadoop se basa en ficheros XML asociados a cada servicio de la plataforma, es decir, encontrarás ficheros de configuración de HDFS, YARN, Hive, etc. por separado. Los ficheros de configuración suelen encontrarse en los servidores donde se ejecuta el servicio, habitualmente en los directorios /etc/conf.

A continuación vamos a ver los principales ficheros de configuración de los servicios más importantes:

- ✓ **Hadoop core (HDFS, YARN y MapReduce):** para configurar estos servicios, existen varios ficheros de configuración en los nodos master donde se ejecuta cada servicio:
 - ◆ **core-site.xml:** en este fichero aparece la configuración común de los componentes core, en ella encontraremos algunas propiedades de configuración como:
 - **fs.defaultFS,** que indica el endpoint de HDFS al que deben apuntar los clientes.
 - **hadoop.security.authentication:** tipo de autenticación requerida (Kerberos, simple).
 - ◆ **hdfs-site.xml:** este fichero contiene la configuración específica de HDFS, como por ejemplo las siguientes propiedades:
 - **fs.namenode.name.dir:** directorio local en el servidor donde se aloja la información del Namenode.
 - **dfs.datanode.data.dir:** directorio local en el servidor worker donde se almacenan los bloques de HDFS.
 - **dfs.namenode.http-address:** dirección en la que se va a arrancar una consola web de monitorización de HDFS.
 - ◆ **yarn-site.xml:** este fichero contiene la configuración específica de YARN, como por ejemplo las siguientes propiedades:
 - **yarn.resourcemanager.scheduler.class:** tipo de Scheduler que se va a utilizar para la gestión de las

- prioridades de las aplicaciones.
 - `yarn.resourcemanager.resource-tracker.address`: dirección del ResourceManager al que se conectarán los NodeManager.
 - `yarn.nodemanager.log-dirs`: directorio en los nodos worker en los que se escribirá el log de la ejecución de las aplicaciones.
 - `yarn.resourcemanager.webapp.address`: dirección en la que se va a arrancar una consola web de monitorización de YARN.
- ✓ **Apache Hive**: utiliza un fichero `hive-site.xml` para almacenar la configuración principal del servicio. Este fichero puede contener centenares de parámetros de configuración, aunque los principales son los siguientes:
- `hive.execution.engine`: indica qué motor se utilizará para la ejecución de consultas, teniendo 3 posibilidades:
 - MapReduce.
 - Tez.
 - Spark.
 - `hive.server2.enable.doAs`: permite ejecutar las queries con el usuario que hizo la petición, en lugar de con el usuario de sistema con el que se ejecuta Hive.
 - `hive.server2.thrift.port`: puerto en el que se levantará el servidor Thrift al que se conectarán los clientes de Hive.
- ✓ **Apache Spark**: utiliza el fichero `spark-defaults.conf`, que no es de tipo XML a diferencia de los ficheros de configuración que suelen tener las herramientas del ecosistema Hadoop. Este fichero sirve para parametrizar los valores por defecto, que se pueden modificar mediante código por las aplicaciones. Algunos de los parámetros más importantes que contiene este fichero son los siguientes:
- `spark.yarn.historyServer.address`: contiene la ruta de YARN donde se ejecuta Spark.
 - `spark.history.ui.port`: puerto en el que se levantará un interfaz web de monitorización de Spark.
 - `spark.eventLog.dir`: ruta en la que se almacenarán los ficheros de log con la ejecución de las aplicaciones.

Estos son algunos de los principales ficheros y parámetros de configuración de Spark. Se pueden modificar de dos formas:

- ✓ Accediendo por consola a los ficheros y modificándolos con un editor tipo `vi`.
- ✓ Mediante una herramienta de administración como Apache Ambari o Cloudera Manager.

En cualquier caso, después de la modificación de algún parámetro, suele ser necesario reiniciar el servicio implicado, lo cual puede llevar un tiempo de pérdida de servicio.

Monitorización

En primer lugar, es preciso conocer que la mayoría de los clústers de Hadoop se ejecutan en sistemas Linux y, por lo tanto, es importante conocer los aspectos clave de la monitorización de Linux. Si el sistema Linux sobre el que se ejecuta Hadoop en cada servidor tiene un cuello de botella o un problema de rendimiento, Hadoop no podrá tener un buen funcionamiento.

Las principales variables que se monitorizan en un sistema Linux son:

- ✓ **Uso de CPU**: es normal encontrar picos de consumo de CPU, aunque éstos no deberían ser constantes o durar más de unos segundos. Es importante revisar la media de consumo además, así como los procesos que consumen procesador. El principal comando para monitorizar el uso de CPU es `top`.
- ✓ **Uso de la memoria**: la memoria es uno de los primeros lugares en los que se debe buscar cuando hay problemas de rendimiento. Si tiene memoria (RAM) inadecuada en el servidor, el sistema puede ralentizarse debido al intercambio excesivo de memoria. El intercambio de memoria significa que el sistema está transfiriendo páginas de memoria a dispositivos de disco para liberar memoria para otros procesos. Los comandos que se utilizan principalmente para monitorizar la memoria son `vmstat`, `meminfo` y `free`.
- ✓ **Almacenamiento en disco**: cuando se trata de monitorizar discos se debe buscar dos cosas: en primer lugar, hay que verificar que no se esté quedando sin espacio: las aplicaciones agregan más datos de forma continua y es inevitable que tenga que agregar más espacio de almacenamiento constantemente. En segundo lugar, observar el rendimiento del disco: ¿hay cuellos de botella debido a un rendimiento lento de entrada/salida del disco? Los comandos utilizados para monitorizar el disco

son fundamentalmente `iostat` y `sar`.

- ✔ **Tráfico de red:** la red es un componente importante de su sistema: si las conexiones de red son lentas, Hadoop funcionará lentamente. Las estadísticas de red simples, como la cantidad de bytes recibidos y enviados, ayudarán a identificar problemas de red. El principal comando para monitorizar la red es `dstat`.

Además de la monitorización de los sistemas Linux sobre los que se ejecuta, Hadoop dispone de varias herramientas de monitorización o administración que veremos a continuación:

- ✔ Dos interfaces sencillas que proporcionan el ResourceManager y el Namenode.
- ✔ Apache Ambari y Cloudera Manager como herramientas de administración y monitorización más complejas.
- ✔ Ganglia como herramienta de monitorización general para clústers de servidores.

Para saber más

En la web oficial de Apache Hadoop, en la [página de documentación](#), encontrarás el significado de todas las variables de configuración de Hadoop.

En el menú de la izquierda, lo encontrarás en la parte inferior, en la zona "Configuration".

Autoevaluación

¿Por qué es importante monitorizar las métricas de los sistemas operativos Linux sobre los que se ejecuta Hadoop?

- Porque toda la configuración está en ficheros XML.
- Porque Linux es un sistema inestable y puede originar problemas de ejecución a Hadoop.
- Porque Hadoop supone que hay muchos servicios ejecutándose sobre la misma máquina, y podría haber conflicto entre los servicios que se podrían visualizar a nivel de sistema operativo.

Incorrecto: dónde se almacena la información es irrelevante para la monitorización.

Incorrecto: Linux es un sistema muy estable. En caso de que tuviera problemas el sistema operativo, desde luego afectarían a Hadoop, pero en este caso, no podemos considerar que sea una respuesta correcta porque Linux no es un sistema inestable.

Correcto: es importante monitorizar las métricas del sistema operativo ya que al fin y al cabo Hadoop se ejecuta sobre él, pidiéndole los recursos de memoria y procesador, y además, Hadoop implica que cada Linux tenga muchos procesos ejecutándose en la misma máquina.

Solución

1. Incorrecto
2. Incorrecto
3. Opción correcta

2.- Interfaz de HDFS: Namenode UI.

Caso práctico

María Robles, la responsable de IT del Banco Español de Inversiones, BEI, tiene el reto de administrar con su equipo la plataforma Hadoop implantada en el banco, que ya dispone de más de 20 usuarios entre científicos de datos y analistas de negocio.

En primer lugar le preocupa conocer el estado de la capa de almacenamiento, que al fin y al cabo es la base de toda la plataforma. Han ingestado 120 terabytes de datos provenientes del mainframe y de otras herramientas, y le preocupa que pueda haber datos corruptos, ficheros con bloques que no tienen un nivel suficiente de replicación, o simplemente, que se puedan caer nodos sin que su equipo se dé cuenta.

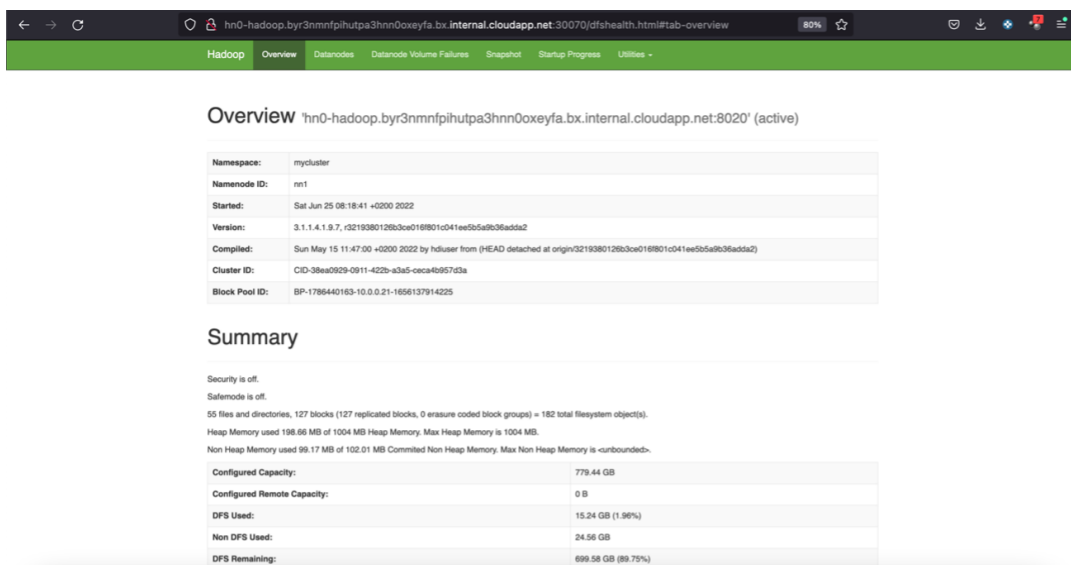
En primer lugar va a estudiar el interfaz que ofrece HDFS por defecto, que se llama Namenode UI.



ds_30 (Dominio público)

HDFS ofrece por defecto una web de administración denominada Namenode UI, o Web UI. Se trata de un servicio web que se arranca en el servidor donde se ejecuta el Namenode, por defecto en el puerto 50070, aunque la ruta se puede configurar en el parámetro `dfs.http.address` del fichero de configuración `hdfs-site.xml`.

Al acceder a esta web se muestra una pantalla de información general:



Overview 'hn0-hadoop.byr3nmfpihutpa3hnn0oxeyfa.bx.internal.cloudapp.net:8020' (active)	
Namespace:	mycluster
Namenode ID:	nn1
Started:	Sat Jun 25 08:18:41 +0200 2022
Version:	3.1.1.4.1.9.7, r3219380128b3ce0168b01c041ee5b5a9c36adda2
Compiled:	Sun May 15 11:47:00 +0200 2022 by hdluser from (HEAD detached at origin/3219380128b3ce0168b01c041ee5b5a9c36adda2)
Cluster ID:	CID-38ea0929-0911-422b-a3a5-ccca4b95743a
Block Pool ID:	BP-1786440163-10.0.0.21-1656137914225

Summary	
Security is off.	
Safemode is off.	
55 files and directories, 127 blocks (127 replicated blocks, 0 erasure coded block groups) = 182 total filesystem object(s).	
Heap Memory used 198.66 MB of 1004 MB Heap Memory. Max Heap Memory is 1004 MB.	
Non Heap Memory used 99.17 MB of 102.01 MB Committed Non Heap Memory. Max Non Heap Memory is <unbounded>.	
Configured Capacity:	779.44 GB
Configured Remote Capacity:	0 B
DFS Used:	15.24 GB (1.96%)
Non DFS Used:	24.56 GB
DFS Remaining:	699.58 GB (89.75%)

Íñigo Sanz (Dominio público)

Accediendo a la pestaña de Datanodes, se obtiene la información sobre los diferentes Datanodes, viendo su dirección IP, cuándo se obtuvo un mensaje de check por última vez, la capacidad y el uso de disco, así como algún dato menos importante:

Summary

Security is off.
Safemode is off.
55 files and directories, 127 blocks (127 replicated blocks, 0 erasure coded block groups) = 182 total filesystem object(s).
Heap Memory used 198.66 MB of 1004 MB Heap Memory. Max Heap Memory is 1004 MB.
Non Heap Memory used 99.17 MB of 102.01 MB Committed Non Heap Memory. Max Non Heap Memory is «unbounded».

Configured Capacity:	779.44 GB
Configured Remote Capacity:	0 B
DFS Used:	15.24 GB (1.96%)
Non DFS Used:	24.56 GB
DFS Remaining:	699.58 GB (89.75%)
Block Pool Used:	15.24 GB (1.96%)
DataNodes usages% (Min/Median/Max/stdDev):	0.00% / 0.00% / 7.82% / 3.39%
Live Nodes	4 (Decommissioned: 0, In Maintenance: 0)
Dead Nodes	0 (Decommissioned: 0, In Maintenance: 0)
Decommissioning Nodes	0
Entering Maintenance Nodes	0
Total Datanode Volume Failures	0 (0 B)
Number of Under-Replicated Blocks	0
Number of Blocks Pending Deletion	0
Block Deletion Start Time	Sat Jun 25 08:28:41 +0200 2022
Last Checkpoint Time	Sat Jun 25 08:18:35 +0200 2022

Íñigo Sanz (Dominio público)

Pinchando sobre alguno de los Datanodes que aparecen en el listado, se puede ver el detalle sobre los bloques que contiene, la capacidad, la ruta de HDFS en el disco local, etc.

Datanode Information

Legend: ✓ In service ● Down ○ Decommissioned ○ Decommissioned & dead ○ In Maintenance & dead

Datanode usage histogram

Disk usage of each Datanode (%)

In operation

Show 25 entries

Node	Http Address	Last contact	Last Block Report	Capacity	Blocks	Block pool used	Version
✓ dn0 hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210 (10.0.0.12:30210)	http://hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210	7s	29m	194.86 GB (8%)	0	44 KB	3.1.1.4.1.8.7
✓ dn1 hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210 (10.0.0.14:30210)	http://hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210	1s	29m	194.86 GB (7.87%)	124	13.24 GB	3.1.1.4.1.8.7
✓ dn2 hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210 (10.0.0.13:30210)	http://hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210	0s	17m	194.86 GB (8%)	0	40 KB	3.1.1.4.1.8.7
✓ dn3 hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210 (10.0.0.11:30210)	http://hadoop-byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30210	0s	167m	194.86 GB (8%)	3	324 KB	3.1.1.4.1.8.7

Showing 1 to 4 of 4 entries

Íñigo Sanz (Dominio público)

En esta misma pantalla, en la pestaña Utilities, se puede acceder a los logs de HDFS en el Datanode:


```

4/4... 2022-06-29 08:58:13,528 INFO impl.FdatasetAsyncDiskService - Deleted BP-1786448163-10.0.0.21-1656137914225 blk_1873742124_1380 URI file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742124 for deletion
2022-06-29 08:58:14,538 INFO impl.FdatasetAsyncDiskService - Deleted BP-1786448163-10.0.0.21-1656137914225 blk_1873742138_1380 URI file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742138
2022-06-29 08:58:35,267 INFO datanode.Datanode - Receiving BP-1786448163-10.0.0.21-1656137914225:blk_1873742138_1380 src:/10.0.0.21:47088 dest:/10.0.0.12:38818
2022-06-29 08:58:35,272 INFO datanode.clienttrace - src:/10.0.0.21:47088 dest:/10.0.0.12:38818 bytes: 80 op: HDFS_WRITE, cli10: DFSClient_30844602625_55925317_1, offset: 0, srvID: 9c4336cc-924b-4182-bb34-260a8669294, blockID: BP-1786448163-10.0.0.21-1656137914225:blk_1873742138_1380, duration(ms): 7886262
2022-06-29 08:58:35,278 INFO datanode.Datanode - PacketResponder: BP-1786448163-10.0.0.21-1656137914225:blk_1873742138_1380, type=LAST_IN_PIPELINE terminating
2022-06-29 08:58:38,558 INFO impl.FdatasetAsyncDiskService - ScheduLink blk_1873742138_1380 replica FinalizedReplica, blk_1873742138_1380, FINALIZED
getDumpres() = 80
getTycatondisk() = 80
getVisibleLength() = 80
getVolume() = /mnt/resource/hadoop/hdfs/data
getBlockURI() = file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742138 for deletion
2022-06-29 08:59:38,557 INFO impl.FdatasetAsyncDiskService - Deleted BP-1786448163-10.0.0.21-1656137914225 blk_1873742138_1380 URI file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742138
2022-06-29 08:59:39,886 INFO datanode.Datanode - Receiving BP-1786448163-10.0.0.21-1656137914225:blk_1873742133_1389 src:/10.0.0.21:58778 dest:/10.0.0.12:38818
2022-06-29 08:59:39,912 INFO datanode.clienttrace - src:/10.0.0.21:58778 dest:/10.0.0.12:38818 bytes: 80 op: HDFS_WRITE, cli10: DFSClient_30844602625_55925317_1, offset: 0, srvID: 9c4336cc-924b-4182-bb34-260a8669294, blockID: BP-1786448163-10.0.0.21-1656137914225:blk_1873742133_1389, duration(ms): 26648836
2022-06-29 08:59:39,916 INFO datanode.Datanode - PacketResponder: BP-1786448163-10.0.0.21-1656137914225:blk_1873742133_1389, type=LAST_IN_PIPELINE terminating
2022-06-29 08:59:38,557 INFO impl.FdatasetAsyncDiskService - ScheduLink blk_1873742133_1389 replica FinalizedReplica, blk_1873742133_1389, FINALIZED
getDumpres() = 80
getTycatondisk() = 80
getVisibleLength() = 80
getVolume() = /mnt/resource/hadoop/hdfs/data
getBlockURI() = file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742133 for deletion
2022-06-29 08:59:38,557 INFO impl.FdatasetAsyncDiskService - Deleted BP-1786448163-10.0.0.21-1656137914225 blk_1873742133_1389 URI file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742133
2022-06-29 09:03:43,313 INFO datanode.Datanode - Receiving BP-1786448163-10.0.0.21-1656137914225:blk_1873742137_1313 src:/10.0.0.21:48772 dest:/10.0.0.12:38818
2022-06-29 09:03:43,327 INFO datanode.clienttrace - src:/10.0.0.21:48772 dest:/10.0.0.12:38818 bytes: 80 op: HDFS_WRITE, cli10: DFSClient_30844602625_55925317_1, offset: 0, srvID: 9c4336cc-924b-4182-bb34-260a8669294, blockID: BP-1786448163-10.0.0.21-1656137914225:blk_1873742137_1313, duration(ms): 5484343
2022-06-29 09:03:43,328 INFO datanode.Datanode - PacketResponder: BP-1786448163-10.0.0.21-1656137914225:blk_1873742137_1313, type=LAST_IN_PIPELINE terminating
2022-06-29 09:03:47,569 INFO impl.FdatasetAsyncDiskService - ScheduLink blk_1873742137_1313 replica FinalizedReplica, blk_1873742137_1313, FINALIZED
getDumpres() = 80
getTycatondisk() = 80
getVisibleLength() = 80
getVolume() = /mnt/resource/hadoop/hdfs/data
getBlockURI() = file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742137 for deletion
2022-06-29 09:03:47,571 INFO impl.FdatasetAsyncDiskService - Deleted BP-1786448163-10.0.0.21-1656137914225 blk_1873742137_1313 URI file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742137
2022-06-29 09:08:31,283 INFO datanode.Datanode - Receiving BP-1786448163-10.0.0.21-1656137914225:blk_1873742141_1317 src:/10.0.0.21:32944 dest:/10.0.0.12:38818
2022-06-29 09:08:31,303 INFO datanode.clienttrace - src:/10.0.0.21:32944 dest:/10.0.0.12:38818 bytes: 80 op: HDFS_WRITE, cli10: DFSClient_30844602625_55925317_1, offset: 0, srvID: 9c4336cc-924b-4182-bb34-260a8669294, blockID: BP-1786448163-10.0.0.21-1656137914225:blk_1873742141_1317, duration(ms): 16788434
2022-06-29 09:08:31,304 INFO datanode.Datanode - PacketResponder: BP-1786448163-10.0.0.21-1656137914225:blk_1873742141_1317, type=LAST_IN_PIPELINE terminating
2022-06-29 09:08:35,592 INFO impl.FdatasetAsyncDiskService - ScheduLink blk_1873742141_1317 replica FinalizedReplica, blk_1873742141_1317, FINALIZED
getDumpres() = 80
getTycatondisk() = 80
getVisibleLength() = 80
getVolume() = /mnt/resource/hadoop/hdfs/data
getBlockURI() = file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742141 for deletion
2022-06-29 09:08:35,594 INFO impl.FdatasetAsyncDiskService - Deleted BP-1786448163-10.0.0.21-1656137914225 blk_1873742141_1317 URI file:/mnt/resource/hadoop/hdfs/data/current/BP-1786448163-10.0.0.21-1656137914225/current/finalized/subdir/subdir1/blk_1873742141

```

Íñigo Sanz (Dominio público)

Volviendo a la pantalla principal, se puede encontrar otras pestañas:

- ✔ **Datanode Volume Failures:** en la que se muestran los errores en disco que se han detectado en HDFS.
- ✔ **Snapshot:** en caso de haberse realizado una copia de la estructura de HDFS para poder restaurarlo, en esta pantalla aparecen todos los puntos generados.
- ✔ **Startup progress:** muestra cómo ha sido el proceso de arranque de HDFS y el estado en el que se encuentra ahora:

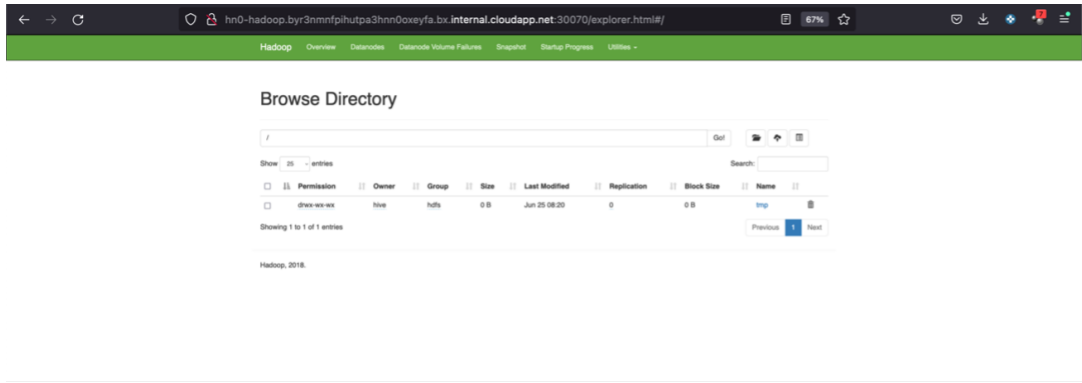
Elapsed Time: 2 sec. Percent Complete: 100%

Phase	Completion	Elapsed Time
Loading fsimage hadoop/hdfs/hamond/current/fsimage_0000000000000000000000 371 B	100%	1 sec
ensuring coding policies (DS)	100%	
inodes (11)	100%	
delegation tokens (DS)	100%	
cache pools (DS)	100%	
Loading edits	100%	0 sec
Saving checkpoint	100%	0 sec
Safe mode	100%	0 sec
awaiting reported blocks (DS)	100%	

Hadoop, 2018.

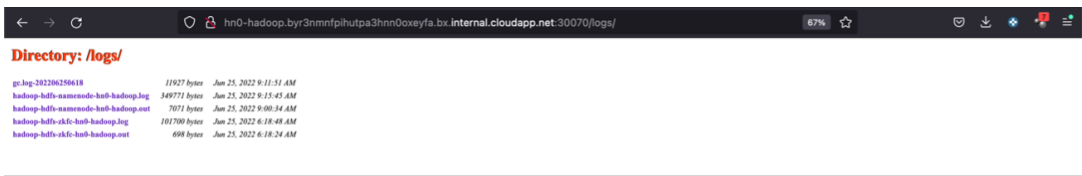
Íñigo Sanz (Dominio público)

- Por último, en la pestaña de **Utilidades**, se puede encontrar diferentes funcionalidades, como las siguientes:
 - ✔ Visualizar el contenido de HDFS:

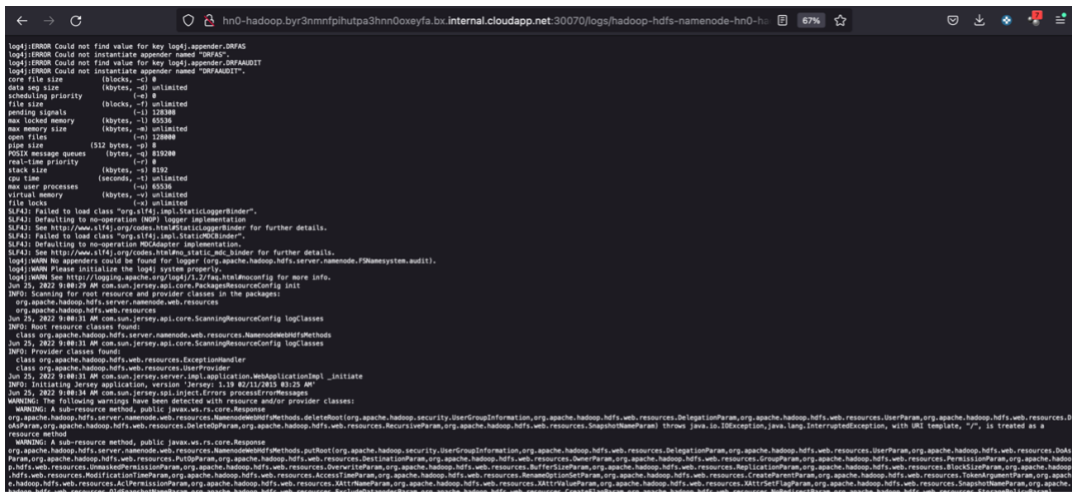


Íñigo Sanz (Dominio público)

Ver los logs de HDFS:



Íñigo Sanz (Dominio público)



Íñigo Sanz (Dominio público)

El fichero de configuración de HDFS:

```
hno-hadoop.byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30070/conf
Este fichero XML no parece tener ninguna información de estilo asociada. Se muestra debajo el árbol del documento.
<configuration>
  <property>
    <name>mapreduce.jobhistory.hist.format</name>
    <value>binary</value>
  </property>
  <property>
    <name>mapred-default.xml</source>
  </property>
  <property>
    <name>fs.s3a.retry.interval</name>
    <value>300ms</value>
  </property>
  <property>
    <name>oozie-default.xml</source>
  </property>
  <property>
    <name>hadoop.proxyuser.hive.group</name>
    <value>*hadoop</value>
  </property>
  <property>
    <name>oozie-site.xml</source>
  </property>
  <property>
    <name>dfs.block.access.token.lifetime</name>
    <value>400</value>
  </property>
  <property>
    <name>hdfs-default.xml</source>
  </property>
  <property>
    <name>mapreduce.application.framework.path</name>
    <value>
      hadoop3 [hdp.version]/mapreduce/mapreduce.tar.gz#framework
    </value>
  </property>
  <property>
    <name>mapred-site.xml</source>
  </property>
  <property>
    <name>mapreduce.job.heap.memory.mb.ratio</name>
    <value>0.5</value>
  </property>
  <property>
    <name>mapred-default.xml</source>
  </property>
  <property>
    <name>mapreduce.map.log.level</name>
    <value>INFO</value>
  </property>
  <property>
    <name>mapred-site.xml</source>
  </property>
  <property>
    <name>dfs.namenode.lazypersist.file.scrub.interval.sec</name>
    <value>3000</value>
  </property>
  <property>
    <name>mapred-site.xml</source>
  </property>
  <property>
    <name>hdfs-default.xml</source>
  </property>
</configuration>
```

Íñigo Sanz (Dominio público)

✔ Y un volcado de los hilos en ejecución en HDFS para poder analizar errores:

```
hno-hadoop.byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net:30070/stacks
Process Thread Dump:
213 active threads
Thread 1195 [IPC Client (1129231526) connection to s45-hadoop.byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net/18.8.8.18:8485 from hdfs]:
  State: TIMED_WAITING
  Blocked count: 17
  Waited count: 18
  Stack:
    java.lang.Object.wait(Native Method)
    org.apache.hadoop.ipc.ClientConnection.waitForWork(Client, java:1818)
    org.apache.hadoop.ipc.ClientConnection.run(Client, java:1822)
Thread 1196 [IPC Client (1129231526) connection to s42-hadoop.byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net/18.8.8.1:8485 from hdfs]:
  State: TIMED_WAITING
  Blocked count: 17
  Waited count: 18
  Stack:
    java.lang.Object.wait(Native Method)
    org.apache.hadoop.ipc.ClientConnection.waitForWork(Client, java:1818)
    org.apache.hadoop.ipc.ClientConnection.run(Client, java:1822)
Thread 1198 [IPC Client (1129231526) connection to s41-hadoop.byr3nmfpihupa3hnn0oxeyfa.bx.internal.cloudapp.net/18.8.8.5:8485 from hdfs]:
  State: TIMED_WAITING
  Blocked count: 17
  Waited count: 18
  Stack:
    java.lang.Object.wait(Native Method)
    org.apache.hadoop.ipc.ClientConnection.waitForWork(Client, java:1818)
    org.apache.hadoop.ipc.ClientConnection.run(Client, java:1822)
Thread 1909 [IPC Parameter Sending Thread #228]:
  State: TIMED_WAITING
  Blocked count: 8
  Waited count: 41
  Stack:
    sun.misc.Unsafe.park(Native Method)
    java.util.concurrent.locks.LockSupport.parkNanos(LockSupport, java:115)
    java.util.concurrent.SynchronousQueue$TransferStack.awaitForWaitFills(SynchronousQueue, java:488)
    java.util.concurrent.SynchronousQueue$TransferStack.transfer(SynchronousQueue, java:382)
    java.util.concurrent.SynchronousQueue.poll(SynchronousQueue, java:943)
    java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor, java:1873)
    java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor, java:1134)
    java.lang.Thread.run(Thread, java:748)
Thread 1878 [IPC Parameter Sending Thread #219]:
  State: TIMED_WAITING
  Blocked count: 8
  Waited count: 35
  Stack:
    sun.misc.Unsafe.park(Native Method)
    java.util.concurrent.locks.LockSupport.parkNanos(LockSupport, java:115)
    java.util.concurrent.SynchronousQueue$TransferStack.awaitForWaitFills(SynchronousQueue, java:488)
    java.util.concurrent.SynchronousQueue$TransferStack.transfer(SynchronousQueue, java:382)
    java.util.concurrent.SynchronousQueue.poll(SynchronousQueue, java:943)
    java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor, java:1873)
    java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor, java:1134)
    java.lang.Thread.run(Thread, java:748)
Thread 1887 [IPC Parameter Sending Thread #218]:
  State: TIMED_WAITING
  Blocked count: 8
  Waited count: 36
  Stack:
    sun.misc.Unsafe.park(Native Method)
    java.util.concurrent.locks.LockSupport.parkNanos(LockSupport, java:115)
    java.util.concurrent.SynchronousQueue$TransferStack.awaitForWaitFills(SynchronousQueue, java:488)
    java.util.concurrent.SynchronousQueue$TransferStack.transfer(SynchronousQueue, java:382)
    java.util.concurrent.SynchronousQueue.poll(SynchronousQueue, java:943)
    java.util.concurrent.ThreadPoolExecutor.getTask(ThreadPoolExecutor, java:1873)
    java.util.concurrent.ThreadPoolExecutor.runWorker(ThreadPoolExecutor, java:1134)
    java.lang.Thread.run(Thread, java:748)
```

Íñigo Sanz (Dominio público)

Autoevaluación

Indica si las siguientes afirmaciones son correctas sobre Namenode UI

Permite monitorizar el estado de HDFS, pero no permite realizar acciones de administración como para o arrancar el servicio, o modificar la configuración

Verdadero Falso

Verdadero

Verdadero: no ofrece funcionalidades para administrar el servicio.

Da muy poca información sobre HDFS, sólo la esencial.

Verdadero Falso

Falso

Falso: ofrece mucha información, prácticamente toda la necesaria para monitorizar el servicio HDFS, aunque es verdad que no tiene un formato muy amigable.

3.- Interfaz de YARN: ResourceManager UI.

Caso práctico

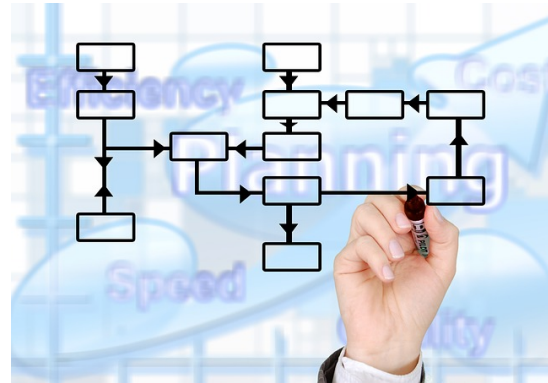
María Robles, la responsable de IT del Banco Español de Inversiones, BEI, tiene el reto de administrar con su equipo la plataforma Hadoop implantada en el banco, que ya dispone de más de 20 usuarios entre científicos de datos y analistas de negocio.

Los usuarios lanzan diferentes trabajos en el clúster, desde consultas a Hive bastante pesadas, por ejemplo, para hacer cálculos sobre inversiones pasadas, o tareas Spark para implementar o ejecutar modelos predictivos.

Esta diversidad de usuarios está creando los primeros problemas, y es que a veces, el clúster se ralentiza mucho y ciertas tareas que son bastante importantes, como los reportes al consejo de administración, no se pueden terminar en la ventana de tiempo que deberían.

María y su equipo van a estudiar cómo monitorizar las aplicaciones que se están ejecutando, en primer lugar, para detectar los momentos en los que el clúster está saturado, y en segundo lugar, para poder averiguar qué tareas son las que están ralentizando la plataforma.

Van a empezar a utilizar interfaz que ofrece YARN por defecto, que se llama ResourceManager UI.



Gerd Altmann (Dominio público)

Al igual que en el caso de HDFS, YARN también ofrece un interfaz web de monitorización que se suele desplegar en el nodo donde se ejecuta el servicio ResourceManager. Esta web permite ver el estado de ejecución de las aplicaciones, el estado de recursos del sistema, o ver el detalle de las aplicaciones y los logs que están generando.

Al acceder a la pantalla principal, se muestra un resumen del sistema:

Cluster Metrics														
Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	V-Cores Used	V-Cores Total	V-Cores Reserved				
0	0	2	0	0.0	102 GB	0.0	0.0	0	32	0				

Scheduler Metrics													
Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority									
Capacity Scheduler	[memory-mb (unit=MB), vcores]	<memory>0.01, <vCores>1	<memory>4096, <vCores>8	0									

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU vCores	Allocated Memory MB	Reserved CPU vCores	Reserved Memory MB	% of Queue	% of Cluster	Progress	Tracking ID	Stacktraced Nodes
application_156813707815_0001	hive	HIVE-4830a5d4-1f04-43b1-aa01-046803600000	TEZ	default	0	Sat Jun 25 10:20:00 +0200 2022	Sat Jun 25 10:20:28 +0200 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_156813707815_0002	hive	HIVE-4830a5d4-1f04-43b1-aa01-046803600000	TEZ	default	0	Sat Jun 25 08:23:00 +0200 2022	Sat Jun 25 08:23:19 +0200 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0

Íñigo Sanz (Dominio público)

En el menú de la izquierda se tienen diferentes opciones, por ejemplo, al pinchar en la opción Nodes se puede ver la información de los nodos worker, pudiendo comprobar cuántos hay, en qué estado se encuentran, la dirección interna del nodo dentro del clúster así como los recursos utilizados y disponibles.

Nodes of the cluster

Cluster Metrics: 2 Apps Submitted, 0 Apps Pending, 0 Apps Running, 2 Apps Completed, 0 Containers Running, 0 B Memory Used, 192 GB Memory Total, 0 B Memory Reserved, 0 V-Cores Used, 32 V-Cores Total, 0 V-Cores Reserved.

Node Labels	Rack	Node State	Node Address	Node HTTP Address	Last health-update	Health-report	Containers	Allocation Tags	Mem Used	Mem Avail	V-Cores Used	V-Cores Avail	Version
/default-rack		RUNNING	wn0-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30050	wn0-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30060	Sat Jun 25 10:01:37 +0000 2022		0		0 B	48 GB	0	8	3.1.14.1
/default-rack		RUNNING	wn1-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30050	wn1-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30060	Sat Jun 25 10:01:11 +0000 2022		0		0 B	48 GB	0	8	3.1.14.1
/default-rack		RUNNING	wn2-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30050	wn2-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30060	Sat Jun 25 10:01:59 +0000 2022		0		0 B	48 GB	0	8	3.1.14.1
/default-rack		RUNNING	wn3-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30050	wn3-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30060	Sat Jun 25 10:02:11 +0000 2022		0		0 B	48 GB	0	8	3.1.14.1

Íñigo Sanz (Dominio público)

En la opción Applications, se puede ver las aplicaciones que se han ejecutado o las que se están ejecutando en este momento. En este ejemplo se puede ver cómo existe una aplicación en ejecución, que se corresponde con una consulta Hive, y dos aplicaciones que se acaban de ejecutar.

All Applications

ID	User	Name	Application Type	Queue	Application Priority	Start Time	Finish Time	State	Final Status	Running Containers	Allocated CPU V-Cores	Allocated Memory MB	Reserved CPU V-Cores	Reserved Memory MB	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
application_1656137970815_0003	hive	HIVE-4630da54-f1c0-43b1-acdf-c569363959b	TEZ	default	0	Sat Jun 25 12:05:17 +0000 2022	N/A	RUNNING	UNDEFINED	1	1	3072	0	0	1.6	1.6		ApplicationMaster	0
application_1656137970815_0002	hive	HIVE-4630da54-f1c0-43b1-acdf-c569363959b	TEZ	default	0	Sat Jun 25 10:26:50 +0000 2022	Sat Jun 25 10:32:36 +0000 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0
application_1656137970815_0001	hive	HIVE-4630da54-f1c0-43b1-acdf-c569363959b	TEZ	default	0	Sat Jun 25 09:23:03 +0000 2022	Sat Jun 25 09:50:15 +0000 2022	FINISHED	SUCCEEDED	N/A	N/A	N/A	N/A	N/A	0.0	0.0		History	0

Íñigo Sanz (Dominio público)

Haciendo clic sobre una aplicación, se puede consultar su detalle de ejecución. Por ejemplo, entrando en el detalle de la segunda aplicación ejecutada, se obtiene la siguiente información:

Application application_1656137970815_0002

Application Overview

User: hive
 Application Name: HIVE-4630da54-f1c0-43b1-acdf-c569363959b
 Application Type: TEZ
 Application Tags: user=admin,hive_20220625072243_28b7cfe-3c2f-4580-aabc-44796e723c88
 Application Priority: 0 (Higher Integer value indicates higher priority)
 Yarn Application State: FINISHED
 Queue: default
 Final Status Reported by AM: SUCCEEDED
 Started: Sat Jun 25 09:26:50 +0000 2022
 Elapsed: 5mins, 44sec
 Tracking URL: History
 Log Aggregation Status: SUCCEEDED
 Application Timeout (Remaining Time): Unlimited
 Diagnostics: Session timed out, lastDAGCompletionTime=1656145629569 ms, sessionTimeoutInterval=300000 ms, Session stats.submittedDAGs=1, successSubDAGs=1, failedDAGs=0, failedDAGs=0
 Unmanaged Application: false
 Application Node Label expression: <not set>
 AM container Node Label expression: <DEFAULT_PARTITION>

Application Metrics

Total Resource Preempted: <memory:0, vCores:0>
 Total Number of Non-AM Containers Preempted: 0
 Total Number of AM Containers Preempted: 0
 Resource Preempted from Current Attempt: <memory:0, vCores:0>
 Number of Non-AM Containers Preempted from Current Attempt: 0
 Aggregate Resource Allocation: 1125967 MB-seconds, 366 vcore-seconds
 Aggregate Preempted Resource Allocation: 0 MB-seconds, 0 vcore-seconds

Attempt ID	Started	Node	Logs	Nodes blacklisted by the app	Nodes blacklisted by the system
appattemp_1656137970815_0002_000001	Sat Jun 25 10:26:50 +0000 2022	http://wn2-hadoop.by3nrmfphupa3hnn0oexyfa.bx.internal.cloudapp.net:30060	Logs	0	0

Íñigo Sanz (Dominio público)

Se puede ver que la aplicación necesitó 5 minutos y 44 segundos para ejecutarse, que lo lanzó el usuario “hive” (es el usuario con el que se lanzan las consultas de Hive por defecto). Asimismo, se puede comprobar que sólo necesitó ejecutarse en un nodo, el nodo wn2, y pinchando sobre el enlace “Logs” se puede consultar el log de ejecución por si fuera necesario analizar un fallo que hubiera podido ocurrir en dicha ejecución:

The screenshot shows a Hadoop job log page. The top part displays the job ID and upload time. Below that, there's a section for 'Log Type: directory.info' showing a list of files and their sizes. The bottom part shows 'Log Type: launch_container.sh' with environment variables and user information.

Íñigo Sanz (Dominio público)

El resto de opciones que aparecen dentro de la opción “Applications” sólo sirven para filtrar las aplicaciones que se muestran según su estado: en ejecución, finalizadas, paradas, etc.

En cuanto a la opción “Scheduler”, muestra el estado de las colas de ejecución que se han configurado en YARN, mostrando la capacidad de cada cola, su ocupación, etc. En el siguiente ejemplo, por ejemplo, se puede ver que hay dos colas configuradas, default y joblauncher:

The screenshot shows the Hadoop Scheduler interface. It displays various metrics for the cluster, including Apps Submitted, Pending, Running, and Completed. Below the metrics, there's a section for 'Application Queues' with a legend and a table showing the status of different queues. The 'default' queue is shown as 'Used' and 'joblauncher' as 'Auto Created Queues'.

Íñigo Sanz (Dominio público)

Haciendo clic sobre una cola, se puede ver su detalle, donde por ejemplo podemos comprobar que la cola

joblauncher tiene una capacidad máxima del 50% de los recursos de YARN:

The screenshot shows the Hadoop YARN web interface. The main heading is "NEW,NEW_SAVING,SUBMITTED,ACCEPTED,RUNNING Applications". The interface is divided into several sections: Cluster Metrics, Cluster Nodes Metrics, Scheduler Metrics, Application Queues, and Tools. The Application Queues section shows the 'joblauncher' queue with a state of 'RUNNING'. The queue status is detailed as follows:

Queue State:	RUNNING
Used Capacity:	<memory:0, vCores:0> (0.0%)
Configured Capacity:	<memory:0, vCores:0>
Configured Max Capacity:	unlimited
Effective Capacity:	<memory:9830, vCores:1> (5.0%)
Effective Max Capacity:	<memory:98304, vCores:16> (50.0%)
Absolute Used Capacity:	0.0%
Absolute Configured Capacity:	5.0%
Absolute Configured Max Capacity:	50.0%
Used Resources:	<memory:0, vCores:0>
Configured Max Application Master Limit:	33.0
Max Application Master Resources:	<memory:33792, vCores:1>
Used Application Master Resources:	<memory:0, vCores:0>
Max Application Master Resources Per User:	<memory:33792, vCores:1>
Num Schedulable Applications:	0
Num Non-Schedulable Applications:	0
Num Containers:	0
Max Applications:	500
Max Applications Per User:	500
Configured Minimum User Limit Percent:	100%
Configured User Limit Factor:	10.0
Accessible Node Labels:	*
Ordering Policy:	FifoOrderingPolicy
Preemption:	disabled
Intra-queue Preemption:	disabled
Default Node Label Expression:	<DEFAULT_PARTITION>
Default Application Priority:	0

Íñigo Sanz (Dominio público)

Por último, en la opción "Tools" disponemos de diferentes utilidades al igual que en el interfaz del Namenode:

- ✔ Ver la configuración de YARN.
- ✔ Ver los ficheros de log.
- ✔ Ver los hilos de ejecución en un momento dado para depurar errores.
- ✔ Ver las métricas actuales en formato JSON.

Autoevaluación

¿Cuál de las siguientes funcionalidades ofrece el ResourceManager UI?

- Ver el total de memoria y núcleos de proceso que YARN puede utilizar y su consumo actual

- Parar tareas que están consumiendo muchos recursos.

- Ver qué aplicaciones se están ejecutando y cuántos recursos está consumiendo cada una.

- Ver qué nodos worker hay en el clúster ejecutando YARN y en qué estado se encuentran.

Mostrar retroalimentación

Solución

1. Correcto
2. Incorrecto
3. Correcto
4. Correcto

4.- Apache Ambari.

Caso práctico

El equipo de IT del Banco Español de Inversiones, BEI, ya puede monitorizar HDFS y las tareas que se ejecutan en el clúster Hadoop y que son lanzadas por diferentes usuarios, tanto usuarios de negocio como aplicaciones o científicos de datos.

Tenían un problema por el que a veces el clúster tenía un rendimiento muy pobre, y gracias al interfaz de usuario de ResourceManager, con el que pudieron identificar la aplicación que consumía muchos recursos y que hacía que el resto de aplicaciones se ejecutaran de una forma muy lenta. Esta aplicación resultó ser unas consultas que un analista de negocio estaba lanzando con Hive, en las que unía múltiples tablas y que generaba consultas que tardaban varias horas en terminar. Pudieron detectarlo con ayuda de los logs que pudieron ver desde el interfaz del ResourceManager, y ayudaron a este analista a optimizar sus consultas.

Ahora bien, necesitan una herramienta que les permita unificar toda la monitorización y sobre todo, poder administrar el sistema, es decir, reiniciar servicios, parar o arrancar nodos, instalar componentes, cambiar la configuración, etc.

Saben que Apache Ambari es una herramienta específica para este propósito, y van a probarla.



[Gerd Altmann](#) (Dominio público)

Apache Ambari tiene como objetivo simplificar la administración de Hadoop para el aprovisionamiento, la administración y el monitoreo de clústeres. Ambari proporciona una interfaz de usuario web de administración de Hadoop intuitiva y fácil de usar respaldada por sus [API RESTful](#).



Apache Ambari

[Apache Software Foundation](#) (Apache License)

Ambari permite a los administradores del sistema:

- ✓ **Instalar** un clúster de Hadoop:
 - ◆ Ambari proporciona un asistente paso a paso para instalar los servicios de Hadoop en cualquier número de hosts.
 - ◆ Ambari maneja la configuración de los servicios de Hadoop para el clúster.
- ✓ **Administrar** un clúster Hadoop, ya que proporciona funcionalidades para iniciar, detener y reconfigurar los servicios de Hadoop en todo el clúster, así como los diferentes nodos o servidores que lo componen.
- ✓ **Monitorizar** un clúster Hadoop, ya que:
 - ◆ Ambari proporciona cuadros de mando para monitorear la salud y el estado del clúster de Hadoop.
 - ◆ Ambari permite definir alertas y notificará cuando se cumpla la condición de la alerta (por ejemplo, un nodo deja de funcionar, el espacio restante en el disco es bajo, etc.).

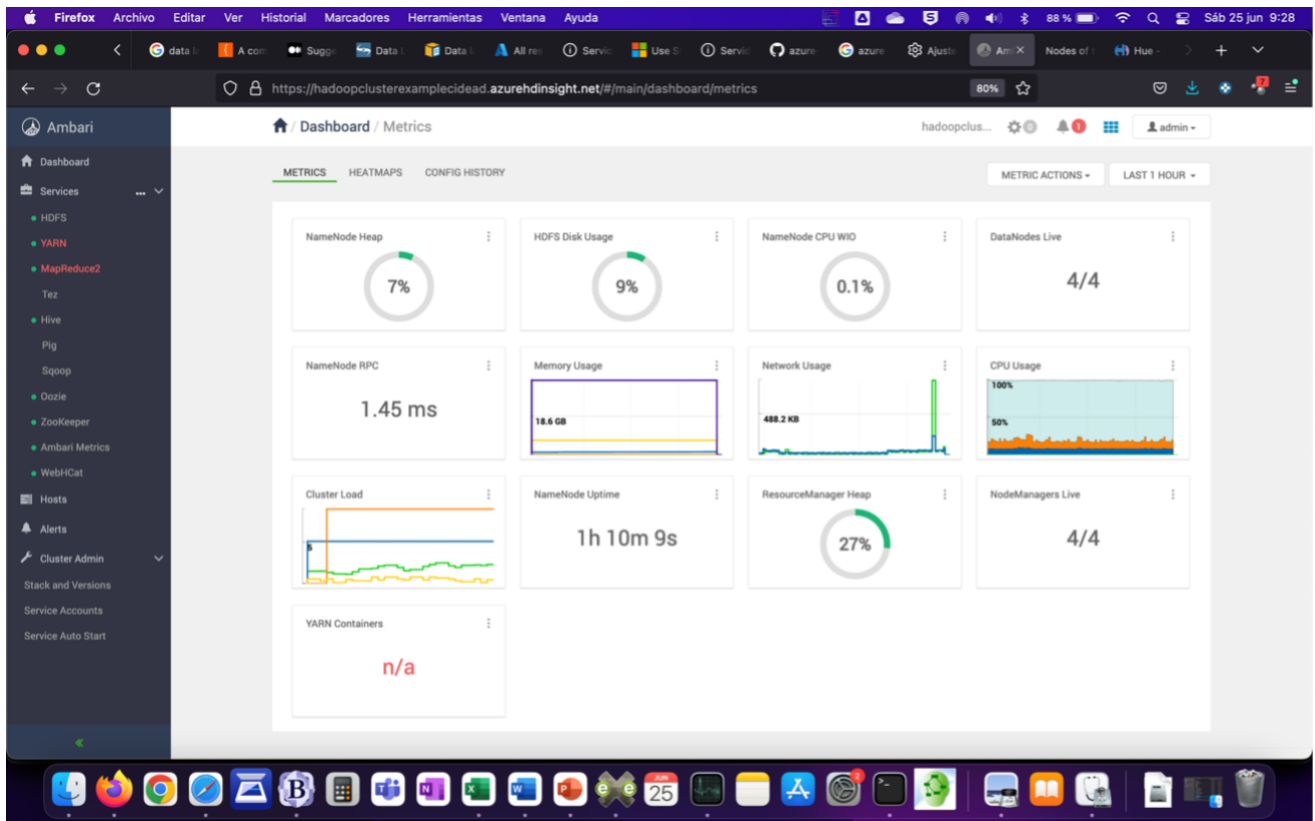
Además, Ambari dispone de un API con el que las aplicaciones pueden utilizar los servicios indicados anteriormente, para poder ejecutar cualquier funcionalidad de administración de forma automática.

A continuación vamos a ver las principales pantallas y funcionalidades de Ambari para la administración y

monitorización de plataformas Hadoop:

Pantalla inicial

Al acceder a Ambari, se obtiene un cuadro de mando general con las principales métricas.

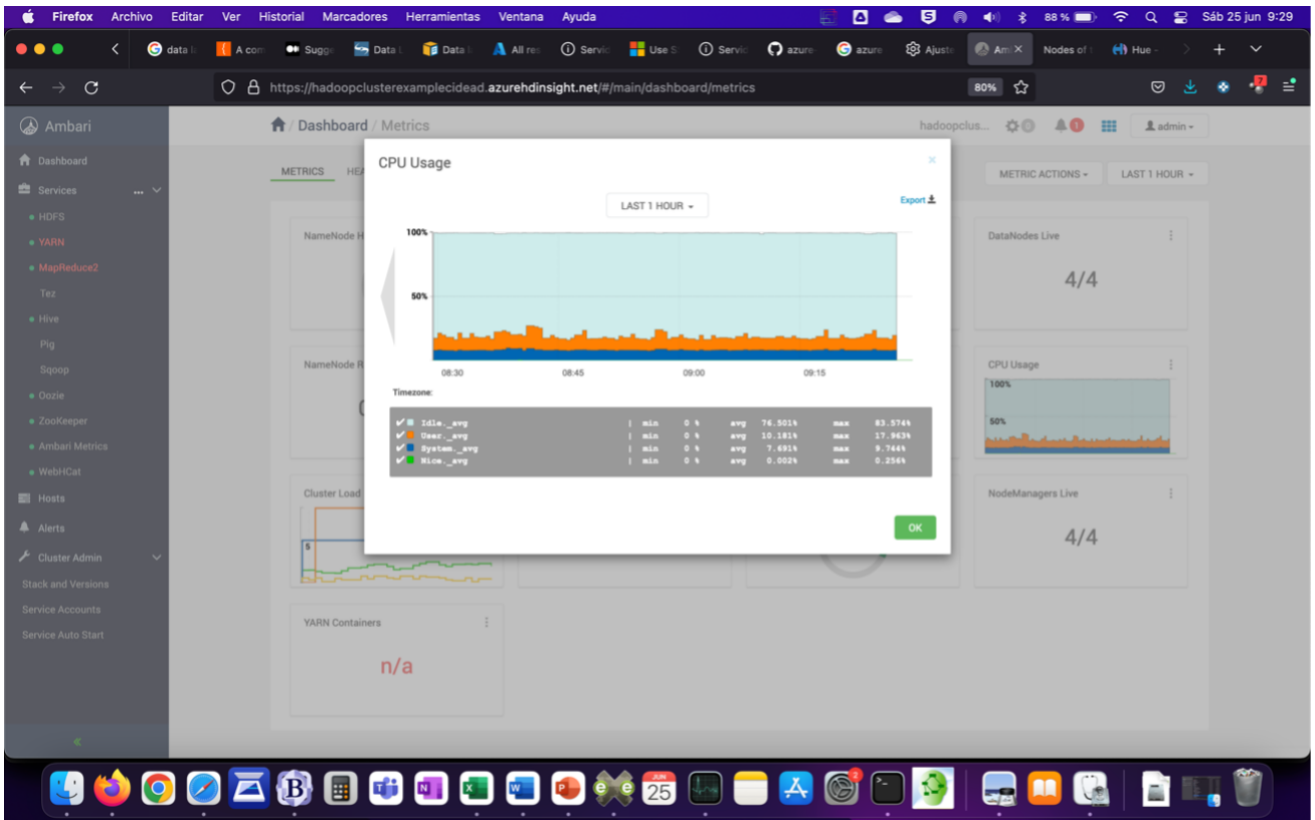


Íñigo Sanz (Dominio público)

Se puede ver algunas métricas importantes:

- ✓ El espacio usado por HDFS del total de espacio disponible, un 9%.
- ✓ El uso de CPU y memoria del Namenode, 0,1% y 7% respectivamente.
- ✓ El número de datanodes y cuántos están en servicio: 4.
- ✓ El uso de red.
- ✓ El uso de CPU general.
- ✓ El número de Nodemanagers y cuántos están activos: 4.
- ✓ Algún dato adicional.

Se puede hacer zoom sobre una métrica para ver el detalle:

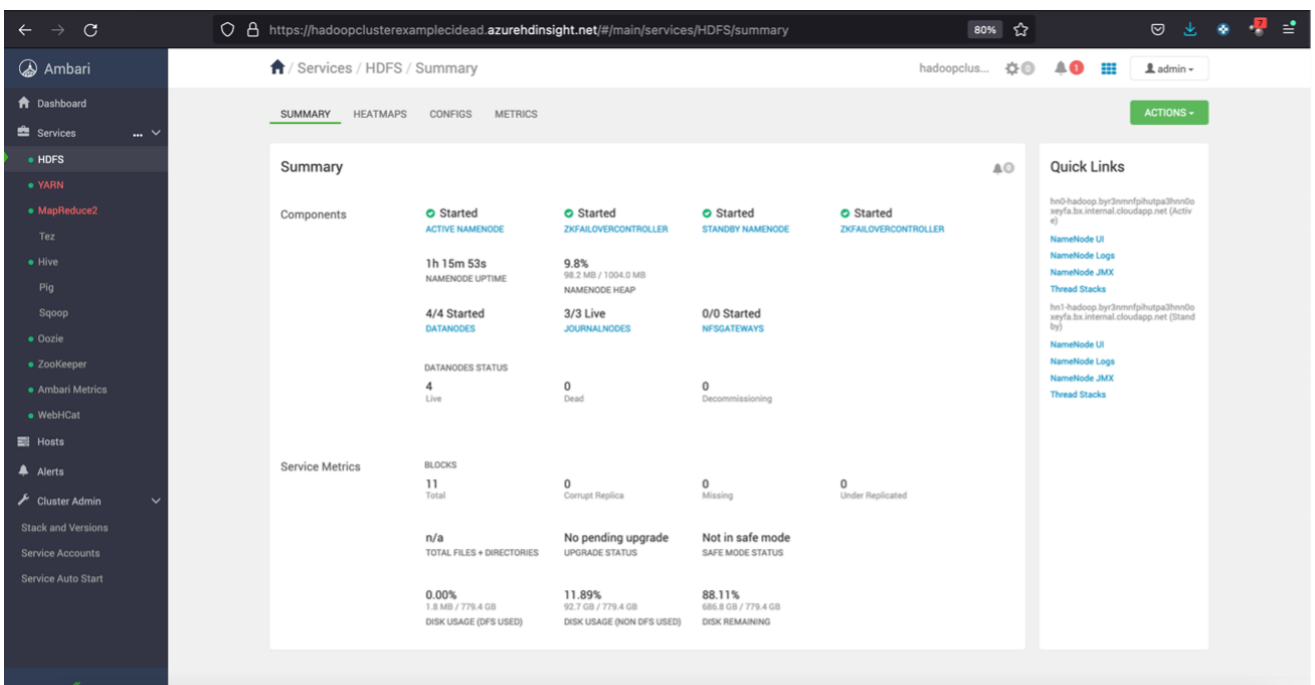


Íñigo Sanz (Dominio público)

En el menú de la izquierda se puede seleccionar qué tipo de acción realizar, se divide en servicios (HDFS, YARN, etc.), hosts (servidores del clúster), alertas, y otras acciones de administración.

Servicios

Entrando en servicios, podemos acceder a cada uno de los servicios o componentes del clúster para administrarlo (parar, arrancar, etc.), ver el estado o cambiar la configuración. Por ejemplo, accedemos a HDFS, y vemos en la primera pantalla un cuadro de mando con el resumen del estado de HDFS.

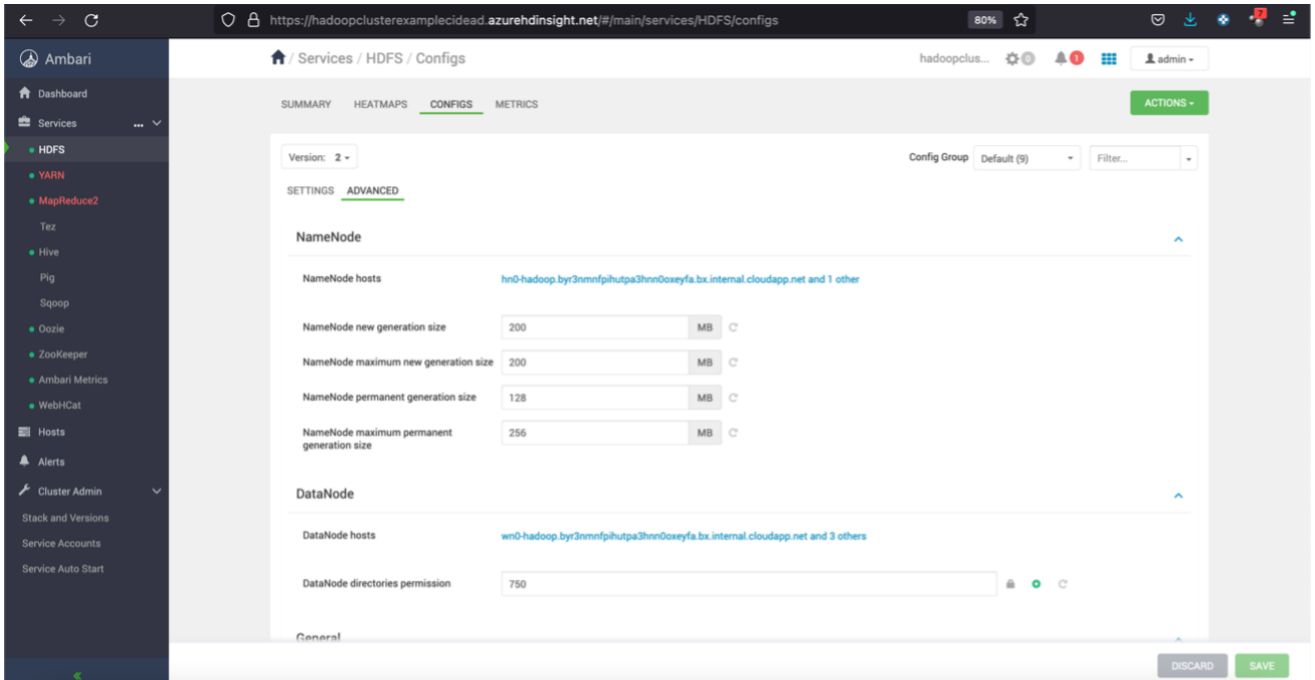


Íñigo Sanz (Dominio público)

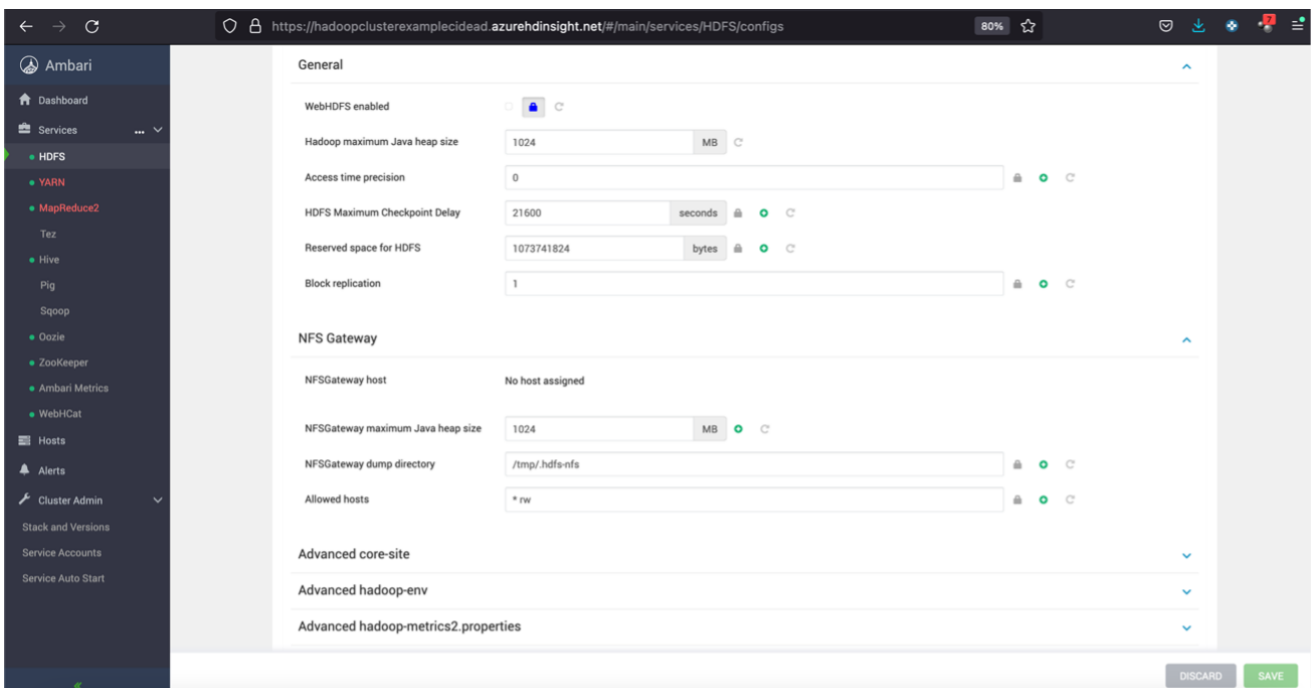
Sobre las métricas que habíamos visto en el cuadro de mando principal, en esta pantalla se añaden algunas importantes como el número total de bloques (11), el estado de las réplicas, por si hubiera algún bloque corrupto o con un número de réplicas más bajos, etc.

En la pestaña de configuración, se puede modificar toda la configuración de HDFS de un modo visual, se puede ver un cuadro general o en la pestaña “Advanced” se puede ver toda la configuración disponible. Algunos parámetros de configuración son:

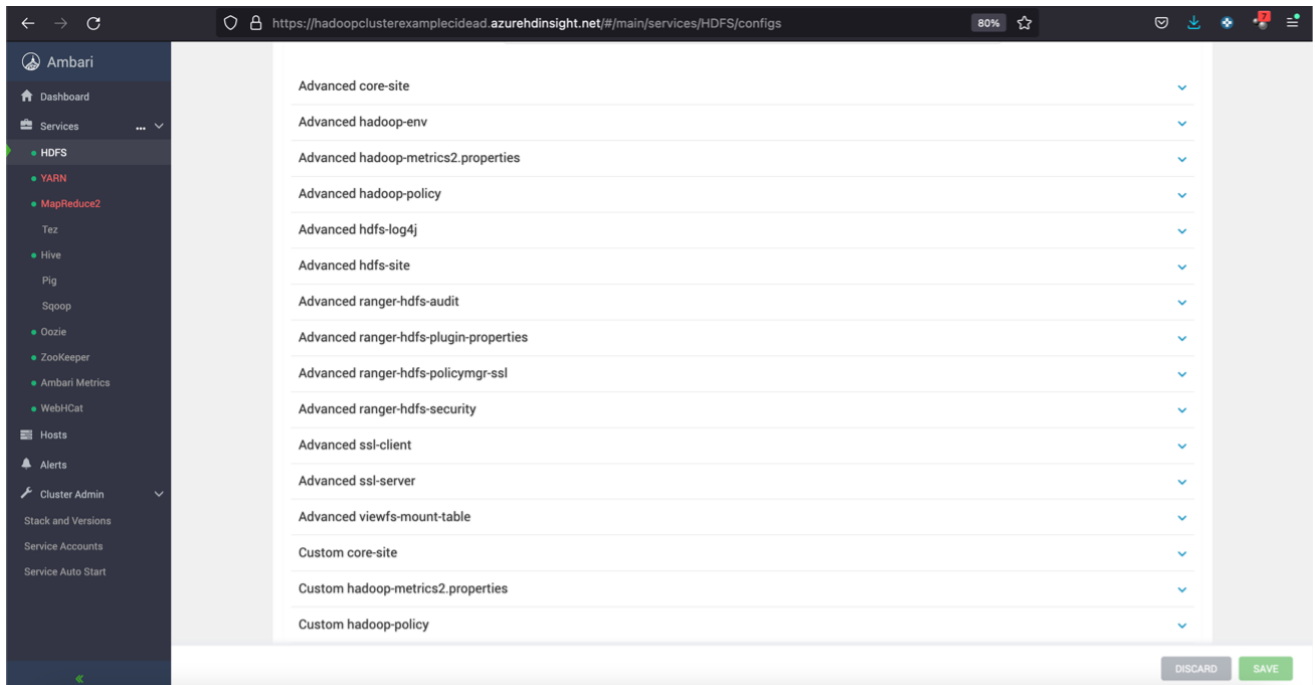
- ✔ Las rutas en el disco local del Namenode y Datanodes donde se enganchará HDFS.
- ✔ Tamaños de memoria de disponible para HDFS en los servidores.
- ✔ El factor de replicación por defecto (1 en nuestro ejemplo).
- ✔ Tiempo máximo de respuesta de los mensajes de status a partir del cual un nodo se marca como parado o defectuoso.



Íñigo Sanz (Dominio público)



Íñigo Sanz (Dominio público)



Íñigo Sanz (Dominio público)

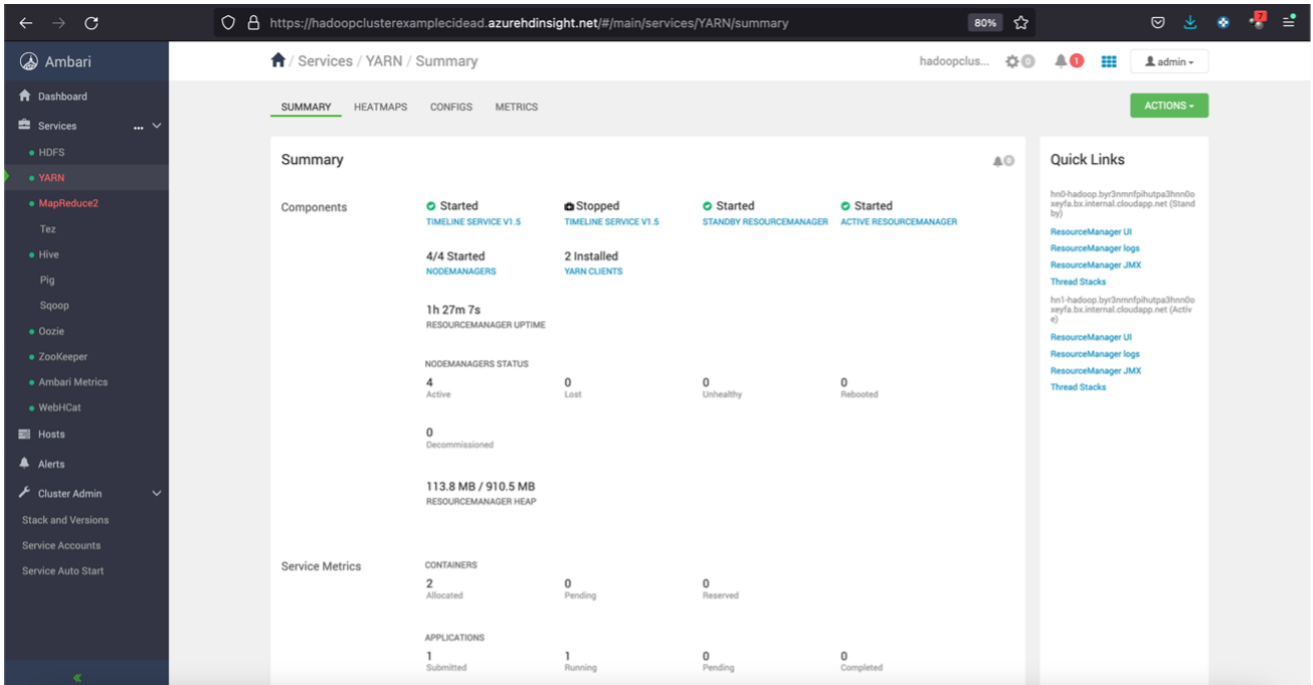
Hay infinidad de variables de configuración. Antes de modificar alguna es conveniente consultar la guía oficial para entender qué implicaciones tiene o para qué sirve el parámetro. En caso de modificar algún parámetro, se requerirá reiniciar los servicios.

En la última pestaña, “Metrics” se puede configurar un cuadro de mando a medida con las métricas que nos resulten interesantes, aunque las que aparecen cargadas por defecto suelen ser las habituales para poder entender la salud de HDFS.

Por otro lado, en la parte superior derecha, se puede ver un botón “Actions” con el que podemos manejar el servicio para parar el servicio, reiniciarlo, etc.

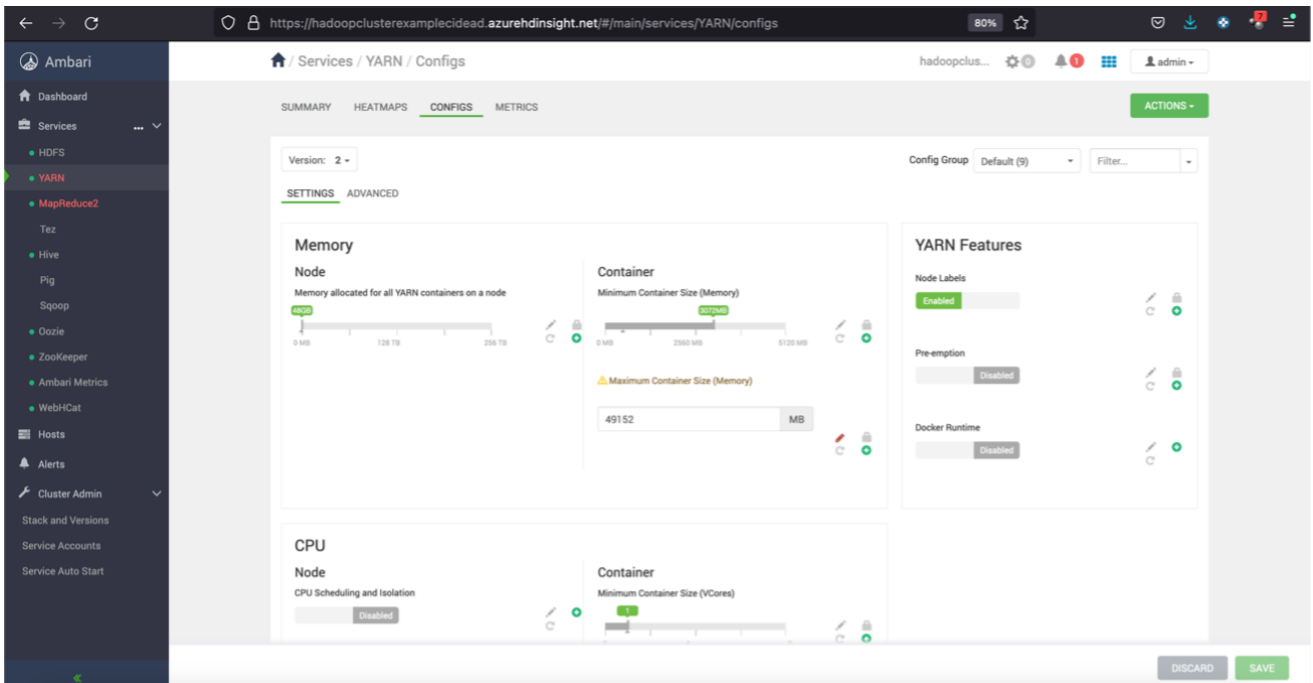
Por último, en la parte derecha aparecen los enlaces a las consolas de administración generales del Namenode, así como a los logs.

Accediendo al servicio YARN obtenemos la misma estructura de pestañas de HDFS, con métricas y configuraciones específicas de YARN, así como las acciones para parar el servicio, reiniciarlo, etc. Por ejemplo, el cuadro de mando general muestra información sobre el estado del ResourceManager y los Nodemanagers, y en la parte inferior indica cuántas aplicación hay ejecutándose en este momento en YARN (1) o cuántos contenedores se están utilizando (2).



Íñigo Sanz (Dominio público)

En cuanto a la configuración, el formato de presentación es similar, con una pantalla simplificada denominada “Settings” así como una pantalla con todos los parámetros de configuración, denominada “Advanced”.



Íñigo Sanz (Dominio público)

The screenshot shows the Ambari configuration interface for YARN. The left sidebar contains navigation options like Dashboard, Services (HDFS, YARN, MapReduce2, Tez, Hive, Pig, Sqoop, Oozie, ZooKeeper, Ambari Metrics, WebHCat), Hosts, Alerts, and Cluster Admin. The main content area is divided into two sections: CPU and GPU.

CPU Configuration:

- Node CPU Scheduling and Isolation:** A slider is set to 100%.
- Container CPU Scheduling and Isolation:** A slider is set to 100%.
- Percentage of physical CPU allocated for all containers on a node:** A slider is set to 100%.
- Number of virtual cores:** A slider is set to 3.

GPU Configuration:

- GPU Scheduling and Isolation:** A slider is set to 100%.
- Container GPU Scheduling and Isolation:** A slider is set to 100%.
- Absolute path of nvidia-smi on NodeManagers:** An empty text field.

At the bottom right, there are 'DISCARD' and 'SAVE' buttons.

Íñigo Sanz (Dominio público)

The screenshot shows the Ambari configuration interface for YARN, specifically the 'CONFIGS' tab. The left sidebar is the same as in the previous image. The main content area shows the configuration for the Resource Manager and Node Manager.

Resource Manager Configuration:

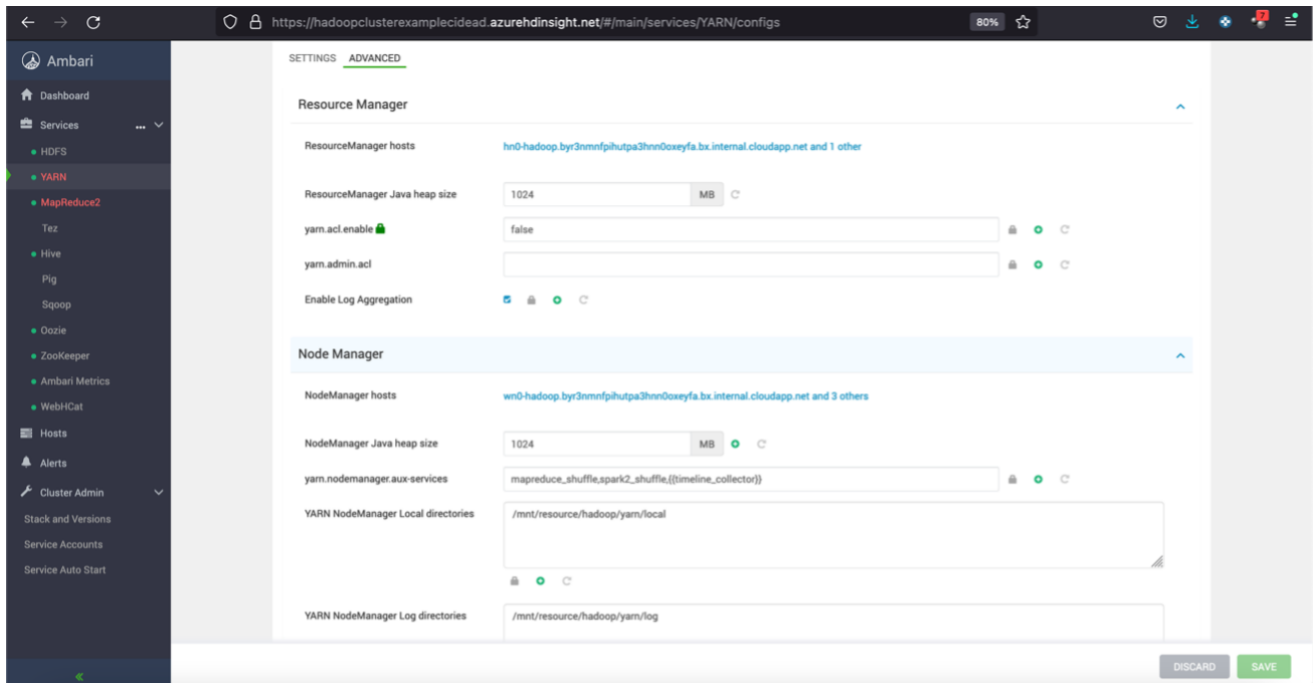
- Resource Manager hosts:** `hn0-hadoop.by3nmnfpihutpa3hnn0oxeyfa.bx.internal.cloudapp.net and 1 other`
- ResourceManager Java heap size:** 1024 MB
- yarn.acl.enable:** false
- yarn.admin.acl:** (empty field)
- Enable Log Aggregation:** checked

Node Manager Configuration:

- Node Manager hosts:** `wn0-hadoop.by3nmnfpihutpa3hnn0oxeyfa.bx.internal.cloudapp.net and 3 others`
- NodeManager Java heap size:** 1024 MB
- yarn.nodemanager.aux-services:** `mapreduce_shuffle,spark2_shuffle,{timeline_collector}`

At the bottom right, there are 'DISCARD' and 'SAVE' buttons.

Íñigo Sanz (Dominio público)

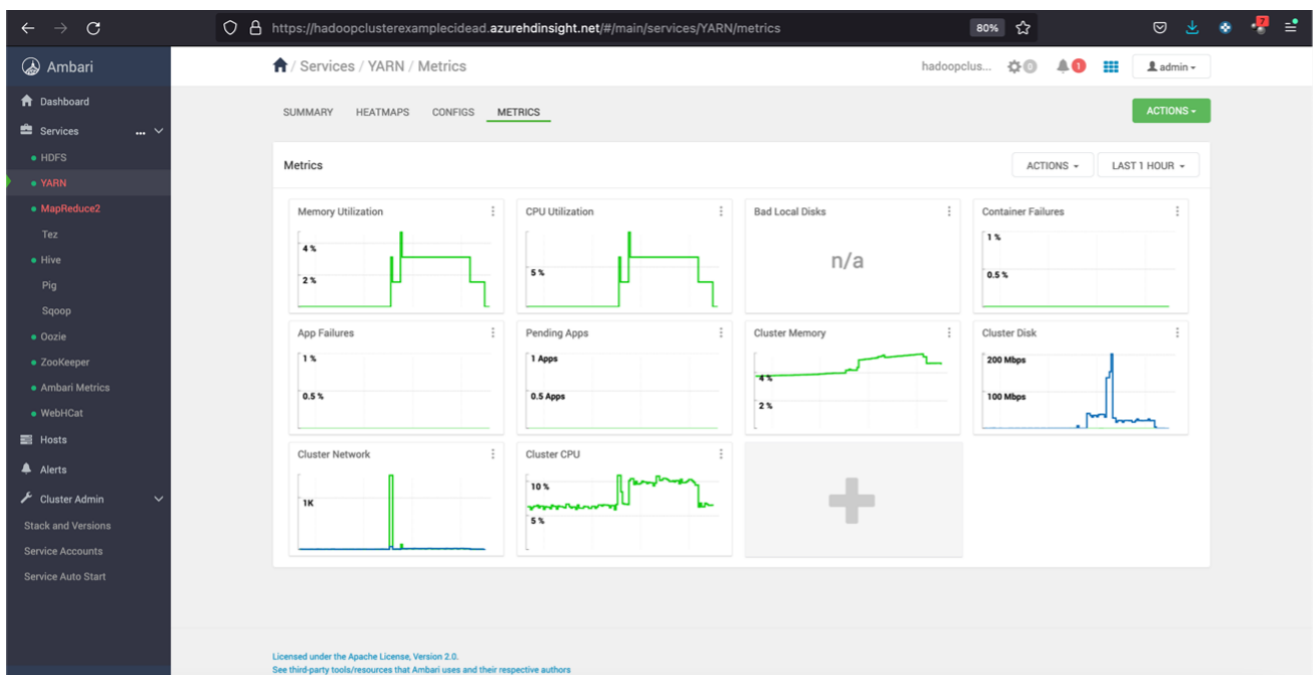


Íñigo Sanz (Dominio público)

Algunos parámetros de configuración importantes son:

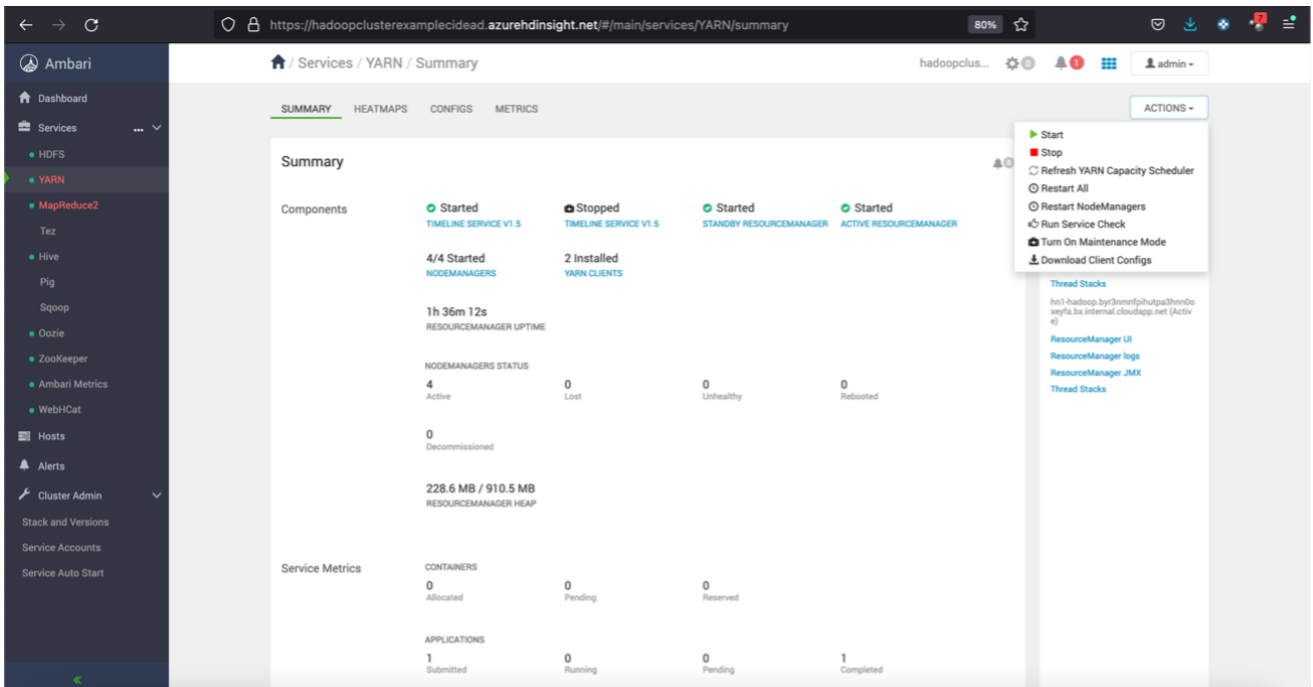
- ✔ El tamaño máximo de los contenedores: 49152 megabytes y 1 vCore en el ejemplo.
- ✔ Los recursos máximos que puede tomar YARN de los distintos nodos worker: 80% de la CPU como máximo y 48 gigabytes de RAM.
- ✔ La ruta de los ficheros de log en los Nodemanager.

Al igual que en el caso de HDFS, se dispone de una última pestaña para montar cuadros de mando a medida con las métricas que más nos interese monitorizar, aunque vienen cargadas las más importantes:



Íñigo Sanz (Dominio público)

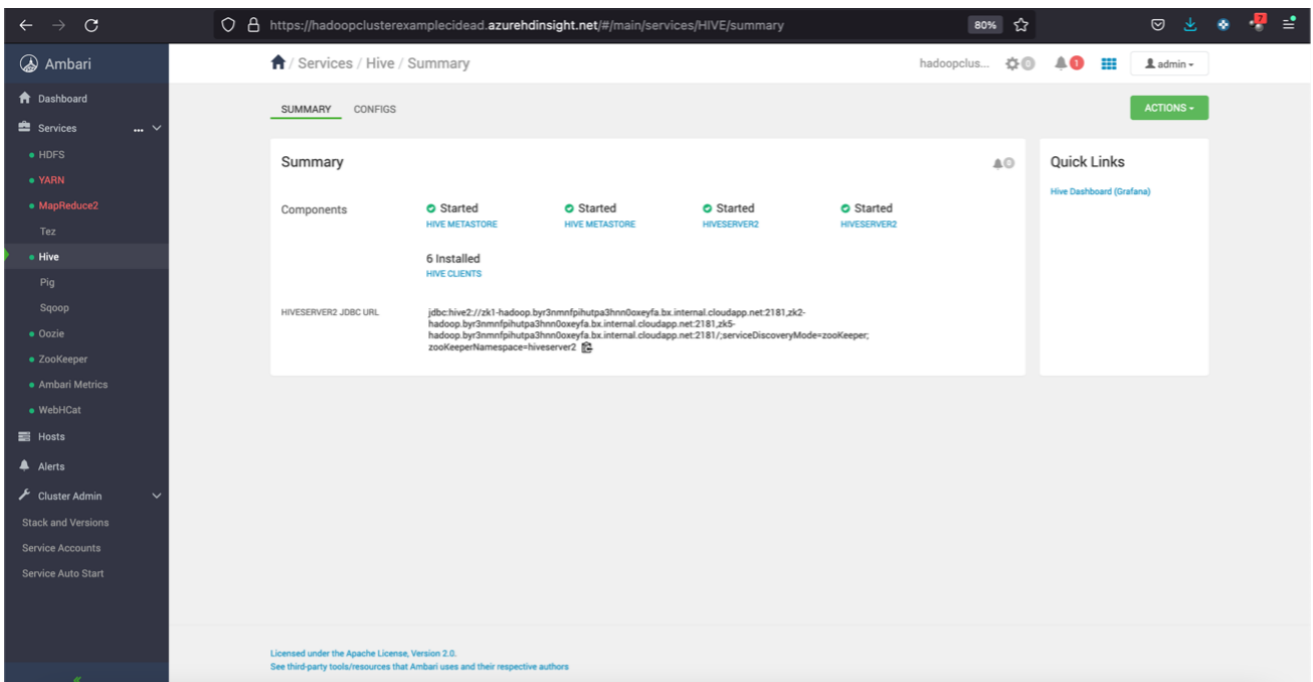
Por último, como ocurría en HDFS, tenemos opciones para arrancar, parar o reiniciar YARN, por ejemplo, tras aplicar algún cambio de configuración, así como los enlaces a las consolas de monitorización por defecto de YARN.



Íñigo Sanz (Dominio público)

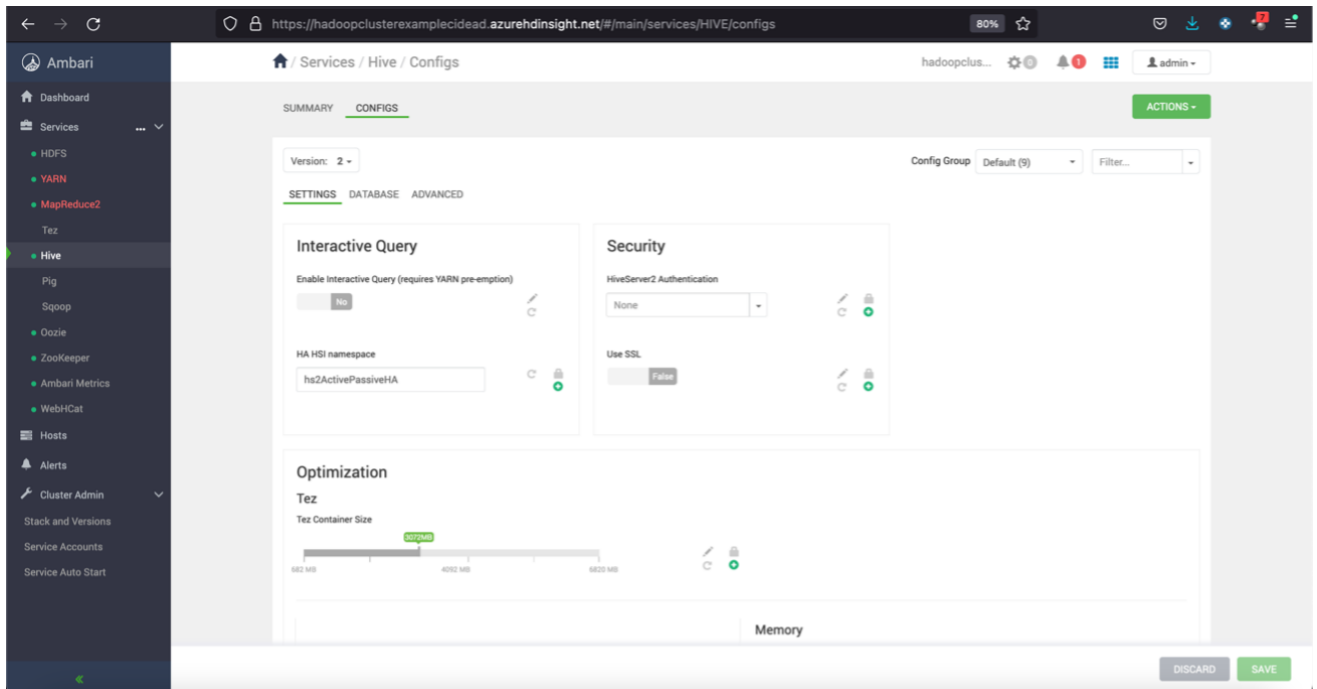
El resto de servicios, como Hive, Tez, Pig, Sqoop, etc. tienen interfaces más sencillas, con menos elementos de monitorización, ya que sus tareas se ejecutan en YARN, y por lo tanto, se monitorizarán desde esa consola. Disponen, además, de toda la configuración para poder realizar modificaciones, así como acciones para parar, arrancar o reiniciar los servicios.

Por ejemplo, en el caso de Hive, nos encontramos este cuadro de mando general:

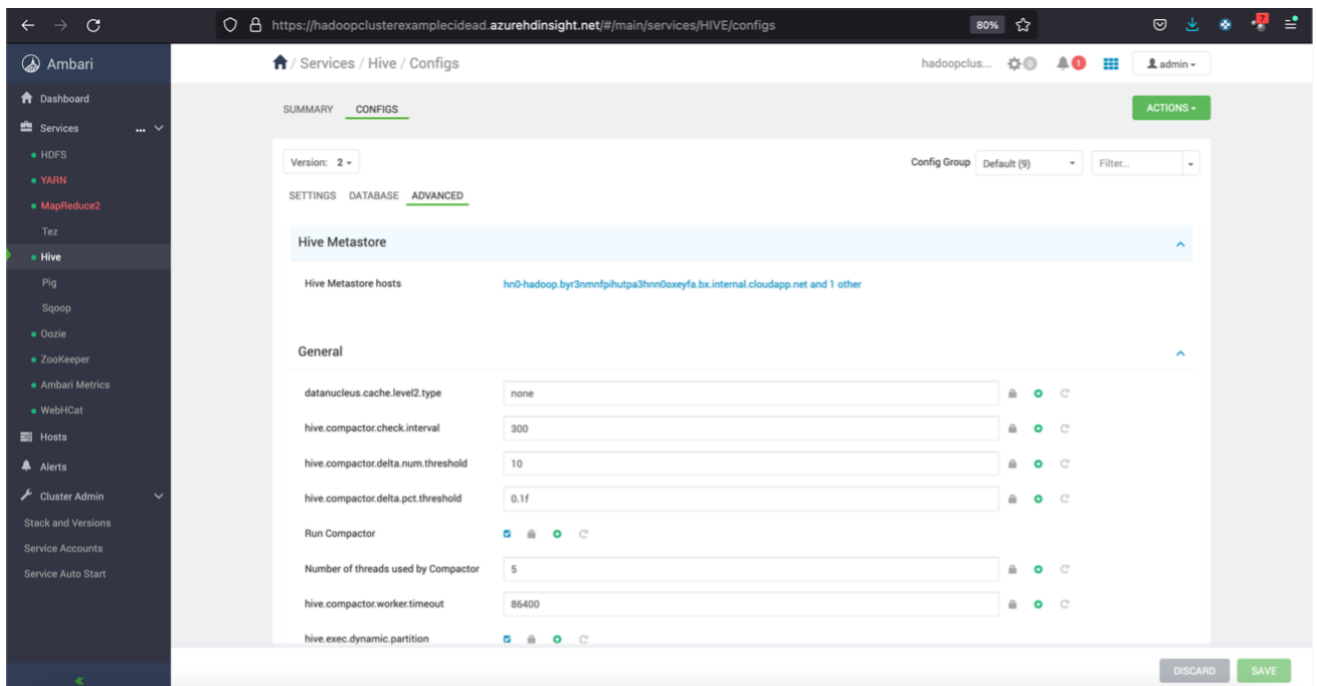


Íñigo Sanz (Dominio público)

Como puedes ver, indica el estado del servicio, pero no muestra datos de rendimiento, ya que sus operaciones, como hemos indicado, se realizan en YARN. En cuanto a la configuración, se dispone del mismo tipo de pantalla, con una ficha simplificada y una pestaña avanzada con toda la configuración disponible:



Íñigo Sanz (Dominio público)



Íñigo Sanz (Dominio público)

Hosts

En cuanto a la opción de Hosts, muestra el estado de los distintos servidores. Es decir, no se tiene una visión de servicio, que puede estar ejecutándose en múltiples servidores, sino una visión de los servidores uno a uno.

Al acceder a la opción, se muestra todos los servidores del clúster:

Ambari Hosts

Name	IP Address	Rack	Cores	RAM	Disk Usage	Load Avg	Versions	Components
hn0-hadoop.byr3nmfpl...	10.0.0.21	/default-rack	4 (4)	31.36GB		8.07	HDInsight-4.1.9.7	20 Components
hn1-hadoop.byr3nmfpl...	10.0.0.22	/default-rack	4 (4)	31.36GB		2.17	HDInsight-4.1.9.7	16 Components
wn0-hadoop.byr3nmfpl...	10.0.0.12	/default-rack	8 (8)	62.81GB		1.00	HDInsight-4.1.9.7	7 Components
wn1-hadoop.byr3nmfpl...	10.0.0.14	/default-rack	8 (8)	62.81GB		0.42	HDInsight-4.1.9.7	7 Components
wn2-hadoop.byr3nmfpl...	10.0.0.13	/default-rack	8 (8)	62.81GB		0.47	HDInsight-4.1.9.7	7 Components
wn3-hadoop.byr3nmfpl...	10.0.0.11	/default-rack	8 (8)	62.81GB		0.12	HDInsight-4.1.9.7	7 Components
zk1-hadoop.byr3nmfpl...	10.0.0.5	/default-rack	2 (2)	3.84GB		1.09	HDInsight-4.1.9.7	4 Components
zk2-hadoop.byr3nmfpl...	10.0.0.6	/default-rack	2 (2)	3.84GB		0.95	HDInsight-4.1.9.7	4 Components
zk5-hadoop.byr3nmfpl...	10.0.0.10	/default-rack	2 (2)	3.84GB		0.91	HDInsight-4.1.9.7	4 Components

Items per page: 10 - 1 - 9 of 9

Licensed under the Apache License, Version 2.0. See third-party tools/resources that Ambari uses and their respective authors

Íñigo Sanz (Dominio público)

Se puede ver que nuestro clúster tiene 9 servidores. Para cada servidor se puede ver la dirección IP interna, la memoria disponible, el uso de disco, la media de CPU utilizada, así como el número de servicios que está ejecutando. En caso de existir alguna incidencia, además, se marcará con un icono como puede verse en el segundo servidor, que indica que está habiendo una incidencia con el servicio Timeline Server, de YARN.

Ambari Hosts

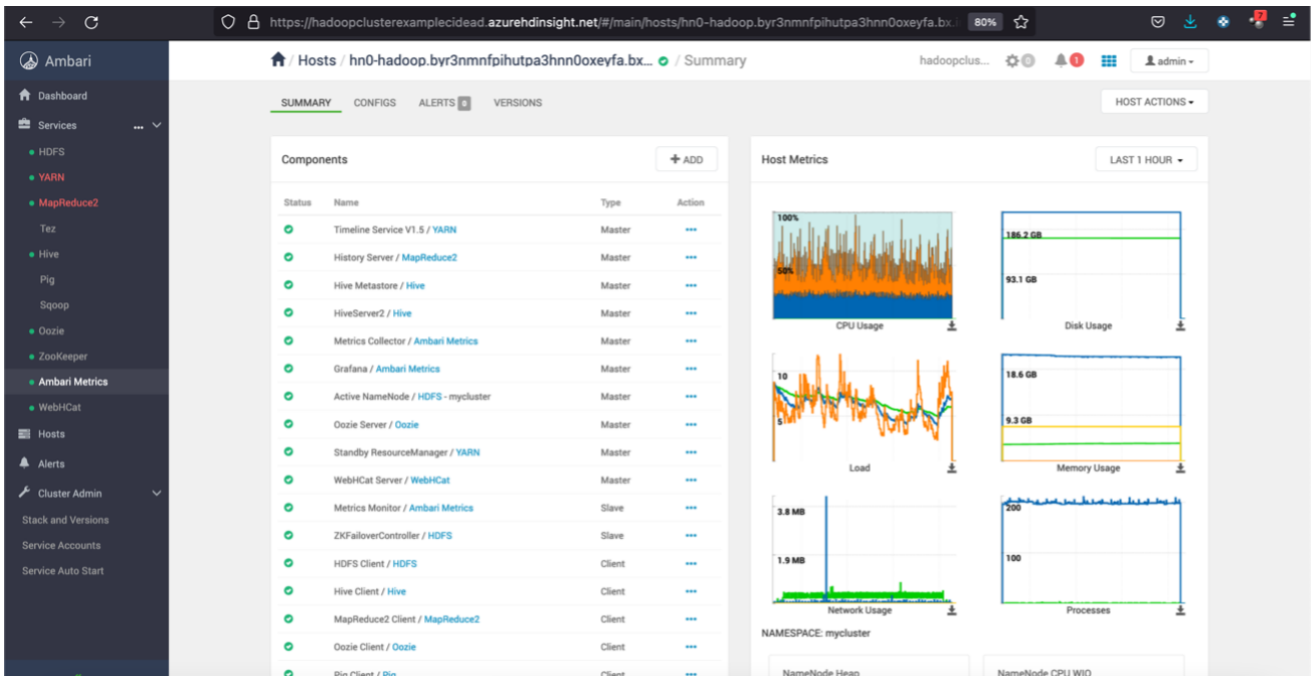
Name	IP Address	Rack	Cores	RAM	Disk Usage	Load Avg	Versions	Components
hn0-hadoop.byr3nmfpl...	10.0.0.21	/default-rack	4 (4)	31.36GB		9.40	HDInsight-4.1.9.7	20 Components
hn1-hadoop.byr3nmfpl...	10.0.0.22	/default-rack	4 (4)	31.36GB		4.83	HDInsight-4.1.9.7	16 Components
wn0-hadoop.byr3nmfpl...	10.0.0.12	/default-rack	8 (8)	62.81GB		0.47	HDInsight-4.1.9.7	7 Components
wn1-hadoop.byr3nmfpl...	10.0.0.14	/default-rack	8 (8)	62.81GB		0.44	HDInsight-4.1.9.7	7 Components
wn2-hadoop.byr3nmfpl...	10.0.0.13	/default-rack	8 (8)	62.81GB		0.31	HDInsight-4.1.9.7	7 Components
wn3-hadoop.byr3nmfpl...	10.0.0.11	/default-rack	8 (8)	62.81GB		0.31	HDInsight-4.1.9.7	7 Components
zk1-hadoop.byr3nmfpl...	10.0.0.5	/default-rack	2 (2)	3.84GB		0.68	HDInsight-4.1.9.7	4 Components
zk2-hadoop.byr3nmfpl...	10.0.0.6	/default-rack	2 (2)	3.84GB		0.64	HDInsight-4.1.9.7	4 Components
zk5-hadoop.byr3nmfpl...	10.0.0.10	/default-rack	2 (2)	3.84GB		0.98	HDInsight-4.1.9.7	4 Components

Items per page: 10 - 1 - 9 of 9

Licensed under the Apache License, Version 2.0. See third-party tools/resources that Ambari uses and their respective authors

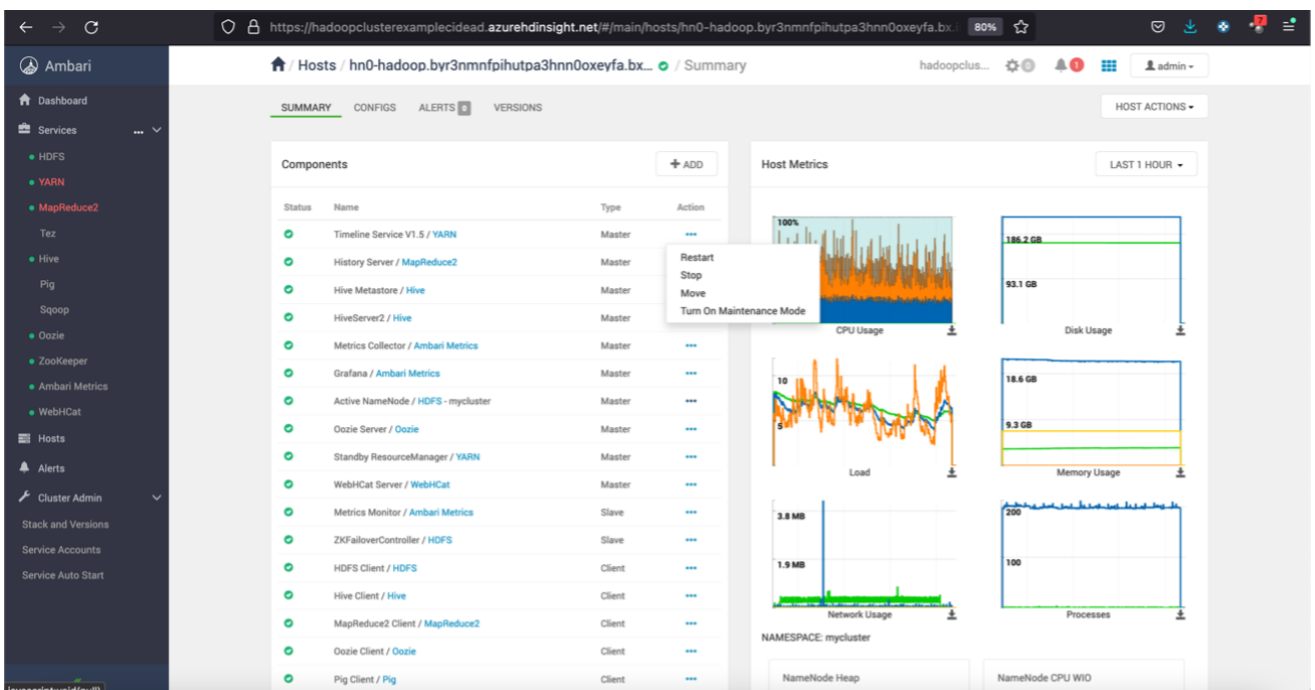
Íñigo Sanz (Dominio público)

Haciendo clic sobre el nombre de un servidor, podemos conocer el detalle de los servicios que está ejecutando, así como su estado. Por ejemplo, al entrar en el primer servidor, que es un nodo maestro de HDFS, que también hace de frontera, con los interfaces de todos los servicios instalados, nos muestra la siguiente información:



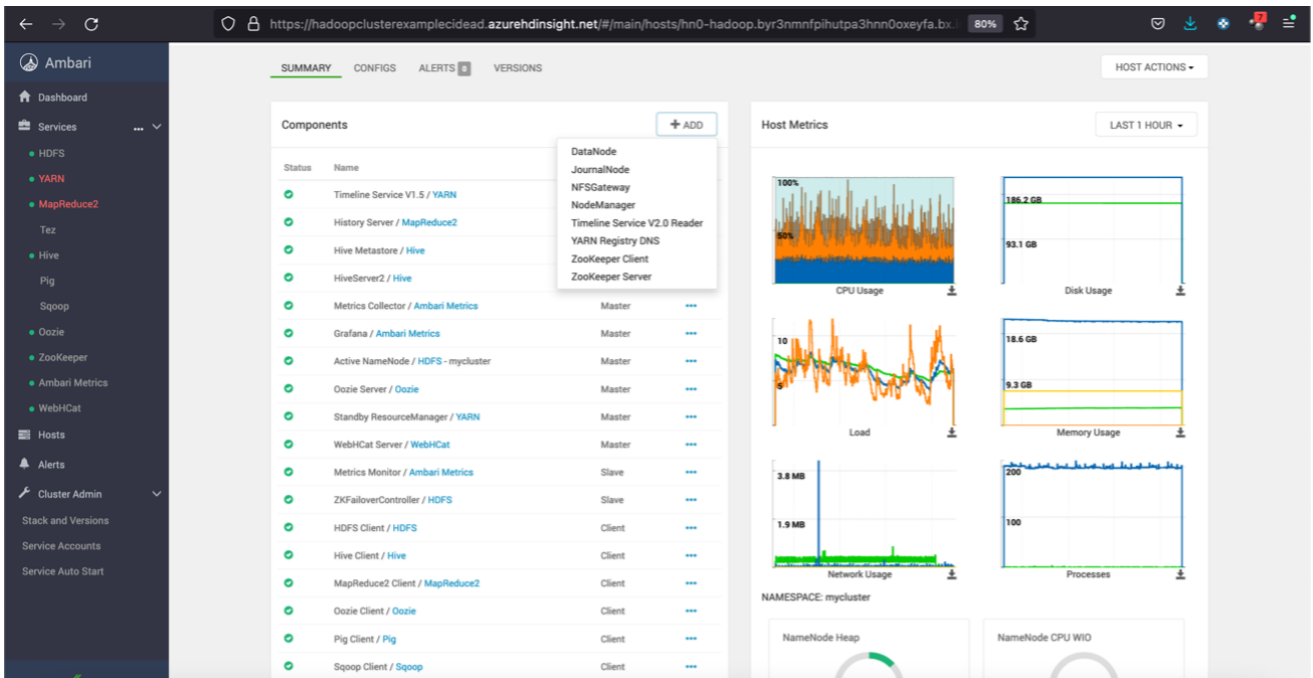
Íñigo Sanz (Dominio público)

Para cada servicio, pinchando en las opciones que aparecen en la parte derecha, se puede parar, reiniciar o mover el servicio a otro servidor.



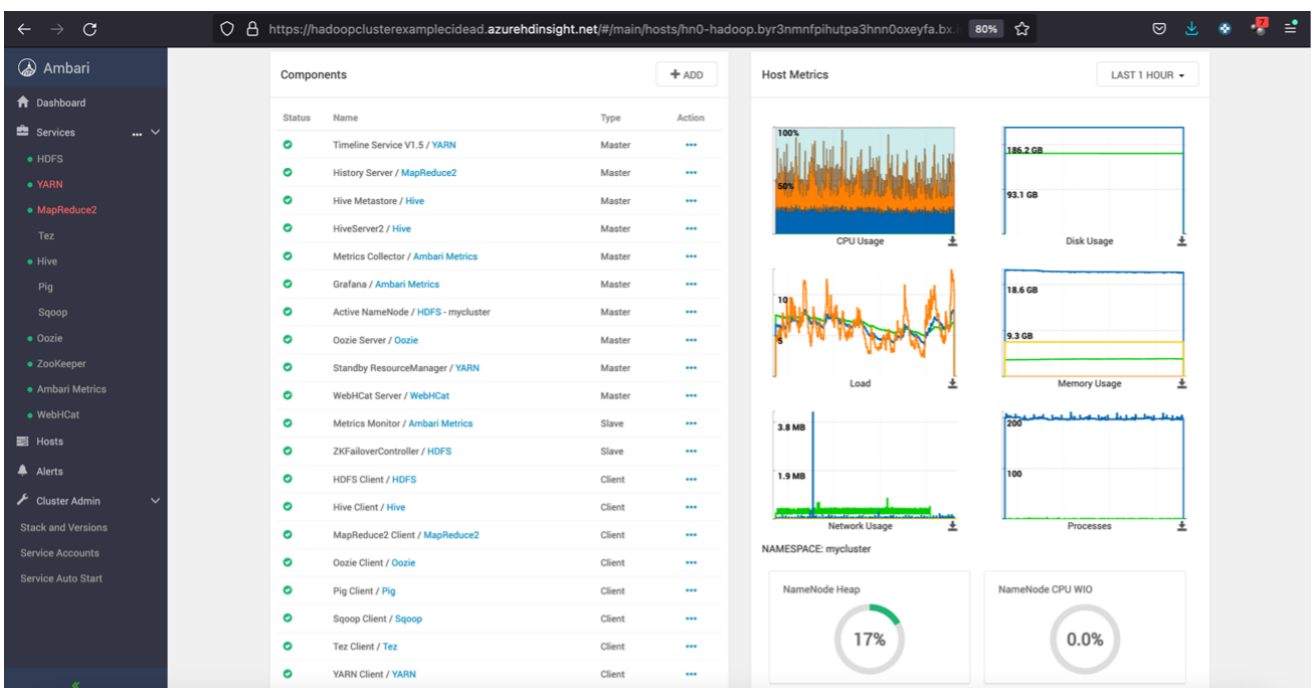
Íñigo Sanz (Dominio público)

Además, se puede añadir un servicio al servidor, por ejemplo, si quisiéramos que este nodo maestro además fuera un Datanode (no recomendable):



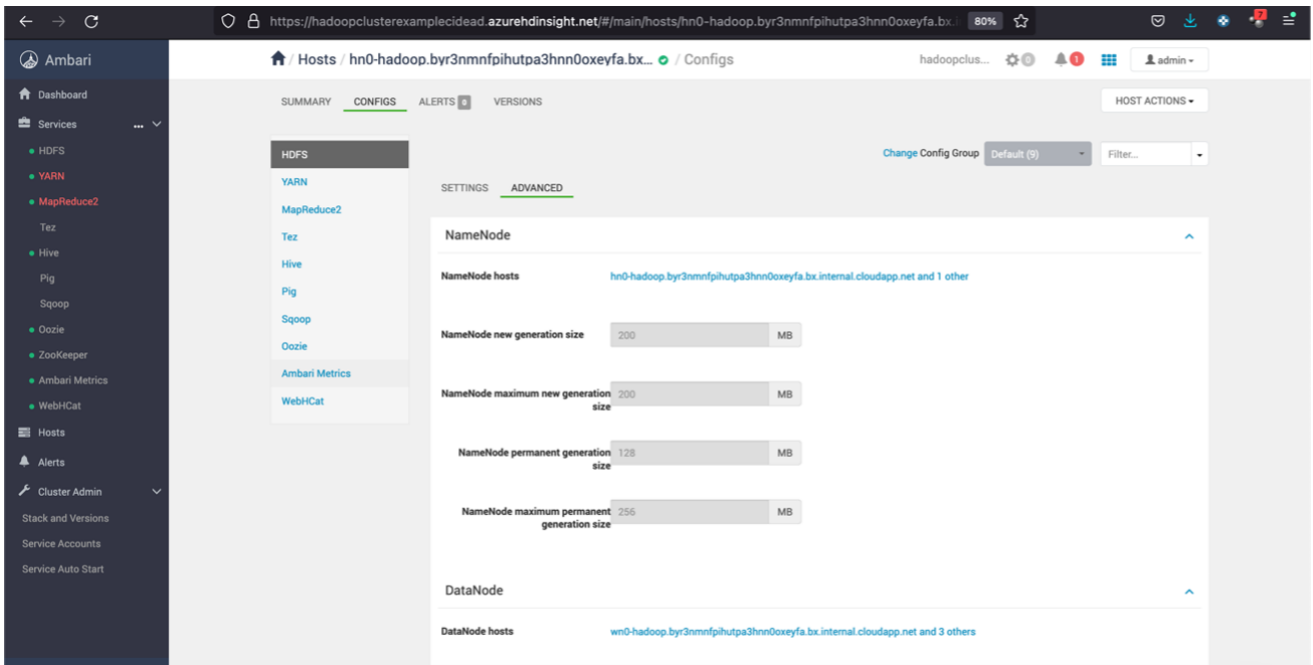
Íñigo Sanz (Dominio público)

En la parte derecha podemos ver diferentes métricas del estado del servidor, donde puede apreciarse que no tiene grandes problemas en cuanto al uso de recursos:



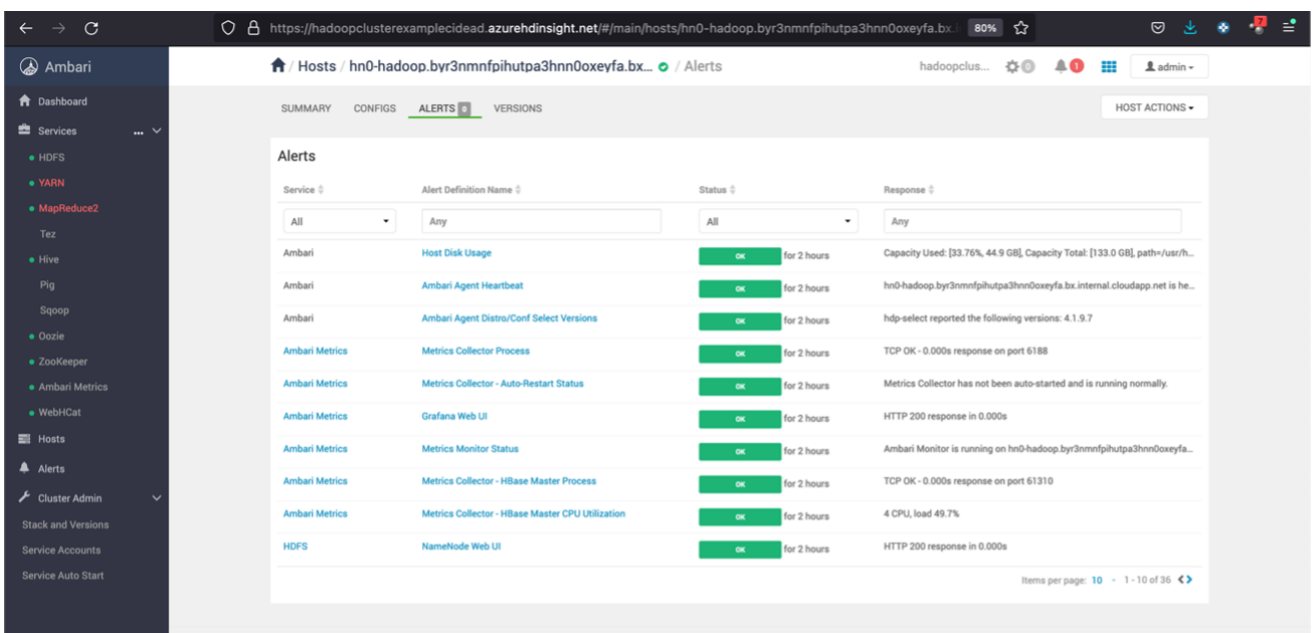
Íñigo Sanz (Dominio público)

En la siguiente pestaña, de configuración, se puede visualizar y modificar la configuración de los servicios de este servidor:



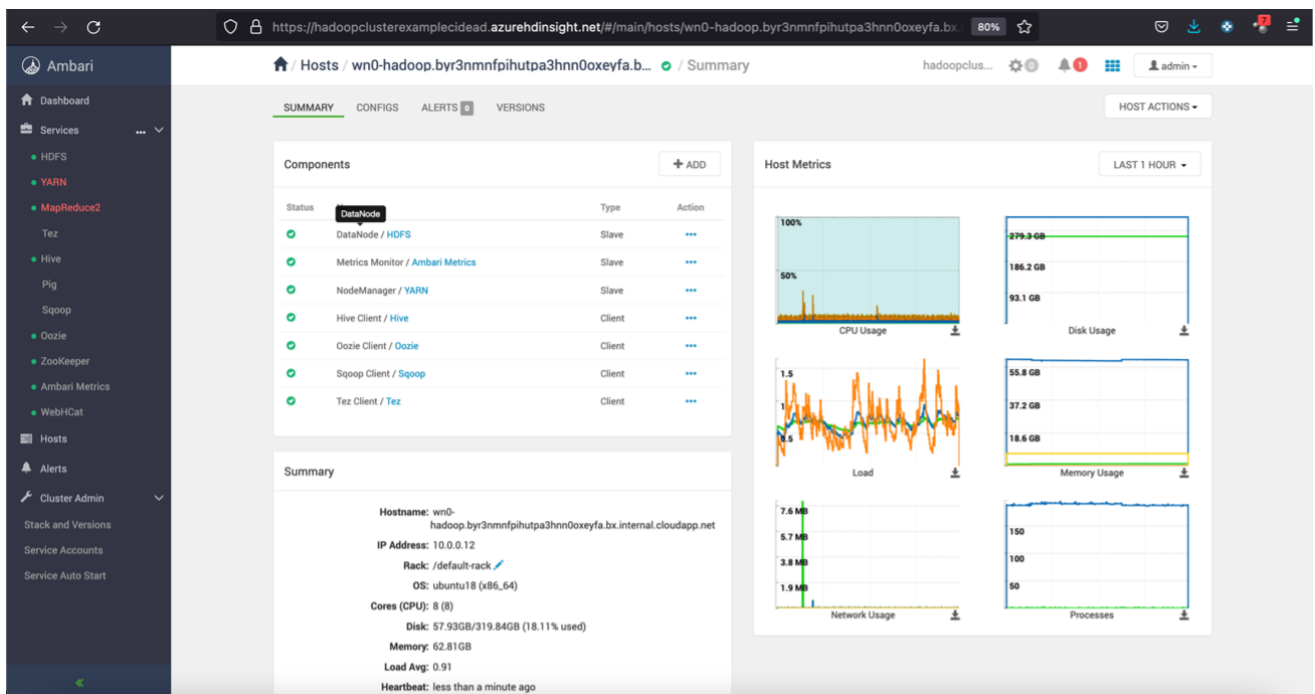
Íñigo Sanz (Dominio público)

En la siguiente pestaña, de alertas, se puede ver si ha habido algún error significativo, o si por el contrario el servidor está funcionando correctamente.



Íñigo Sanz (Dominio público)

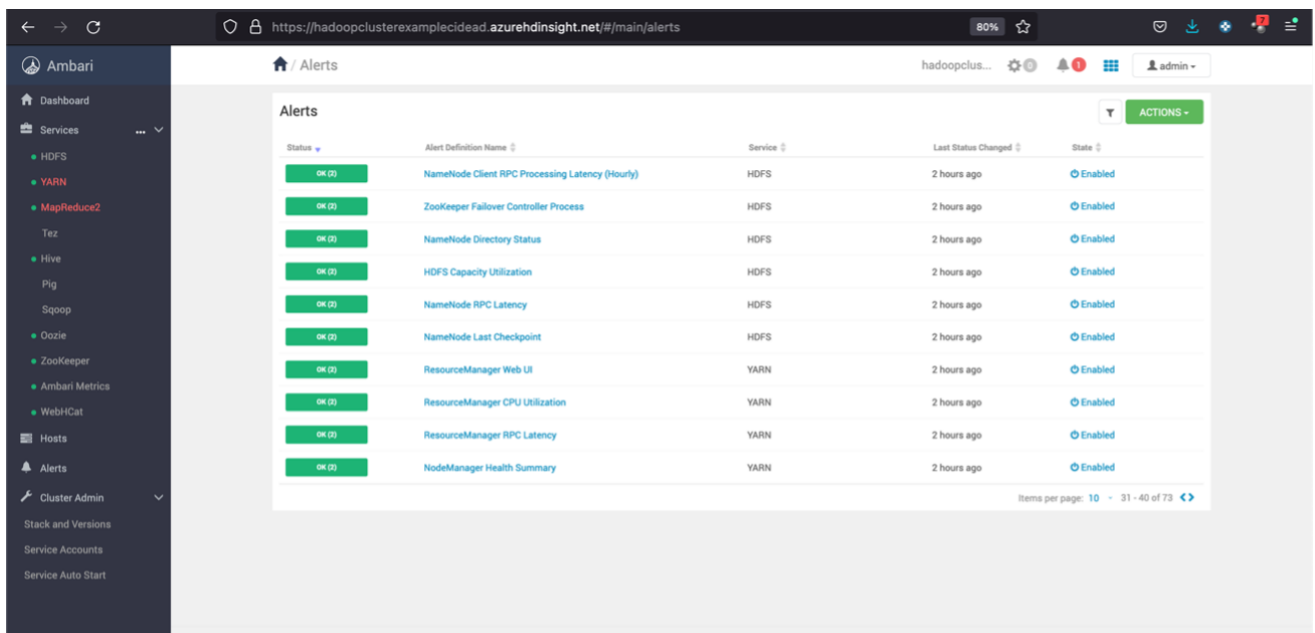
En el caso de que accedamos a un nodo worker, muestra menos servicios, como es lógico, porque no tiene interfaces o servicios maestro instalados, pero el resto de estructura de pestañas es similar:



Íñigo Sanz (Dominio público)

Alertas

En la opción de alertas, muestra un listado de todas las comprobaciones periódicas, indicando si hay algún fallo en alguna de ellas:

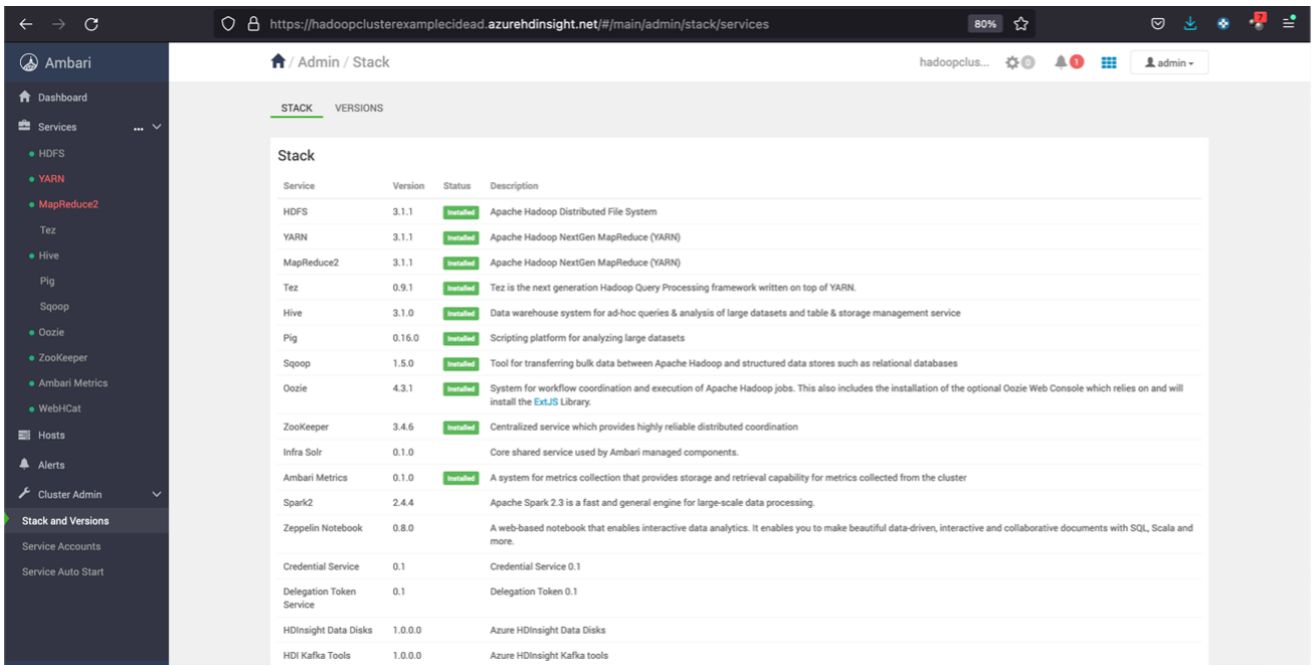


Íñigo Sanz (Dominio público)

Administración del clúster

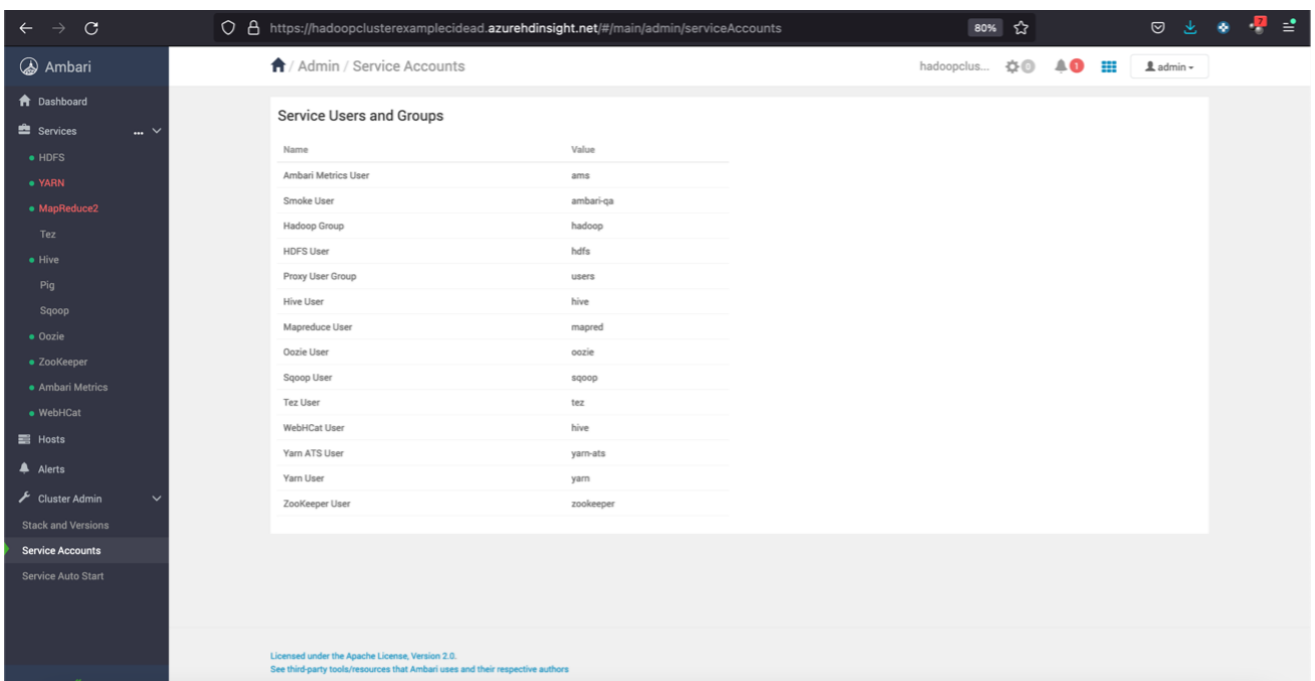
En la opción de Administración del clúster, se puede gestionar aspectos como los componentes y versiones de los componentes utilizados, los usuarios de sistema operativo de los diferentes servicios, u opciones para arranque automático de los servicios.

Pantalla de componentes y versiones:



Íñigo Sanz (Dominio público)

Pantalla de usuarios de sistema operativo de los servicios:



Íñigo Sanz (Dominio público)

Inicio automático de los servicios:

← → ↻ https://hadoopclusterexamplecloud.azurehdinsight.net/#/main/admin/serviceAutoStart 80% ☆

Ambari / Admin / Service Auto Start hadoopclus... admin

Ambari services can be configured to start automatically on system boot. Each service can be configured to start all components, masters and workers, or selectively.

Auto Start Settings Enabled

Service	Components	Auto Start
Ambari Metrics	Metrics Collector	<input checked="" type="checkbox"/>
	Grafana	<input checked="" type="checkbox"/>
	Metrics Monitor	<input checked="" type="checkbox"/>
HDFS	DataNode	<input checked="" type="checkbox"/>
	HDFS Client	<input checked="" type="checkbox"/>
	JournalNode	<input checked="" type="checkbox"/>
	NameNode	<input checked="" type="checkbox"/>
	ZKFailoverController	<input checked="" type="checkbox"/>
Hive	Hive Client	<input checked="" type="checkbox"/>
	Hive Metastore	<input checked="" type="checkbox"/>
	HiveServer2	<input checked="" type="checkbox"/>
MapReduce2	History Server	<input checked="" type="checkbox"/>
	MapReduce2 Client	<input checked="" type="checkbox"/>
Oozie	Oozie Client	<input checked="" type="checkbox"/>
	Oozie Server	<input checked="" type="checkbox"/>

CANCEL SAVE

Íñigo Sanz (Dominio público)

Autoevaluación

Indica si las siguientes afirmaciones son verdaderas o falsas.

Ambari permite simplificar la instalación de un clúster Hadoop, ofreciendo una funcionalidad para hacer la instalación a modo de asistente

Verdadero Falso

Verdadero

Verdadero: Ambari dispone de un asistente para ayudar en la instalación y configuración inicial de un clúster.

Ambari me permite definir alertas para recibir notificaciones en caso de que se caiga un servidor del clúster.

Verdadero Falso

Verdadero

Verdadero: Ambari tiene una funcionalidad para definir alertas basada en reglas.

Con Ambari se puede saber el porcentaje de CPU utilizado en los servidores, el porcentaje de memoria o de disco.

Verdadero Falso

Verdadero

Verdadero: Ambari ofrece muchas métricas, algunas de las cuales son las que se indican en la pregunta.

4.1.- Ejemplo: parada de nodos worker

Como sabes, Hadoop es capaz de sobreponerse a la parada o caída de los nodos worker, dando servicio igualmente:

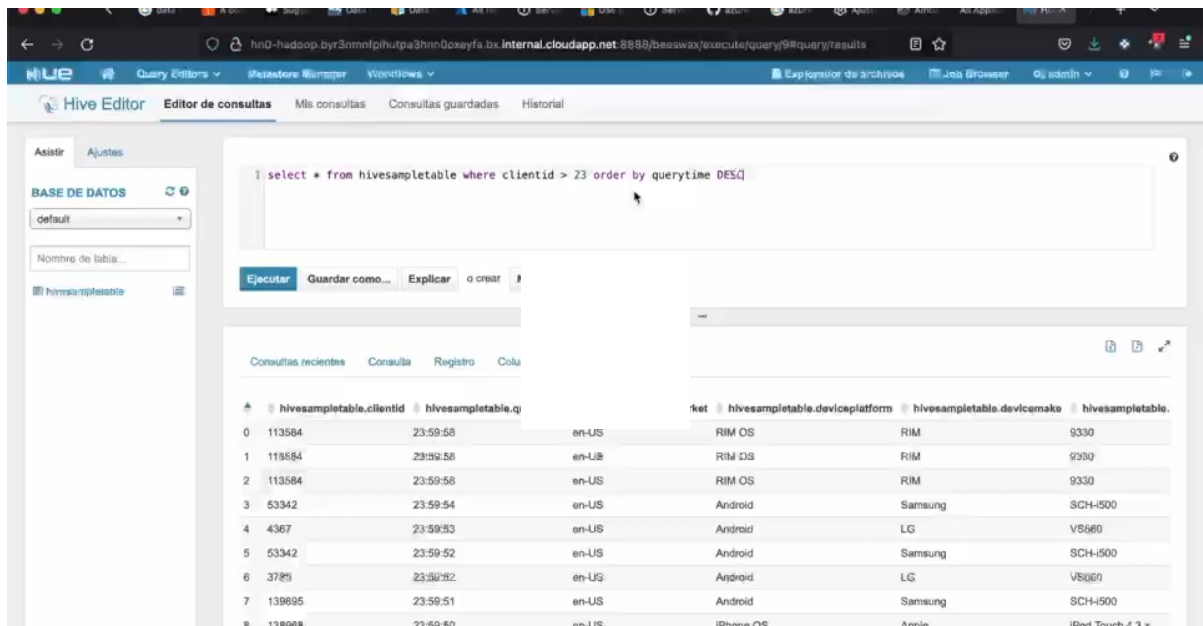
- ✓ En el caso de HDFS, los datos de los nodos que se paran se cogen de las réplicas que están en otros nodos del clúster.
- ✓ En el caso de YARN, los trabajos se distribuyen entre los servidores que están activos.

A continuación se va a mostrar un vídeo en el que, utilizando Apache Ambari, se van a parar la mitad de los nodos worker, simulando una caída por fallo de los servidores, y veremos cómo seguimos siendo capaces de ver los datos y de ejecutar aplicaciones, en este caso, consultas Hive, sin pérdida de servicio.

Situación inicial:

- ✓ Disponemos de un clúster con 4 nodos worker.
- ✓ Estamos realizando consultas en Hive a una tabla que contiene datos que están en un fichero en la ruta /hive/warehouse/managed/hivesampletable.

Veamos cómo Hadoop es capaz de sobreponerse a este tipo de situaciones:



The screenshot shows the Hive Editor interface. The query editor contains the following SQL query:

```
select * from hivesampletable where clientid > 23 order by querytime DESC
```

Below the query editor, there are buttons for "Ejecutar", "Guardar como...", "Explicar", and "o crear". The results pane shows a table with the following columns: clientid, hivesampletable.q, en-US, RIM OS, hivesampletable.deviceplatform, hivesampletable.device make, and hivesampletable. The table contains 9 rows of data.

	clientid	hivesampletable.q	en-US	RIM OS	hivesampletable.deviceplatform	hivesampletable.device make	hivesampletable
0	113584	23:59:58	en-US	RIM OS	RIM	9330	
1	113584	23:59:58	en-US	RIM OS	RIM	9330	
2	113584	23:59:58	en-US	RIM OS	RIM	9330	
3	53342	23:59:54	en-US	Android	Samsung	SCH-I500	
4	4367	23:59:53	en-US	Android	LG	VS660	
5	53342	23:59:52	en-US	Android	Samsung	SCH-I500	
6	3789	23:59:52	en-US	Android	LG	VS660	
7	139895	23:59:51	en-US	Android	Samsung	SCH-I500	
8	138968	23:59:50	en-US	iPhone OS	Apple	iPod Touch 4.3.x	

00:00

02:45

Íñigo Sanz (Dominio público)

En el vídeo podrás encontrar los siguientes momentos relevantes:

- ✓ 24": se para manualmente uno de los 4 nodos worker.
- ✓ 1'20": se lanza la consulta Hive con un nodo parado y sigue funcionando.
- ✓ 1'40": se para manualmente otros de los 4 nodos worker, quedando sólo 2 activos.
- ✓ 2'29": se lanza la consulta con dos nodos parados y sigue funcionando.

5.- Cloudera Manager.

Caso práctico

El equipo de IT del Banco Español de Inversiones, BEI, comenzó a utilizar Apache Ambari para la administración de su cluster Hadoop, ya que inicialmente tenía una distribución de Hortonworks. Sin embargo, tras la compra de Hortonworks por parte de Cloudera, y la continuación de las versiones, hace unos meses tomaron la decisión de migrar todo el clúster a Cloudera.

El cambio no supuso un esfuerzo grande, pero a partir de ahora tienen que utilizar Cloudera Manager para administrar el clúster en lugar de Apache Ambari. En principio parece que no habría muchos cambios, pero veamos si son aplicaciones similares.



[Gerd Altmann](#) (Dominio público)

Cloudera Manager fue la primera herramienta de administración de Hadoop propiamente dicha. Fue desarrollada por Cloudera e incluida en su distribución de Hadoop, denominada Cloudera CDH, desde su origen en el año 2009.

En ese momento, no había ninguna herramienta alternativa en el ecosistema Apache, ya que Apache Ambari se lanzó en el año 2013.

Cloudera Manager siguió ofreciéndose en las distribuciones Cloudera y se sigue incluyendo hoy en día. De hecho, la desaparición de Hortonworks y del resto de distribuciones ha hecho que Apache Ambari se haya descontinuado, ya que la única distribución comercial de Hadoop, la de Cloudera, incluye Cloudera Manager en lugar de Ambari.

Sin embargo, ambas herramientas son muy similares en cuanto a funcionalidad. Las dos ofrecen capacidad de instalación, administración y monitorización de clústers, e incluyen un interfaz de usuario con una experiencia prácticamente idéntica.

Veamos algunas pantallas y funcionalidades de Cloudera Manager en un clúster de demostración de Cloudera.

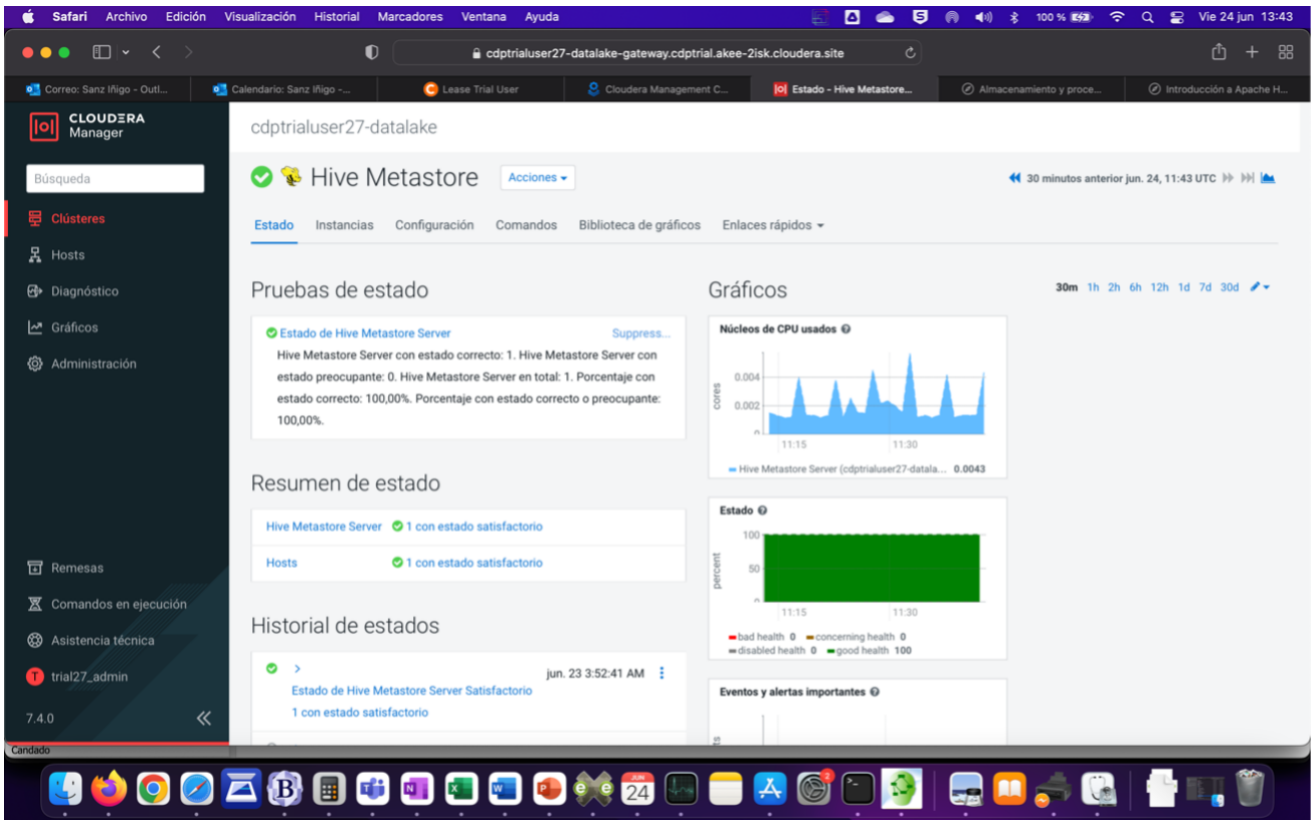
La organización de opciones del menú de Cloudera Manager es muy parecida a la de Apache Ambari, separando los servicios (denominados Clústeres en Cloudera Manager) de los servidores donde está desplegado Hadoop (denominado Hosts). Al acceder a la primera pantalla, se ve un resumen de los servicios instalados en el clúster, así como las principales métricas.

Íñigo Sanz (Dominio público)

Haciendo clic en la opción “Clústeres”, se puede ver los diferentes servicios de instalados:

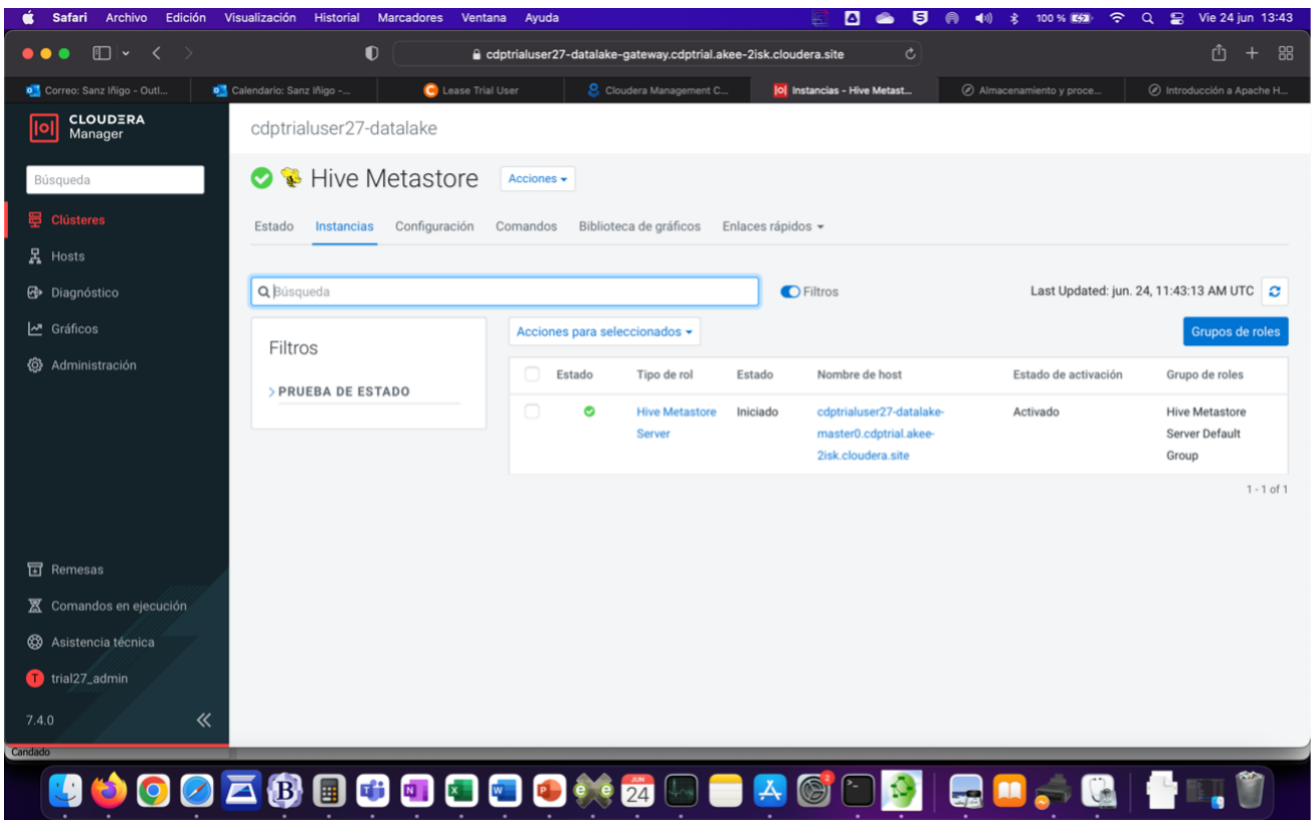
Íñigo Sanz (Dominio público)

Seleccionando uno de ellos, por ejemplo, Hive, se tiene una primera pantalla, denominada “Estado” con la información general del servicio: resumen del estado, historial, así como las principales métricas de monitorización.



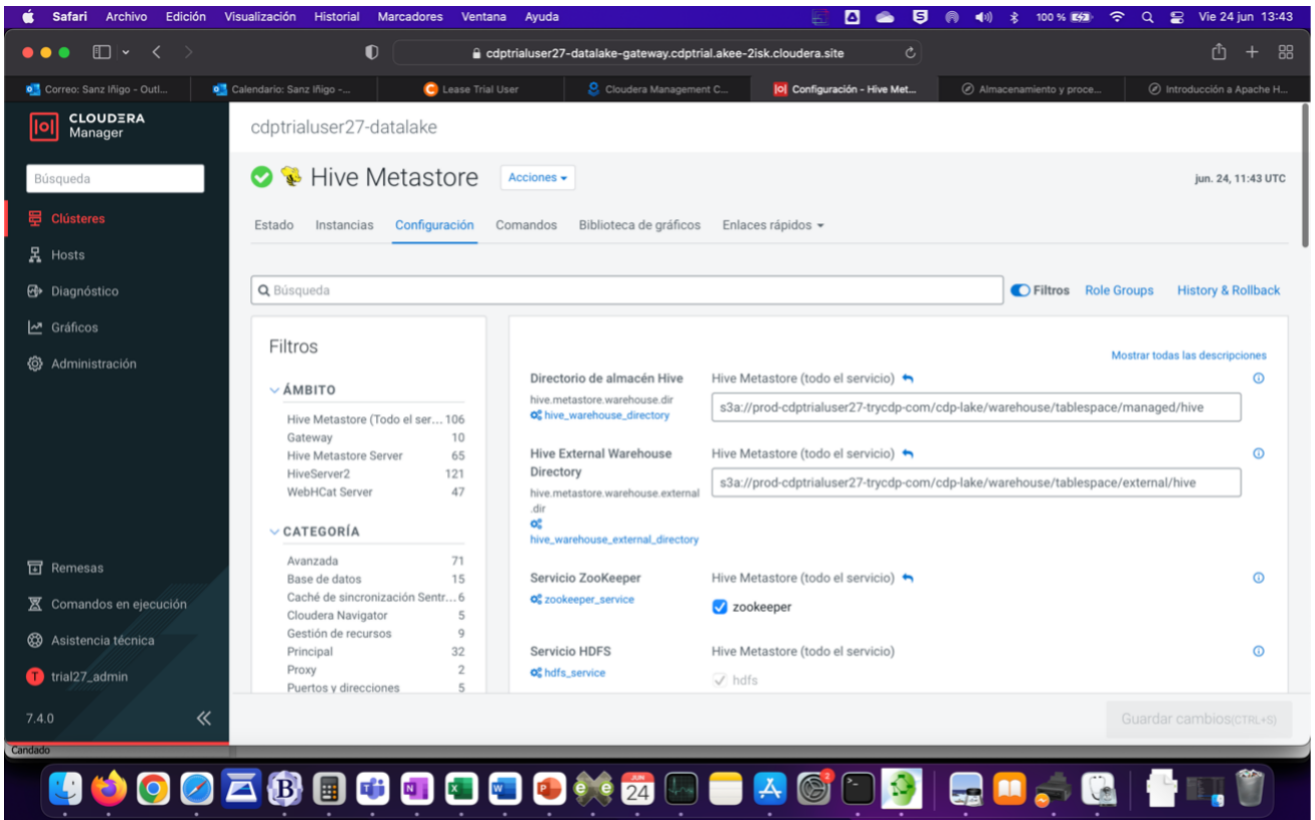
Íñigo Sanz (Dominio público)

Se puede navegar entre las pestañas, por ejemplo, accediendo a la pestaña Instancias se puede ver los servidores donde se encuentra desplegado el servicio:



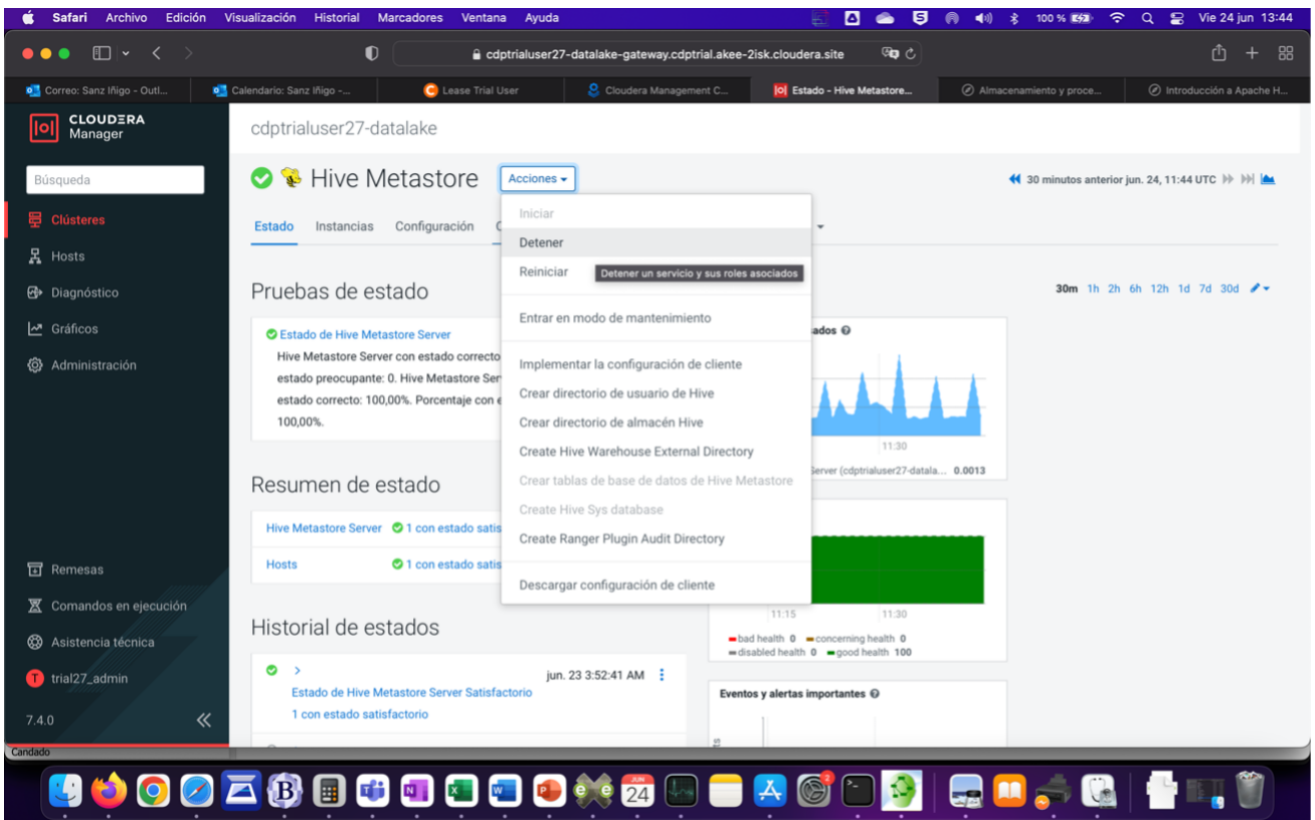
Íñigo Sanz (Dominio público)

En la pestaña "Configuración", se puede ver y modificar los parámetros de configuración del servicio.



Íñigo Sanz (Dominio público)

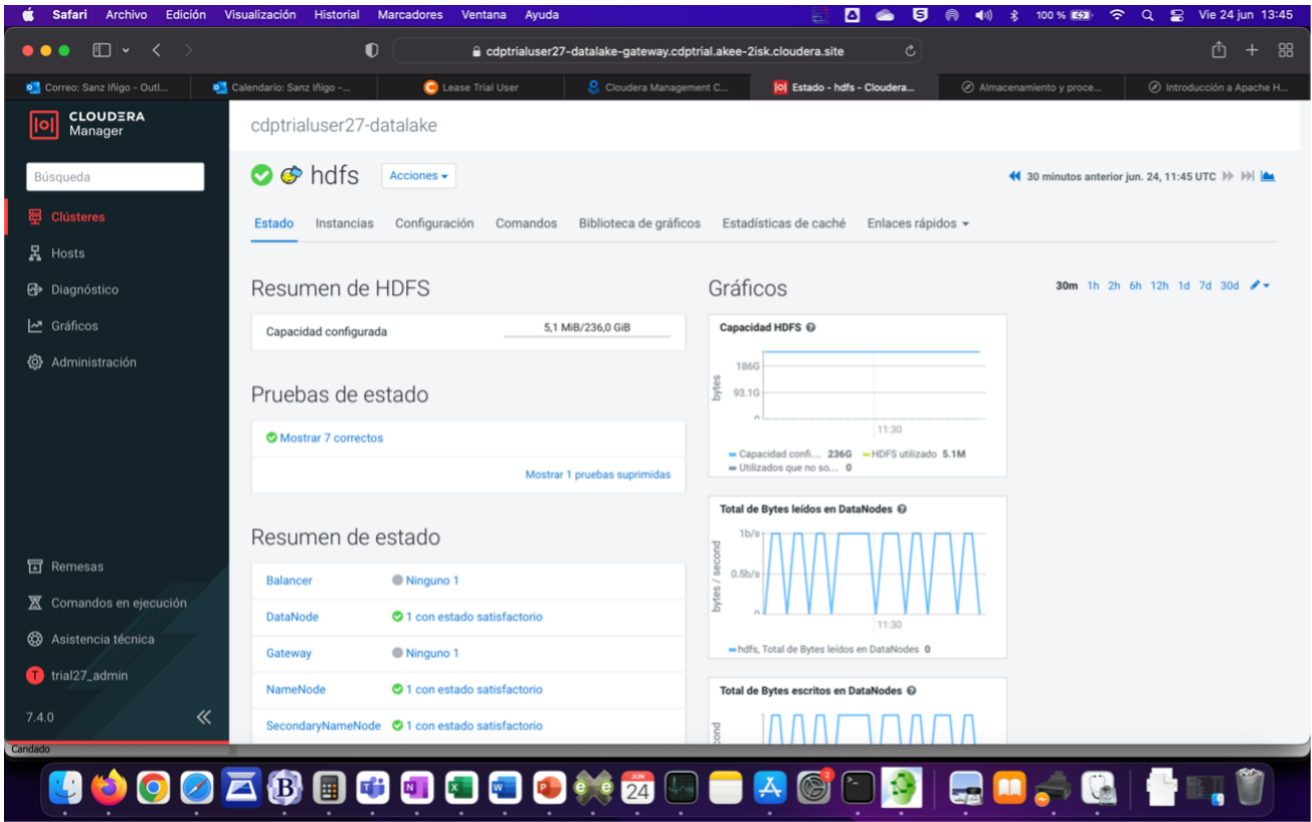
Asimismo, existe un botón “Acciones” que ofrece la posibilidad de detener, reiniciar o realizar otras actividades sobre el servicio.



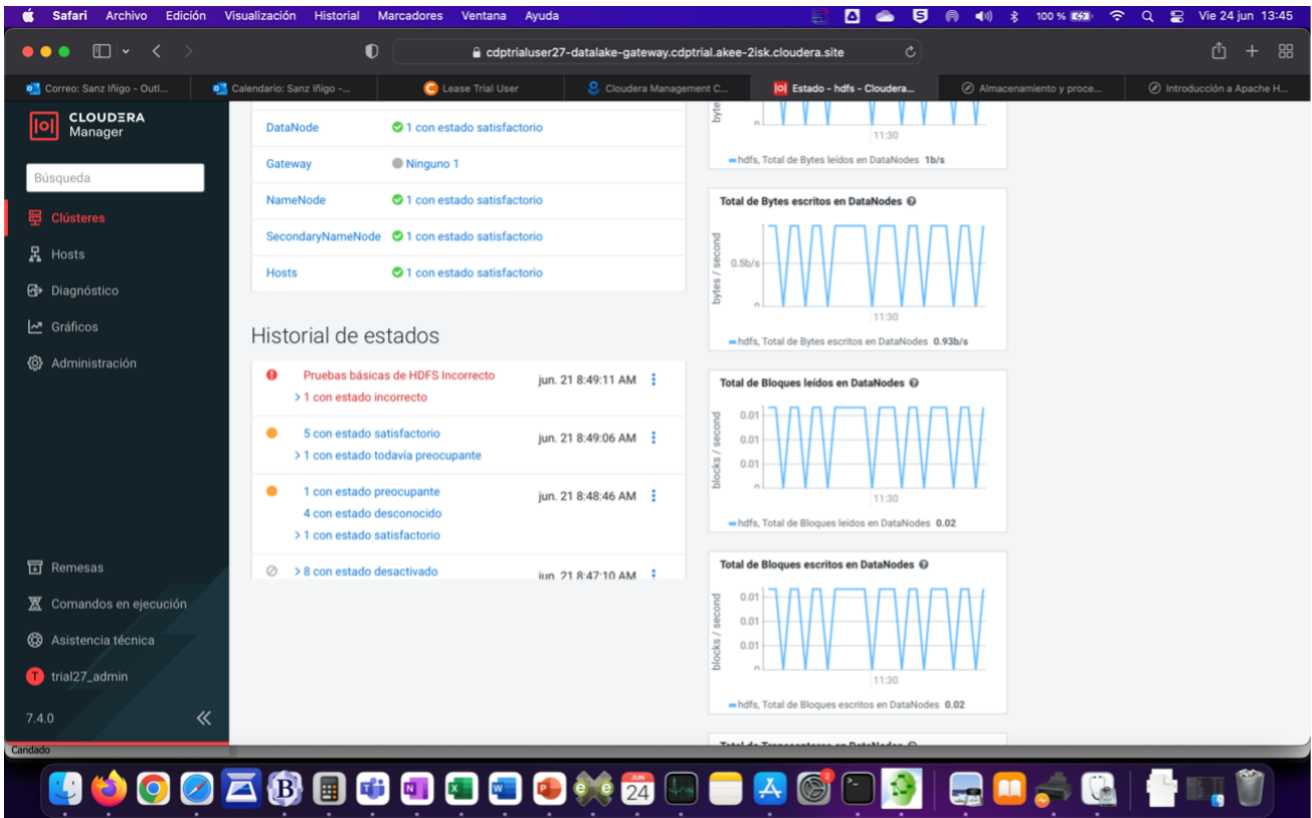
Íñigo Sanz (Dominio público)

A continuación se muestra las pantallas de administración del servicio HDFS, donde se puede comprobar que las opciones son similares a las aparecidas en Hive, o idénticas a las que se ofrecen en Apache Ambari.

Por ejemplo, la pantalla inicial de Estado muestra las métricas fundamentales de HDFS, así como el estado de los servicios:

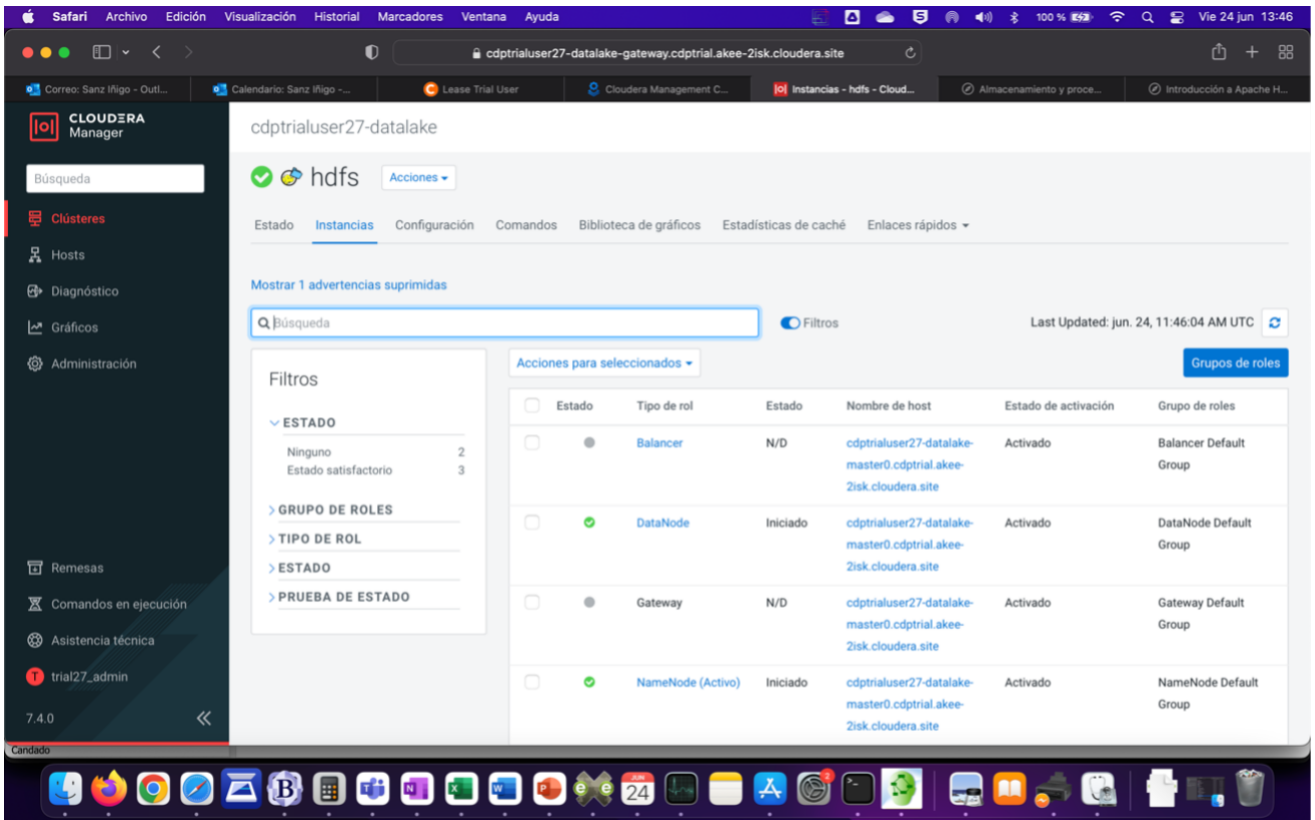


Íñigo Sanz (Dominio público)



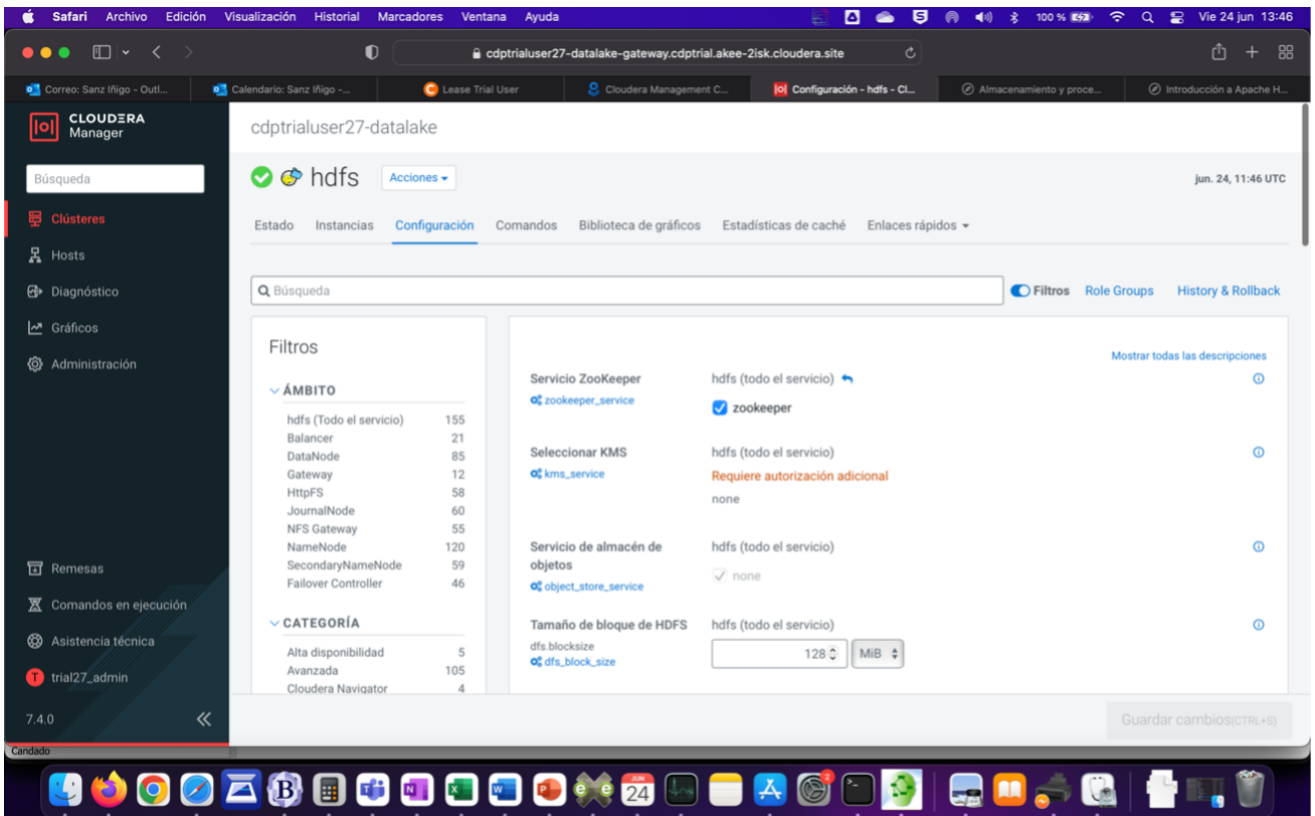
Íñigo Sanz (Dominio público)

En la pestaña de instancias, se puede comprobar que HDFS se encuentra desplegado en diferentes nodos, con un Namenode y un Datanode.



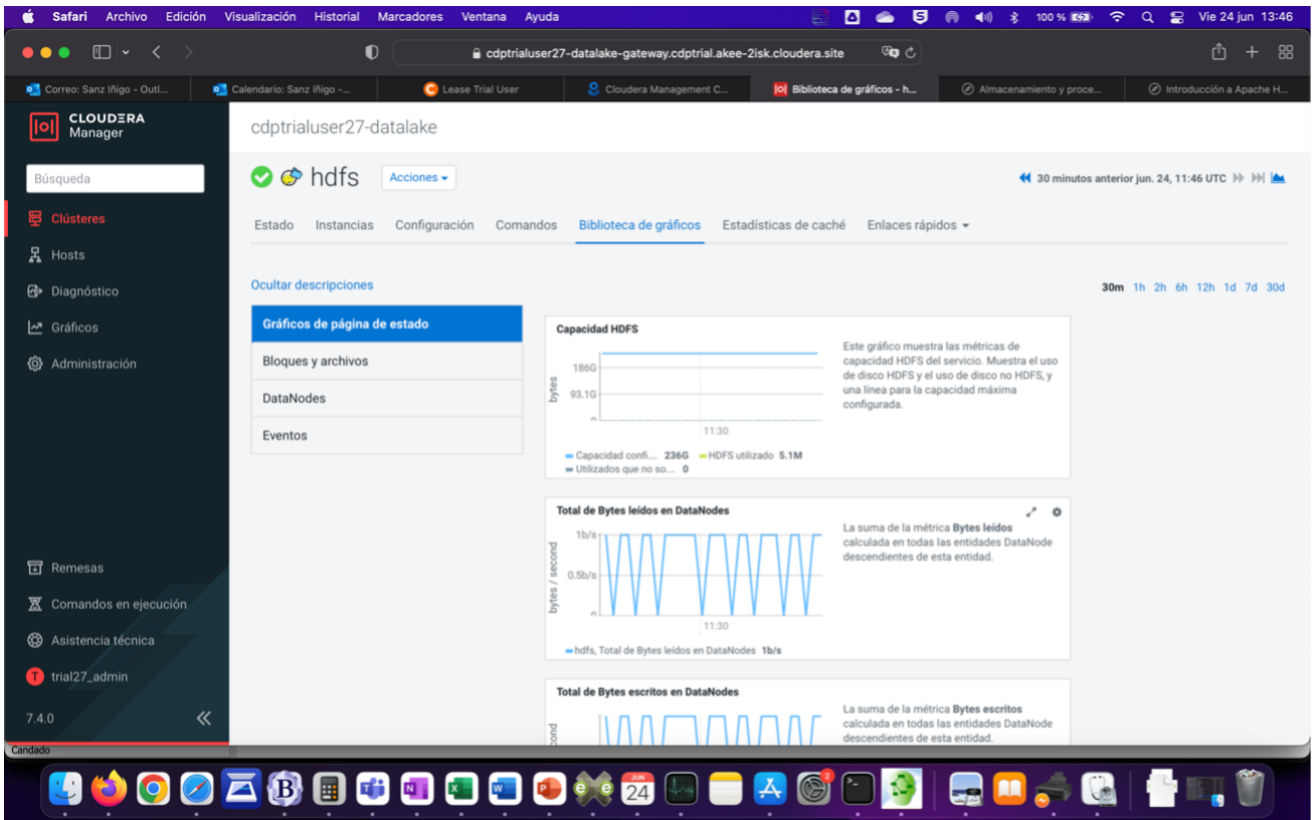
Íñigo Sanz (Dominio público)

En cuanto a la configuración, es similar a Hive, apareciendo las opciones relacionadas con HDFS:



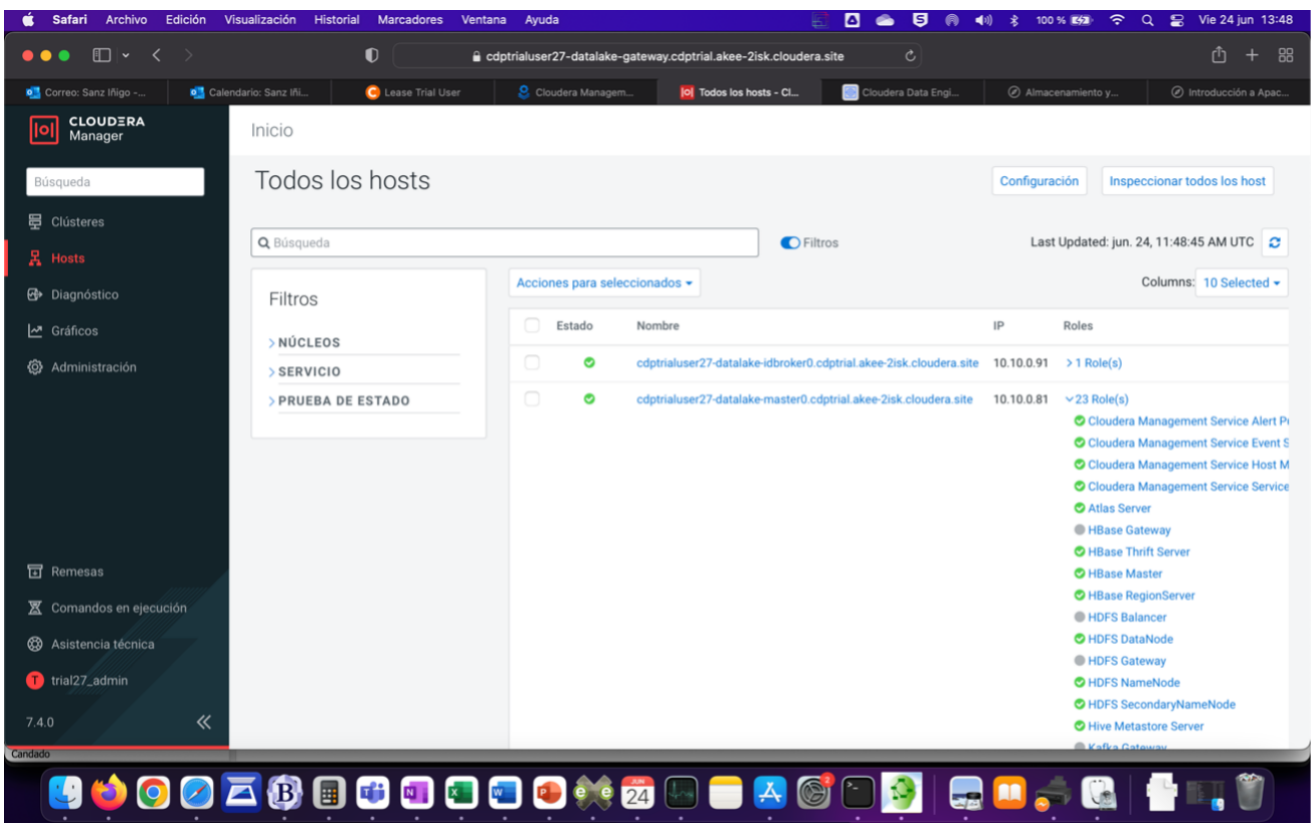
Íñigo Sanz (Dominio público)

La pestaña de “Biblioteca de gráficos” muestra la información de las métricas agrupadas por tipo:



Íñigo Sanz (Dominio público)

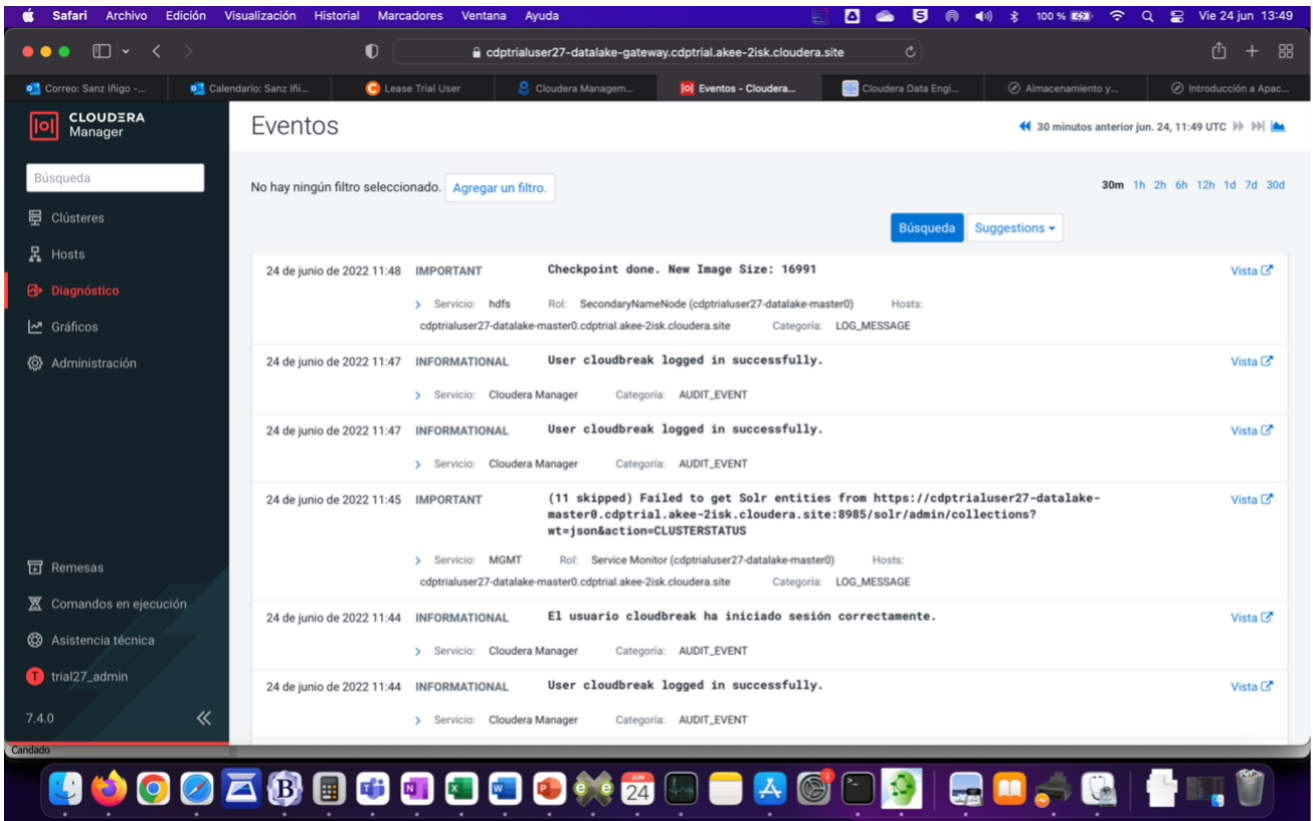
En cuanto a la opción de menú de Hosts, muestra los servidores que se están utilizando para el clúster. En este ejemplo, que se está utilizando un entorno de demostración de Cloudera, tenemos un clúster con dos servidores únicamente.



Íñigo Sanz (Dominio público)

En la opción de “Diagnóstico”, se muestran los mensajes de estado y las diferentes comprobaciones que se

han realizado recientemente sobre los diferentes servicios y nodos del clúster:



Íñigo Sanz (Dominio público)

En la opción “Gráficos” permite generar cuadros de mando a medida, y en la opción de Administración se dispone de opciones para la gestión de usuarios, versiones de los componentes, etc.

Para saber más

Si deseas obtener más información de Cloudera Manager, puedes acceder a la web oficial en [este enlace](#).

Asimismo, en la página de Cloudera tienes muchos vídeos de demostración de diferentes funcionalidades de Cloudera Manager. Puedes verlos en [este enlace](#).

Autoevaluación

Indica si las siguientes afirmaciones son verdaderas o falsas.

Cloudera Manager permite arrancar y parar los servicios del clúster.

Verdadero Falso

Verdadero

Verdadero: permite administrar todos los servicios de Hadoop instalados.

Cloudera Manager es una herramienta de trabajo para los data scientists.

Verdadero Falso

Falso

Falso: Cloudera Manager es una herramienta de trabajo para los administradores de sistemas.

6.- Ganglia.

Caso práctico

El equipo de IT del Banco Español de Inversiones, BEI, ha utilizado tradicionalmente otro sistema de monitorización de los diferentes clústers que había para otras aplicaciones, como los clústers de servidores web o de un grid de HPC.

Dado que conocen bien esa herramienta, que se llama Ganglia, se preguntan si podrían utilizarla para el clúster Hadoop, ya que en su día a día dominan su uso y podrían implementar cuadros de mando específicos para Hadoop, integrándolos en los existentes y centralizando la monitorización en una única herramienta.

Veamos cómo es Ganglia y si permite monitorizar clústers Hadoop.



[Thomas Ulrich](#) (Dominio público)

Ganglia es una herramienta opensource que permite la recogida de métricas de un sistema y su monitorización. No es una herramienta específica de Hadoop, ya que su propósito es ayudar en la monitorización de cualquier tipo de clúster, pero puede resultar útil en entornos en los que no se disponga de Apache Ambari o Cloudera Manager.

Ganglia se puede ejecutar, por lo tanto, en los nodos del clúster, de modo que Hadoop pueda enviar los datos de métricas a los agentes de Ganglia o recogerlos directamente del sistema operativo en el que se ejecutan los servicios de Hadoop en cada nodo. Además, Ganglia se puede integrar con Nagios para montar un sistema de alertas sobre las métricas recogidas en Ganglia.

Ganglia recopila métricas como el uso de la CPU y el espacio libre en el disco y también puede ayudar a monitorizar los nodos que están caídos.

Arquitectura de Ganglia

Hay cuatro componentes principales en un sistema de monitoreo de Ganglia:

- ✔ **gmond:** es un demonio que se ejecuta en cada nodo del clúster, cuyo trabajo es recopilar los datos de métricas de cada nodo. Cada nodo ejecuta el demonio gmond y el nodo recibirá métricas del resto de los nodos del clúster (todos los nodos se comunican con todos los nodos). Esto significa que el proceso de recogida de métricas (gmetad, que se explica a continuación) necesita solo un nodo para obtener las métricas del clúster y también en caso de un fallo de un nodo, Ganglia sigue proporcionando servicio.
- ✔ **gmetad:** este es el demonio que sondea los nodos en busca de datos de las métricas. Puede obtener un volcado de métricas para todo el clúster desde cualquier nodo del clúster. El demonio gmetad crea tablas denominadas RRD para almacenar los datos de métricas.
- ✔ **RRDtool:** este componente almacena los datos de las métricas recogidas por el demonio gmetad en cada nodo.
- ✔ **gweb:** esta es la interfaz web para visualizar las métricas recopiladas por el sistema de monitoreo de Ganglia a través de los datos almacenados en las bases de datos de RRD. Puede ver métricas específicas mediante gráficos y también crear gráficos personalizados profundizando en los detalles de una métrica o host específico. El proceso gweb es en realidad un programa PHP que se ejecuta en

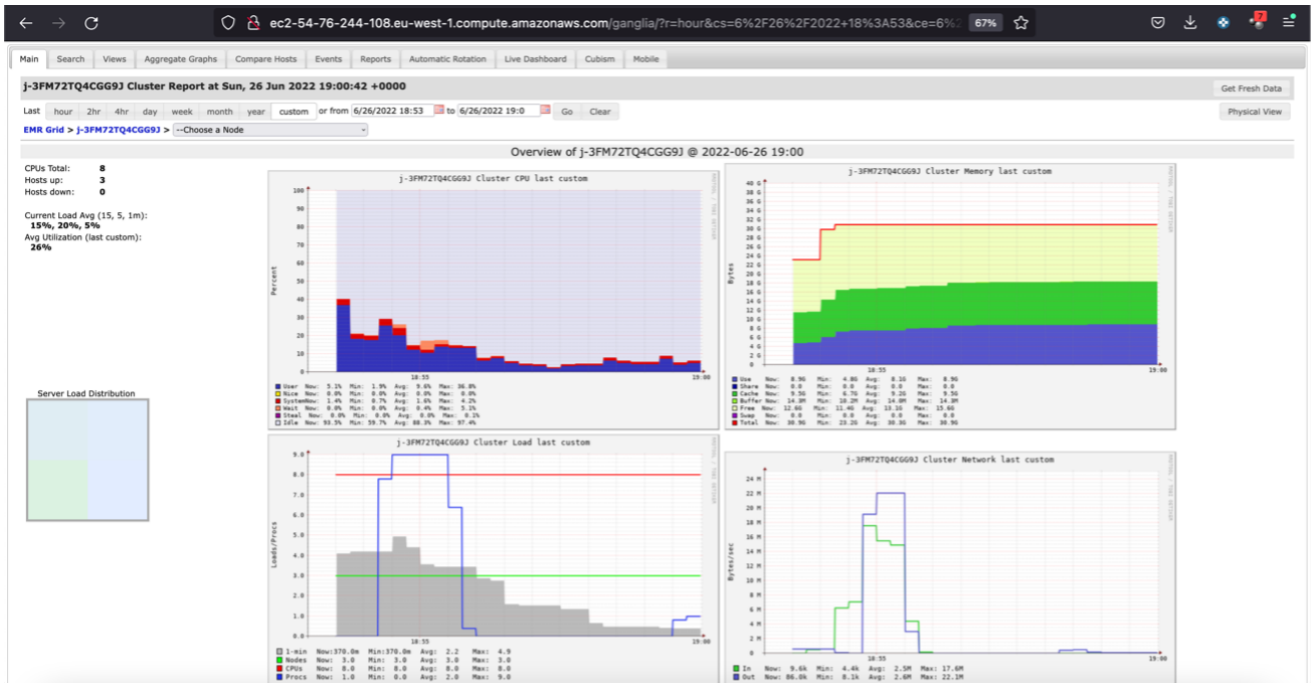
un servidor web Apache.

Funcionalidad de Ganglia

La instalación de Ganglia consiste en instalar los agentes en cada uno de los nodos y levantar gweb, el interfaz web, para la visualización de las métricas.

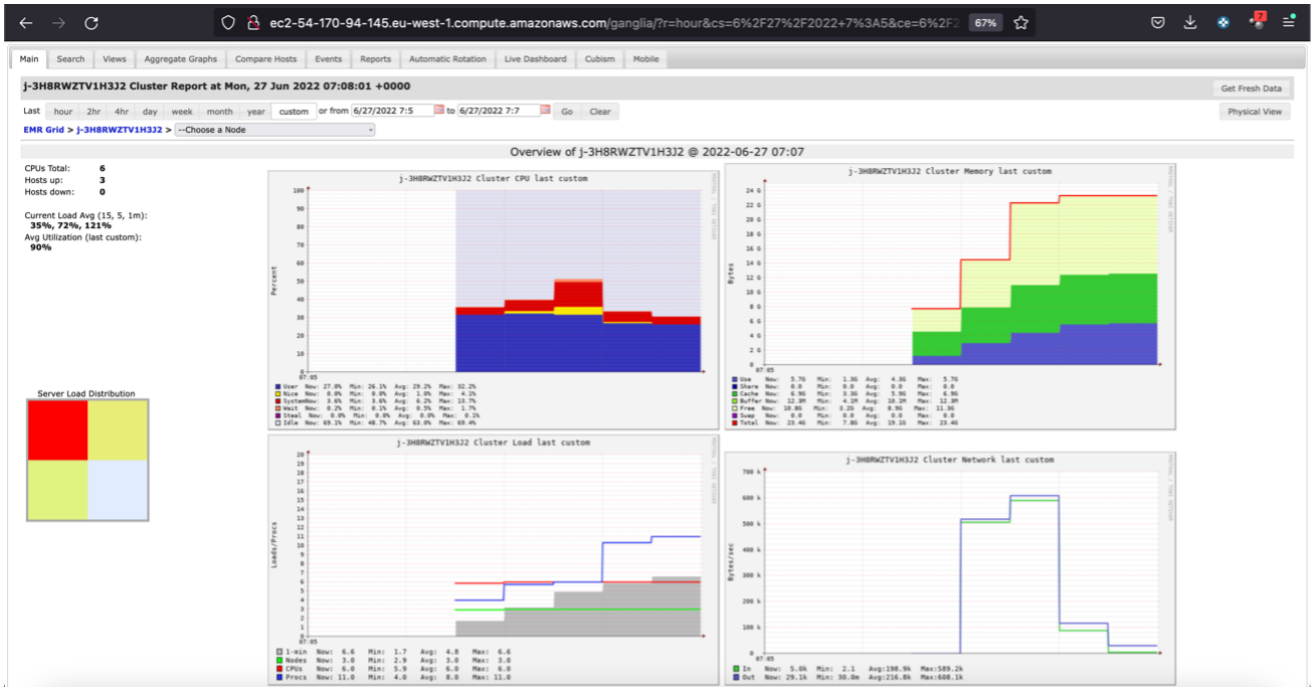
A continuación vamos a conocer las principales pantallas y funcionalidades que ofrece gweb, y para ello, vamos a visualizar cómo sería alguna pantalla de Ganglia para un clúster de 3 nodos de Hadoop en Amazon Web Services.

Al acceder a la pantalla principal, se muestran las principales métricas agregadas de los 3 nodos:



Íñigo Sanz (Dominio público)

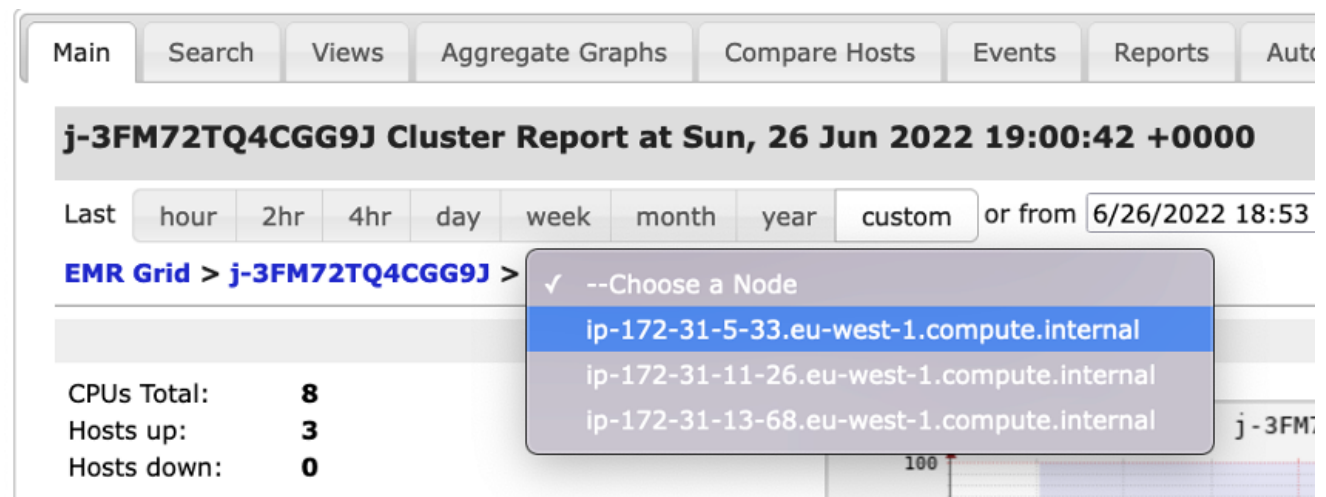
Se puede ver que muestra como gráfico la carga de CPU, de memoria o red. En caso de que algún nodo tenga alguna métrica por encima de los umbrales establecidos, se muestra en el mapa de la izquierda en rojo.



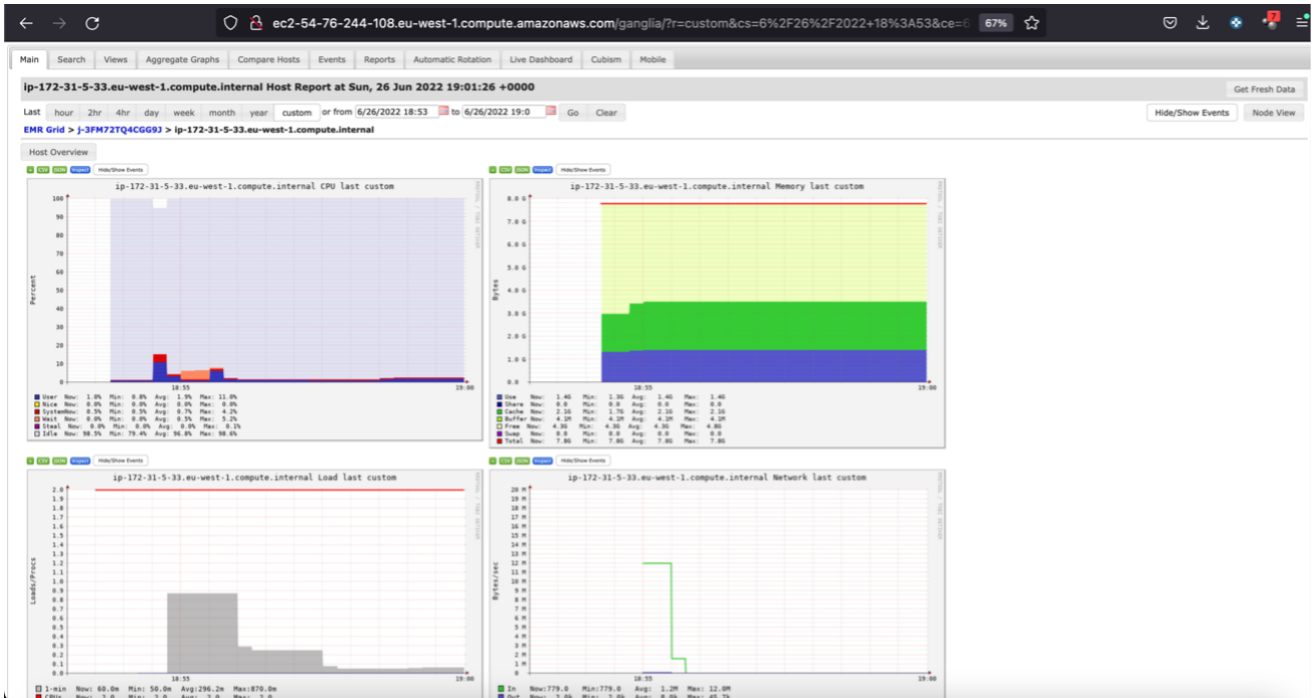
Íñigo Sanz (Dominio público)

Este mapa, representado en forma de cuadrado, suele contener un cuadrado más pequeño por cada nodo. En nuestro ejemplo hay sólo 3 cuadrados al haber sólo 3 nodos, pero en un sistema de producción con múltiples nodos, se divide en cuadrados más pequeños, y permitiendo en un vistazo rápido conocer el estado de los nodos del clúster.

En el desplegable que aparece en la parte superior izquierda, se puede elegir un nodo concreto del clúster para visualizar únicamente sus métricas:

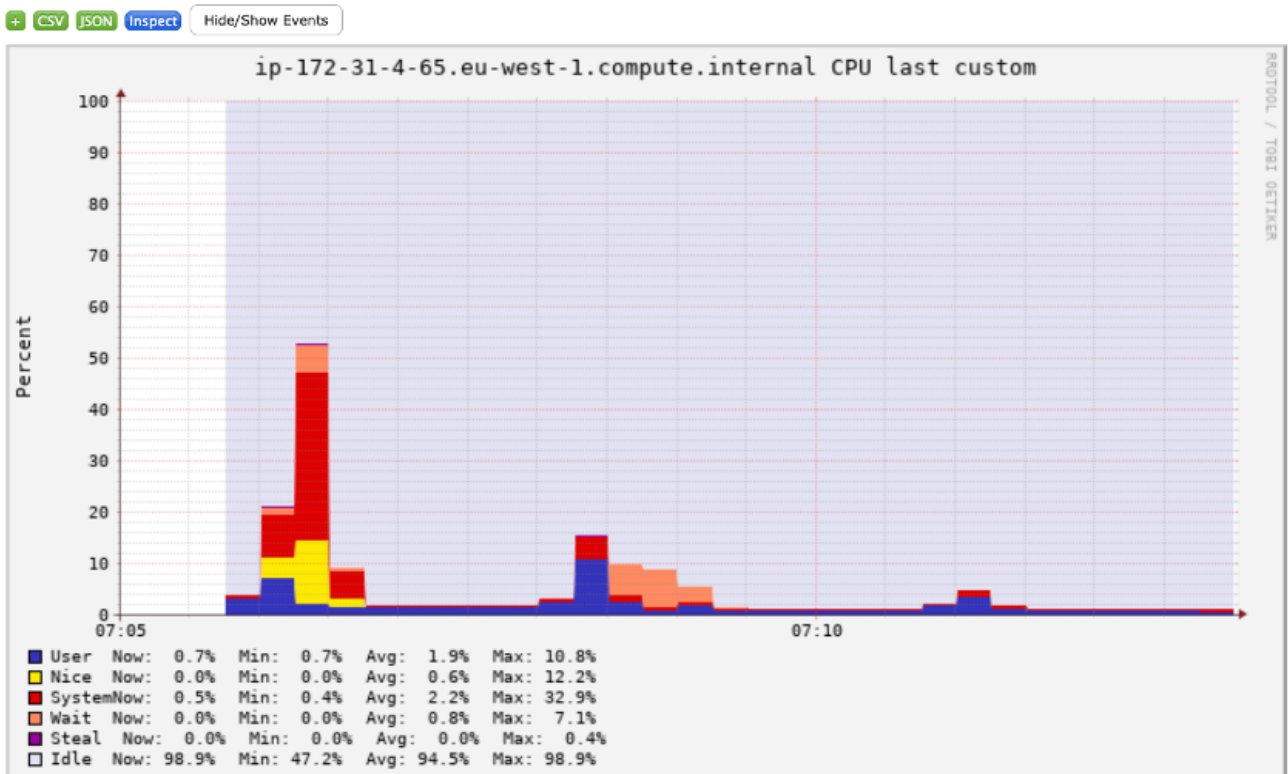


Íñigo Sanz (Dominio público)



Íñigo Sanz (Dominio público)

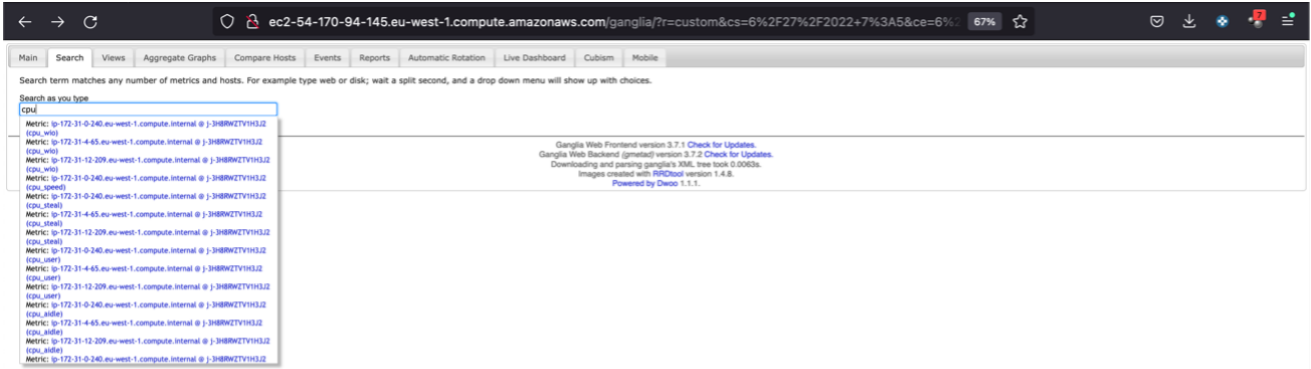
Todas los gráficos ofrecen la posibilidad de exportar los datos en CSV o JSON, como se puede ver en los botones verdes que aparecen en la parte superior del gráfico:



Íñigo Sanz (Dominio público)

O pulsando en el botón “+” se puede definir los umbrales de aviso y añadir el gráfico a un dashboard a medida que se puede construir en la pestaña “Views”.

En la pestaña “Search” se puede buscar por cualquier nodo o métrica, por ejemplo, buscando las métricas que contengan “cpu”



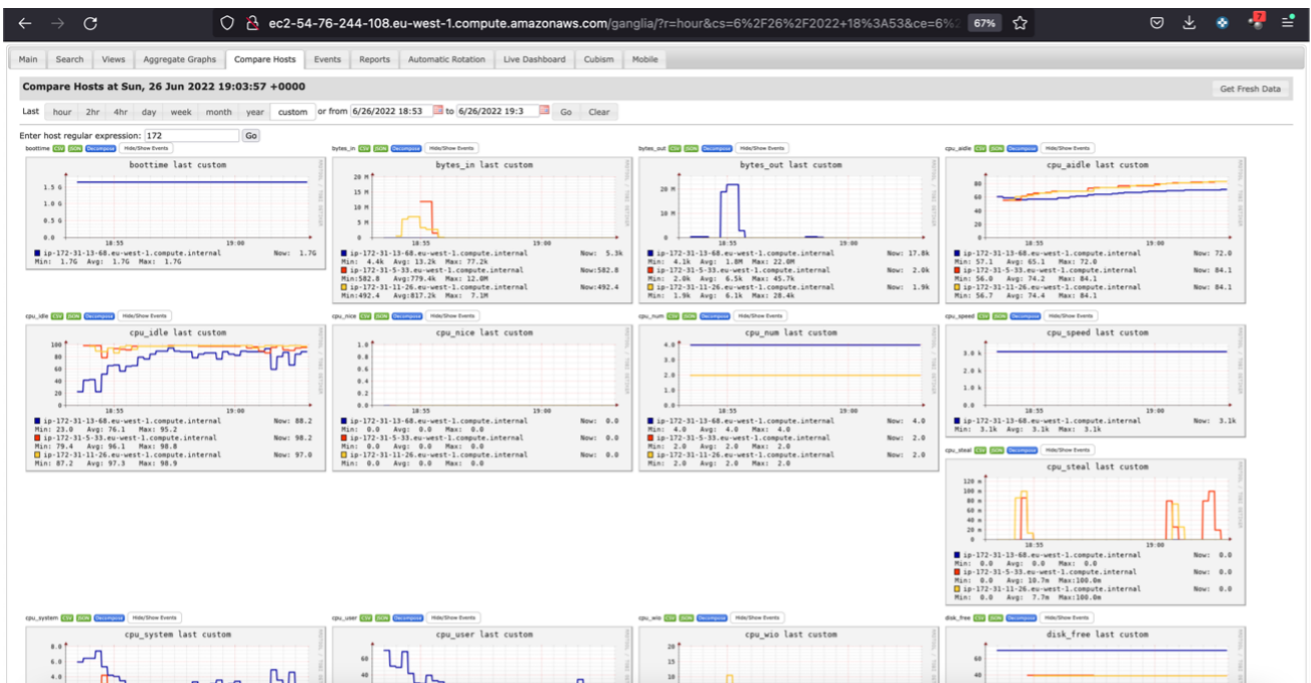
Íñigo Sanz (Dominio público)

O buscando los nodos que contengan el nombre “172” (todos los nodos del clúster del ejemplo tienen un nombre que empieza por 172).



Íñigo Sanz (Dominio público)

Las capacidades que ofrece Ganglia para la monitorización de sistemas distribuidos es muy amplia, pudiendo generar métricas agregadas, cuadros de mando a medida e infinidad de posibilidades. Por ejemplo, en la siguiente imagen se muestra un cuadro de mando comparando diferentes métricas de los 3 nodos que componen el clúster. Este tipo de cuadros de mando puede resultar muy útil para detectar problemas en algunos de los nodos:



Íñigo Sanz (Dominio público)

Sin embargo, pese a que Ganglia es un buen sistema de monitorización, la realidad es que en los clústers en los que se dispone de herramientas como Ambari o Cloudera Manager, éstas reemplazan por completo a Ganglia, ya que permiten no sólo monitorizar, sino también administrar el clúster en una única herramienta.

Para saber más

Si quieres conocer más información sobre Ganglia, puedes acceder a su página oficial en [este enlace](#).

Autoevaluación

¿Qué diferencias hay entre Ganglia y Ambari o Cloudera Manager?. Selecciona las opciones correctas:

- Ganglia no permite modificar parámetros de configuración de Hadoop.

- Ganglia no permite realizar acciones en Hadoop como parar un servicio o un nodo.

- Ganglia ofrece métricas de uso de CPU en los nodos, mientras que Ambari o Cloudera Manager no.

Mostrar retroalimentación

Solución

1. Correcto
2. Correcto
3. Incorrecto

