JESUÏTES El Clot
Escola del Clot

M013: Projecte de Desenvolupament d'Aplicacions Mutiplataforma

Implantar una solució de BigData amb un clúster Apache Hadoop

Curs: 2019-20

CFGS: DAM2

Alumne: Arnau Subirós Puigarnau

Data: 02-06-2020

Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

INDEX

pàgina

Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

1. Introducció al projecte

Inicialment aquest projecte consistia amb un clúster Hadoop utilitzant 3 Raspberry PI. Maluradament degut al confinament del COVID-19 no ha pogut ser així, i s'ha optat per buscar alternatives.

- Com que el portàtil té 16GB RAM s'ha pensat en crear 3 màquines virtuals de 4GB de RAM cadascuna ,deixant limitadament 4GB al host amfitrio (pot funcionar mentre el host carregui mol la memòria, ja que es bloquejaria.
    - En aquesta opció primer es fa fer amb sistema operatiu CentOS, després es volia fer amb Raspbian (simulant la Raspberri Pi) però té arquitectura ARM incompatible amb VirtualBox i VMWare en què m'he instal·lat Raspberry Pi Desktop (una distribució de Debian amb interfície de Raspberry Pi dissenyat per poder-ho emular en maquines virtuals, però amb limitacions). S'ha fet proves però s'ha trobat moltes incompatibilitats de sistema i problemes de diferents repositoris.

- També s'ha intentat fer-ho amb Dockers. Primer des de el sistema operatiu actual (Window 10). Que com que no disposo de la llicència Windows 10 Pro, Dockers s'havia utilitzat Dockers Toolbox on el servidor Dockers estaria ubicat en una màquina de Linux (una versió lleugera)en VirtualBox. S'ha fet proves amb altres sistemes operatius (Ubuntu 20.04 core i Centos 6) instal·lats en maquines virtuals. Però en intentar crear un clúster de diverses nodes, no s'ha aconseguit i s'ha descartat l'opció.

- Hem centrat en el sistema operatiu Centos6. On s'ha creat 3 tipus de clúster
    - El bàsic per practicar on nomes hi havia 1 node (màster-slave)
    - Un clúster hadoop d'1 master i 2 slaves o s'ha ampliat l'ecosistema Hadoop, instal.lant i configurant Hive,Spark i Ambari(no operatiu)
    - Clúster Hadoop d'un 1 master i 2 slaves utilitzant Ambari (des de 0), ja que en l'anterior tot i comentar les variables d'entorn, com que ja hi havia configuracions establertes, generava errors.

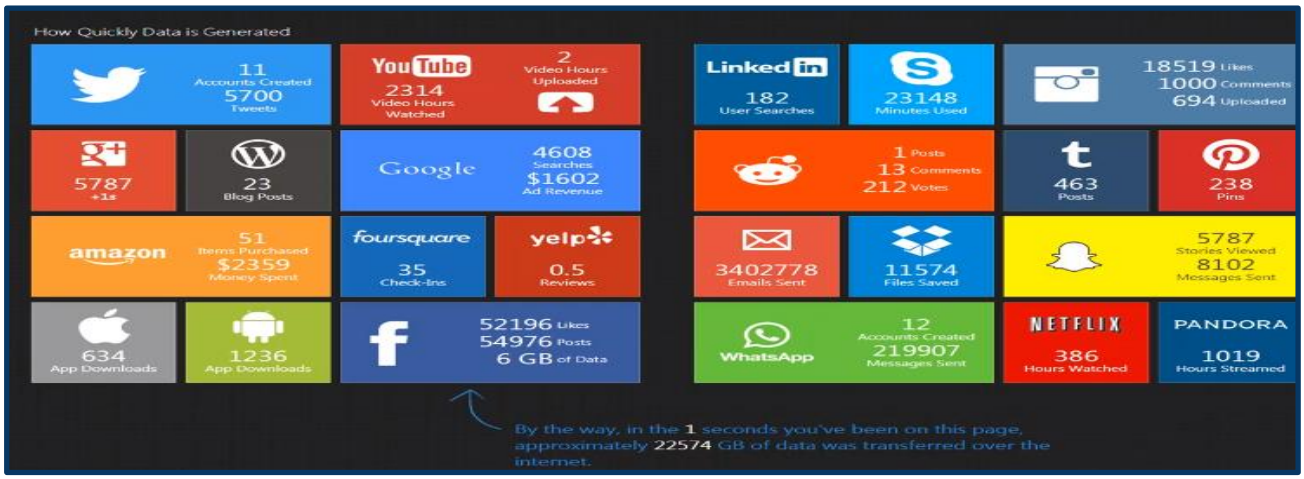
Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

1.1. Introducció a Big Data

Big Data és la convergència d'enormes quantitats de dades tan estructurades (ex: Base de dades relacionals en forma de taules Oracle, MySQL, etc.) com sense estructurar (dades de tweets, de Facebook, de blogs…)

- Es creen *Petabytes* [1] de dades diàriament
  - Xarxes socials
  - Mòbils
  - Sensors
  - Dades científiques ...



Hi ha una enorme quantitat de dades dels quals volem processar i extreure la informació. Quan parlem de Big Data parlem de les 3 V:

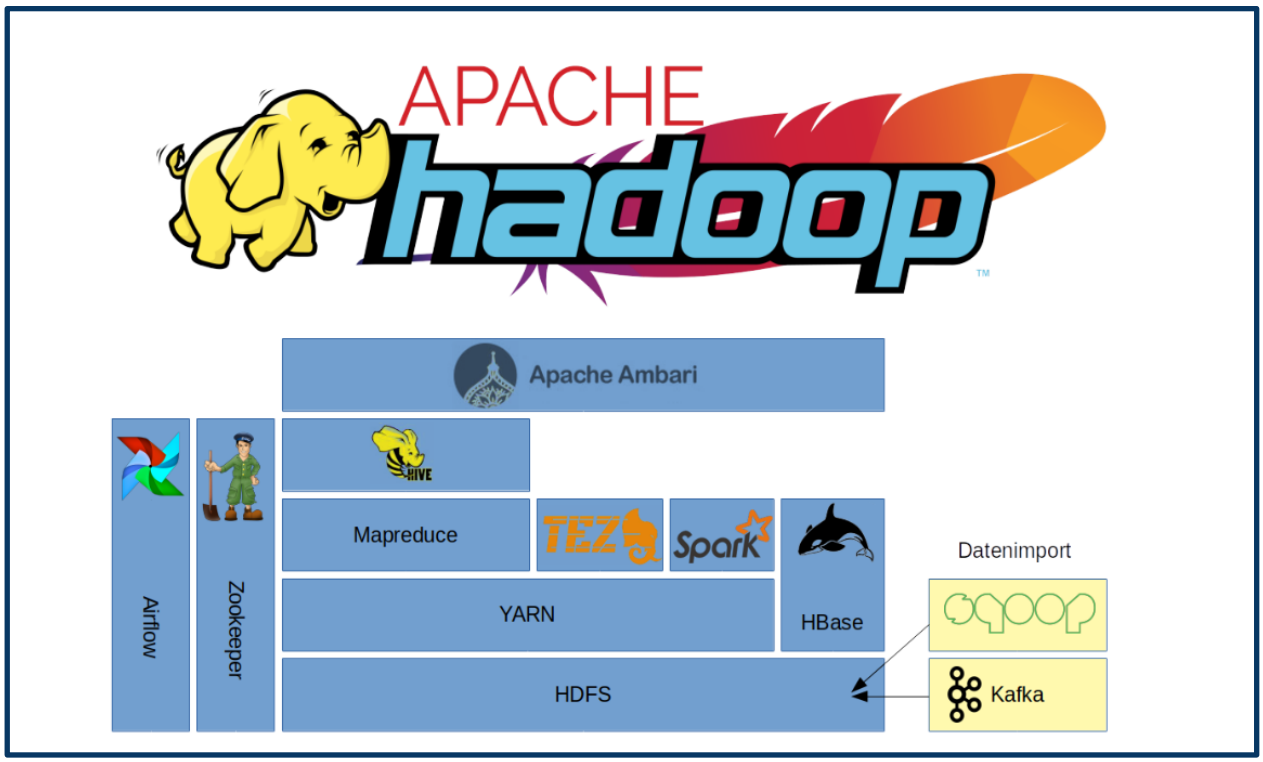
- Volum (Terabytes, Petabytes…)
- Varietat (d'informació: estructurat, no estructurat)
- Velocitat

Com que les tecnologies tradicionals no poden fer front aquesta quantitat d'informació, és necessari utilitzar noves tecnologies. De grans servidors a entorns distribuïts.

[1] **Nota**: 1 Petabyte són 1.000.000 GB o dit d'una altra manera, 1e+6GB

1.1.1. Tecnologia Big Data : Apache Hadoop

Apache Hadoop és un framework (basat en Java) que permet el processament distribuït de grans conjunts de dades en grups de computadores utilitzant models de programació simples. Està dissenyat per a escalar des de servidors individuals a milers de màquines, cadascuna de les quals ofereix computació i emmagatzematge local.



Nom i Cognoms **Data**

Arnau Subirós Puigarnau 02-06-2020

1.2. Apache Hadoop

Apache Hadoop és un entorn distribuït on el seu "core està format per 2 components bàsics:

- *dades*
- *processos*

- És un sistema de tipus clúster que es pot escalar d'acord amb les nostres necessitats(de pocs nodes a milers de màquines), a més es pot utilitzar un hardware relativament econòmic.
- Implementa processament en paral·lel a través de nodes de dades en un sistema de fitxers distribuït.
- Està dissenyat per executar-se en servidors de baix cost i disposa una gran tolerància a les fallades.
- Subministra llibreries Open Source per la computació distribuïda utilitzant diversos components:
  - *Hadoop Common*
  - *MapReduce*
  - *Hadoop Distributed File System (HDFS)*

1.2.1. HDFS

HDFS és el sistema d'arxius que utilitza Hadoop com a sistema d'emmagatzematge per als arxius.

- És un sistema distribuït, escalable i portàtil
- Permet connectar nodes (màquines de baix cost) les quals formen els clústers, en aquests nodes s'emmagatzemarà la informació de forma distribuïda
- Les aplicacions o comandos s'executaran seguint el model MapReduce. Pel fet que Hadoop tracta amb grans volums de dades es veurà afectat per la velocitat de transferència, sigui de xarxa o de velocitat d'escriptura en disc.

- Fa possible que els càlculs s'executin en la màquina on resideixen les dades en lloc de moure les dades on es realitzen la petició d'aquests.

Característiques generals dels Hadoop HDFS

- ❑ Una escriptura i moltes lectures.
- ❑ Tolerància a fallades i recuperació automàtica en cas de fallada.
- ❑ Coherència en les dades.
- ❑ Replicació de dades.
- ❑ Processament de les dades on estan emmagatzemats, en lloc de moure'ls.
- ❑ Facilitat de processament a través de sistemes heterogenis de maquinari i sistema operatiu.
- ❑ Escalabilitat per a processar i emmagatzemar grans quantitats de dades.

HDFS gestiona l'emmagatzematge en el clúster, dividint els fitxers en blocs i emmagatzemat copies duplicades a través dels nodes (mida del bloc 128 MB).

Per defecte es replicarà en 3 nodes diferents.

En crear un Clúster Hadoop hi haurà:

- Un node que actuarà com a *"màster"* de dades. Només té metadades i gestionarà les dades dels nodes *"slaves*".
- La resta de nodes són *"slaves*" que contenen les dades.

Disposa d'uns fitxers (metadades) que gestionen els canvis que es produeixen al clúster de HDFS que són:

- edits_000xxxxx → Son els canvis que es produeixen dins la base de dades de HDFS
- edits_inprogress_xxxx →A on s'escriuen les dades en temps real
- fsimage_00000xxxxx → Conte una còpia d'un moment en el temps en el sistema de fitxers

Quan s'arrenca el **HDFS** es llegeix l'últim **fsimage** que es carrega a la memòria del **Namenode**, no es toca sinó que es construeix el fitxer **edits_inprogres**s (es canvien les dades en memòria i no al **fsimage** original per guanyar rendiment)

- ❑ Cada cert temps se sincronitza amb el **Secondary Namenode** (la qual cosa s'ha anat guardant amb el que hi havia en aquella imatge)

- ❑ Aquests fitxers no es toquen

```
[hadoop@node1 current]$ ls -ls
total 2080
   4 -rw-rw-r--. 1 hadoop hadoop      42  5 mai 10:52 edits_0000000000000000001-0000000000000000002
   4 -rw-rw-r--. 1 hadoop hadoop      42  5 mai 11:53 edits_0000000000000000003-0000000000000000004
1024 -rw-rw-r--. 1 hadoop hadoop 1048576  5 mai 11:53 edits_0000000000000000005-0000000000000000005
1024 -rw-rw-r--. 1 hadoop hadoop 1048576  5 mai 12:00 edits_inprogress_0000000000000000006
   4 -rw-rw-r--. 1 hadoop hadoop     325  5 mai 10:39 fsimage_0000000000000000000
   4 -rw-rw-r--. 1 hadoop hadoop      62  5 mai 10:39 fsimage_0000000000000000000.md5
   4 -rw-rw-r--. 1 hadoop hadoop     325  5 mai 11:53 fsimage_0000000000000000004
   4 -rw-rw-r--. 1 hadoop hadoop      62  5 mai 11:53 fsimage_0000000000000000004.md5
   4 -rw-rw-r--. 1 hadoop hadoop       2  5 mai 12:00 seen_txid
   4 -rw-rw-r--. 1 hadoop hadoop     215  5 mai 10:39 VERSION
[hadoop@node1 current]$
```

1.2.2. PROCESSOS

És el component de processament de Hadoop. Consisteix en un framework de programació (llibreries i entorn d'execució) que treballa sobre HDFS i es basa en l'ús de dos tipus de funcions:

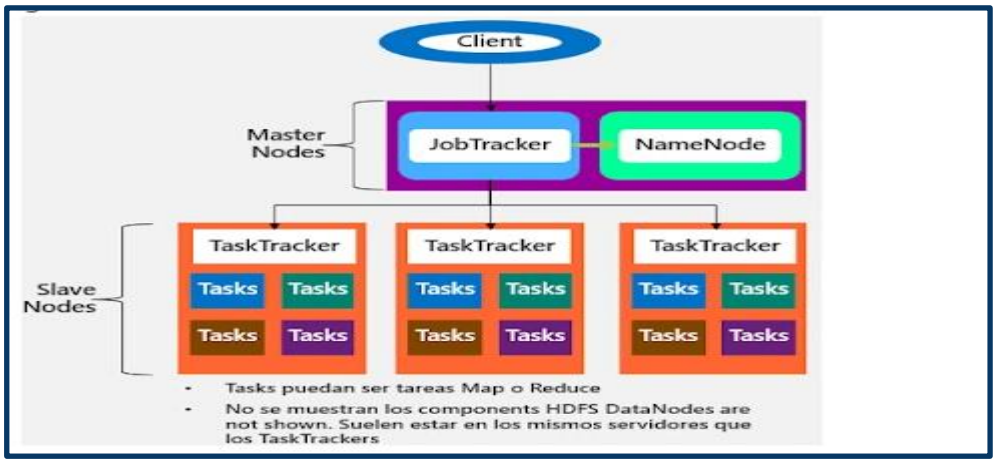
- **Map** – *Divideix* la tasca d'entrada en subtasques i les executa entre diferents nodes.
- **Reduce** – *L*a funció *"Reduce"* recull les respostes de les subtasques en cada subnode i les combina i agrupa per a obtenir la resposta final.

Resource Manager(segons les versions de Hadoop)

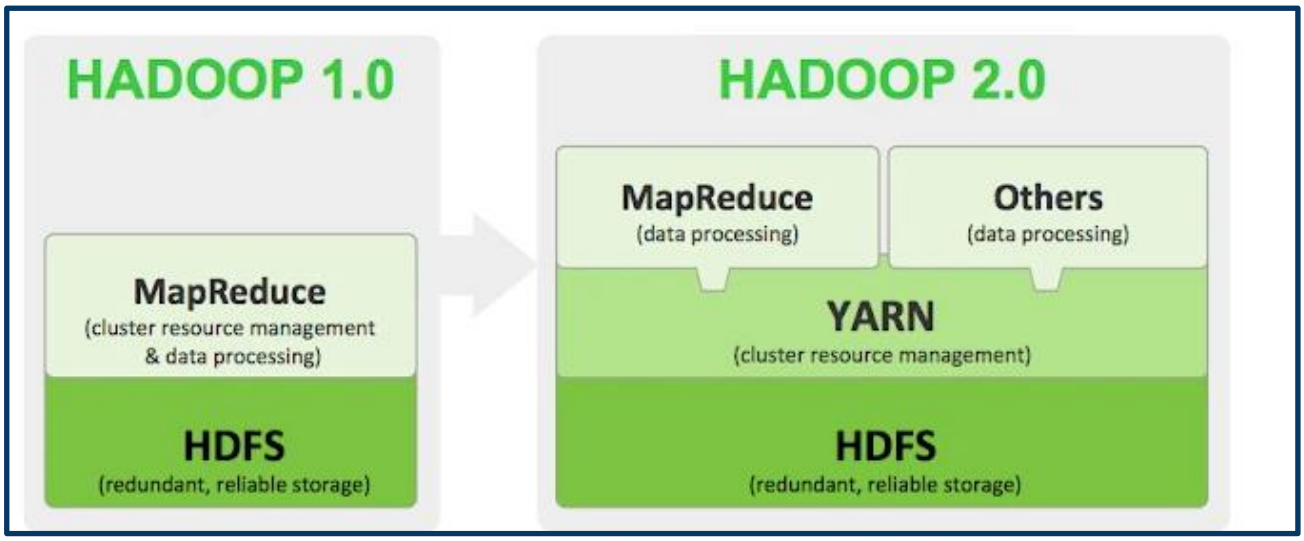
- **Hadoop 1.x :** Utilitza Map Reduce V1

  Cada màquina d'un clúster Hadoop té un servidor MapReduce que es diu TaskTracker. Al seu torn, hi ha un gestor de Jobs per cada clúster, el JobTracker, que s'encarrega de dividir cada procés a realitzar en subprocessos, i distribuir la computació d'aquests subprocessos entre diferents màquines del clúster, enviánt als TaskTrackers de cadascuna d'elles el job que li correspon realitzar.

  Nota: si queia el Job Tracker o se saturava, el clúster deixava de funcionar.



- **Hadoop 2.x [2]:** A partir de l'any 2011 amb la versió 2 incorpora YARN (Yet Another Resourge Manager), un gestor del clúster millorat que separa les tasques que realitzava el Job Tracker.



En la versió 2 es va desenvolupar una nova versió de ***MapReduce***,[3] molt diferent de l'anterior, ja que va deixar d'estar integrada en el Core (s'executa com una aplicació independent). Gràcies a això, ara és possible l'accés al sistema de fitxers de Hadoop des d'altres entorns de programació i execució

[2] És la versió que he utilitzat

[3] És un algoritme de processament de dades que implementa processos en paral.lel.De forma simple distribueix les tasques a través dels nodes d'un clúster executant la funció "map" .

- La funció "map" estudia el problema, ho divideix en parts i les envia a diferents màquines perquè totes les parts puguin executar-se de forma simultanea
- Els resultats d'aquest procés paral·lel és recullen i es distribueixen a través dels diferents servidors que executen una funció "reduce" que agafa els resultats de les parts i les recombina per obtenir una resposta simple

- **Hadoop 3.x :** La primera versió 3.0.3 va aparèixer l'any 2018. Per a un clúster, la gestió de recursos utilitza YARN, amb totes les característiques.

1.2.3. Altres productes implicats (ecosistema Hadoop)

1.2.3.1. **Hbase**: Una base de dades orientada a valors/clau que s'executa sobre HDFS. Aquesta base de dades no relacional és de tipus columnar i de codi obert. Distribuïda i escalable utilitzat en l'emmagatzemant de Big Data

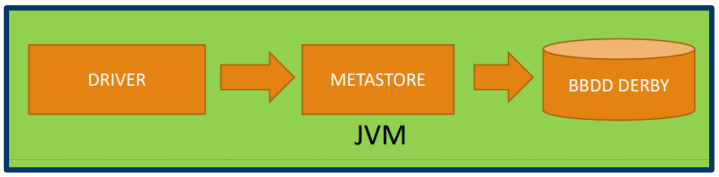
1.2.3.2. **Hive** : Permet accedir a HDFS com si fos una Base de dades

Hive implementa una variant al SQL, anomenada HQL (Hive QL). Quan s'executen consultes HQL, Hive converteix la consulta HQL a un treball MapReduce que és executat per a obtenir les dades.

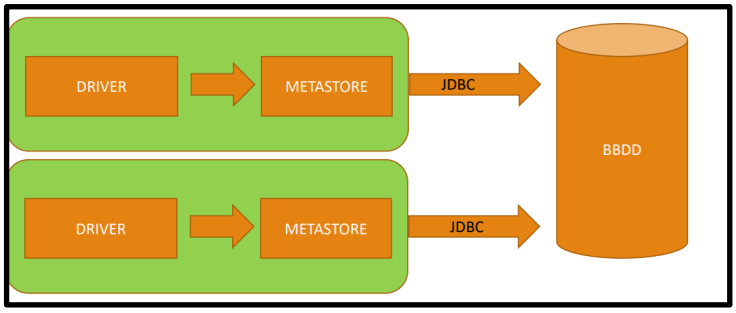
- ❑ Simplifica el desenvolupament i gestió amb Hadoop
- ❑ HIVE té un metastore que és un repositori central per les metadades de Hive.

Hi ha 3 tipus HIVE:

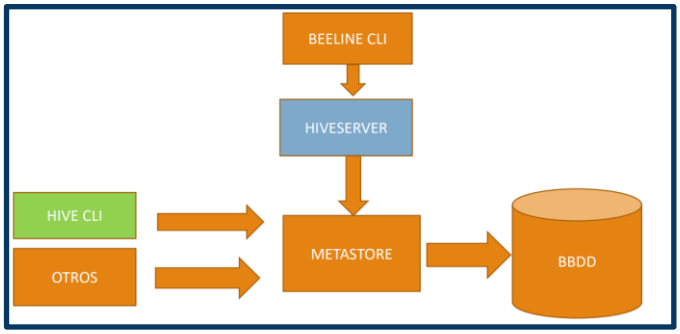
- **Encastat**: Tot està a la mateixa maquina virtual JAVA , però només ho pot utilitza 1 usuari)



- **Local**: Més potent que el metode "Encastat". JVM executa el programa + el servei de metastore i despréss via JDBC accedim a la base de bades, per exemple MySQL. Aquest mètode ens permet tenir múltiples usuaris però per defecte tot està a la mateixa maquina .



- **Remot** On tenim un servei metastore que es accedit per un servidor anomenat Hive Server. Aquest Hive Server pot ser accedit per determinats clients utlitzant beeliene o hive cli.



1.2.3.3. Pig: Llenguatge de alt nivel (de tipus scripting) pe r gestionar els fluxes de dades i execucions d'aplicacions sobre Hadoop.

- Treballa amb paral.lel i permet gestionar gran quantitat d'informacio
- Es un compilador que genera comandos MapReduce

1.2.3.4. Mahout: Entorn d'aprenantatge de màquines implementat en Hadoop.

- Machine Learning
- Data Mining

1.2.3.5. Zookeeper : és un servei per mantenir la configuració , coordinació i aprovisionament d'aplicacions distribuïdes .No nomes funciona amb Hadoop(no és un producte de Hadoop, sino de Apache), però és molt útil en aquesta arquitectura

- Elimina la complexitat de la gestió distribuïda en la plataforma
- Funciona amb Java
- Indispensable si es vol utilitzar Apache Ambari

1.2.3.6. Sqoop: Eines dissenyades per transferir dades massives des de Hadoop a altres entorns com base de dades relacionals (gestors de dades estructurades).

- Sqoop ofereix connectors per integrar Hadoop amb altres sistemes com Oracle o SqlServer

1.2.3.7. Flume: és un servei distribuït per distribuir, afegir i recollir grans quantitats de informació (especialment registres) com fitxers de logs, paquets de Twitter,etc.

- Té una arquitectura de tipus streaming amb un flux de dades molt potent i personalitzable

1.2.3.8. Cassandra: és un sistema de gestió de base de dades NoSQL distribuït de tipus columnar.

- Dissenyat per a manejar grans quantitats de dades en molts servidors bàsics, proporcionant alta disponibilitat sense un sol punt de falla.

1.2.3.9. Ambari : és una eina molt potent que serveix p er instal·lar, configurar, mantenir i monitoritzar Hadoop des d'una plataforma gràfica.

- Podrem instal·lar Hadoop des de 0, configurar-ho, gestionar-lo i monitorar-lo
- Simplifica enormement la gestió d'Hadoop

1.2.3.10. HUE : És un entorn que ens permet treballar de forma més amigable amb un entorn Hadoop. És una eina gràfica amb una sèrie de característiques

- Té editor per treballar amb Hive,Impala,Pig

- Té un Dashboard
- Te un Scheduler (Planificador)
- Es un navegador de components
- Podem navegar pels directoris HDFS
- Accedir al metastore de Hive

1.2.3.11. Spark : És un motor eficient de processament de dades a gran escala.

- Implementa processament en temps real al contrari que MapReduce
- Es molt més ràpid que MapReduce
- Treballa de forma massiva en memòria
- Pot funcionar stand-alone (no necessita Hadoop)
- Spark pot treballar tant en Java,Scala ,Python i R
- Permet l'anàlisis de gran quantitat de dades
- Integra diferents entorns com Base de Dades NoSQL, Real Time, machine learning, anàlisis de grafos
- Tambe podriem mesclar aplicacions Spark i MapReduce para batch i Real Time.
- Soport diferents fonts de dades
  - hive
  - json
  - cassandra
  - csv
  - rdbms, etc

- Spark constitueix un "core" i un conjunt de llibreries com a més importants:
  - Spark SQL
  - Spark Streaming
  - Spark Mlib
  - GraphX

1.2.3.11.2. Spark CORE: És el motor base per al processament en escala i distribuït.

- Esta programat amb Scala però hi ha APIS per Python, JAVA i R

- S'encarrega de:
  - Gestió de la memòria
  - Recuperació envers les fallades
  - Planificació, distribució de treballs del clúster
  - Motorització del treball

- ❏ Accedeixes als sistemes d'emmagatzemant

- Utilitza una estructura de dades especial denominada RDD:
  - ❏ Realitzen processos
  - ❏ Són col·leccions de registres immutables i particionades que a més a més poden ser utilitzades en paral·lel
  - ❏ Poden contenir qualsevol classe d'objectes de Java, Python, Scala o personalitzats

1.2.4. Distribucions Hadoop

Hi ha diferents empreses que ofereixen solucions empaquetades per Hadoop, entre les més importants hi ha:

Distribucions Hadoop	
Nom	**Producte**
Amazon Web Service (AWS)	Amazon Elastic MapReduce
Cloudera	Cloudera Enterprise
Hortonworks	Hortonworks Data Plataform
Intel	Intel Distribution for Apache Hadoop
MapR Technologies	MapR M3 - MapR M7
Microsoft	Windows Azure HDInsight
Pivotal Software	Pivotal HD
Teradata	Teradata Open Distribution Hadoop (TDH)

2. Introducció al Clúster Hadoop

Un clúster de Hadoop no és més que un grup de computadores connectades entre si a través de LAN. Ho fem servir per a emmagatzemar i processar grans conjunts de dades.

- ❏ Els clústers d' Hadoop tenen una sèrie de maquinari bàsic connectat entre si.
- ❏ Es comuniquen amb una màquina d'alta gamma que actua com un mestre.
- ❏ Aquests mestres i esclaus implementen la computació distribuïda sobre l'emmagatzematge de dades distribuïdes.

El clúster de Hadoop té una arquitectura *"master-slave"*

- ➢ **Master :** és una màquina amb una bona configuració de memòria i CPU. Té 2 "demons (serveis executant-se en segon pla) que són:
  - ○ ***NameNode***
    - ■ Administra l'espai de noms del sistema d'arxius
    - ■ Regula l'accés als arxius per part del client
    - ■ Emmagatzema metadades de dades reals (ruta d'arxius, núm.de blocs, ubicació, etc.)
    - ■ Executa operacions d'espai de noms del sistema d'arxius com obrir, tancar, reanomenar arxius i directoris.
  - ○ ***Resource Manager***
    - ■ Arbitra recursos entre nodes competidors
    - ■ Realitza un seguiment dels nodes vius i morts

- ➢ **Esclau :** és una màquina de configuració normal on hi ha 2 demons executant-se en màquines Slave que són:

- **DateNode**
  - Emmagatzema les dades
  - Realitza operacions de lectura, escriptura i processament de dades
  - Després de la instrucció d'un mestre, crea, elimina i replica blocs de dades.

- **NodeManager**
  - Executa serveis en el node per a verificar el seu estat i informa el mateix a Resource Manager.
  - Podem escalar fàcilment el clúster d' Hadoop agregant-li més nodes. Per tant, ho diem un clúster amb escala lineal. Cada node agregat augmenta el rendiment del clúster.

2.1. Tamany del clúster

- **Cluster pseudistribuit (1 node)** s'implementa tot en una sola màquina i els clústers de múltiples nodes s'implementen en diverses màquines.

  - Tots els "daemons" com NameNode, DataNode s'executen en la mateixa màquina. En un clúster de Hadoop d'un sol node.

  - Tots els processos s'executen en una instància de JVM. L'usuari no necessita realitzar cap ajust de configuració. L'usuari d' Hadoop només necessita establir la variable JAVA_HOME. El factor predeterminat per al clúster Hadoop de node únic és un.

- **Clúster Hadoop de múltiples nodes**, té una arquitectura *"master-slave"*

  Màquina Master s'executa :
  - "daemon" NameNode
  - "daemon" ResourceManager
  - La màquina ha de ser un servidor potent

  Màquina Slave s'executa :
  - "daemon"DataNode
  - "daemon" NodeManager
  - Les màquines poden ser simples i barates
  - En múltiples nodes, les màquines "slaves" poden estar presents en qualsevol ubicació, independentment d'ubicació física del servidor mestre

Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

3. Clúster Hadoop (CentOS 6)

En aquest apartat mostrarem la part pràctica amb exemples. S'ha utilitzat VirtualBox on s'ha virtualitzat diverses màquines amb S.O CentOs 6

S'ha pensat que era més didàctic i entenedor fer-ho per fases.

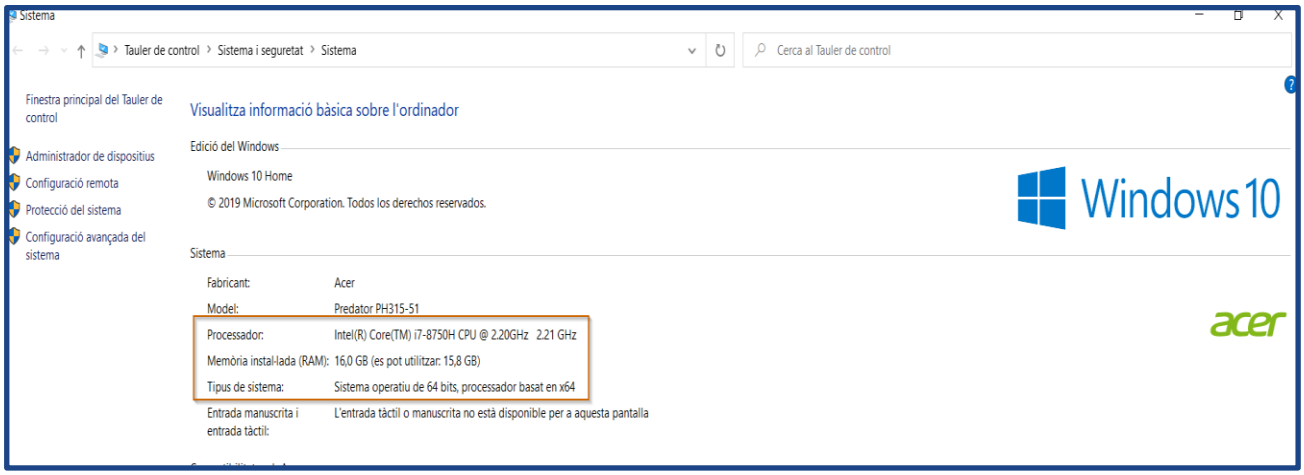
- ❏ Clúster pseudodistribuït (1 node)
- ❏ Clúster real (1 master i 2 slaves) : Exemples pràctics
- ❏ Clúster real (1 master i 2 slaves) amb l'ampliació ecosistema Hadoop (Hive, Spark i Ambari) . Exemples pràctics
- ❏ Clúster Hadoop (1 master i 2 slaves) utilitzant Ambari (visualització des del navegador de Windows [utilitza la mateixa xarxa local])

3.1. Clúster Hadoop pseudodistribuït [4]

3.1.1. Requisits de hardware

Abans de començar la instal·lació hem de saber quanta memòria RAM i quants nuclis té el host amfitrió (Windows 10) abans de crear la màquina virtual CentOS.

- PC amfitrió (Windows 10) disposa de 16 GB de memòria RAM i 6 nuclis. Farem servir *4 GB* [5] de memòria RAM i 2 nuclis. Revisem que la unitat D: on estaran ubicades les màquines virtuals, disposa de suficient espai



3.1.2. Versions del software(per tema de compabilitats)

[4] Aquest clúster Hadoop se li diu pseudistribuit ja només hi haurà 1 node on una part farà de master i l'altre de slave

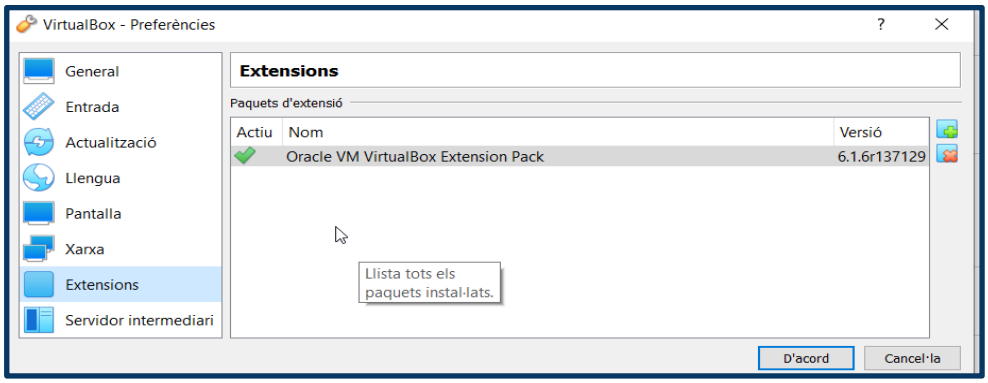
[5] *En aquest cas podríem fer servir més memòria RAM, però com ja es veurà posteriorment aquesta màquina es clonarà dos cops i ens quedarà 4 GB pel host amfitrió).*

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

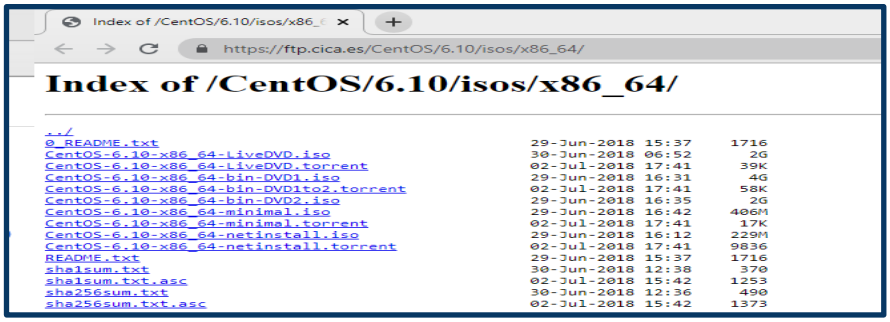
A continuació mostraré les versions dels programes que s'han instal·lat (incloent-hi VirtualBox, ja que pot ser que en versions diferents s'hagi de fer modificacions)

- ❑ versió del VirtualBox : **6.1**

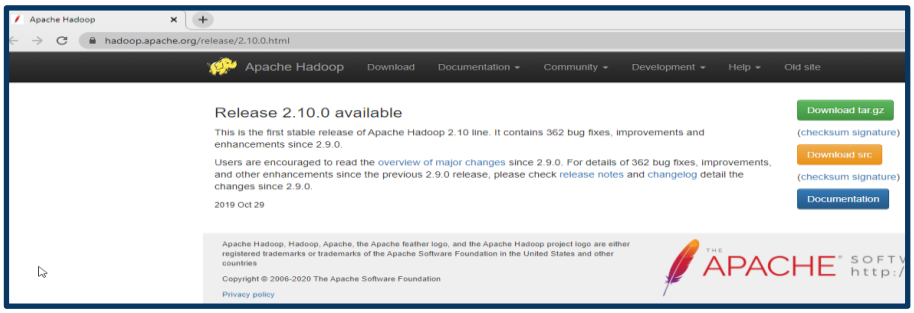
- ❑ versió de les Guestt Additions : **6.1.6r1317129**



- ❑ Versió del sistema operatiu: **CentOS 6.10 de 64 bits**
  https://ftp.cica.es/CentOS/6.10/isos/x86_64/

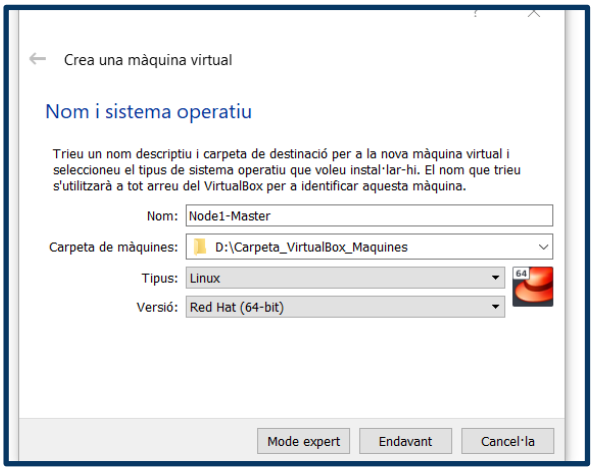


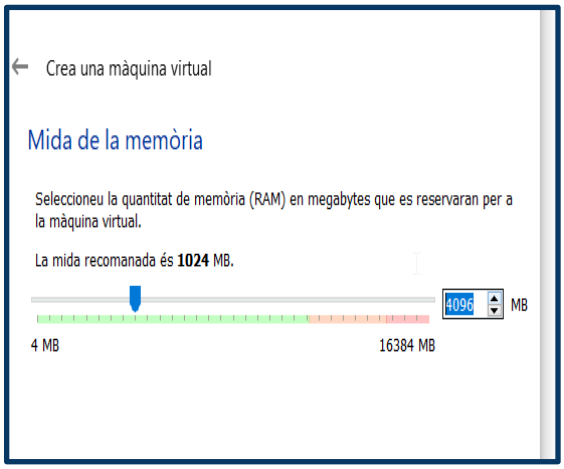
- ❑ Versió de Hadoop : **2.10.0**
  https://hadoop.apache.org/release/2.10.0.html

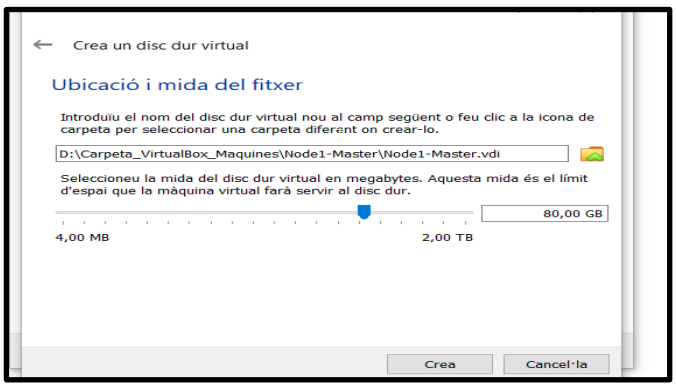


3.1.3. Instal·lació de CentOS 6.10

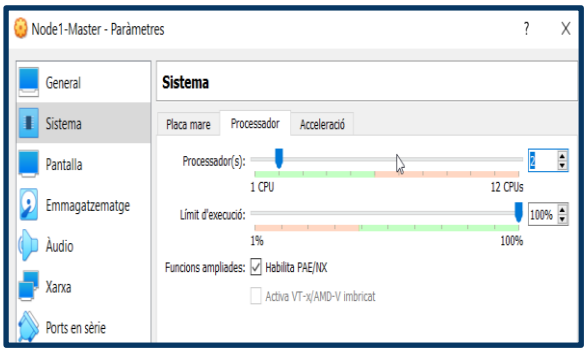
A continuació mostrarem les captures de com hem configurat la màquina i la instal·lació del seu sistema operatiu , afegint posteriorment les *"Guest Additions"* per poder tenir una millor resolució de pantalla.

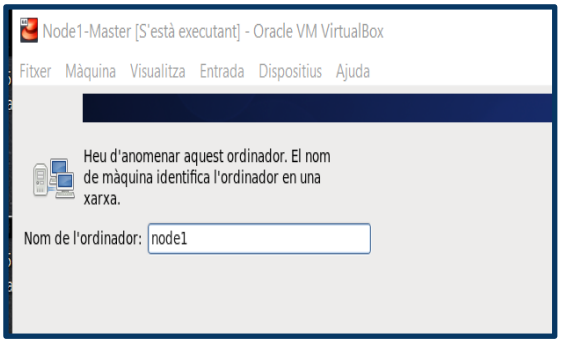


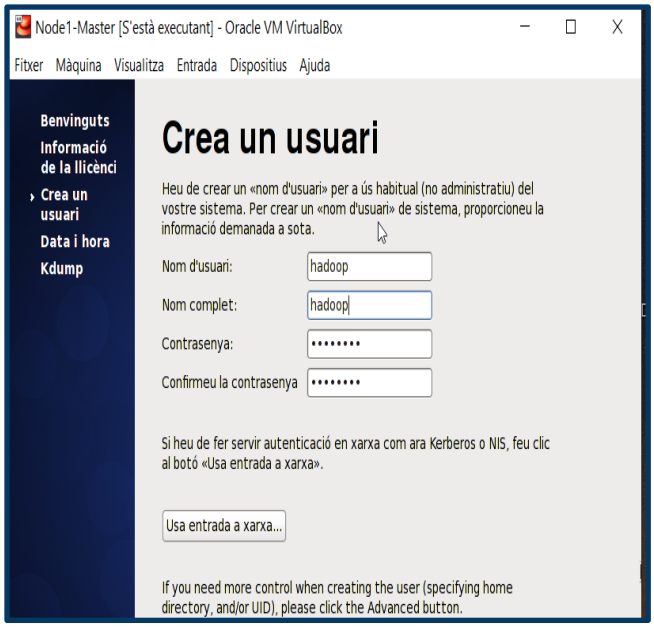


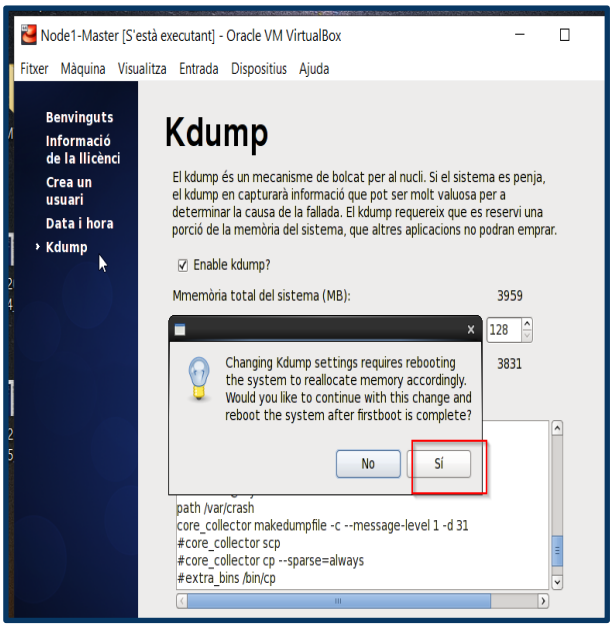


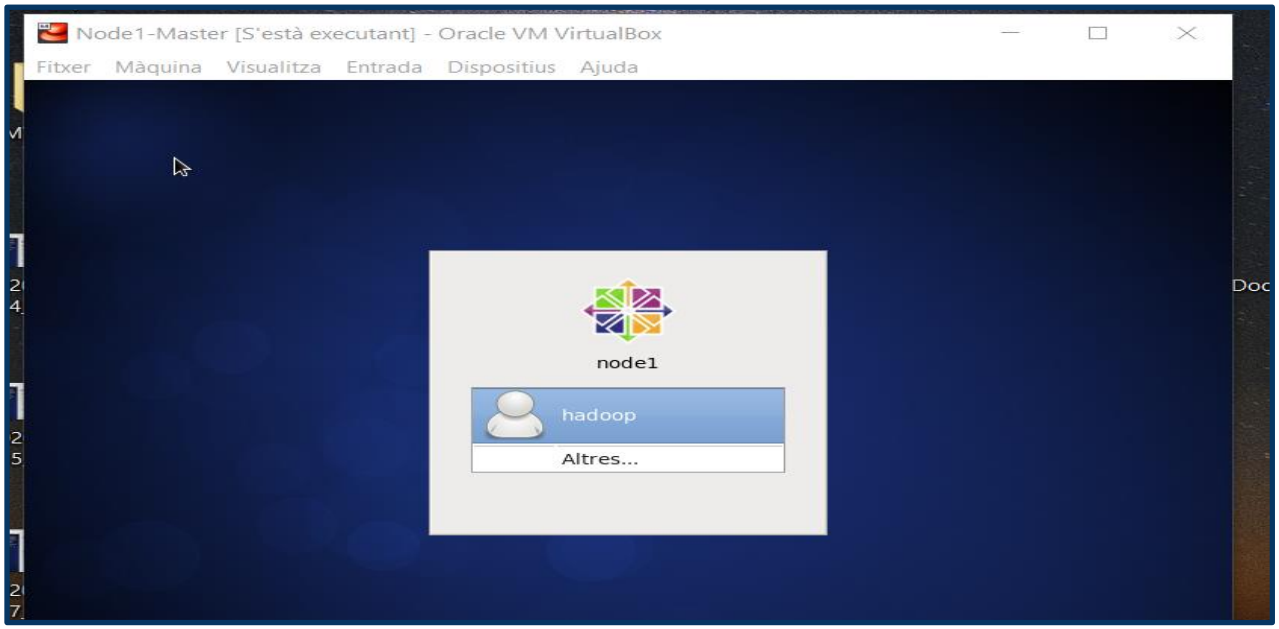
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





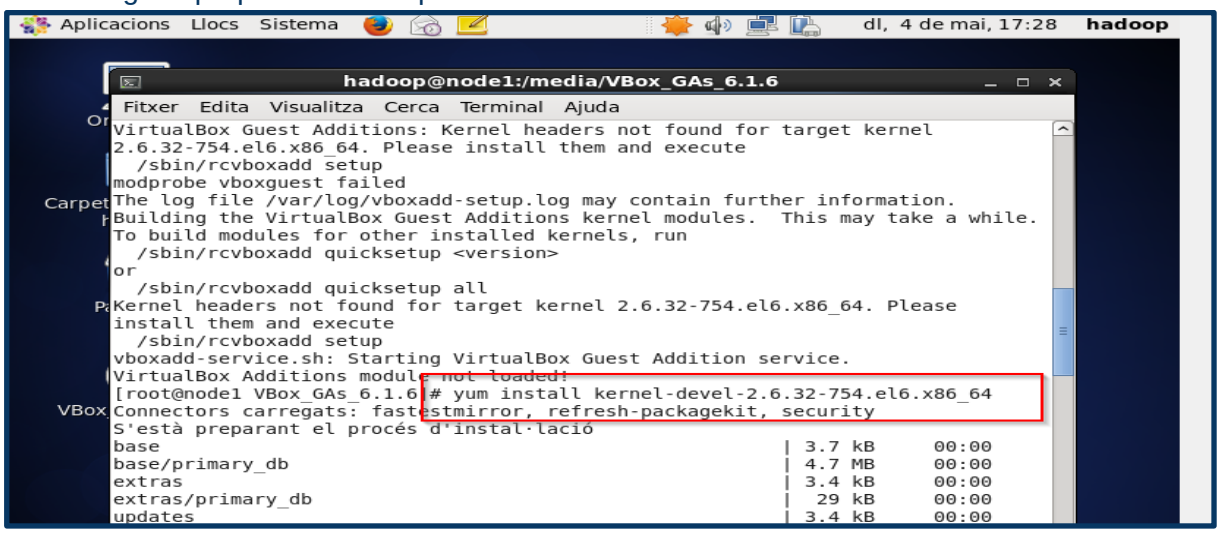


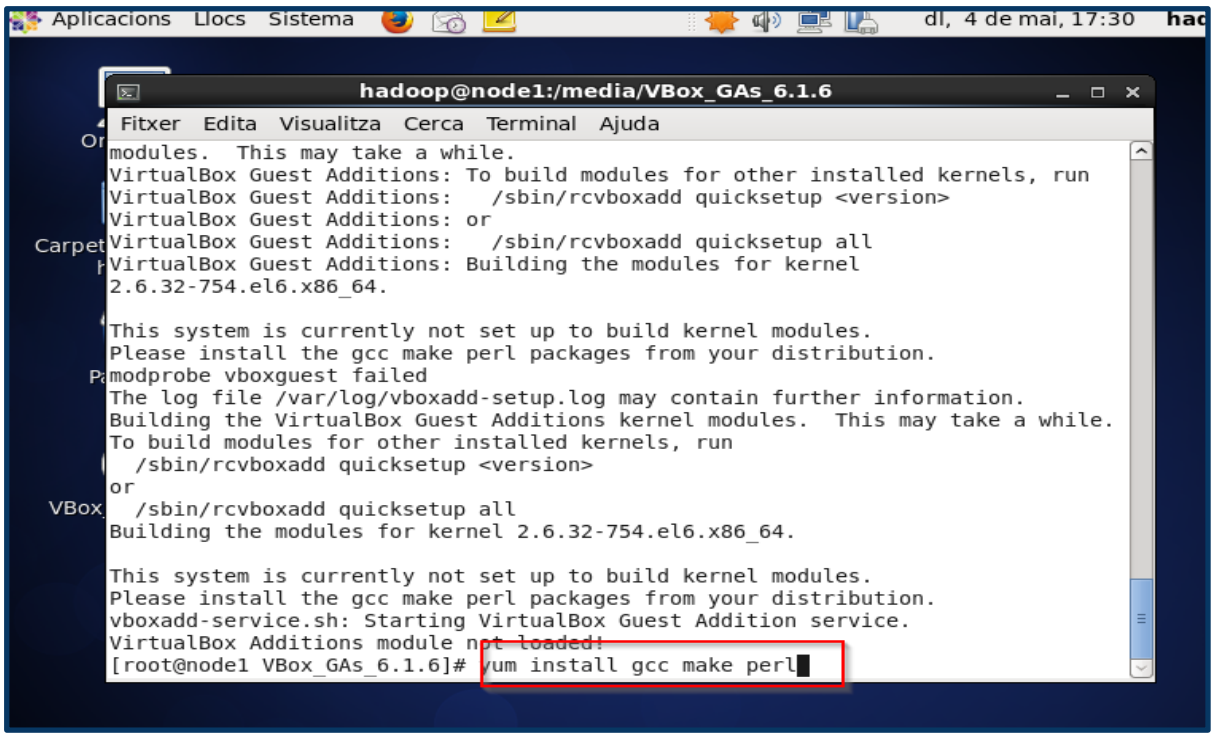




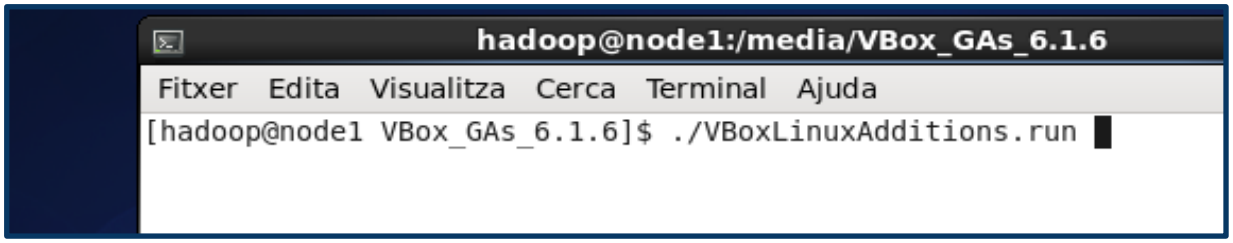
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Afegim les Guest Additions (abans haurem d'actualitzar el repositori *yum* i instal·lar alguns paquets com es pot veure a continuació.





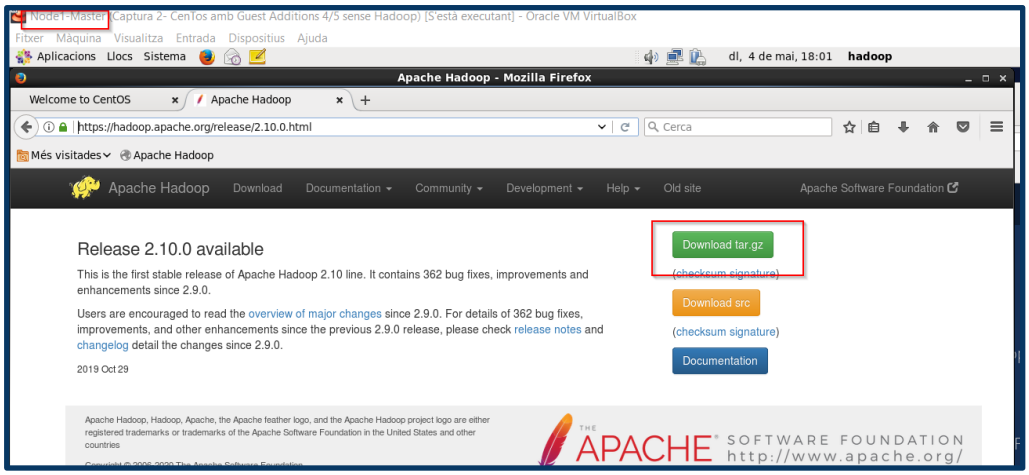
Executem el script per instal.lar les Guest Additions i posteriorment reiniciem la màquina virtual



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

3.1.4. Instal·lació i configuració Hadoop 2.10.0

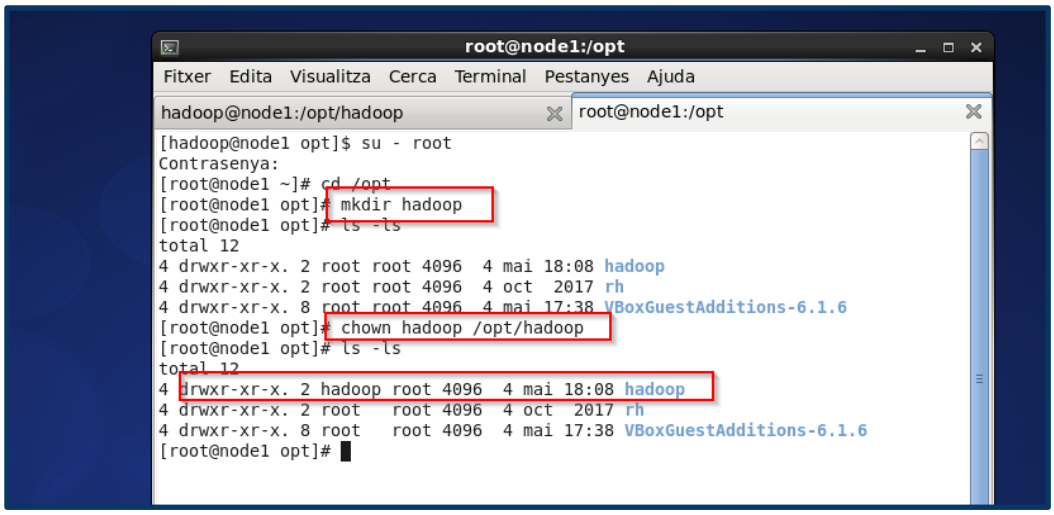
- Desde la màquina virtual tenint connexió a Internet accedim al navegador Firefox per instal·lar Hadoop 2.10.0



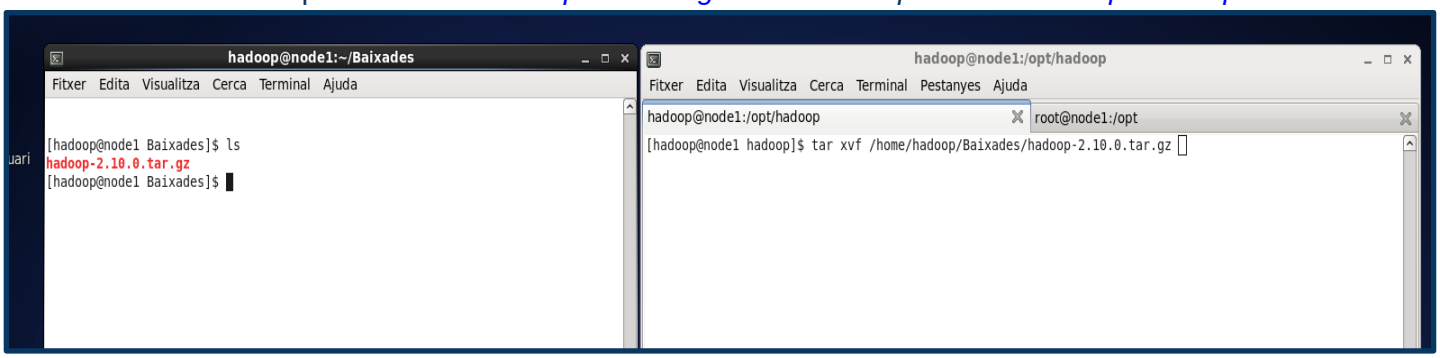
- Crearem un directori anomenat "**hadoop"** dins del directori **/opt**[6] i donant-li permisos a l'usuari hadoop per poder-hi accedir.

[6] Podríem instal·lar-ho en un altre directori, però per tema de bones pràctiques s'ha instal·lat a /Opt, ja que és a on instal·lem les aplicacions opcionals manualment.

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

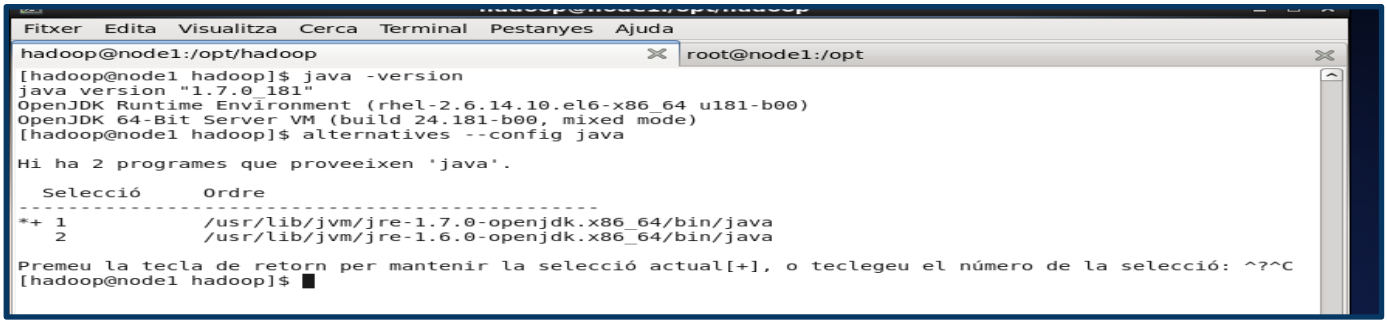


- Descomprimim l'arxiu *hadoop-2.10.tar.gz al directori que hem creat /opt/hadoop*



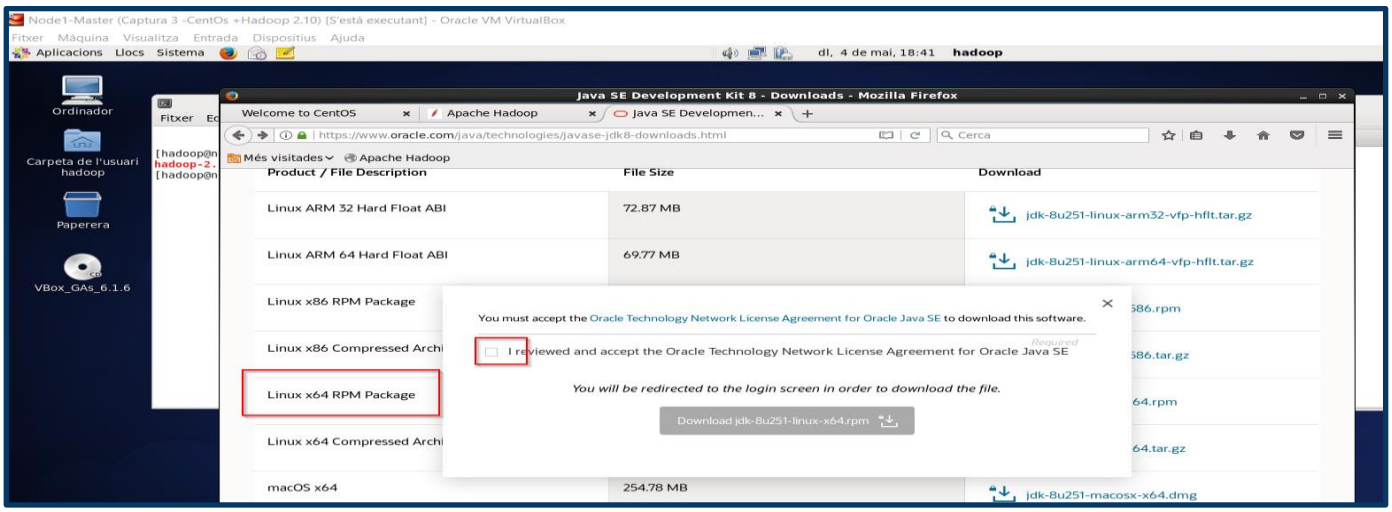
- Un cop fet això hem de revisar quina versió de JAVA tenim (i si és JRE o JDK) utilitzant la comanda *"java -version"*

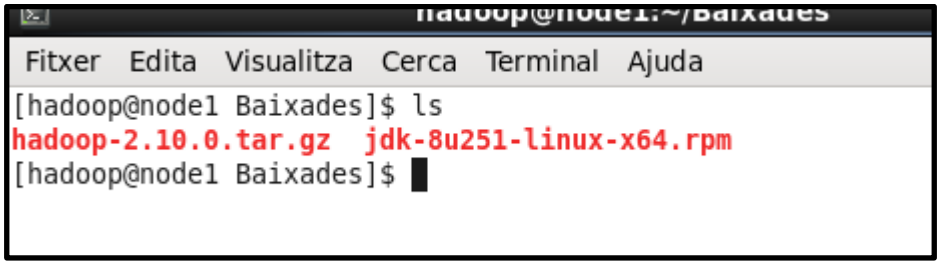
Important *: Utilitzant Hadoop 2.x necessitem mínim JDK 7, però utilitzarem JDK 8(que és el mínim si utilitzéssim Hadoop 3.x)*



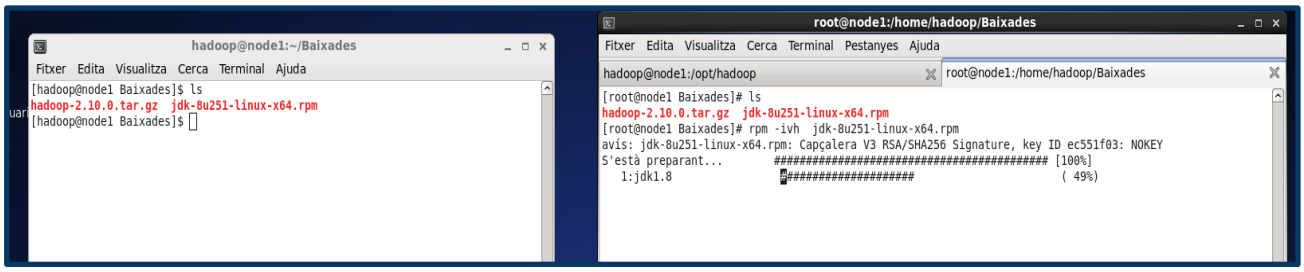
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Veient que no tenim el JDK, anem a la pàgina oficial d'Oracle i ens descarreguem el JDK 8 en format RPM

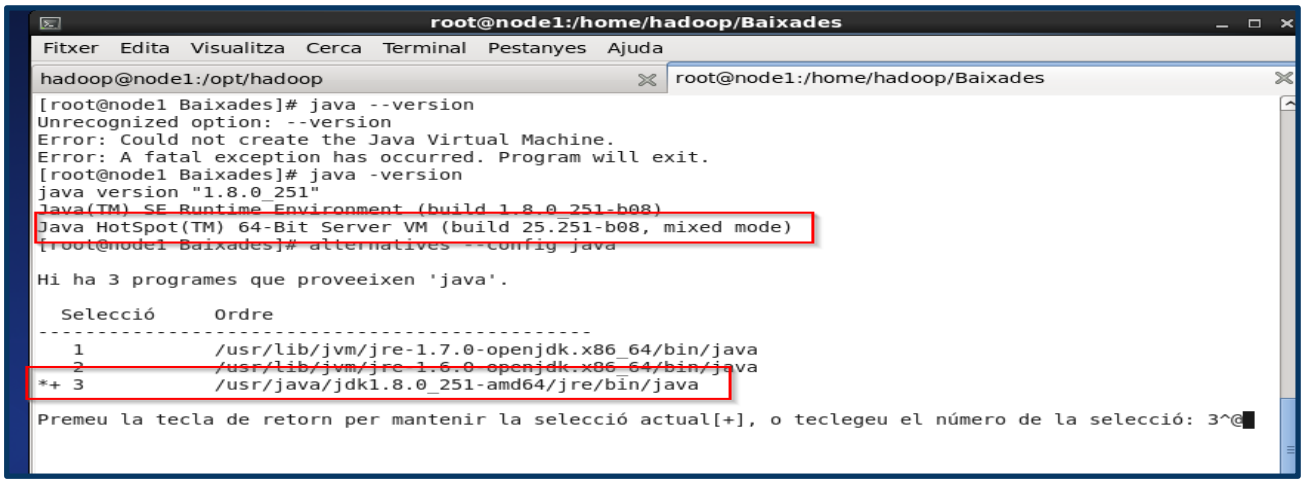




- Instal·lem el paquet RPM amb el comando *" rpm -ivh jdk-8u251-linux-x64.rpm"*

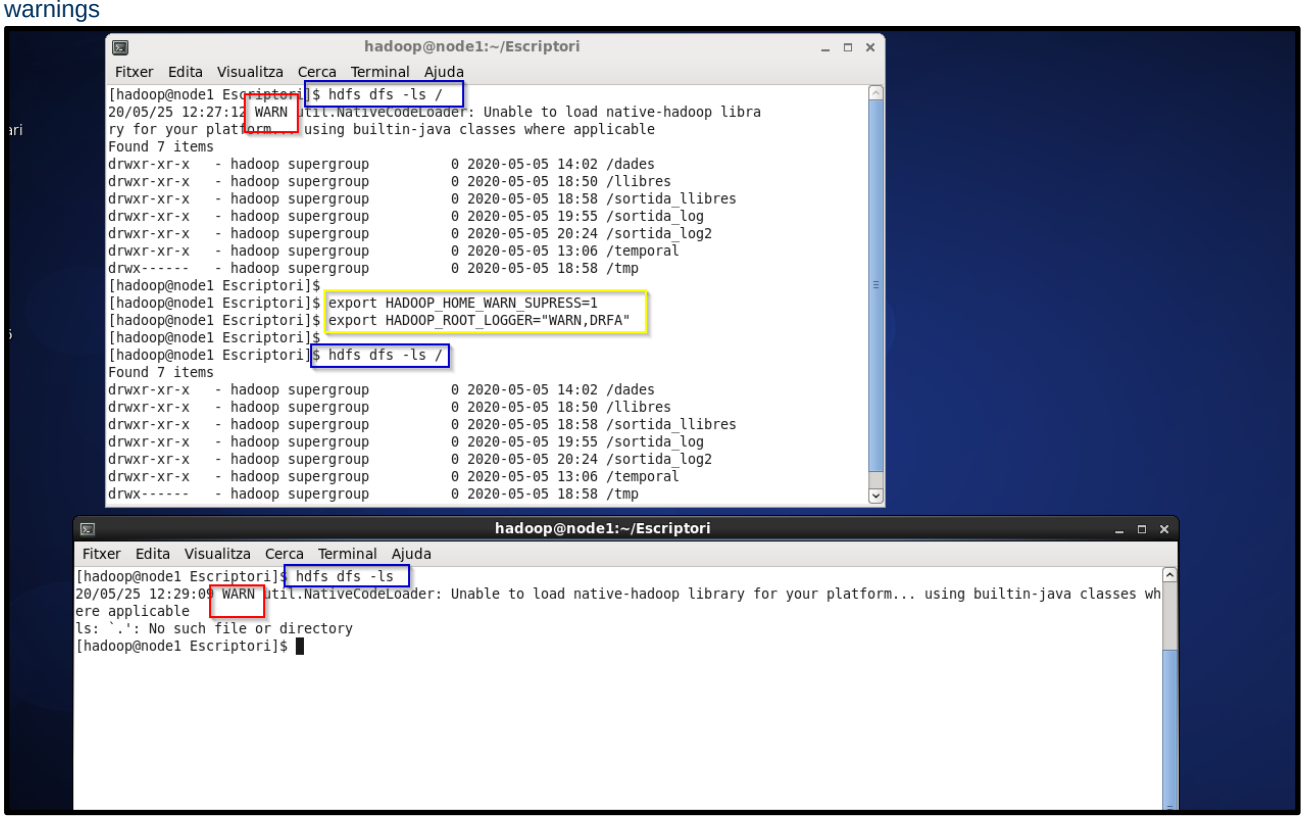


- Tornem a revisar les versions disponibles de JAVA i confirmem que la que ens interessa està en ús.



- Posteriorment degut a que hem instal·lat JDK de 64 bits ens apareixeran warnings al executar comandos HDFS, ja que les llibreries natives de Hadoop per defecte en aquesta versió és de 32 bits.

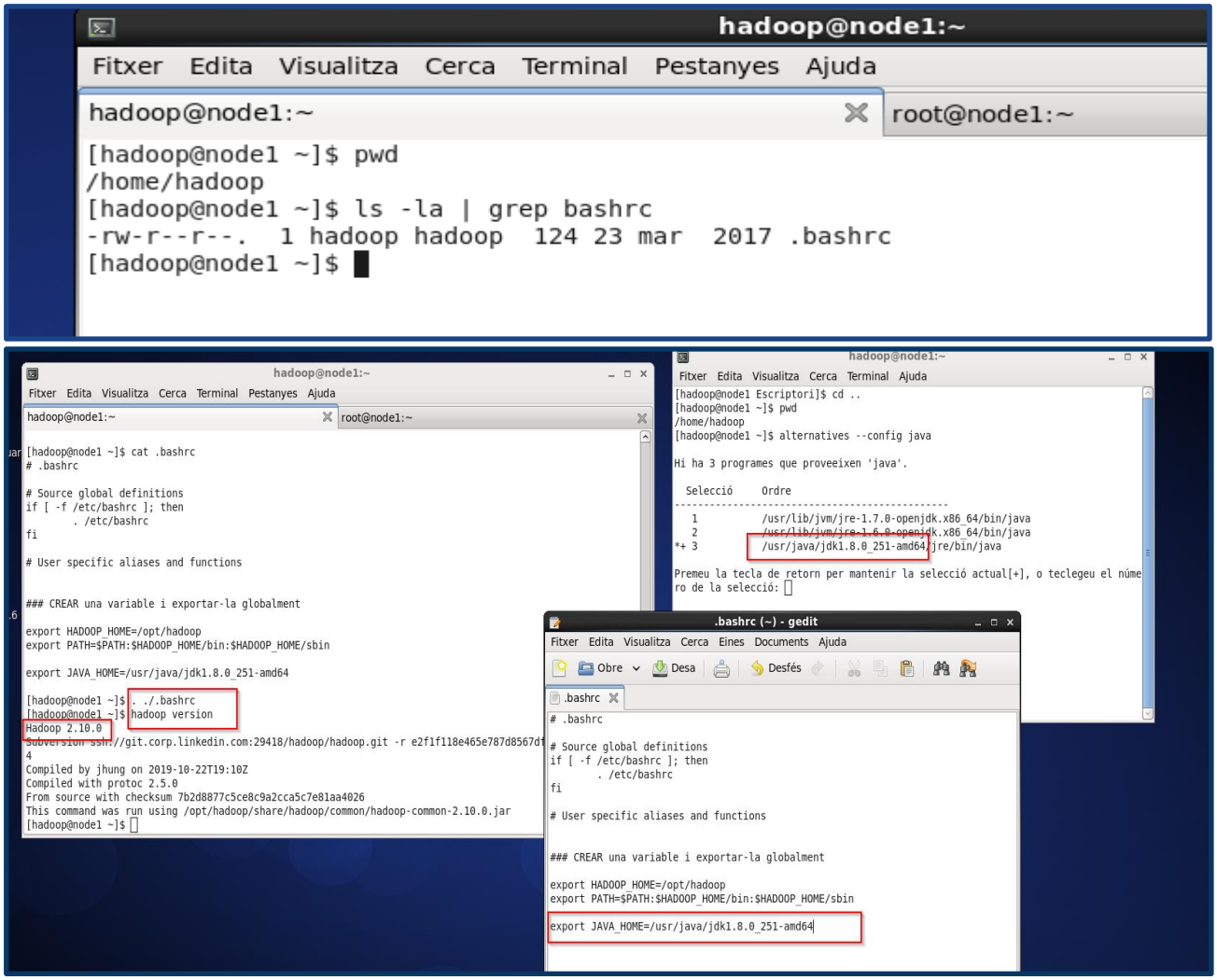
Nota: En el nostre cas hem fet una prova per eliminar-los de forma manual però no ho hem afegir a l'arxiu ~/.bashrc i obrim un nou terminal ens apareixerà novament. Per aquest motiu a continuació es podran veure captures amb o sense warnings



- Un cop instal·lat Hadoop hem de tenir en compte els directoris més importants.
  - **/bin** : Estan ubicats els binaris per llençar els processos a Hadoop
  - **/etc** **:** És el directori de configuració (que posteriorment haurem de modificar)

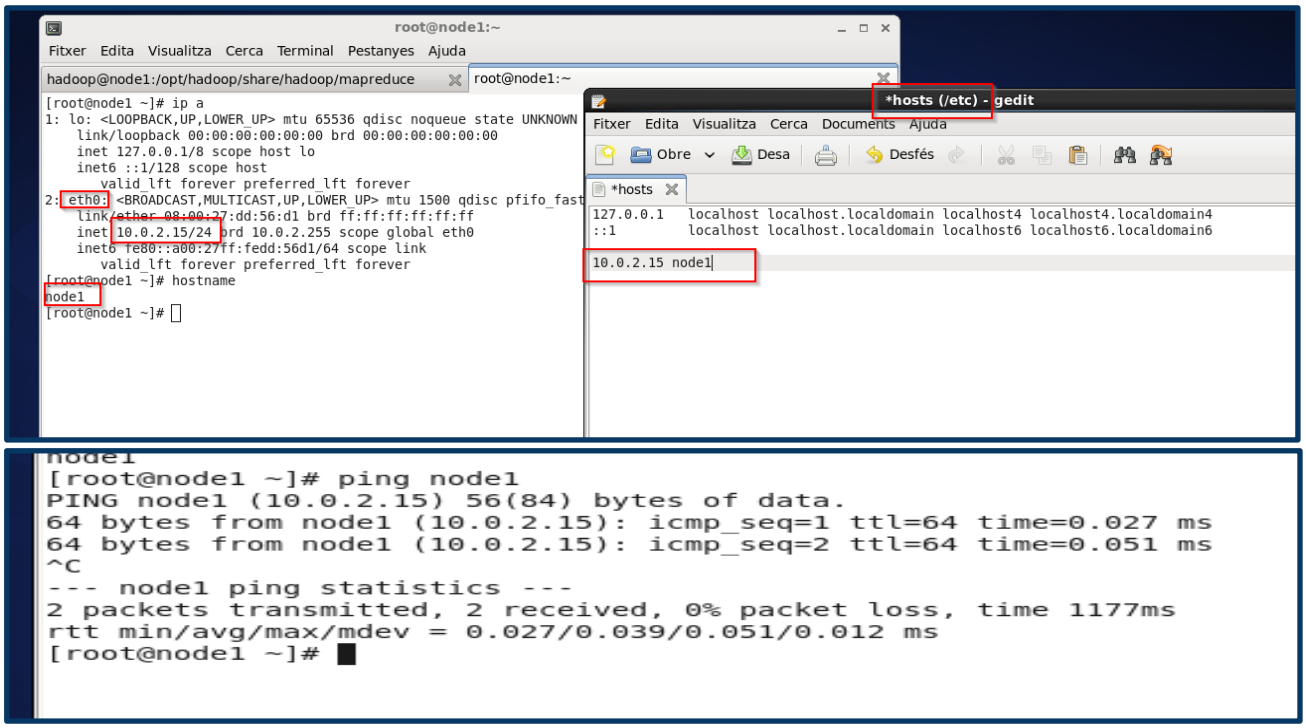
- **/lib** : Conté les llibreries natives
- **/sbin** : Scrips que em permeten parar,arrencar hdfs…
- **/share** : On hi ha la paqueteria de Hadoop, exemples…

❑ El següent punt important és configurar *les variables d'entorn de Hadoop i Java* per poder treballar de manera directa amb els executables. Per això haurem de modificar l'arxiu ocult anomenat ~/.bashrc

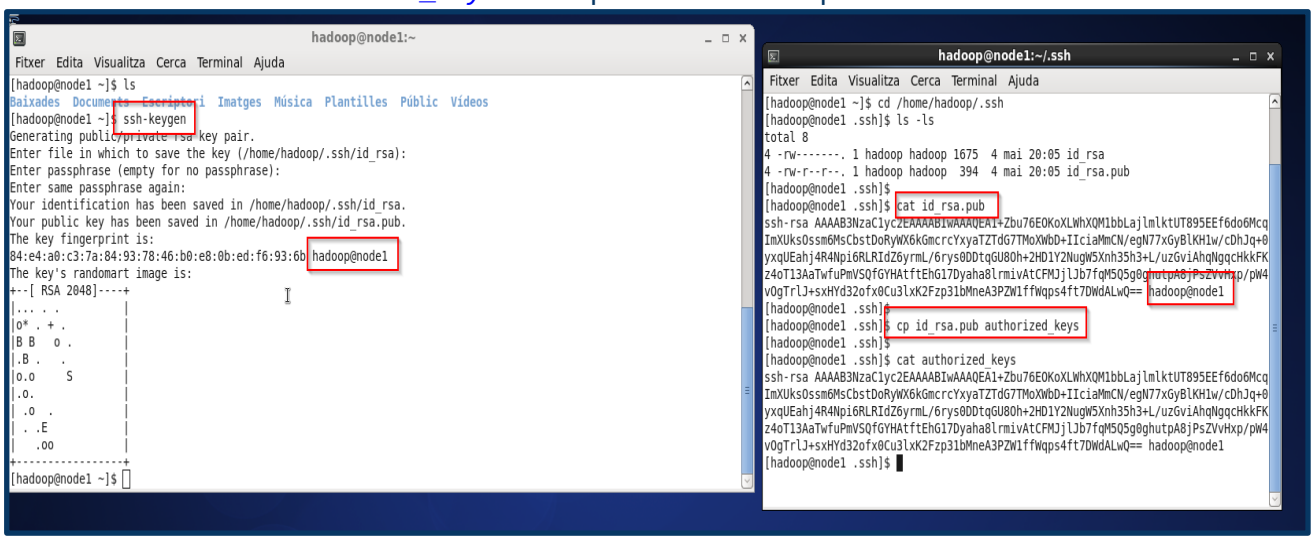


❑ Respecte a la xarxa, en aquest cas d'un sol node, farem servir el mode NAT, però així i tot haurem d'accedir a l'arxiu */etc/hosts* , ja que no tenim servidor DNS i quan utilitzem SSH ho voldrem amb el nom i no la IP

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

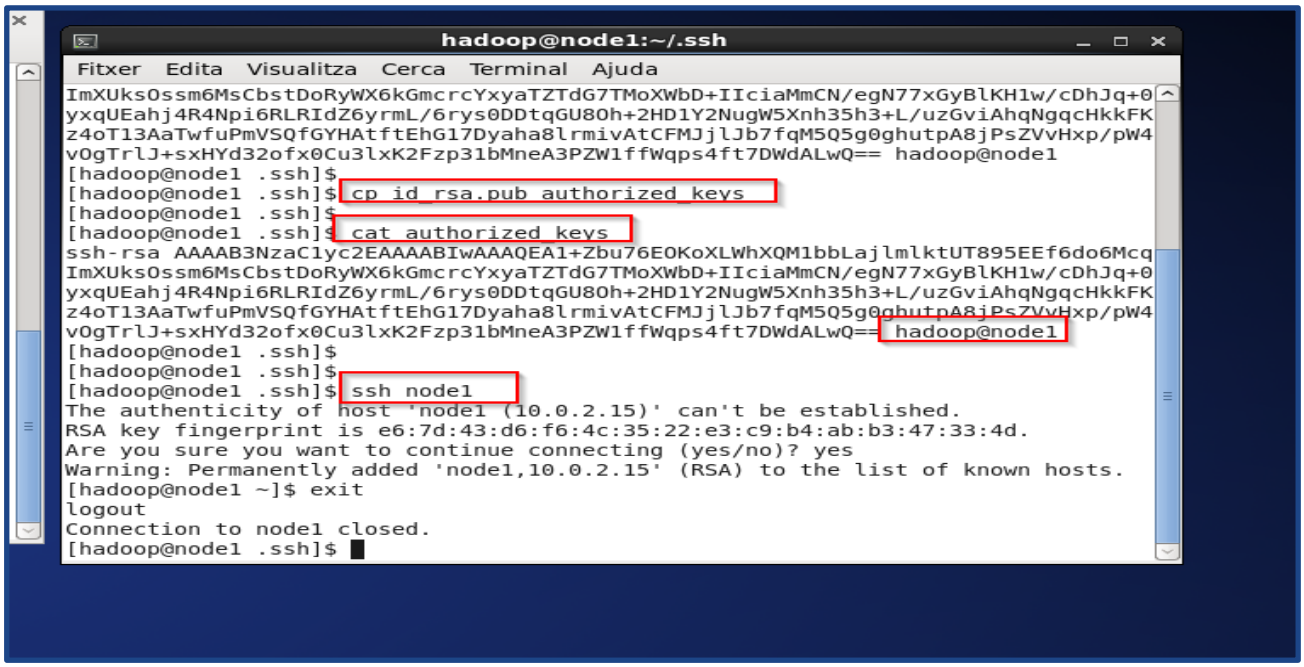


- Un cop modificat l'arxiu /etc/hosts hem de configurar el SSH, ens interessa accedir-hi sense password, per aquest motiu haurem de generar un parell de claus (pública i privada) i crear l'arxiu "*authorized_keys*[7]" on dipositarem la clau publica.



- Fem una prova per accedir per SSH sense contrasenya (des de la mateixa maquina).

[7] Aquest punt és molt important en la creació d'un clúster en diversos nodes (ja es mostrarà més endavant)

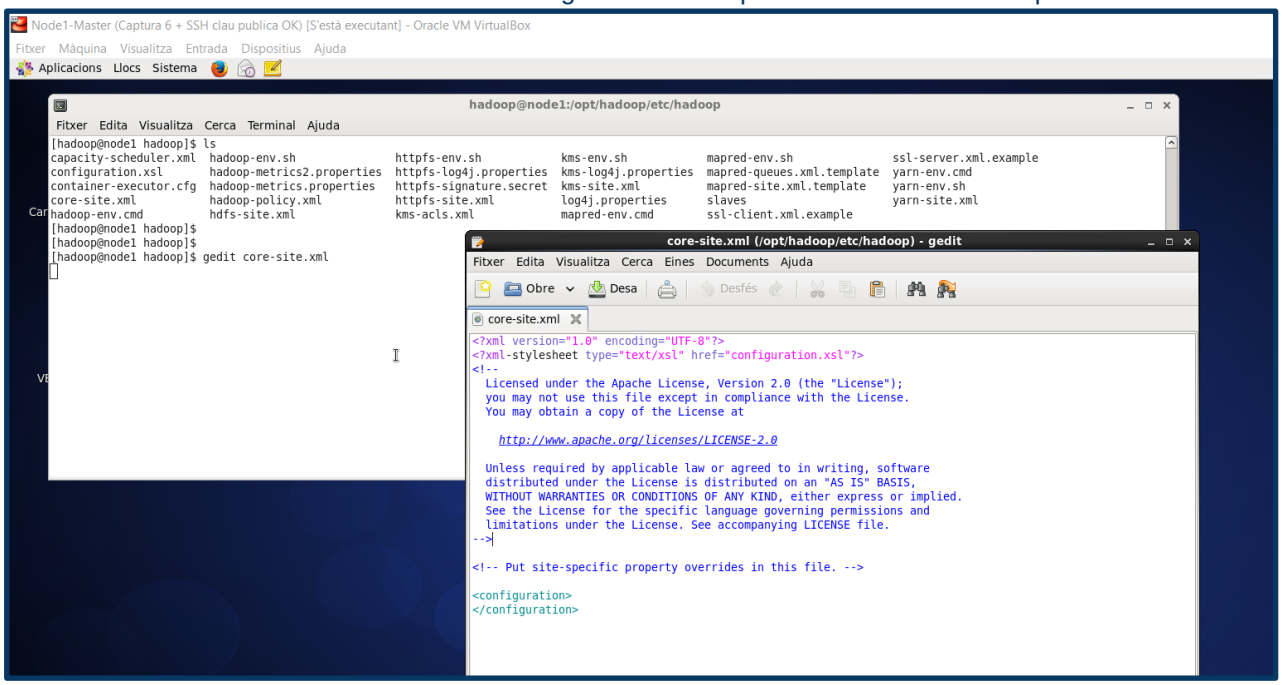


3.1.4.1. Configuració de les dades

Com ja hem comentat abans els 2 components bàsics d'un clúster Hadoop són les "dades" i els "processos"

- ❏ Primer farem la configuració de les dades amb algun exemple pràctic (creació d'un directori HDFS)
- ❏ Seguidament haurem de fer la configuració dels processos amb algun exemple amb MapReduce
- ❏ Per la configuració de les dades i els processos ens interessa modificar 4 arxius XML ubicats a **/opt/hadoop/etc/hadoop**
    - ➢ **core-site.xml** : Arxiu de configuració general del Clúster
    - ➢ **hdfs-site.xml** : Arxiu de configuració per les dades HDFS
    - ➢ **mapred-site.xml.template** : Arxiu de configuració per MapReduce
    - ➢ **yarn-site.xml** : Arxiu de configuració el mode de processos Yarn

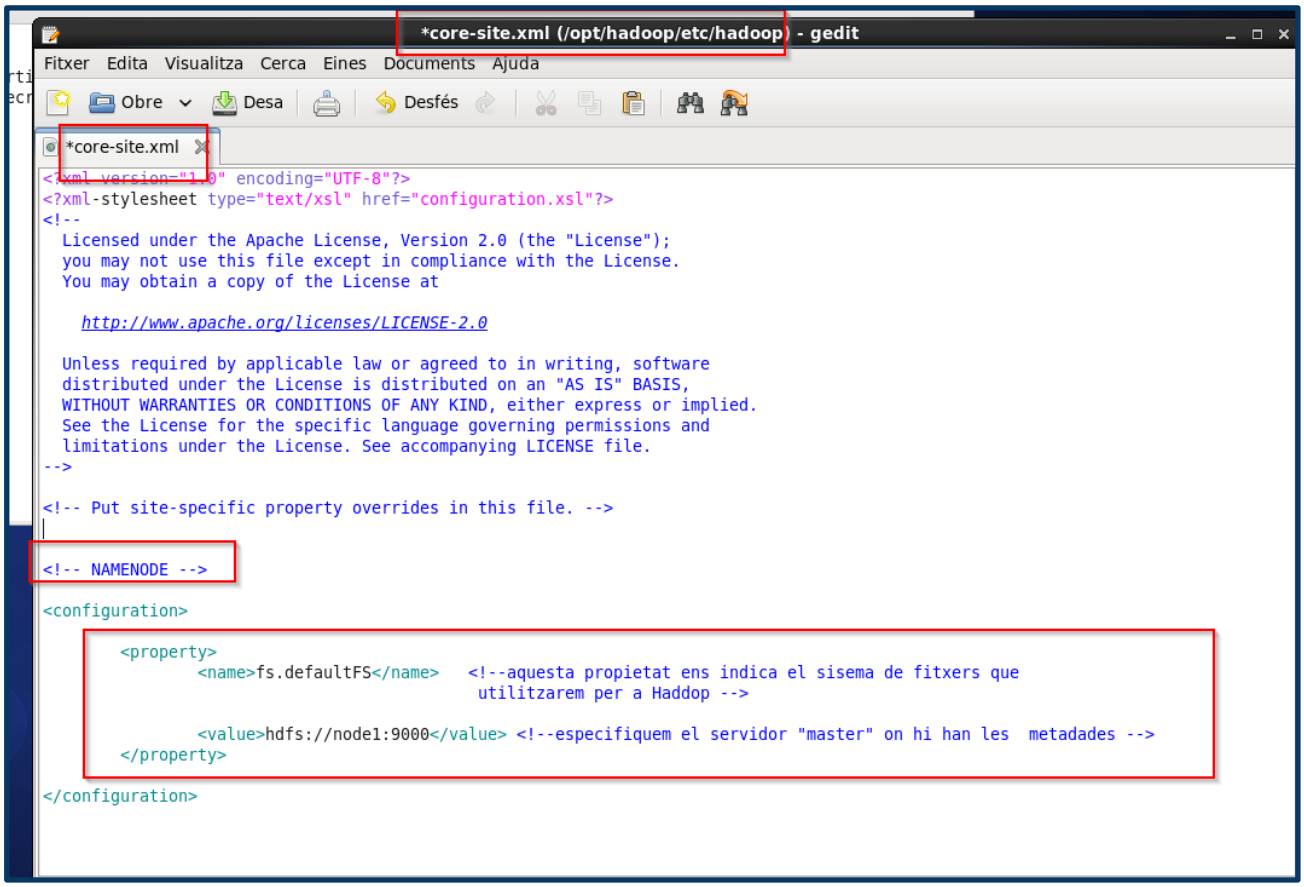
- Arxiu **core-site.xml**[8] **:** Li direm la configuració bàsica pel nostre Clúster Hadoop



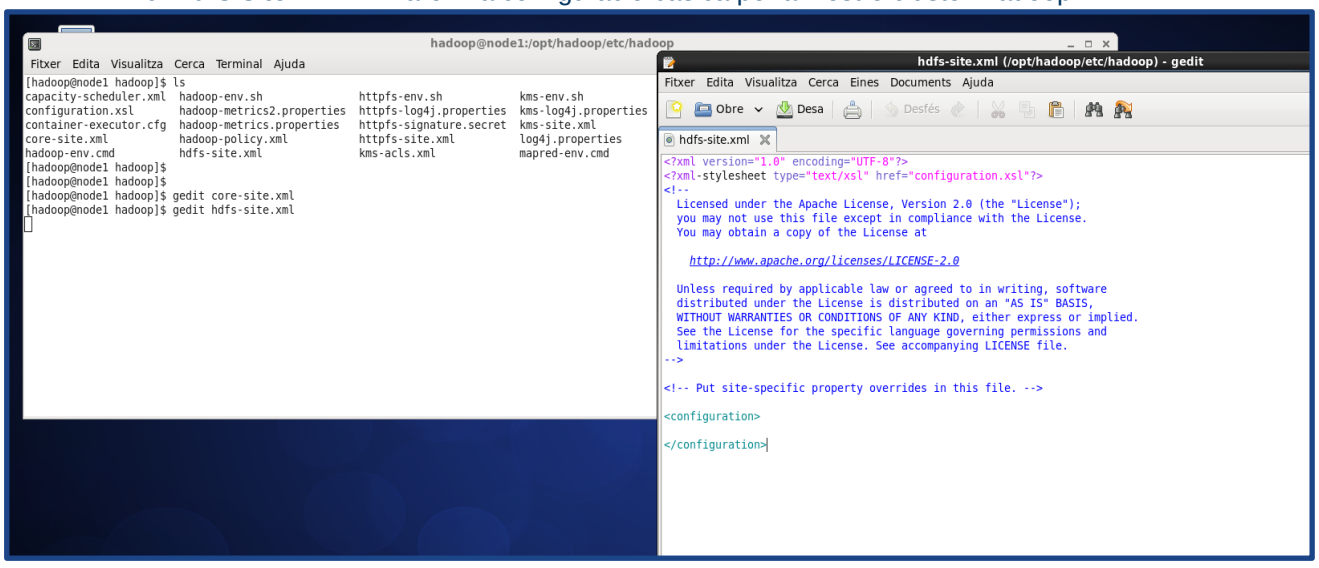
farem una configuració bàsica

[8] Per defecte aquest arxiu està buit

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

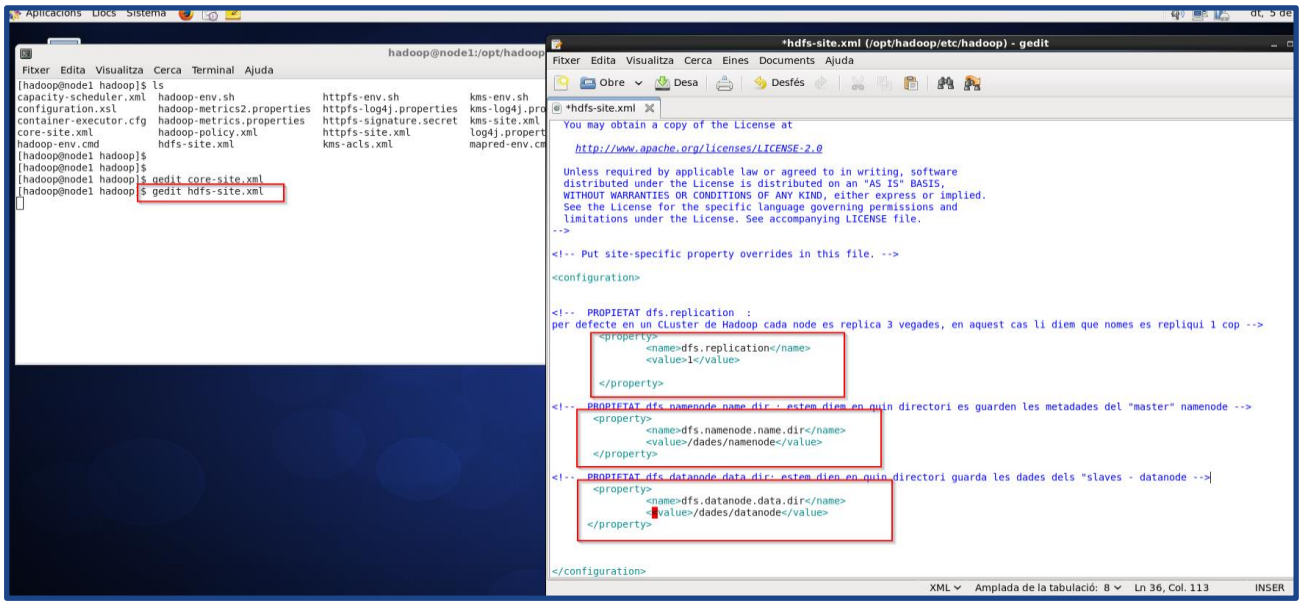


- Arxiu **hdfs-site.xml**[9] **:** Farem la configuració bàsica per al nostre clúster Hadoop



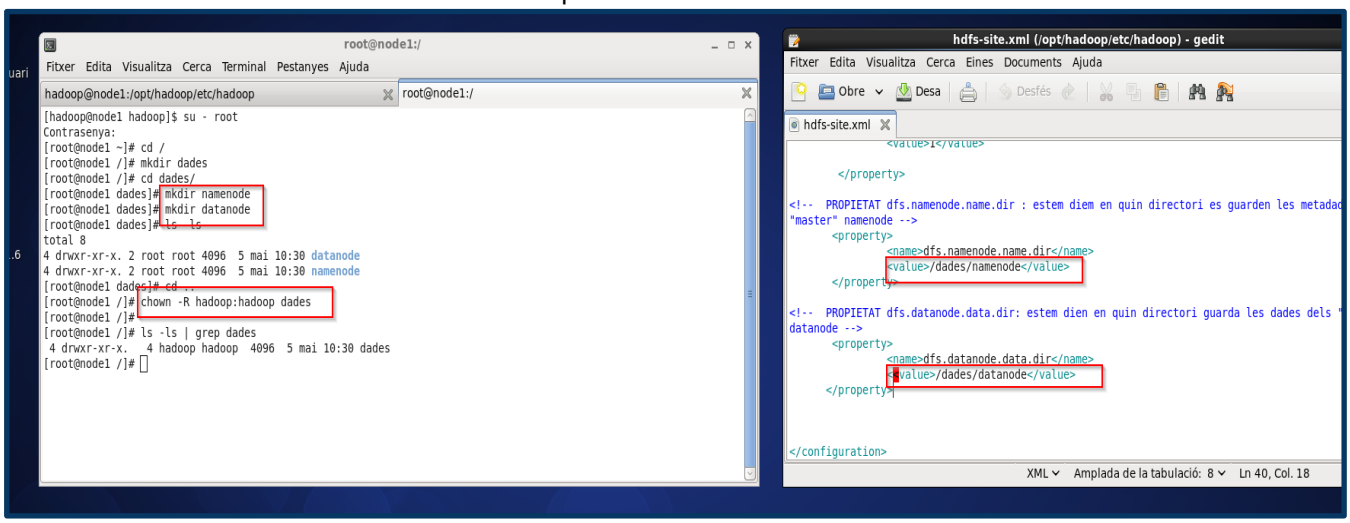
[9] Per defecte aquest arxiu està buit

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



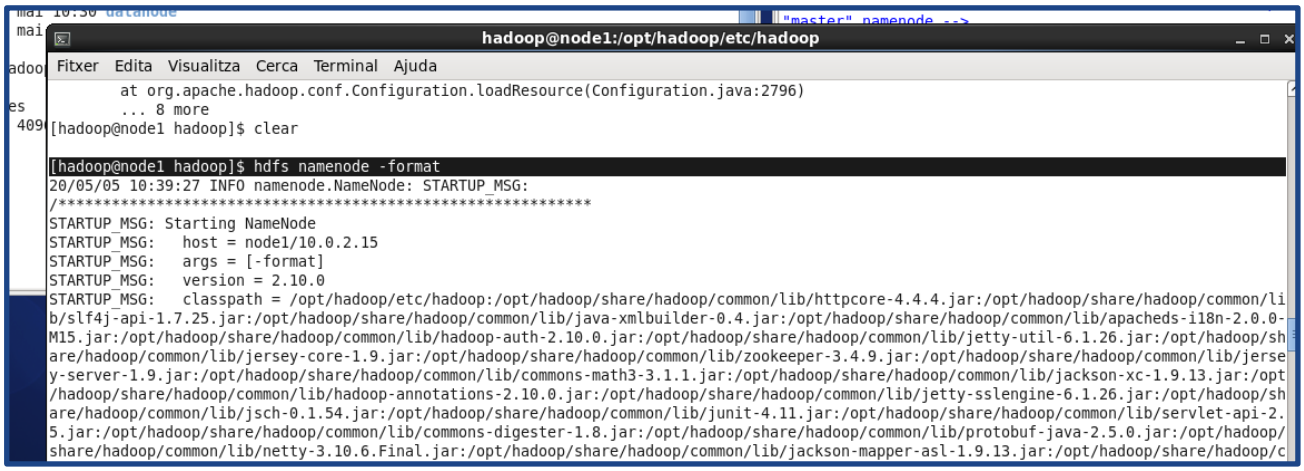
- ❏ Un cop configurats aquests 2 arxius haurem d'arrencar HDFS.

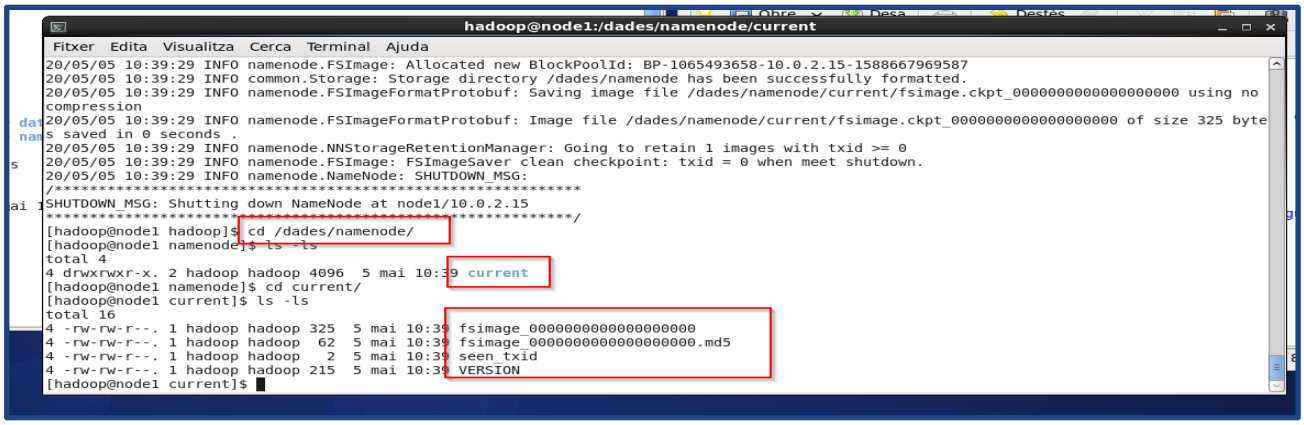
Primer haurem de crear els directoris que hem dit en les propietats del NameNode i DataNode */dades/namenode* i */dades/datanode* i donar-li permisos a l'usuari



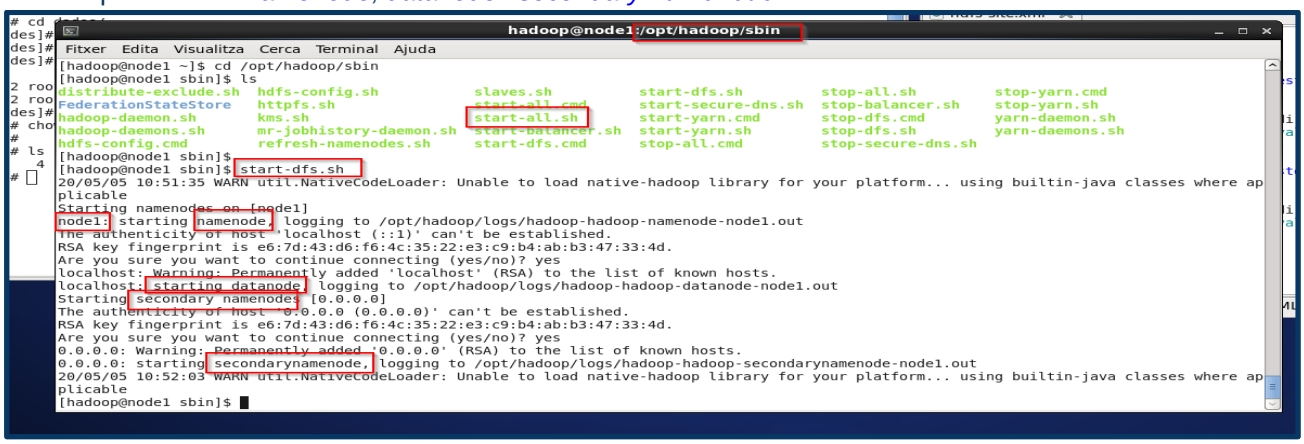
- ❏ Després utilitzant el comando "*hdfs namenode -format*" crearem les metadades al directori /dades/namenode

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





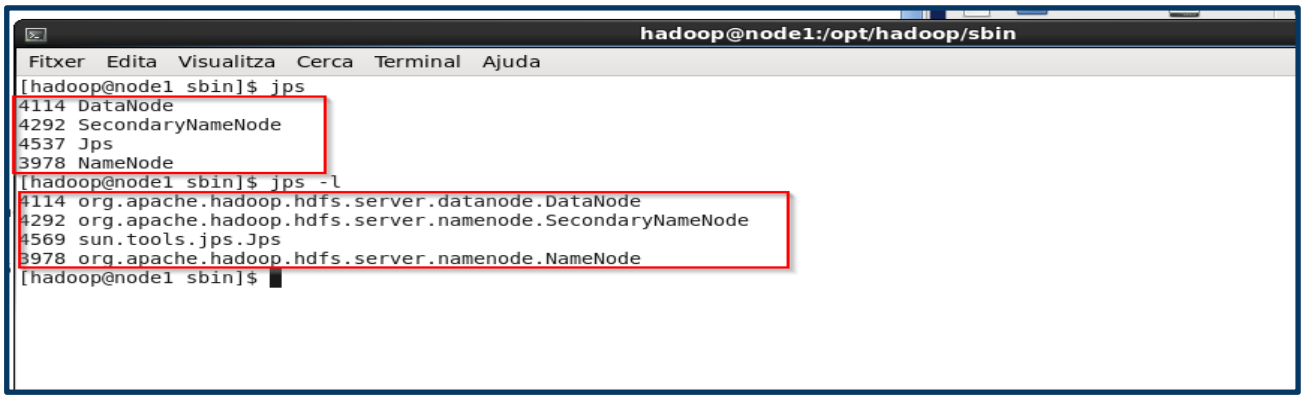
- Després utilitzant el script **start-dfs.sh**[10] ubicat al directori /opt/hadoop/sbin arrencarà el clúster (només dades) segon hem especificat en l'arxiu de configuració. Podrem observar que arrencarà 3 processos: "*namenode*, *datanode* i *secondary namenode*".



- Hi ha un comando de Java (dins del JDK) molt útil anomenat **jps** per saber quins processos s'estan executant a la màquina virtual Java (JVM)

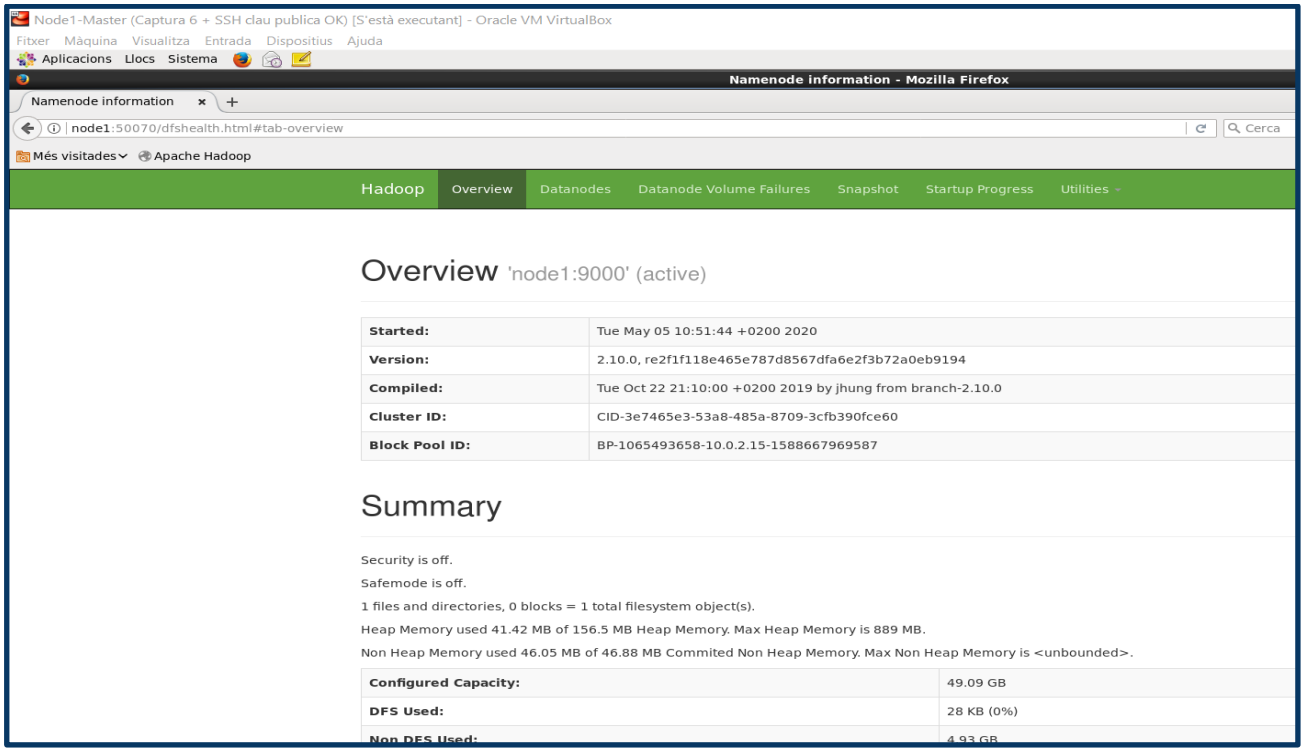
- Utilitzant **jps -l** ens dóna més informació de la classe Java que s'està executant

[10] L'arrencada del clúster es fa desde el "master" i via SSH enviarà comandes als "slaves"



- Obrim un navegador per veure la interfície d'usuari web del Namenode[11] (per defecte utilitza el port 50070 . Com que anteriorment ja havíem associat el nom de la màquina amb la IP estàtica a l'arxiu /etc/hosts accedirem pel nom del host i el port.

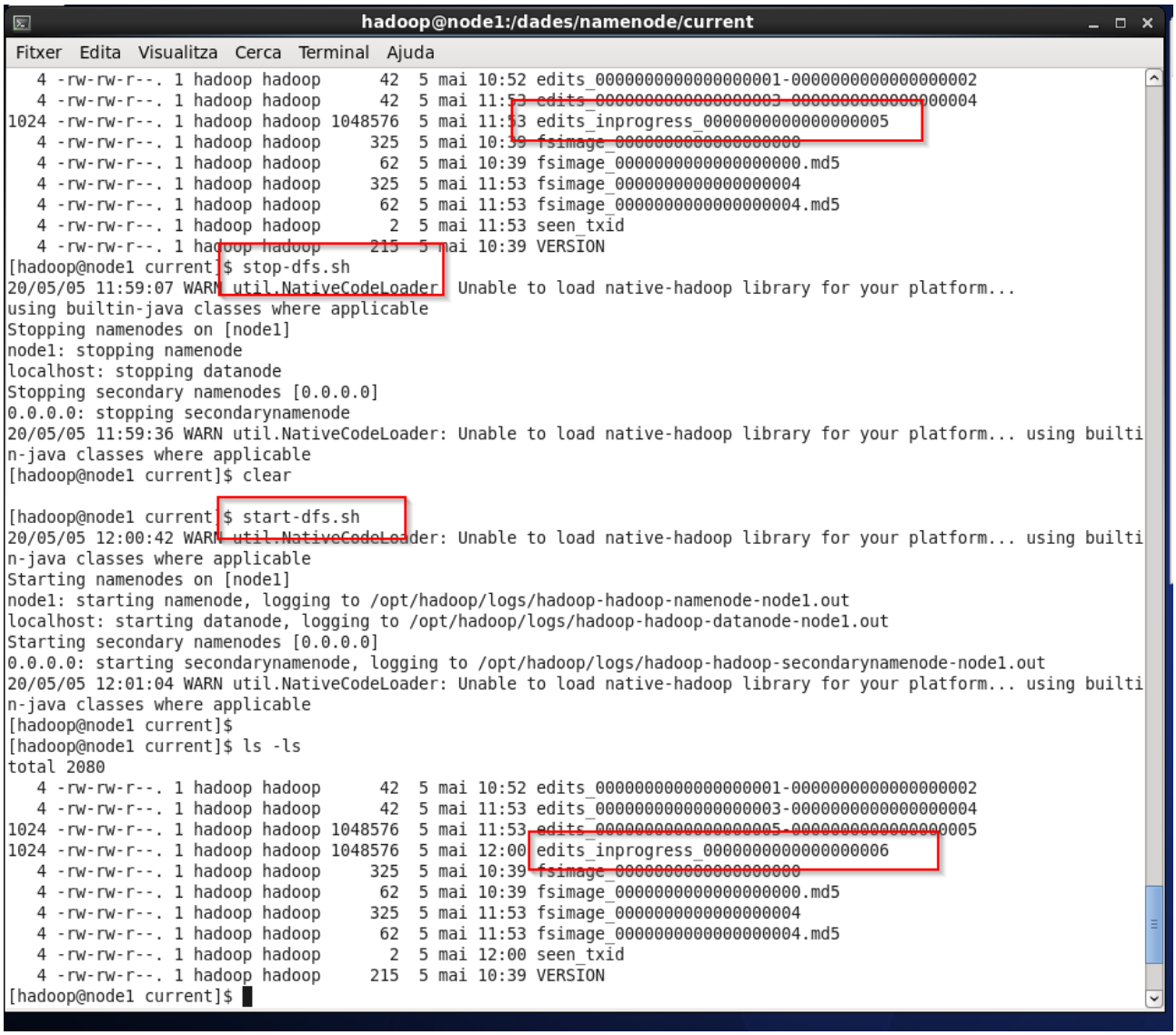
node1:50070



- Parem el clúster utilitzant el comando **stop-dfs.sh**

[11] Si utilitzem una versió de Hadoop 3.x, el port per defecte canvia, seria el 9870

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

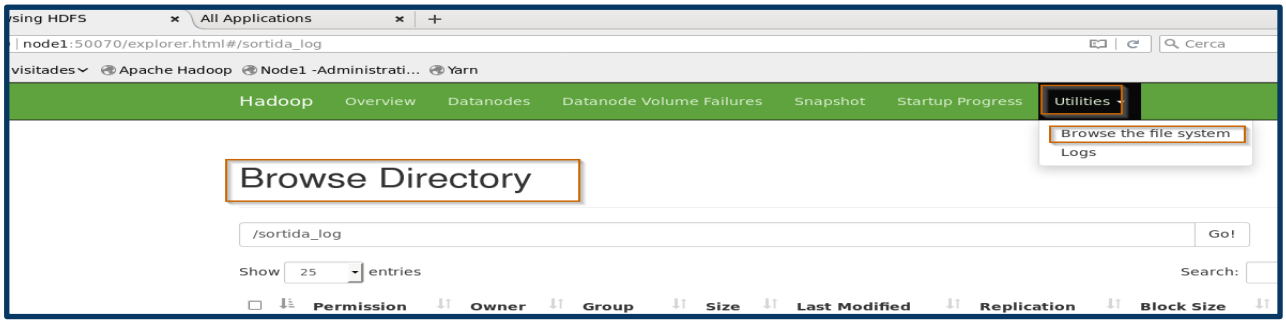


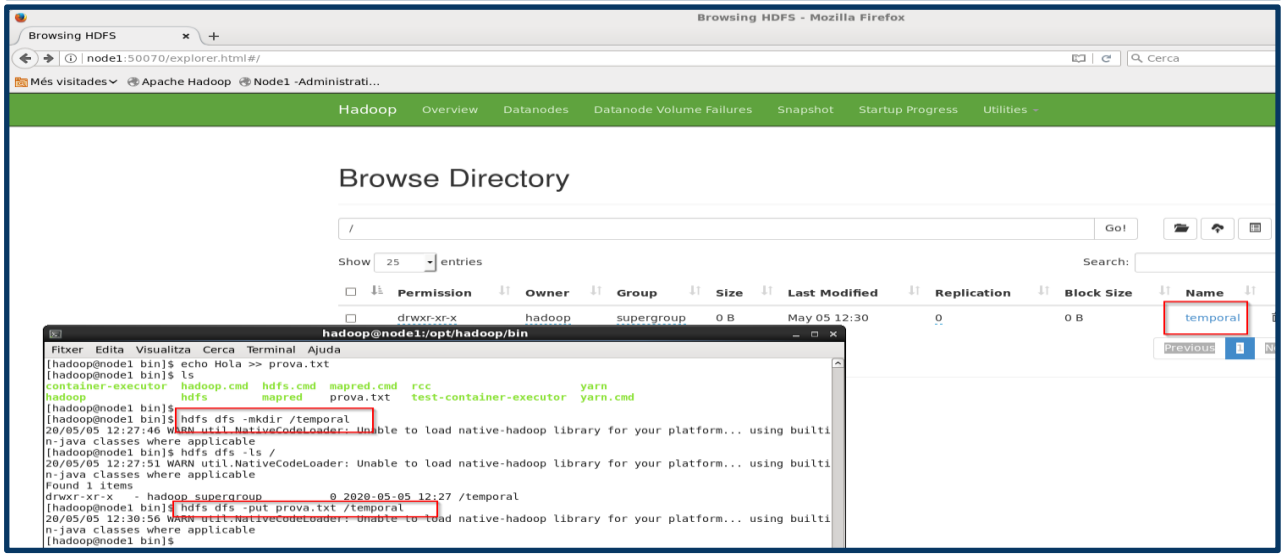
3.1.4.1.1. Exemples pràctics amb HDFS

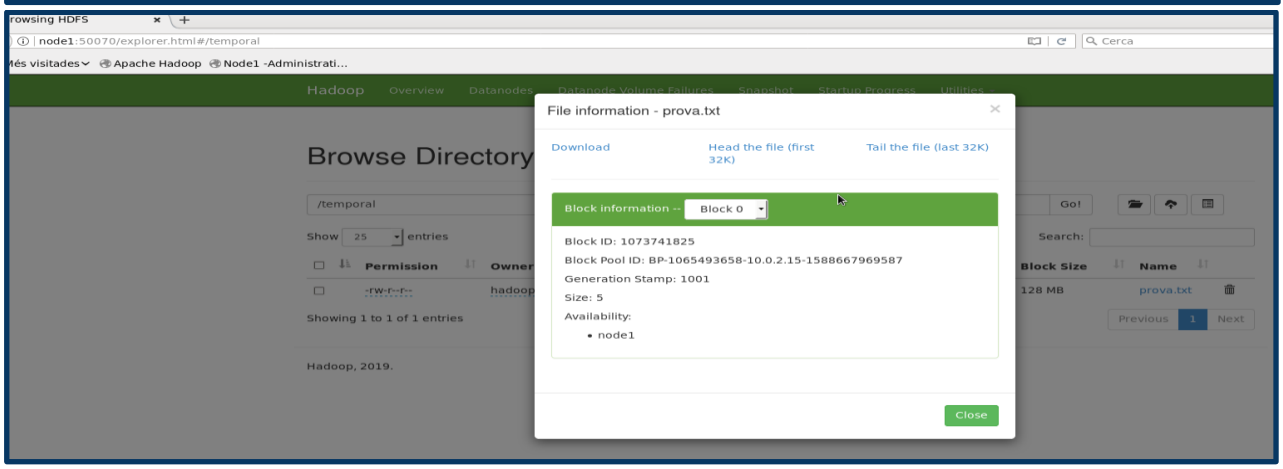
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Crearem un directori HDFS anomenat "temporal" i utilitzant la comanda "put" pujarem un fitxer de txt del sistema d'arxius local al directori temporal de HDFS. Posteriorment ho podrem visualitzar des de la interfície d'usuari web del Namenode on seleccionarem l'opció "Browse the file system" ubicat en la pestanya "Utilities"

  - hdfs dfs -mkdir /temporal
  - hdfs dfs -put prova.txt /temporal

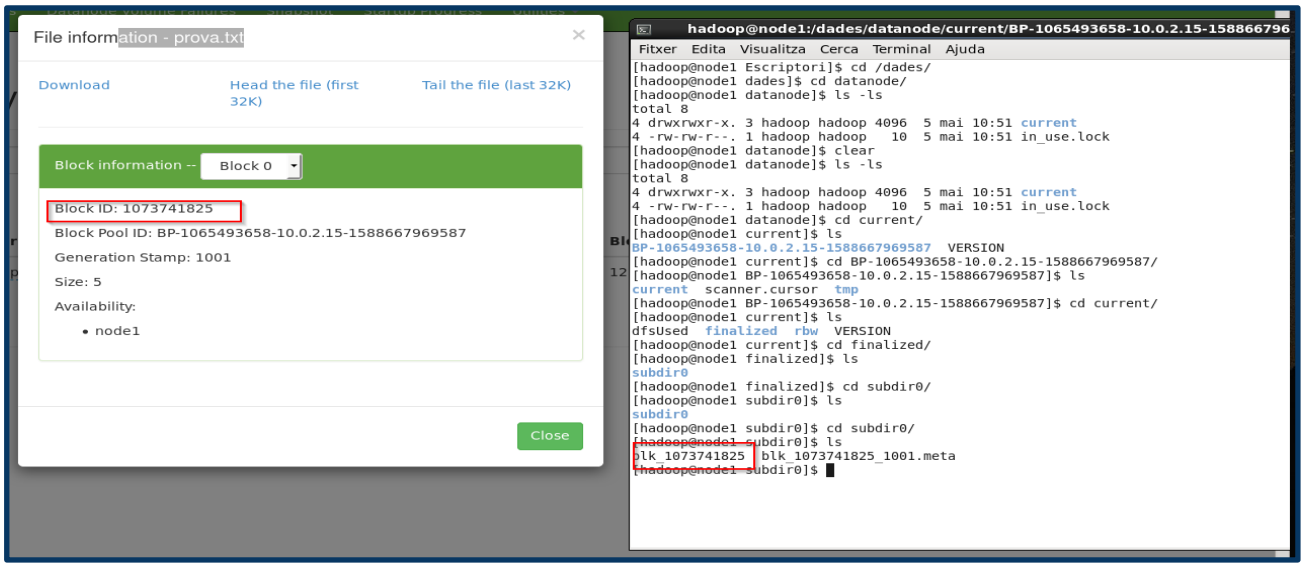




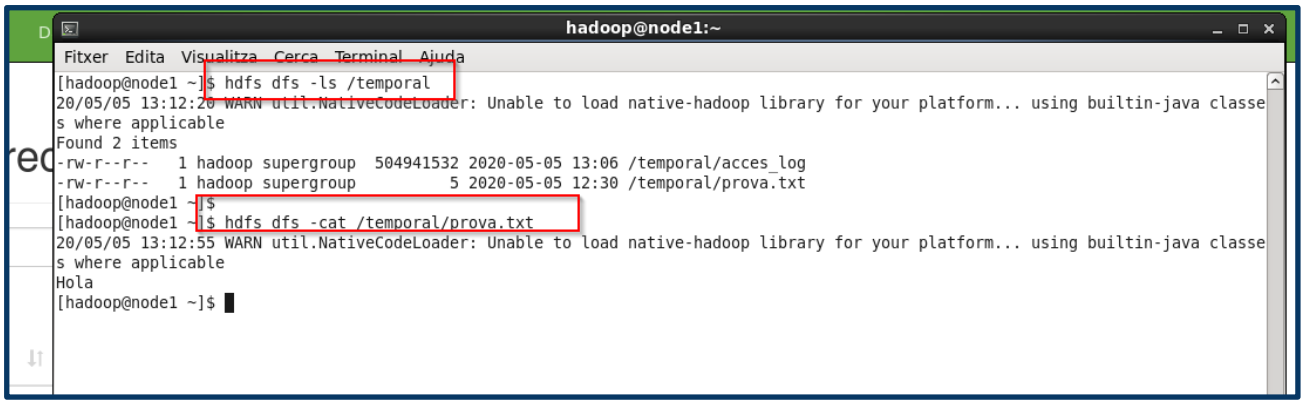


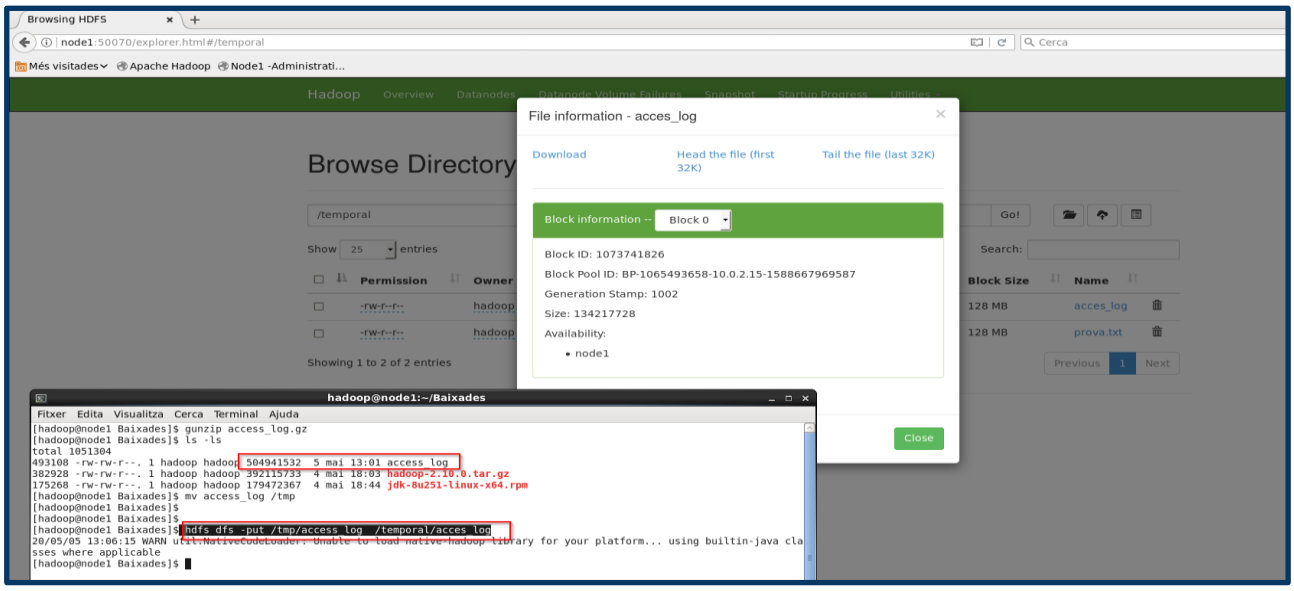
El ID del block i la seva ubicació en el sistema d'arxius local dins del directori */dades/datanode*

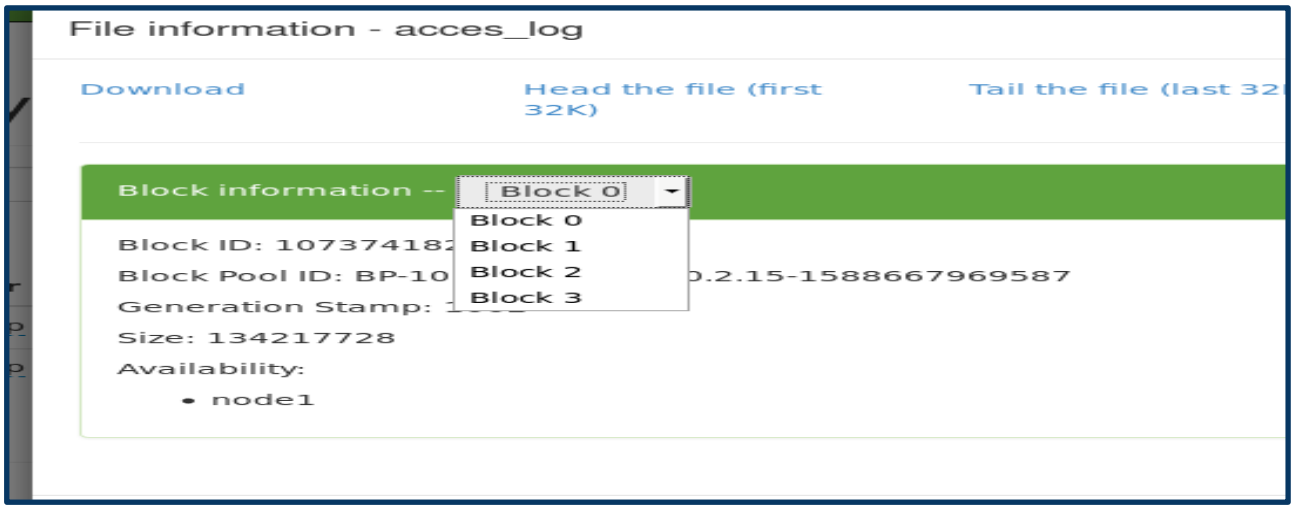
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



- Farem una prova, amb un fitxer més gran (> de 500 MB) on ja no es guarda en 1 block(128 MB) sino en 4.



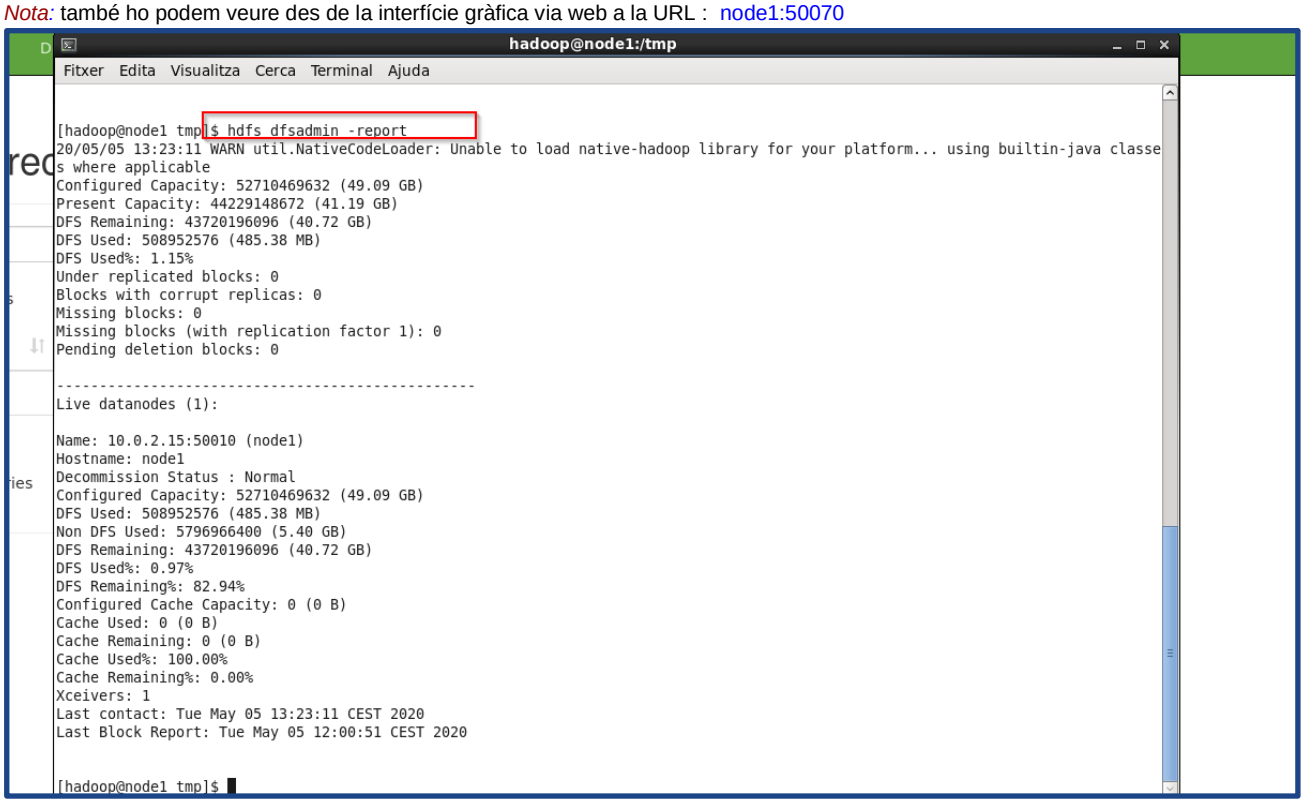




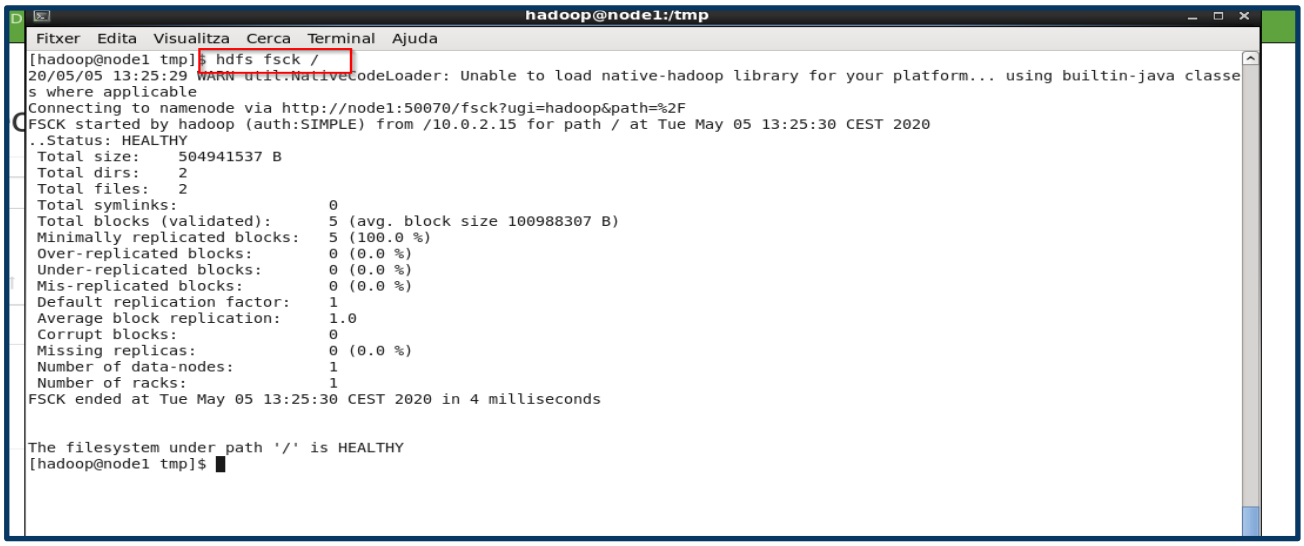
❑ A continuació mostrarem una sèrie de comandes útils per l'administració del clúster (via terminal)

- hdfs dfsadmin -report : On ens diu la informació bàsica el sistema d'arxius i estadístiques

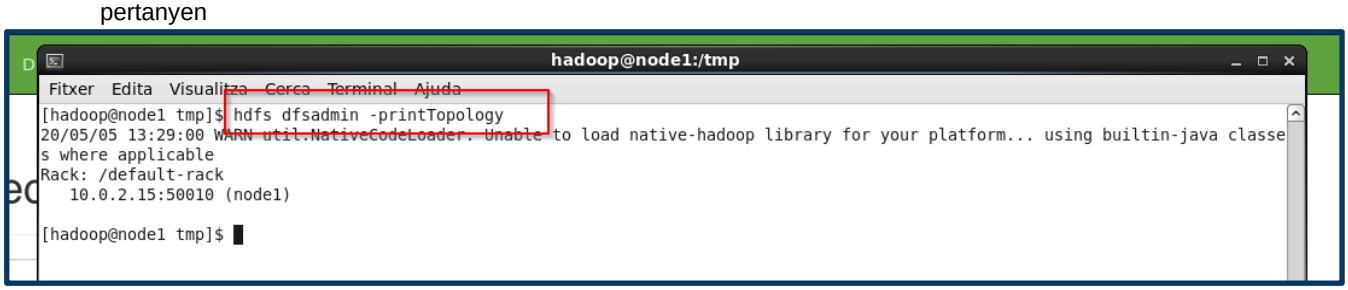
Nota: també ho podem veure des de la interfície gràfica via web a la URL : node1:50070



- hdfs fsck / : Ens fa un check del sistema de fitxers des de l'arrel



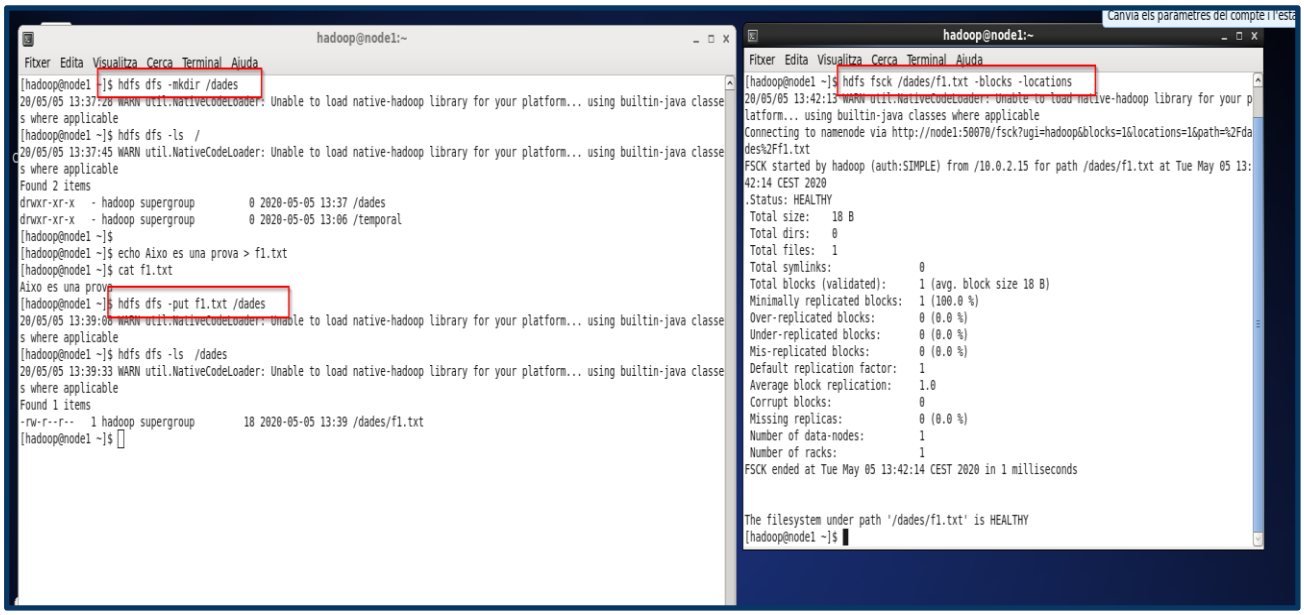
- hdfs dfsadmin -printTopology : Ens indica el nombre de nodes que tenim dins la màquina i a quin rack (físic) pertanyen



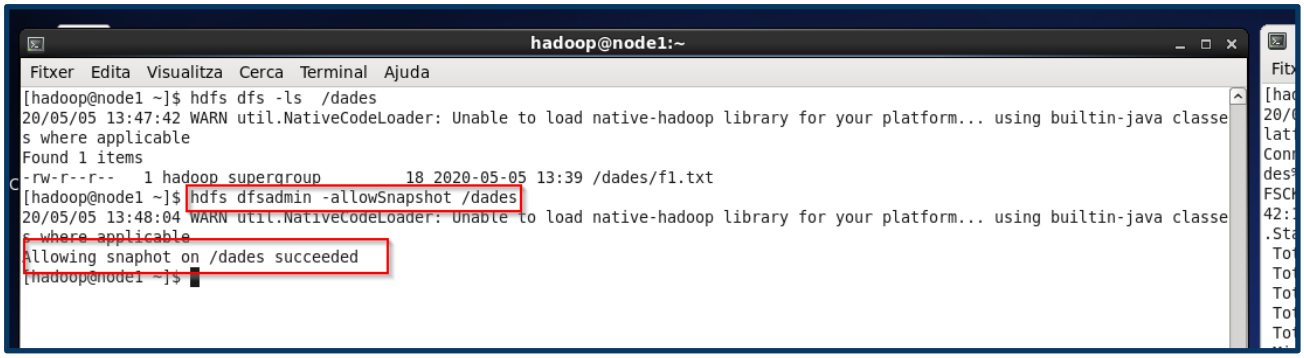
3.1.4.1.2. HDFS Snapshot

- ❏ És una foto del sistema de fitxers HDFS en un moment determinat.
- ❏ Serveix per fer Backups, recuperacions, manteniment d'un històric, etc.
- ❏ A continuació farem un exemple pràctic de HDFS Snapshot
- ❏ Crearem un directori anomenat dades i inserirem un arxiu anomenat f1.txt
  - hdfs dfs -mkdir /dades
  - hdfs dfs -put f1.txt /dades

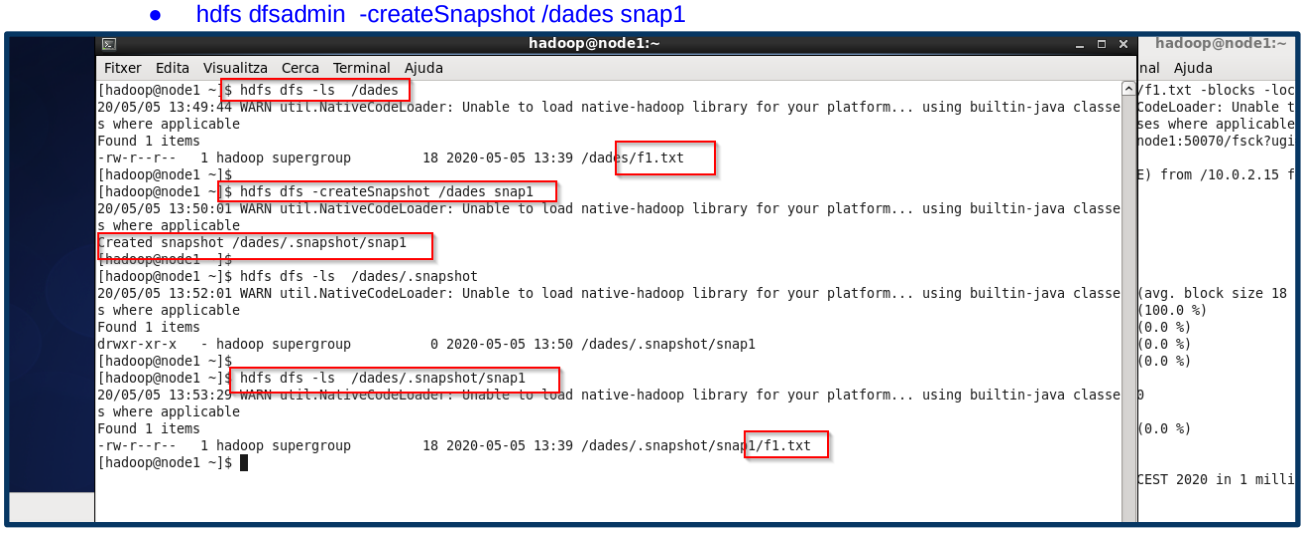
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



- ❑ Activem *snaphot* al directori "*dades*"
  - hdfs dfsadmin -allowSnapshot /dades



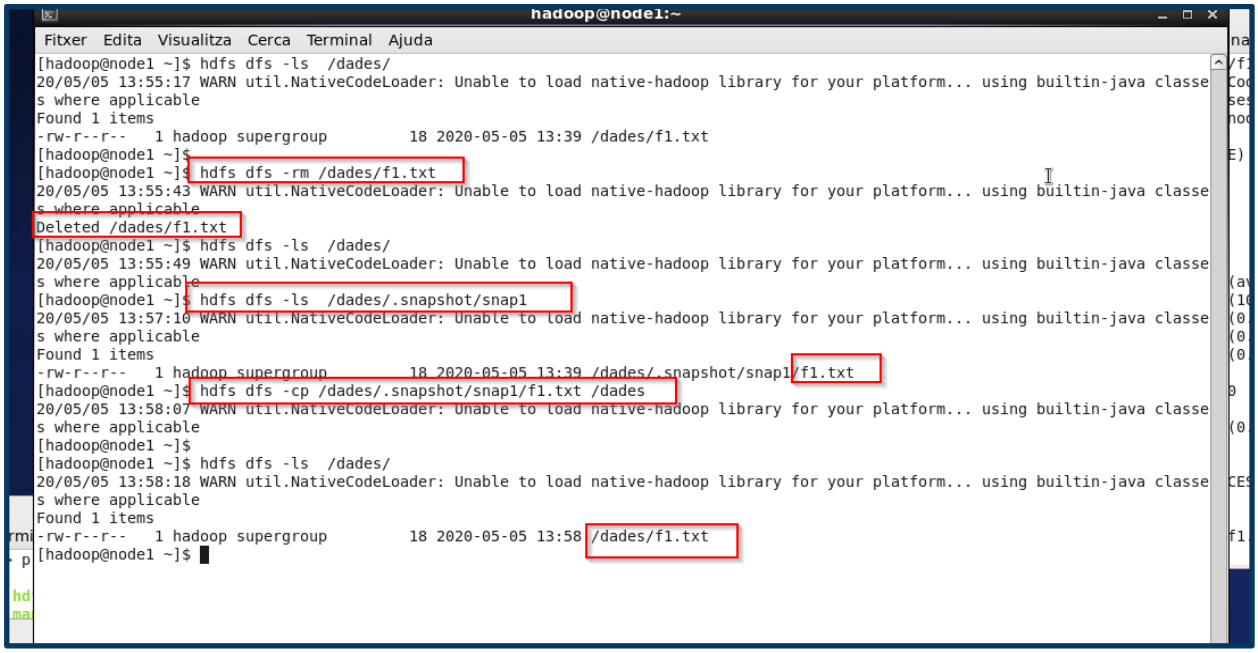
- ❑ Creacio del Snapshot anomenat "snap1"
  - hdfs dfsadmin -createSnapshot /dades snap1



Nom i Cognoms | **Data**

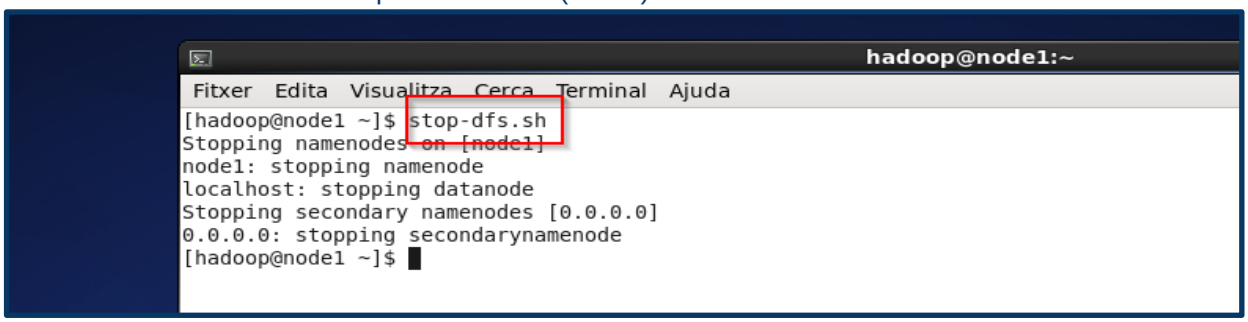
Arnau Subirós Puigarnau | 02-06-2020

- ❑ Ja tenim una còpia. Ara farem una prova i eliminarem l'arxiu f1.txt.
- ❑ Seguidament visualitzem el contingut del snapshot creat on hi ha l'arxiu eliminat
- ❑ Finalment fem una còpia de l'arxiu i ja el tornem a tenir com abans.
  - hdfs dfs -rm /dades/f1.txt
  - hdfs dfs -ls /dades/.snapshot/snap1
  - hdfs dfs -cp /dades/.snapshot/snap1/f1.txt /dades



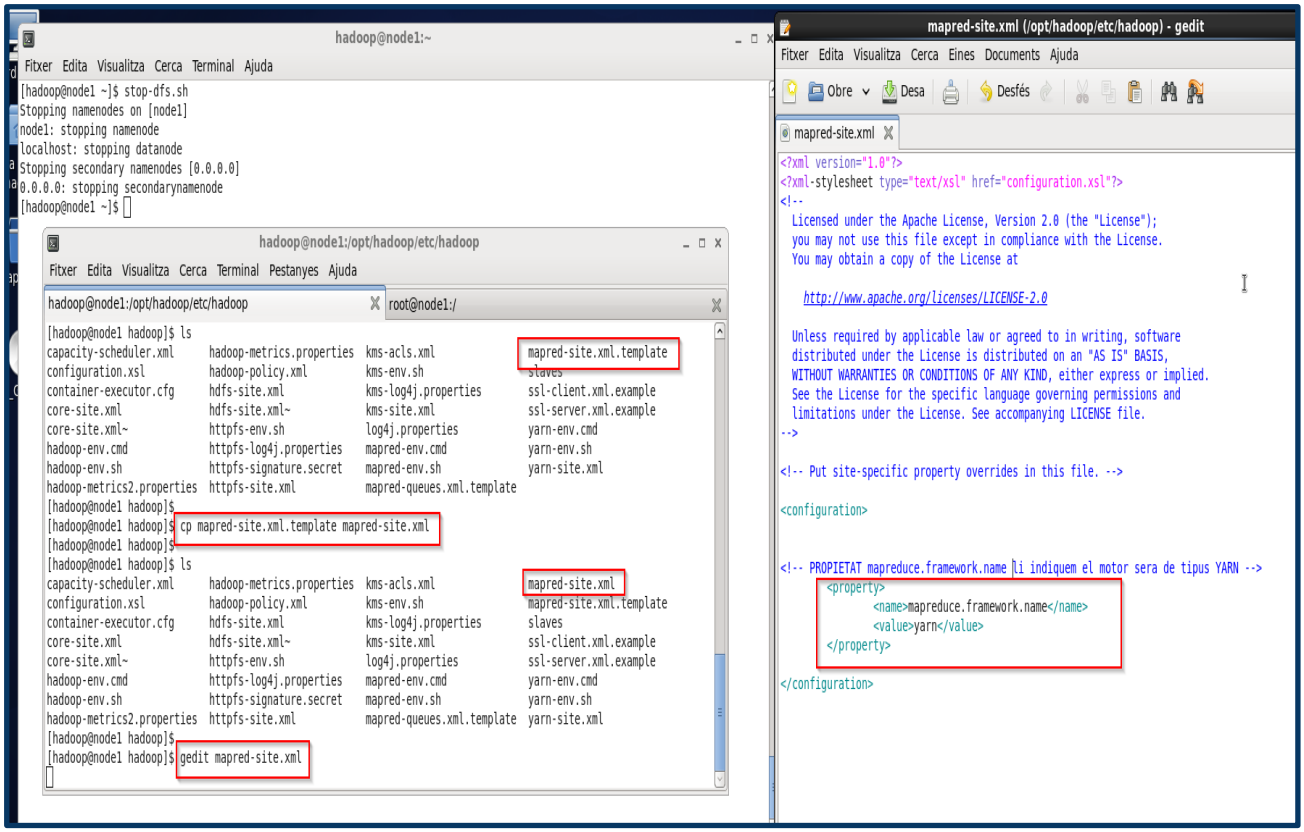
3.1.4.2. Configuració dels processos (Yarn)

- ❑ Primer de tot haurem de parar el clúster (dades)



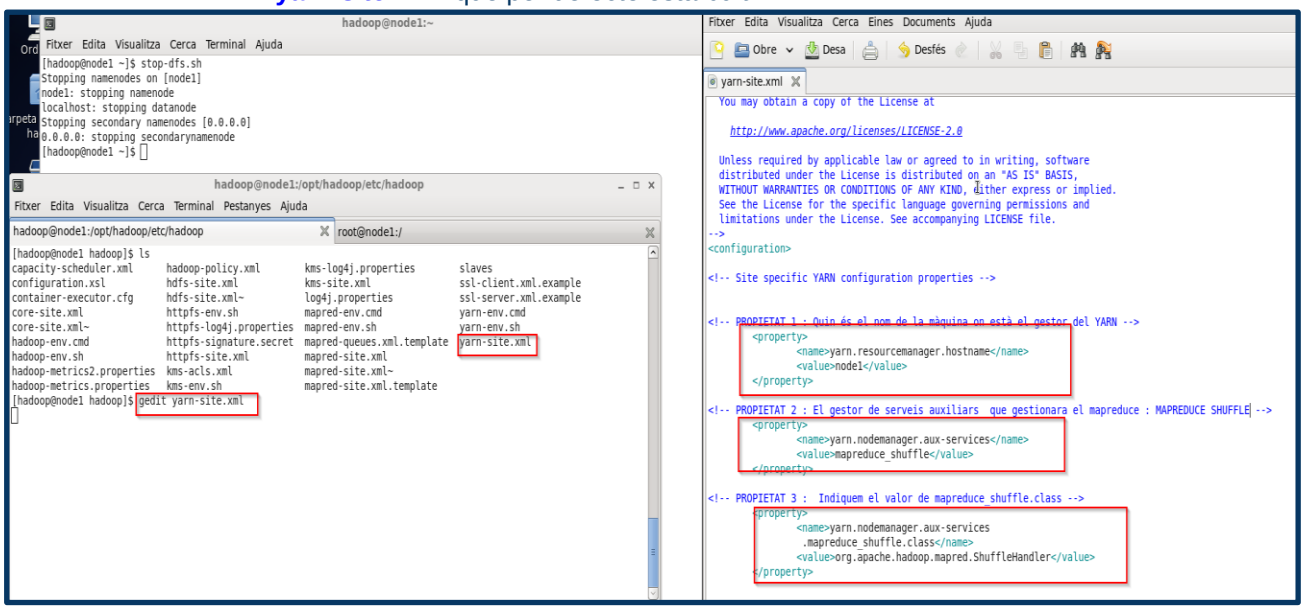
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Accedim als fitxers de configuració ubicats a /opt/hadoop/etc/hadoop
- E n aquest cas haurem de modificar 2 arxius: yarn-site.xml i mapred-site.xml.template
- Els arxius templates (plantilles) no els hem de modificar, haurem de copiar-lo canviant el nom de l'arxiu (sense acabar amb template :mapred-site.xml)
- Editem l'arxiu **mapred-site.xml** que per defecte està buit i afegim la propietat mapreduce.framework.name per indicar que el motor serà de tipus Yarn

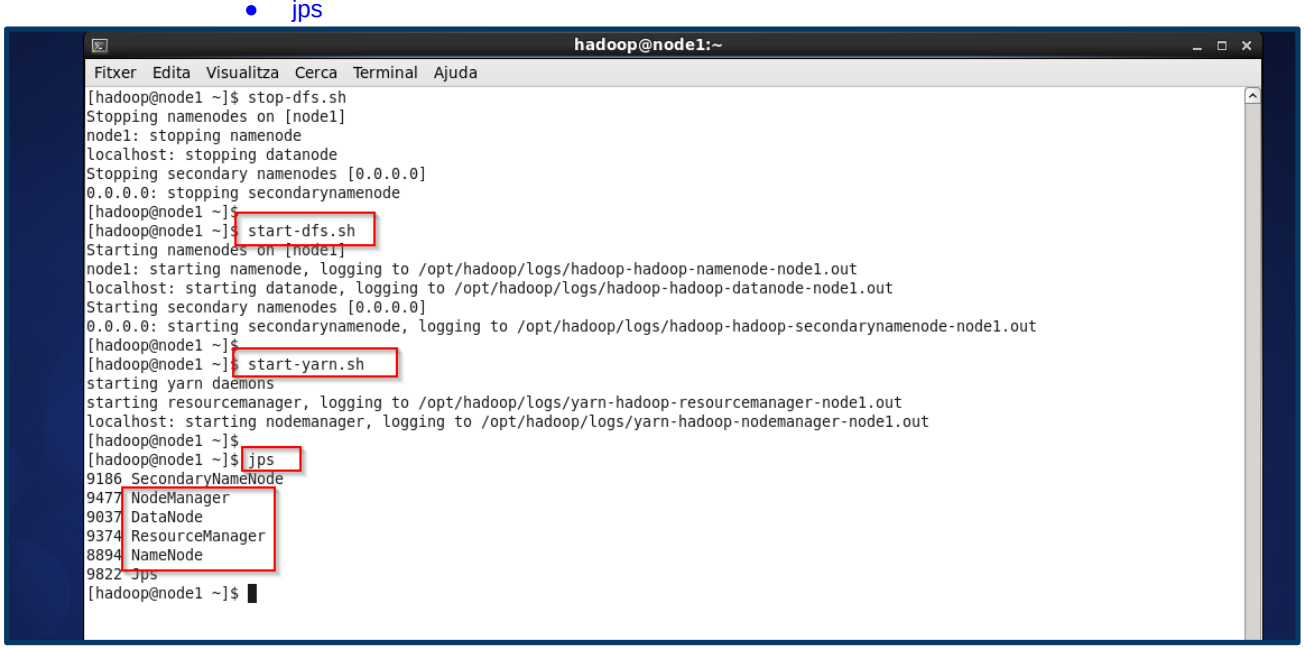


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Editem l'arxiu **yarn-site.xml** que per defecte està buit.

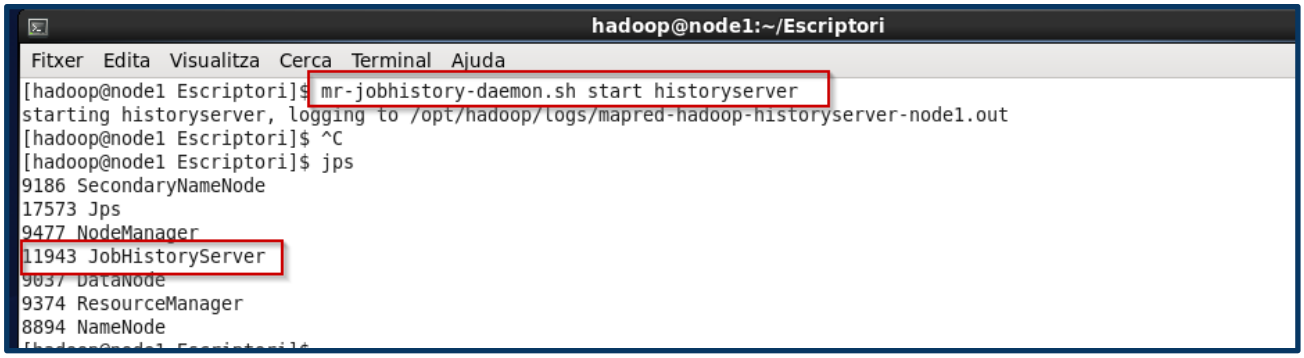


- Guardem els canvis i arrenquem les 2 parts el clúster, primer la part de les dades i després la part dels processos. i visualitzem els processos actius de JVM.
  - start-dfs.sh
  - start-yarn.sh
  - jps

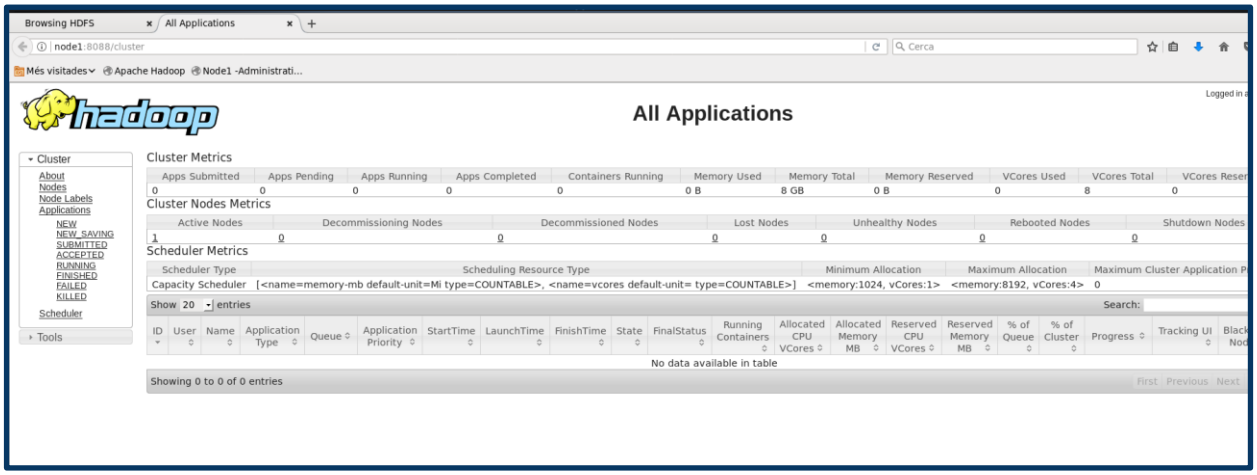


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Si volem el historic dels "jobs" llençats haurem d'activar un altre servei.
  - mr-jobhistory-daemon.sh start historyserver



- Accedim a la interfície web de Resource Manager per defecte el port és el 8088



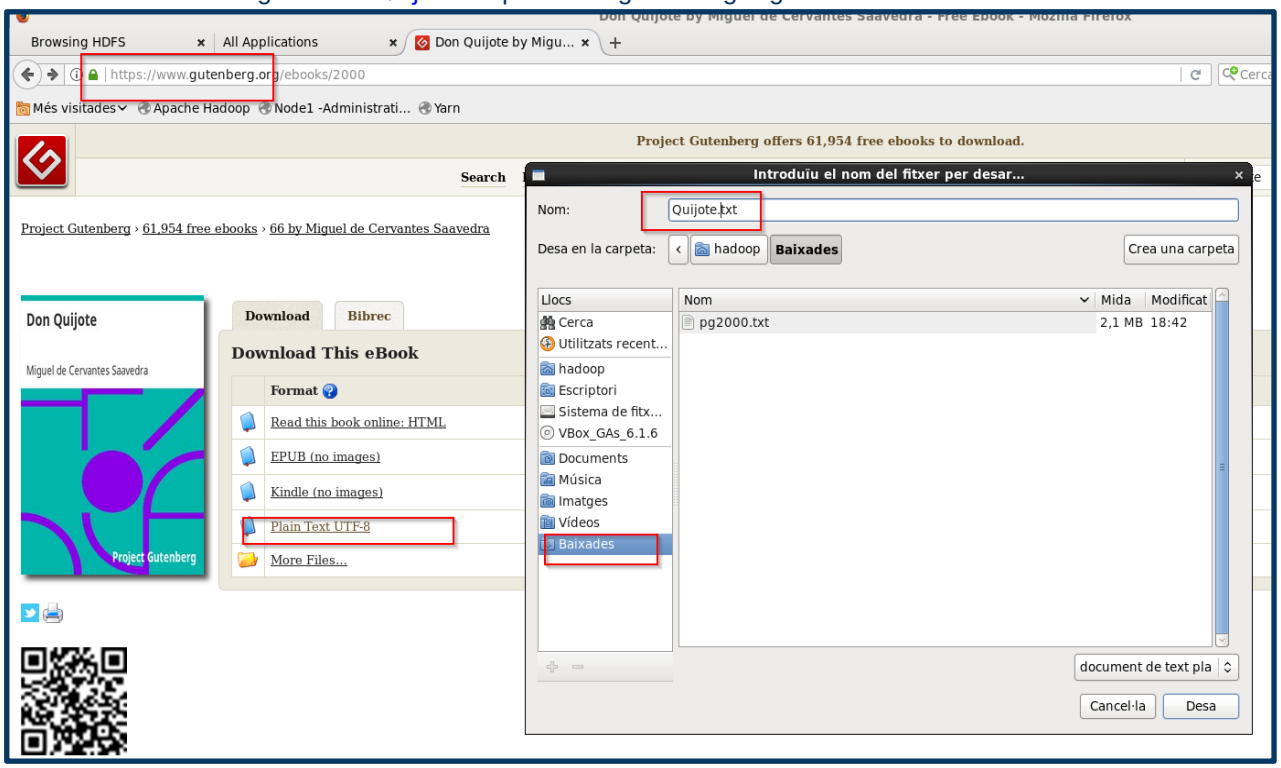
Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

3.1.4.2.1. Exemples Mapreduce

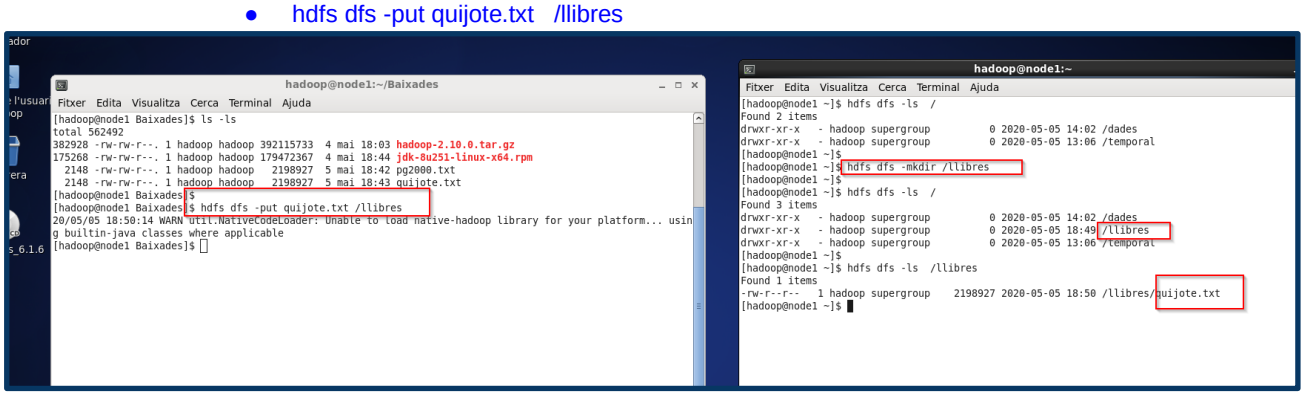
❑ Utilitzant el jar d'exemples Mapreduce

❑ Ens descarreguem "El Quijote" https://www.gutenberg.org/



❑ Crearem un directori HDFS anomenat "llibres" i pujarem l'arxiu

- hdfs dfs -mkdir /llibres
- hdfs dfs -put quijote.txt /llibres

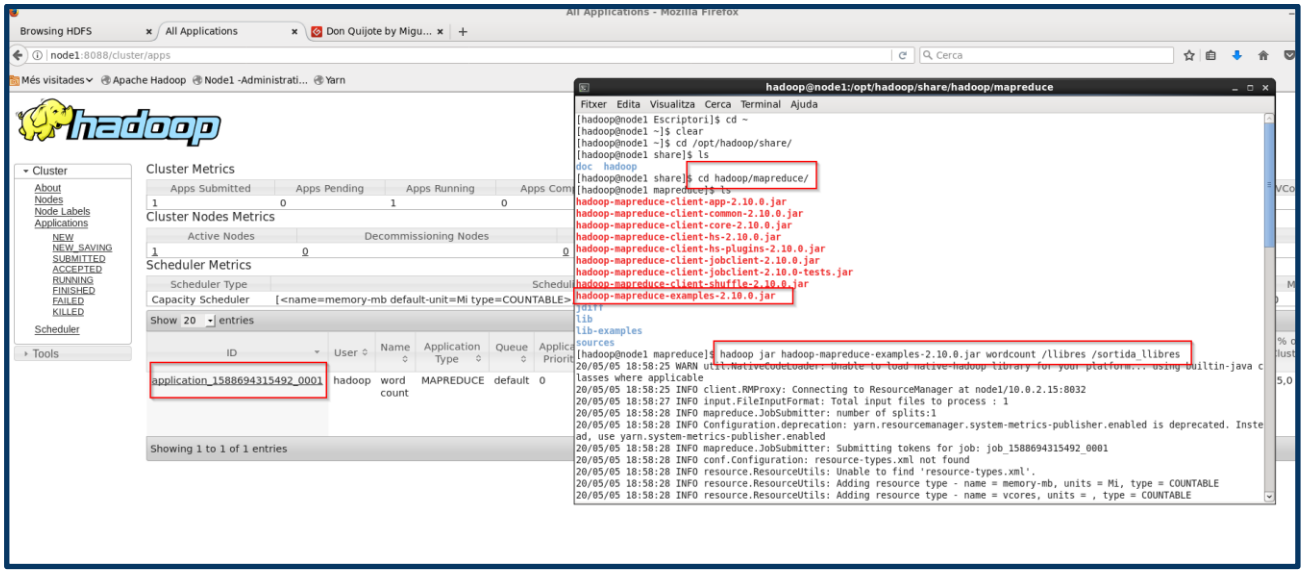


❑ Un cop hem pujat l'arxiu el quijote.txt, crearem un nou fitxer perquè indiqui el número de paraules que té i quantes vegades apareix cada paraula

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

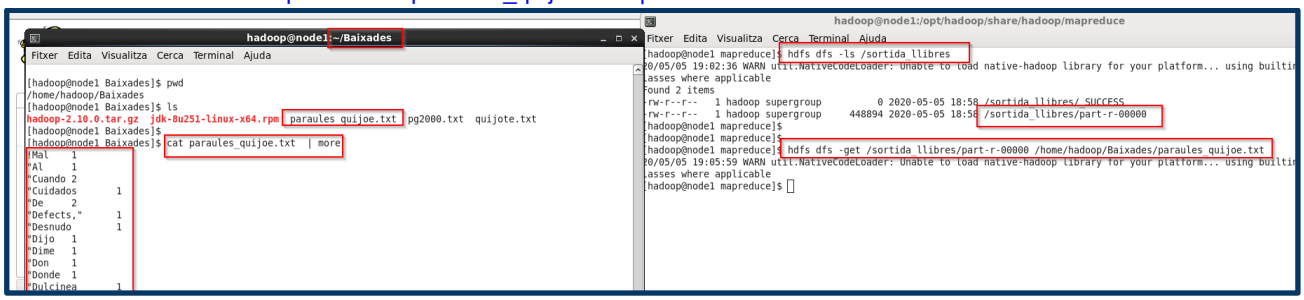
- ❏ Utilitzem el jar d'exemples de mapreduce ubicat a */opt/hadoop/share/hadoop/mapreduce* anomenat *hadoop-mapreduce-examples-2.10.jar* (un dels exemples és el *wordcount)* que busca el contigut del directori seleccionat deixant les paraules a un altre directori desti(si no està creat ,es crea)

- hdfs jar *hadoop-mapreduce-examples-2.10.jar wordcount /llibres /sortida_llibres*



- ❏ Confirmem que l'arxiu de sortida s'ha creat al nou directori sortida_llibres i farem un "get" per baixar-lo i poder-lo llegir (li direm paraules_qujote.txt i l'ubiquem al directori Baixades)

- dfs dfs -ls /sortida_llibres
- hdfs dfs -get /sortida_llibres/part-r-00000 /home/hadoop/Baixades/paraules_quijote.txt
- cat /home/hadoop/Baixades/paraules_quijote.txt | more



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

❑ Utilitzant codi JAVA

❑ Farem un exemple de MapReduce desde codi JAVA amb l'arxiu anomenat ComptarParaules.java

*ComptarParaules.java

```
import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.util.GenericOptionsParser;
/**
* <p>Comptem  el número de paraules que  apareixen en un document utilitzant MapReduce.
El codi té un mapper, reducer, i el programa principal.</p>
*
*/
public class ComptarParaules {
 /**
 * <p>
 * El mapper s'exte de la interface org.apache.hadoop.mapreduce.Mapper.
 * al executar ejecutar Hadoop , es rep cada linia del fiter d'entrada com un input
 * La funció map retorna per cada paraula(word) un (word,1) com sortida </p>
 */
 public static class TokenizerMapper
 extends Mapper<Object, Text, Text, IntWritable>{

 private final static IntWritable one = new IntWritable(1);
 private Text word = new Text();

 public void map(Object key, Text value, Context context
 ) throws IOException, InterruptedException {
 StringTokenizer itr = new StringTokenizer(value.toString());
 while (itr.hasMoreTokens()) {
 word.set(itr.nextToken());
 context.write(word, one);
 }
 }
 }
```

*ComptarParaules.java

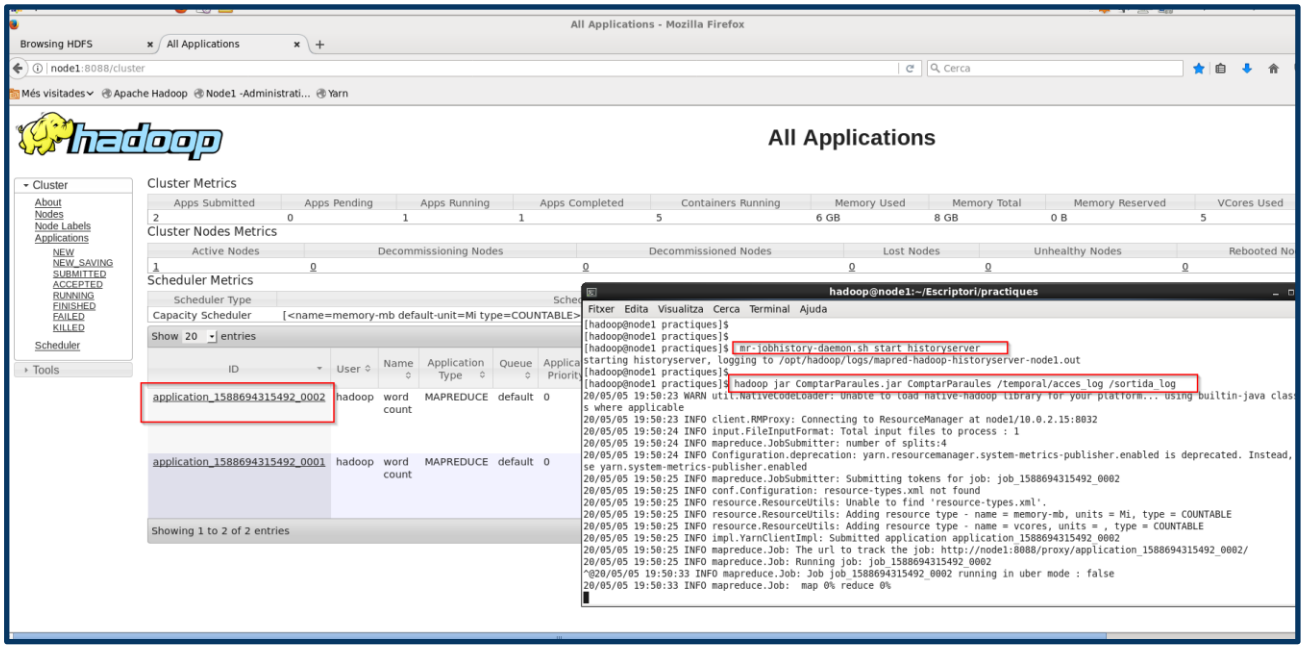
```
 }

 /*
La funció reduce rep tots els valors que té la mateixa clau com entrada i retorna la clau i el
numero de ocurrencies com a sortida  */
 public static class IntSumReducer
 extends Reducer<Text,IntWritable,Text,IntWritable> {
 private IntWritable result = new IntWritable();
 public void reduce(Text key, Iterable<IntWritable> values,
 Context context
 ) throws IOException, InterruptedException {
 int sum = 0;
 for (IntWritable val : values) {
 sum += val.get();
 }
 result.set(sum);
 context.write(key, result);
 }
 }
 /**
 * La entrada es qualsevol fitxer
 */
 public static void main(String[] args) throws Exception {
 Configuration conf = new Configuration();
 String[] otherArgs = new GenericOptionsParser(conf, args).getRemainingArgs();
 if (otherArgs.length != 2) {
 System.err.println("Us: ComptarParaules <in> <out>");
 System.exit(2);
 }
 Job job = Job.getInstance(conf, "word count");
 job.setJarByClass(ComptarParaules.class);
 job.setMapperClass(TokenizerMapper.class);

 /****  ****/
 //job.setCombinerClass(IntSumReducer.class);

 job.setReducerClass(IntSumReducer.class);
 job.setOutputKeyClass(Text.class);
 job.setOutputValueClass(IntWritable.class);
 FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
 FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
 System.exit(job.waitForCompletion(true) ? 0 : 1);
 }
}
```

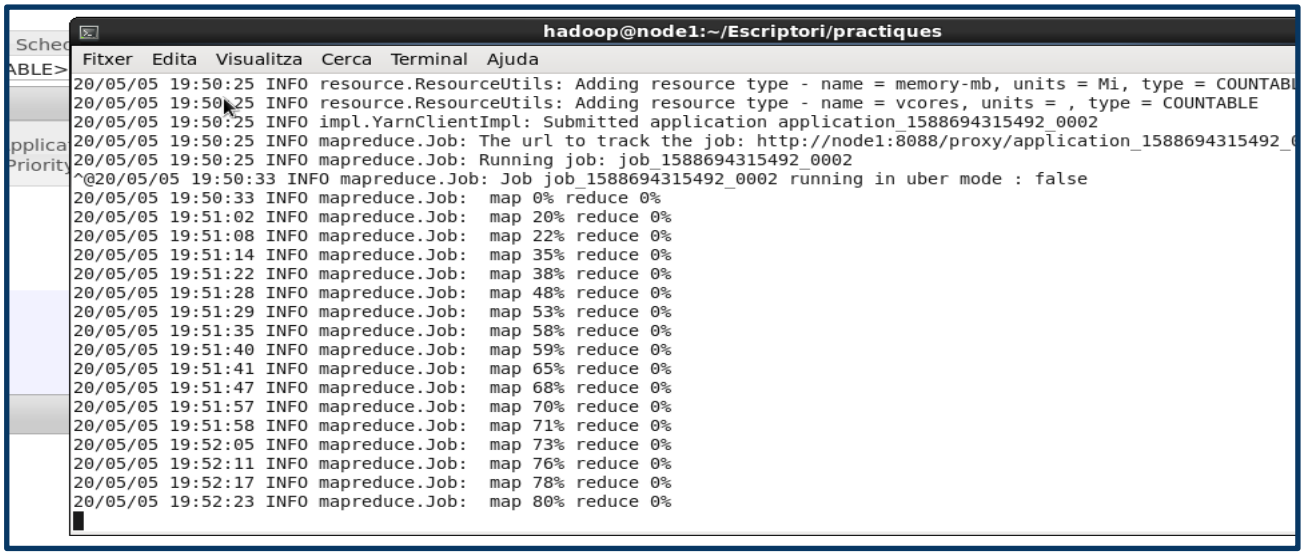
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- ❑ Com que no tenim IDE de compilació de JAVA (Eclipse, Netbeans,etc) farem la compilació de l'arxiu amb hadoop que ens generarà 3 classes (1 pel Main, un altre pel Mapper i l'altre pel Reducer)
  - hadoop com.sun.tools.javac.Main ComptarParaules.java

- ❑ Hadoop només admet fitxers jar. Haurem de crear un fitxer jar
  - jar cf ComptarParaules.jar Comp*.class

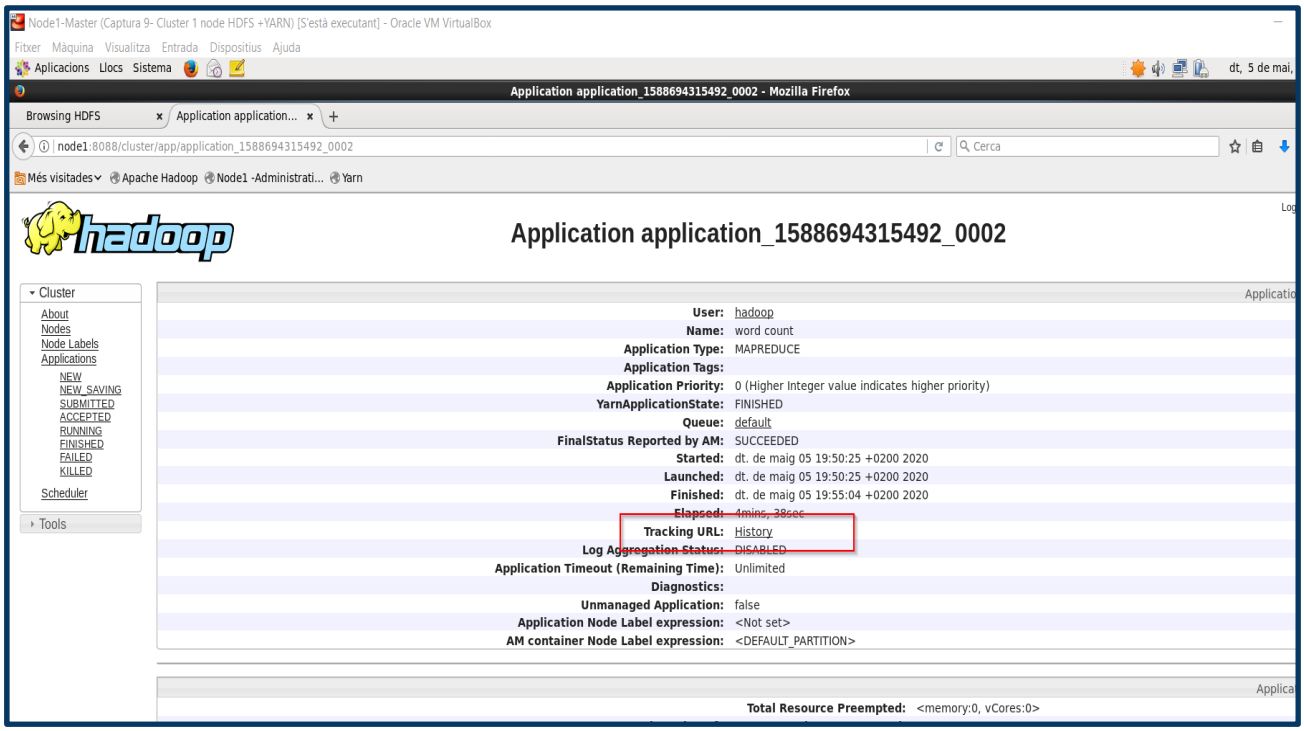
- ❑ Executem el programa ComptarParaules i el resultat el dipositara a */temporal/acces_log/sortida_log*
  - hadoop jar ComptarParaules.jar ComptarParaules /temporal/acces_log/sortida_log

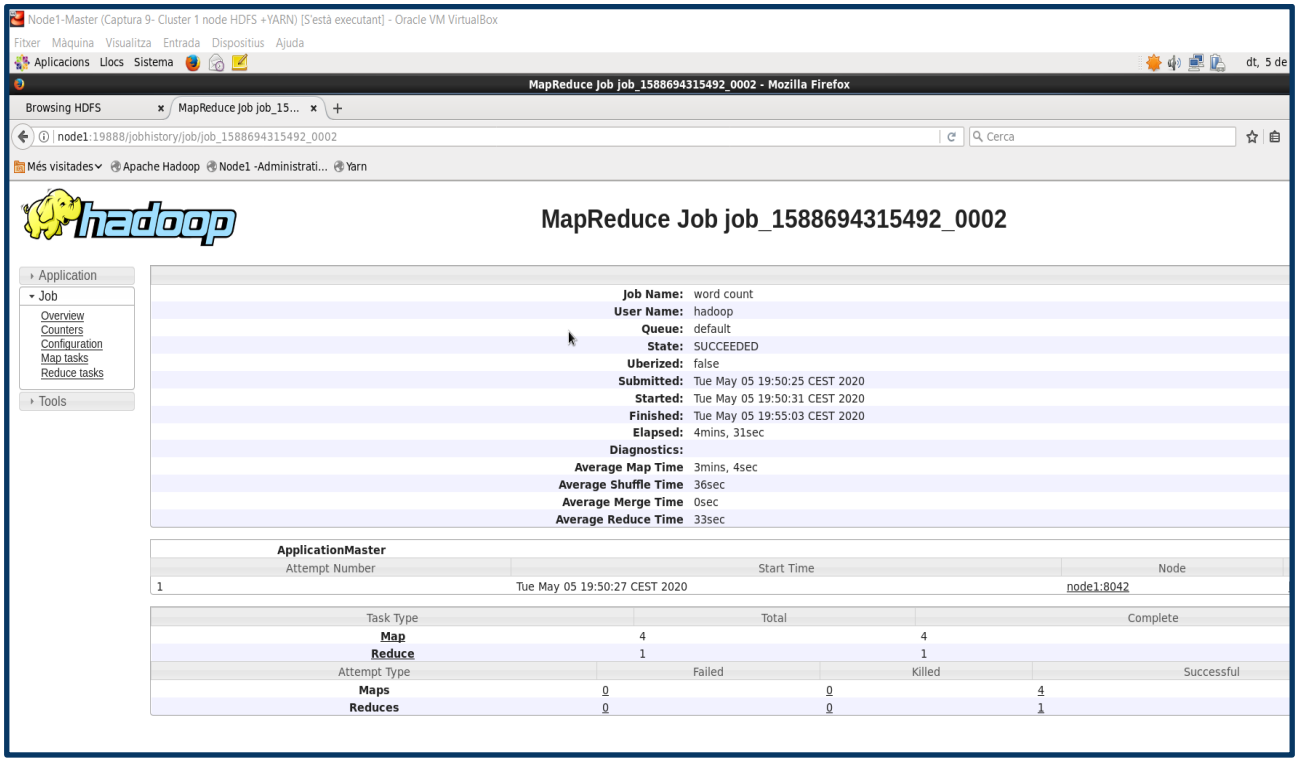




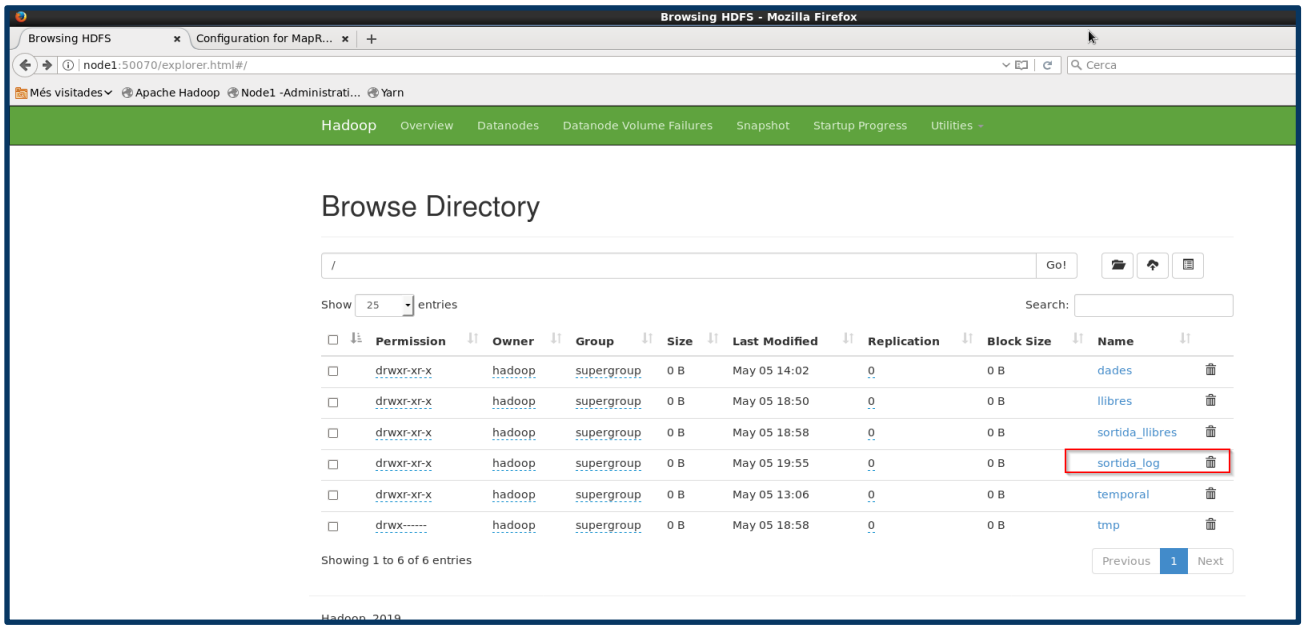
- ❑ L'opció "history" la tindrem activa, ja que anteriorment hem iniciat el servei start historyserver)
  - mr-jobhistory-daemon.sh start historyserver

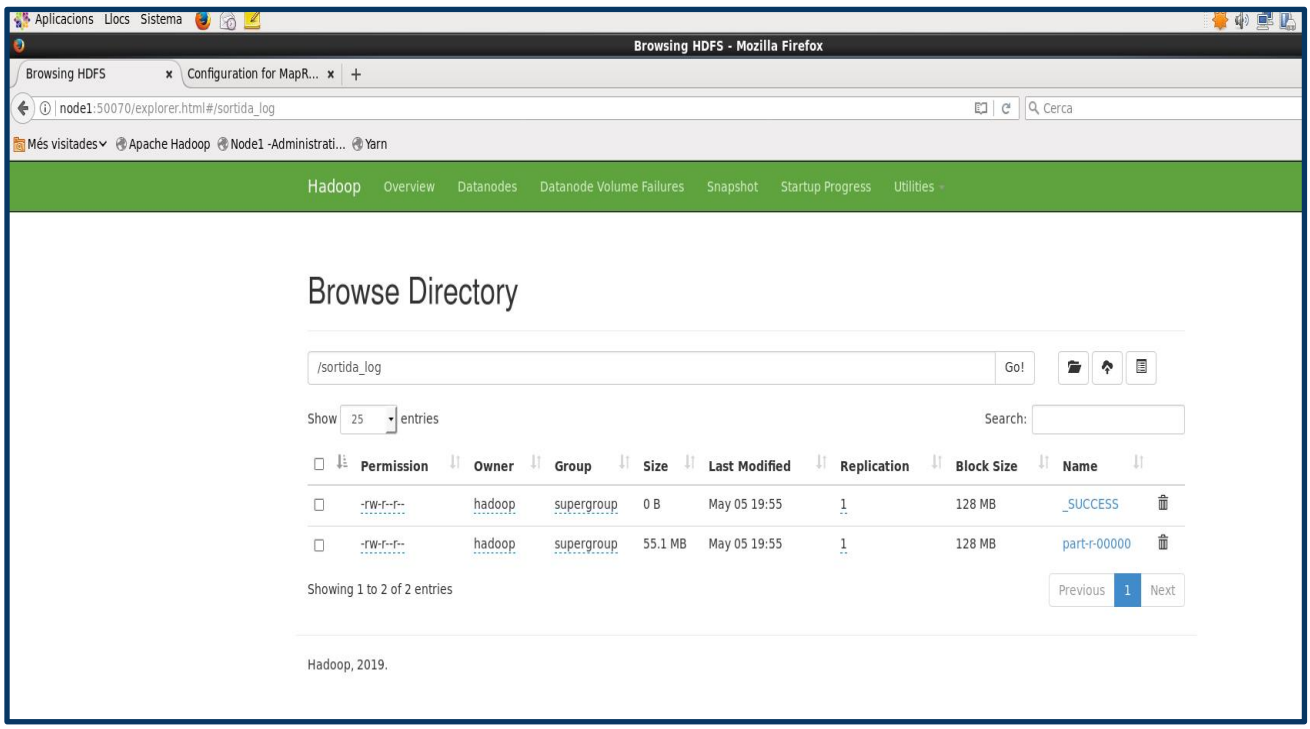
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

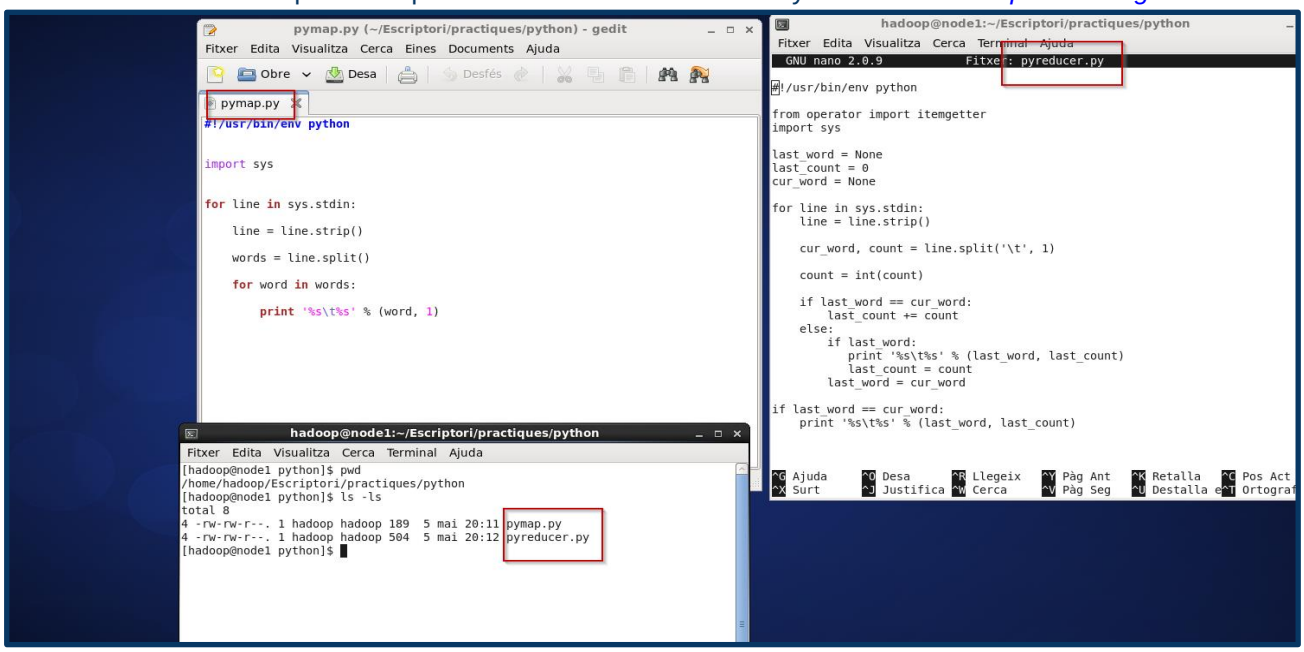




❑ Utilitzant codi Python

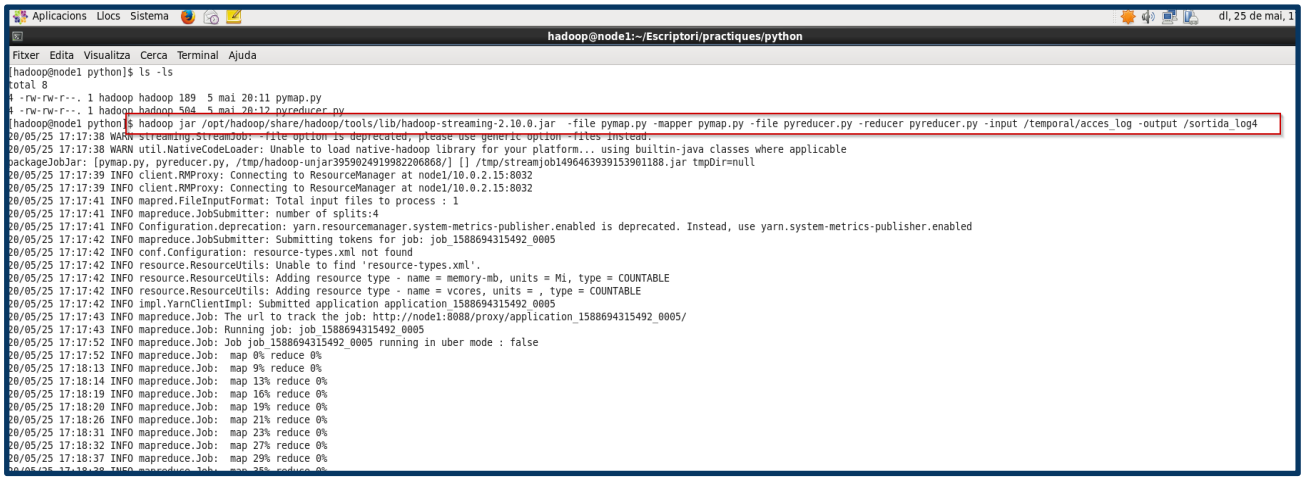
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Farem un exemple de MapReduce desde utilitzant codi Python amb *Hadoop streaming* [12]



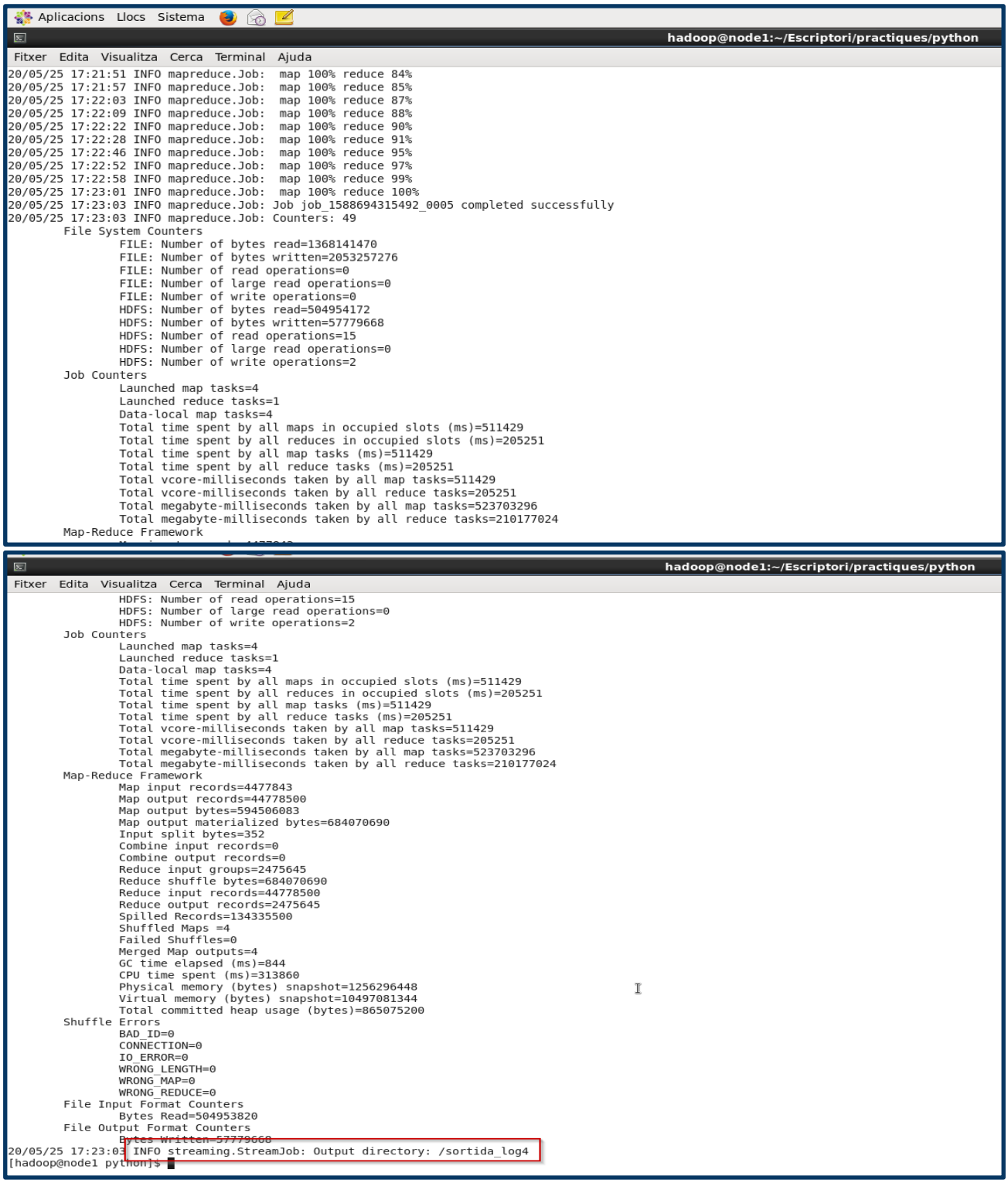
- Executem l'aplicació amb Hadoop streaming especificant: el mapper ,el reducer, l'arxiu d'origen de les dades i a quina carpeta volem ubicar el resultat(el directori destí no fa falta que existeixi, es crea automàticament)

- *hadoop jar /opt/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.10.0.jar -file pymap.py -mapper pymap.py -file pyreducer.py -reducer pyreducer.py -input /temporal/acces_log -output /sortida_log4*

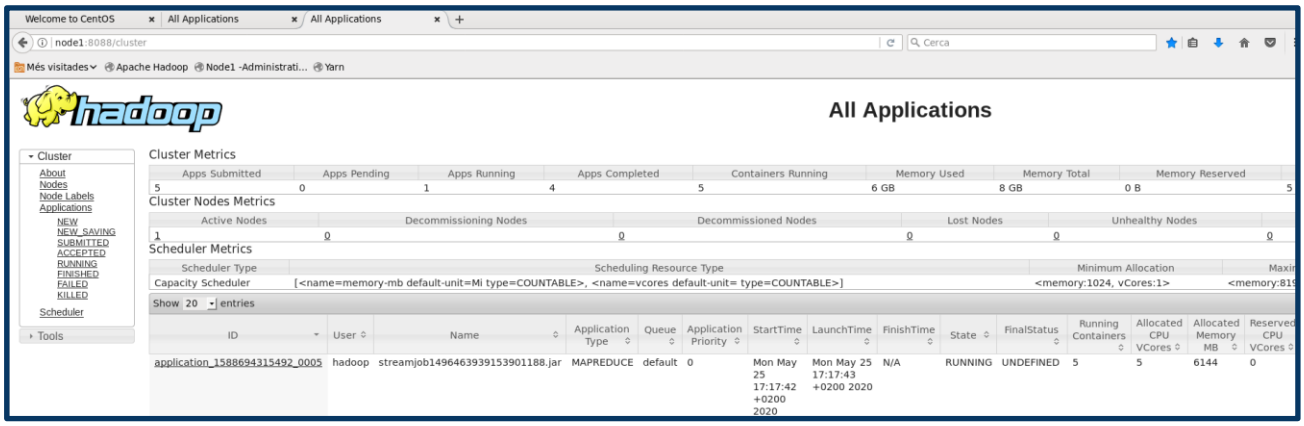


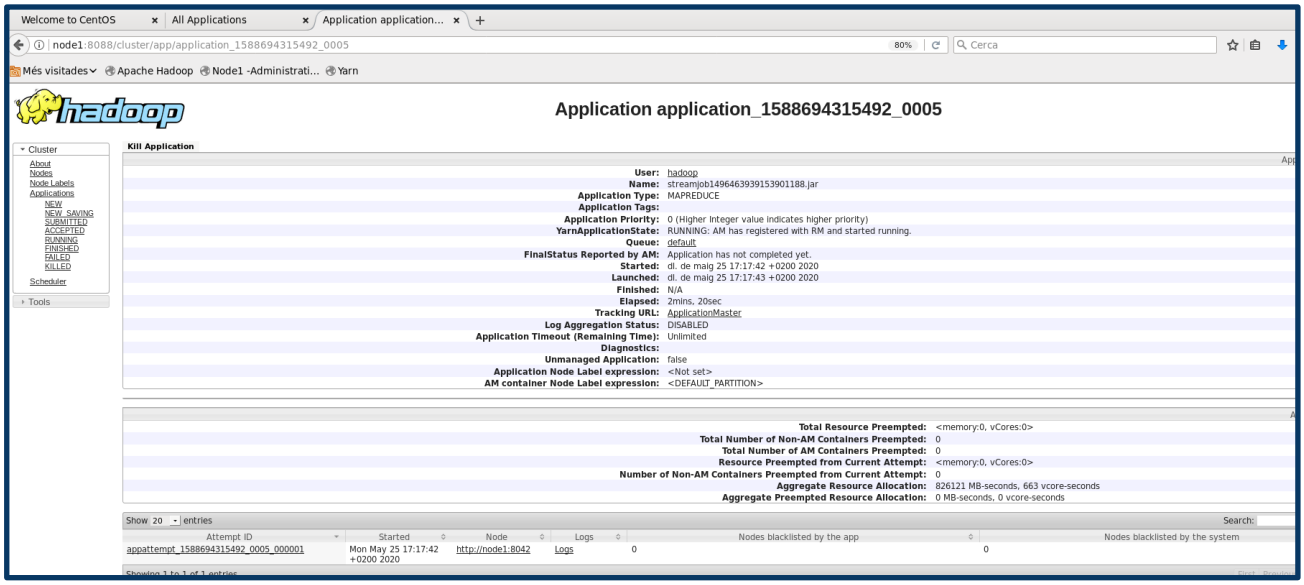
[12] Ens permet fer mappers i reduccer amb un altre llenguatge que no sigui Java.S'uneix el mapper i el reduccer i ho llença com si fos un "proces Java"

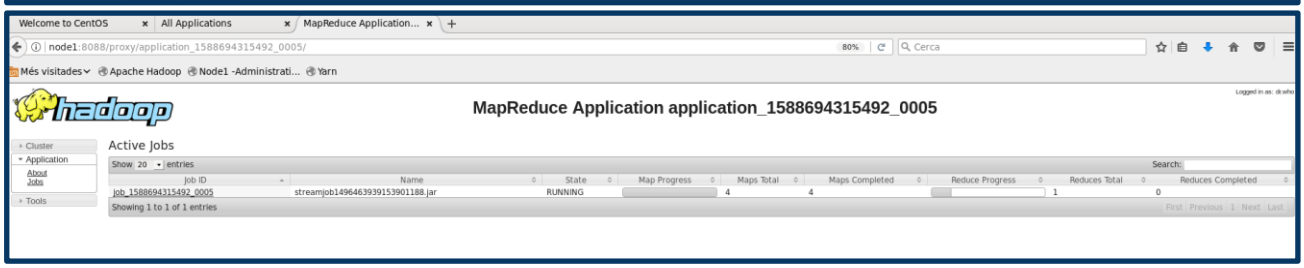
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



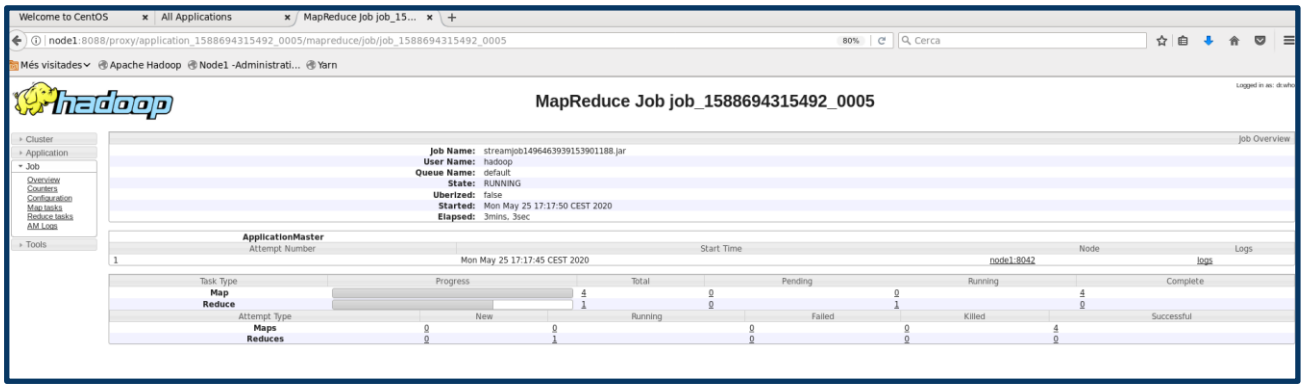
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

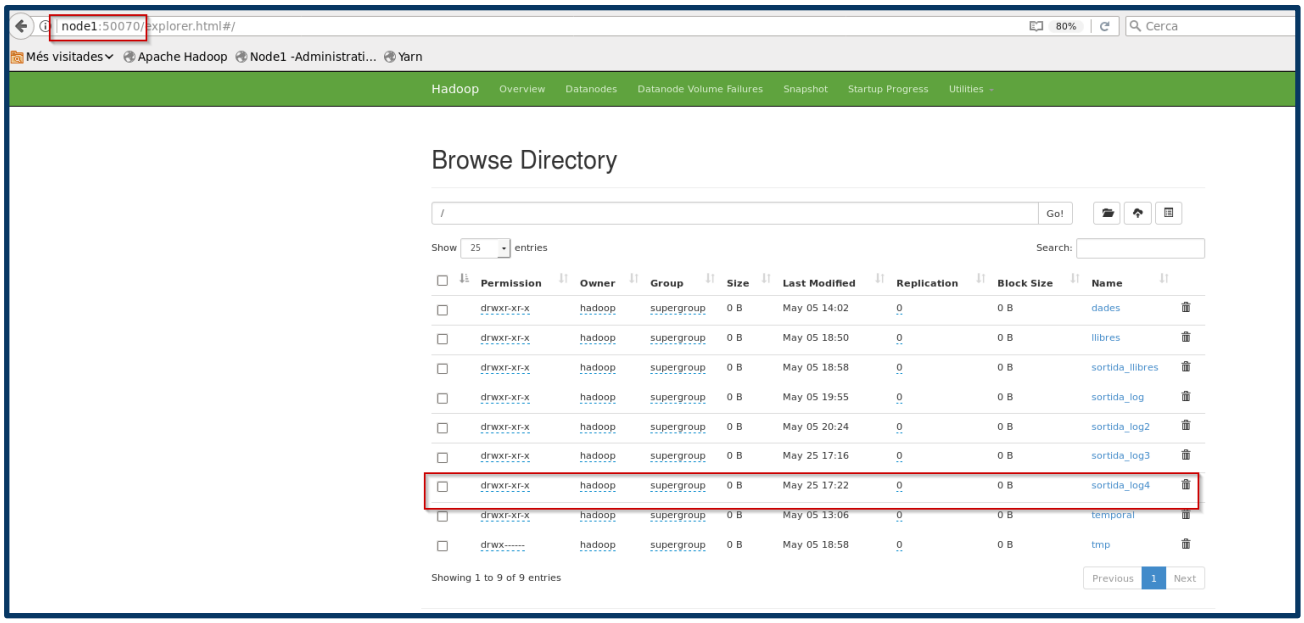


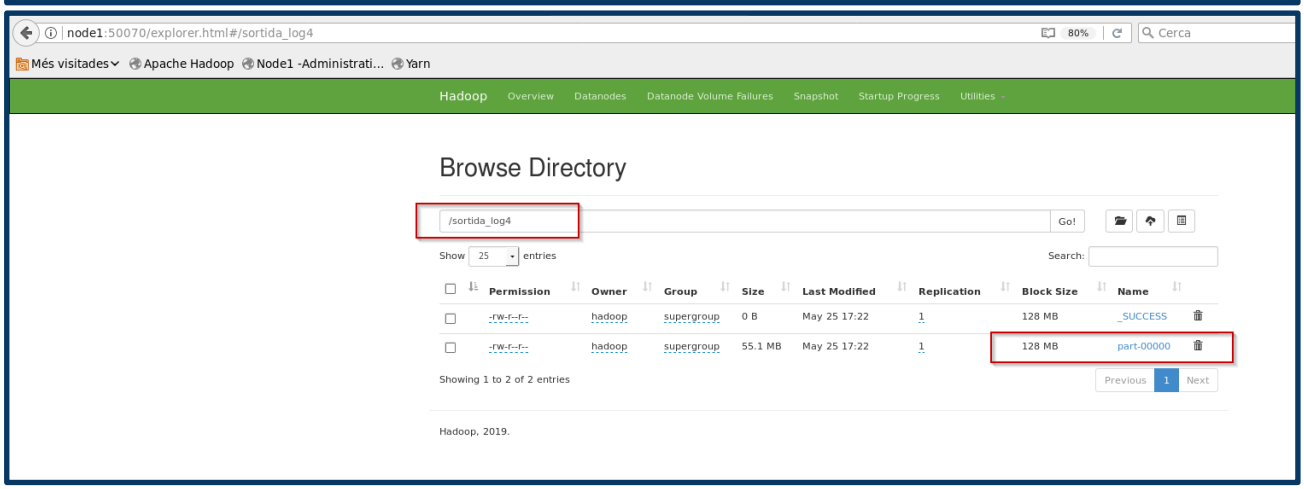




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020







Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

3.2. Clúster Hadoop (diversos node s)

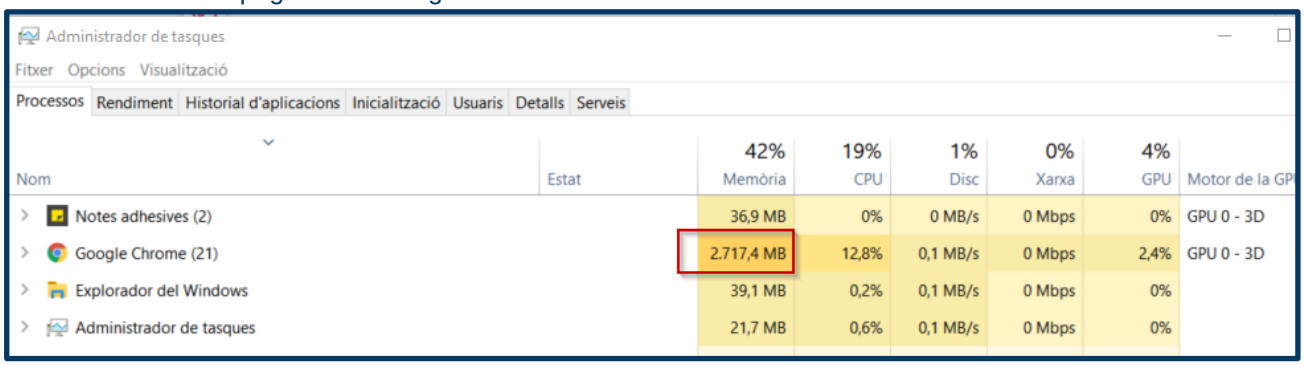
3.2.1. Configuracions prèvies

Per no tornar a instal·lar el sistema operatiu, i diversos paquets que anteriorment hem hagut de descarregar, clonarem la màquina virtual on havíem fet les proves amb el clúster d' Hadoop pseudodistribuit però haurem de tenir en compte:

- Canviar el hostname
- Configurar la xarxa
- Configurar el SSH amb claus públiques

Nota: Hem de tenir en compte que quan tinguem les 3 màquines virtuals enceses (seran 4 GB /host), només tindrem 4 GB disponibles en el host amfitrió, o sigui que tindrem que limitar la memòria RAM. Si obrim el navegador, hauria de ser el indispensable. (Tot i que normalment utilitzo Google Chrome consumeix més memòria que altres navegadors com Firefox)

Actualment tenint 6 pàgines de Google Chrome obertes em consumeix 2.7 GB de memòria RAM

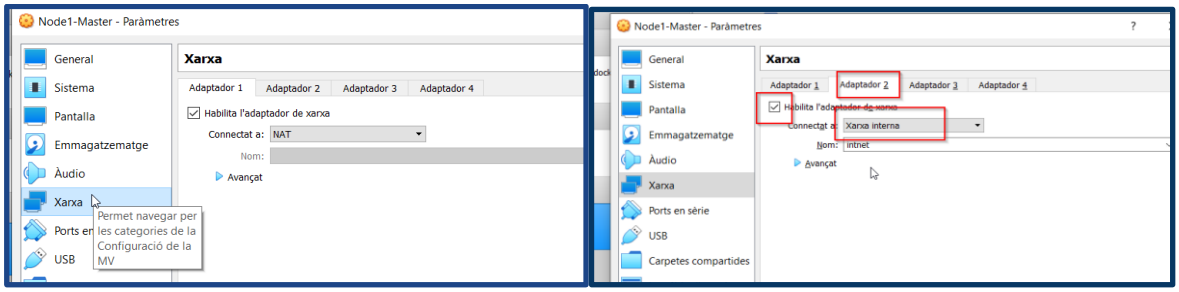


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- ❏ Hem de tenir en compte que abans de clonar les màquines hem de parar el clúster de Hadoop (tant dades com processos) i parar la màquina virtual.

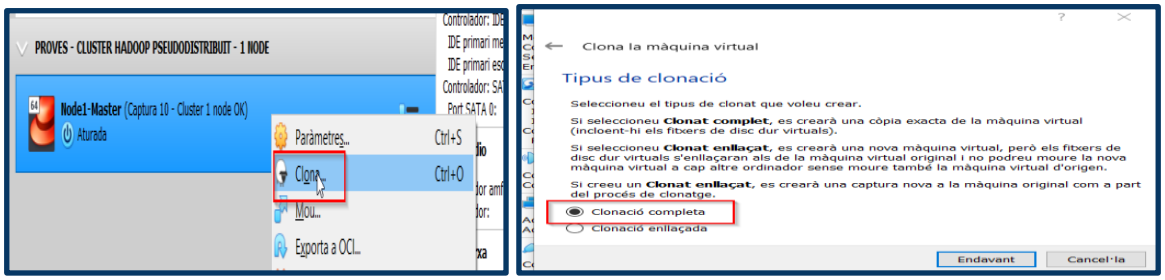
  - [root@node1~]# stop-yarn.sh
  - [root@node1~]# stop-dfs.sh
  - [root@node1~]# init 0

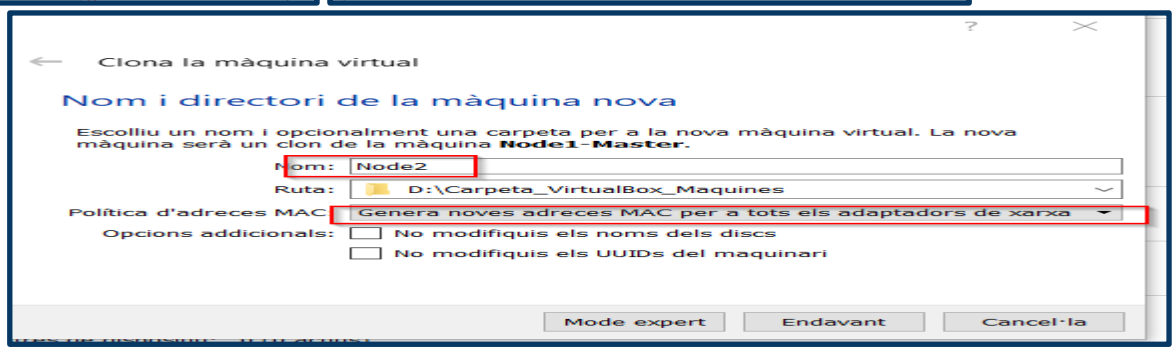
- ❏ Abans afegirem un nou adaptador de xarxa.
  - Adaptaro 1 : NAT per conentant-se a Internet
  - Adaptador 2 : Xarxa interna per la comunicació entre les màquines del clúster.

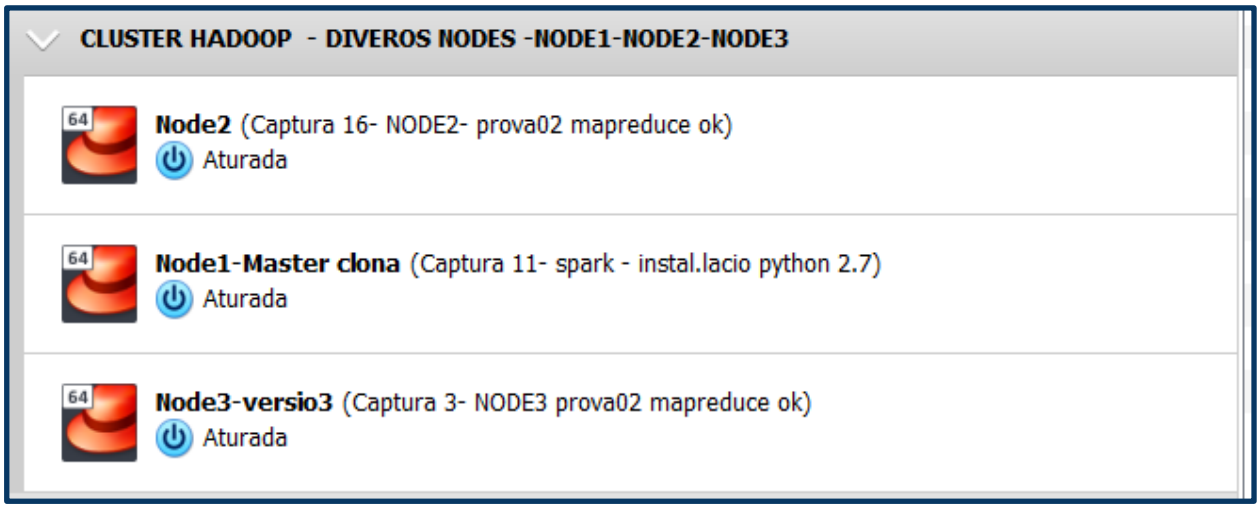


- ❏ Accedirem al VirtualBox i farem la clonació de la màquina node1 ,3 cops per tenir una màquina "template" i poder fer altres clonacions en cas d'errors o modificacions.

  Important: Al clonar les màquines hem de generar noves MAC pels adaptadors de xarxa







3.2.2. Configuració de la xarxa

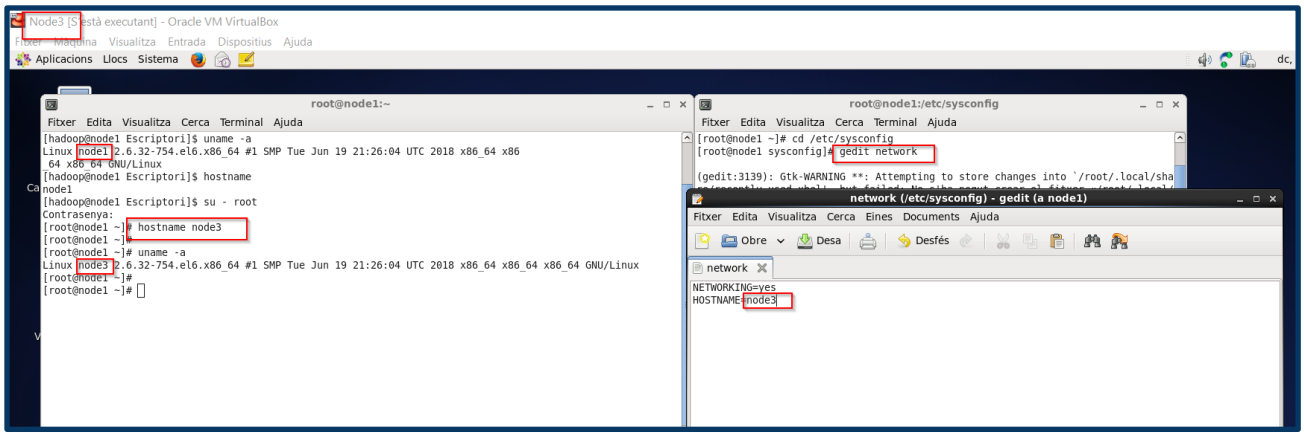
- Un cop ja tenim les màquines clonades, haurem de canviar el nom de les màquines associant-la a una IP estàtica (de la xarxa interna), ja que no disposem de servidor DNS

NODE 2

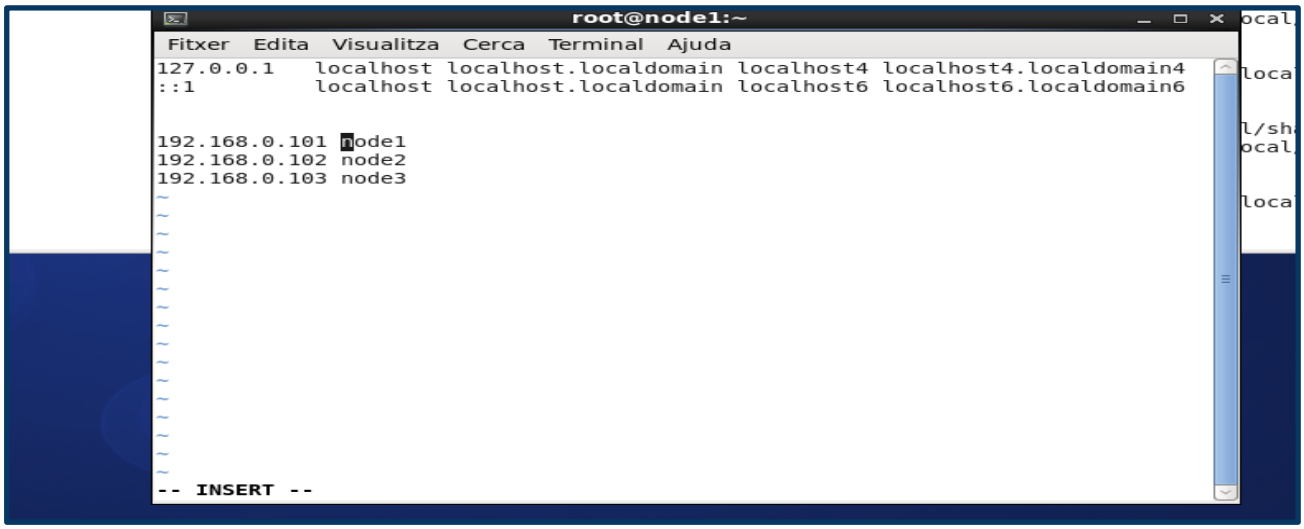
- [root@node1~]# hostname node2
- [root@node2~]# gedit /etc/sysconfig/network
- [root@node2~]# gedit /etc/hosts

NODE 3

- [root@node1~]# hostname node3
- [root@node3~]# gedit /etc/sysconfig/network
- [root@node1~]## gedit /etc/hosts

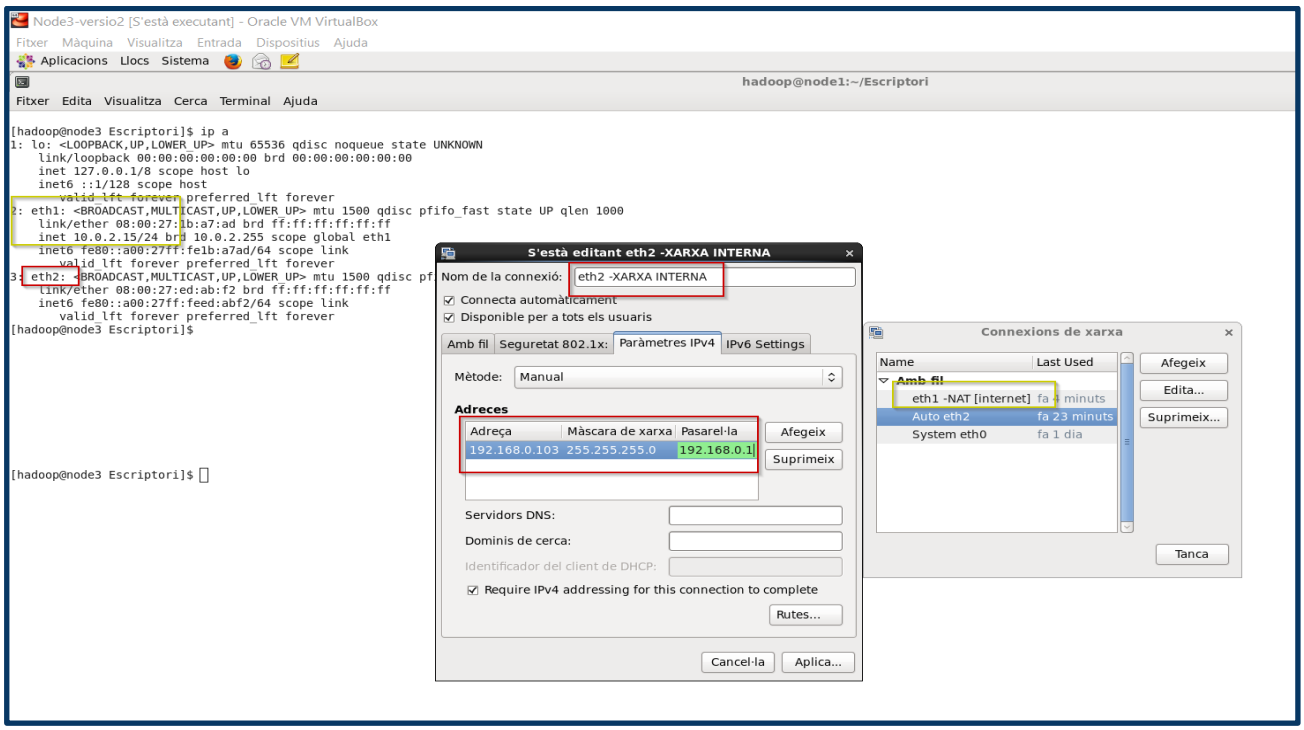


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



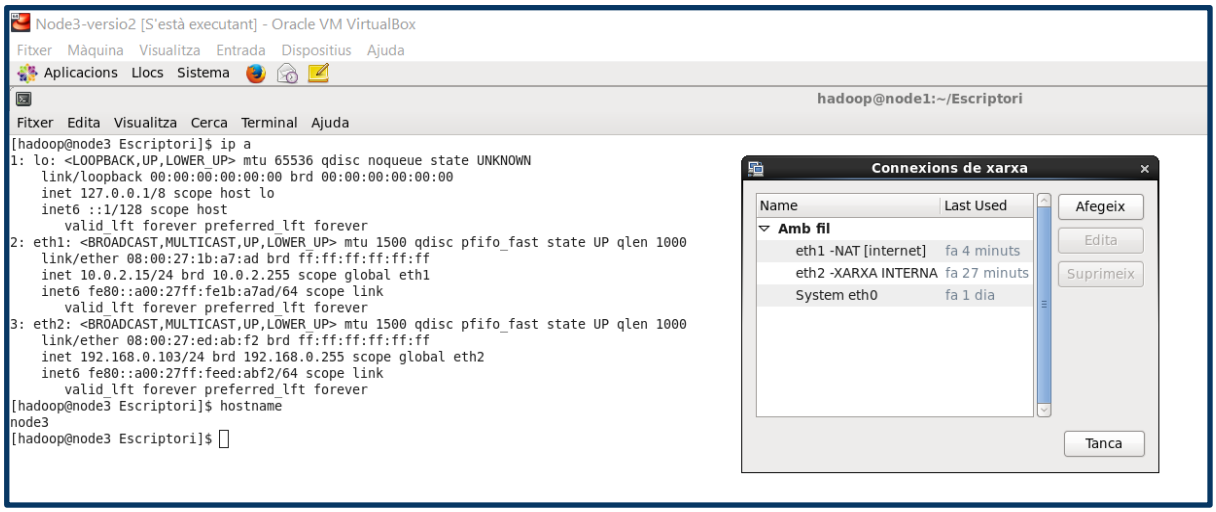
- ❑ Accedim a la interfície de Network Manager per afegir manualment les IP estàtiques . La IP de cada node serà la que anteriorment hem configurat a l'arxiu /etc/hosts

NODE 3

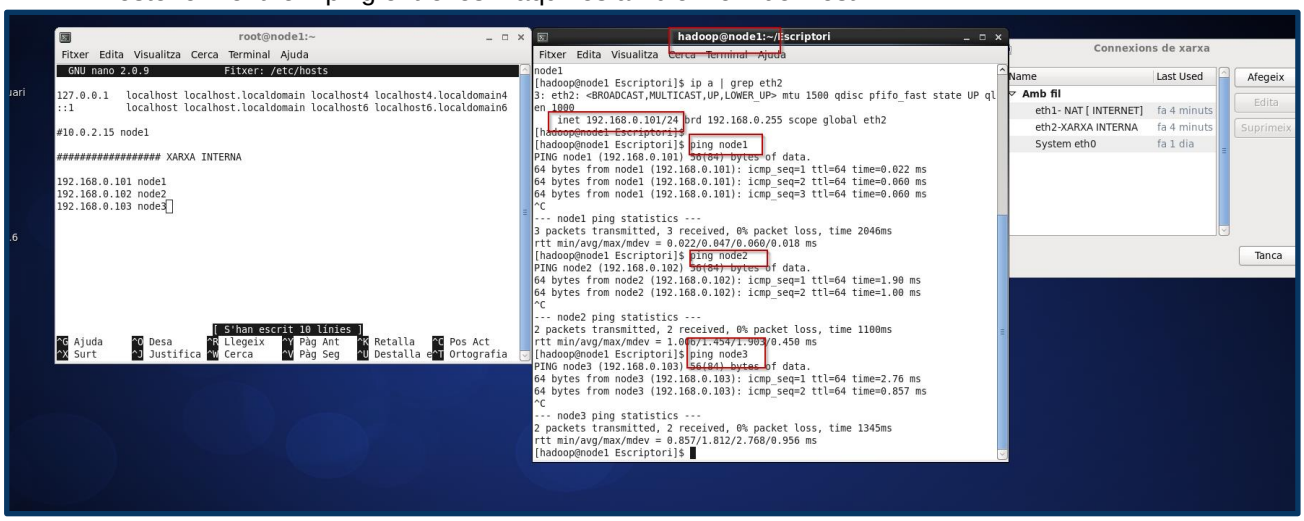


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

NODE 3



- ❑ Posteriorment fem ping entre les màquines amb el nom del host.



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

3.2.3. Configuració del SSH

- E n la carpeta oculta ~/.ssh eliminen tots els fitxers per evitar errors (ja que les màquines estan clonades) i ho tornem a generar pels 3 nodes.

NODE1

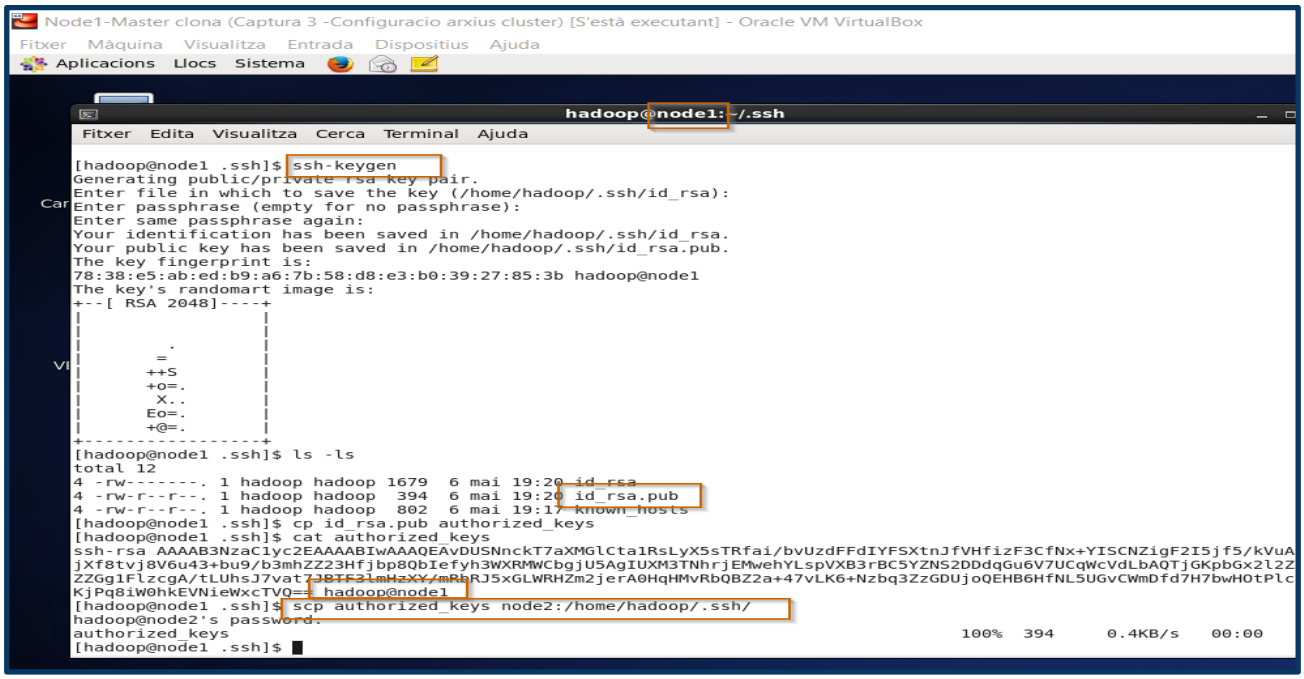
```
[hadoop@node1 .ssh]# rm *
[hadoop@node1 .ssh]# ssh-keygen
[hadoop@node1 .ssh]# cp id_rsa.pub authorized_keys
[hadoop@node1 .ssh]# cat authorized_keys
[hadoop@node1 .ssh]# scp [13]authorized_keys  node2:/hadoop/.ssh
```

NODE2

```
[hadoop@node2 .ssh]# rm *
[hadoop@node1 .ssh]# ssh-keygen
[hadoop@node1 .ssh]# cat id_rsa.pub  >> authorized_keys
[hadoop@node1 .ssh]# cat authorized_keys
[hadoop@node1 .ssh]# scp authorized_keys  node3:/hadoop/.ssh
```

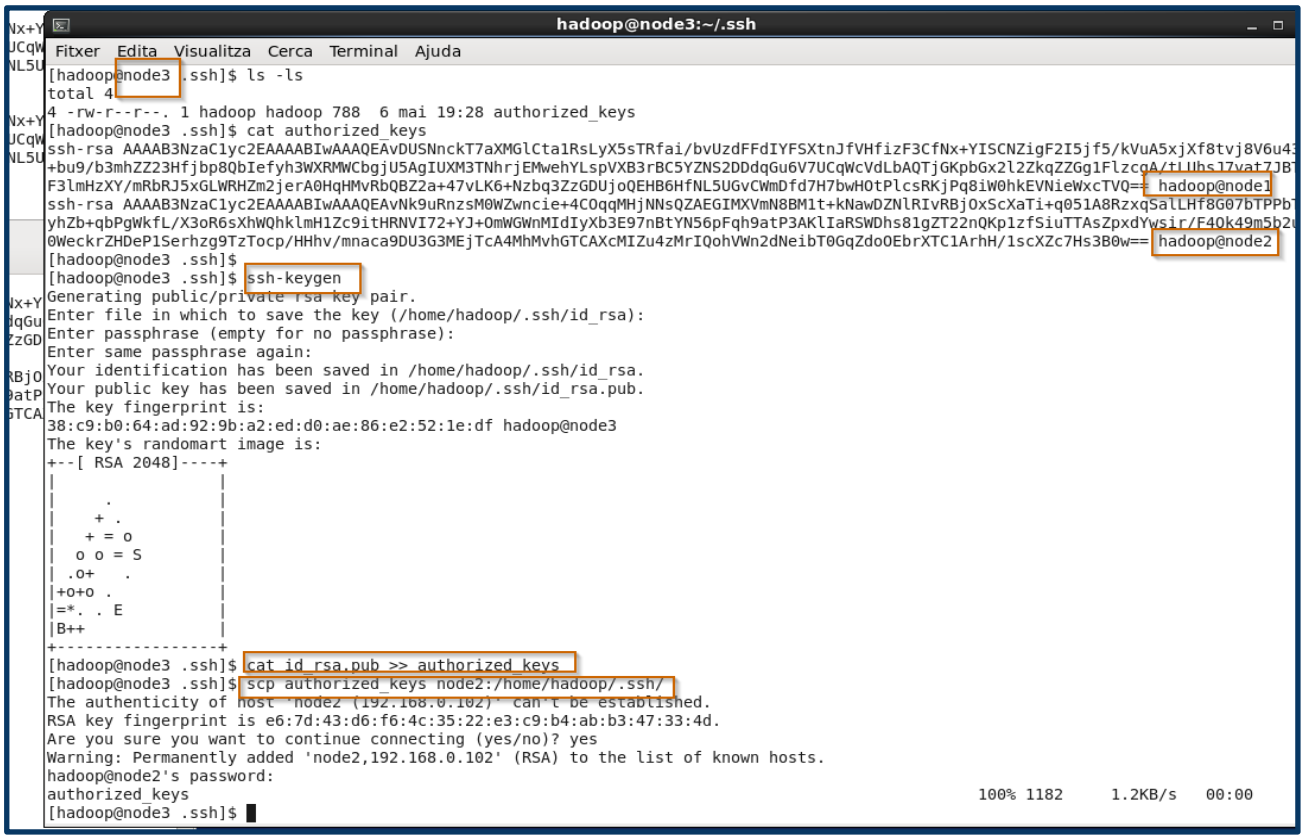
NODE3

```
[hadoop@node3 .ssh]# rm *
[hadoop@node3 .ssh]# ssh-keygen
[hadoop@node3 .ssh]# cat id_rsa.pub >> authorized_keys
[hadoop@node3 .ssh]# cat authorized_keys
[hadoop@node3 .ssh]# scp authorized_keys  node2:/hadoop/.ssh
[hadoop@node3 .ssh]# scp authorized_keys  node1:/hadoop/.ssh
```



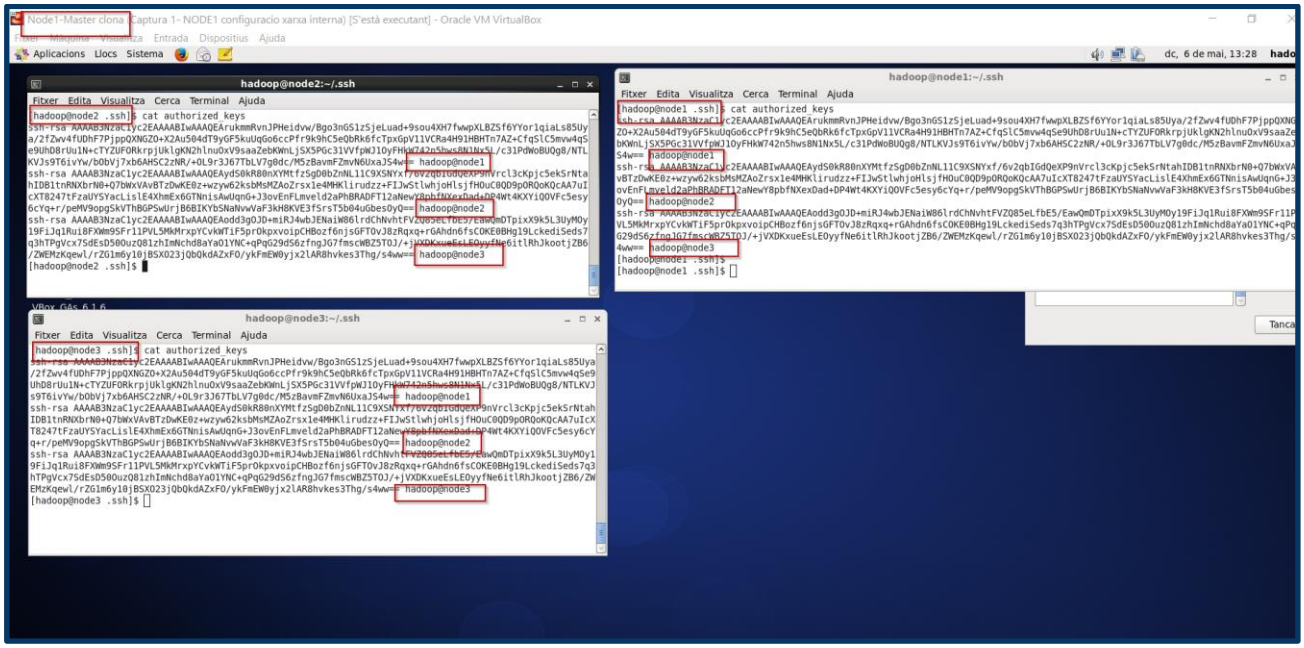
[13] **scp** el fem servir per fer una copia d'un fitxer utilitzant SSH

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



- Revisem en cada node tenim tots les claus públiques a l'arxiu "authorized_keys" i podem establir connexió via SSH entre nodes sense password.

Nota: en aquest casnomés, només està habilitat per l'usuari "hadoop", els altres usuaris necessitarien el password.



3.2.4. Configuració del clúster

❑ Abans de començar la configuració del clúster, hem de tenir en compte:

- El mateix usuari en tots els nodes del clúster (en el nostre cas, no suposarà cap problema, ja que les màquines estan clonades i fan servir el mateix usuari "hadoop")
- Els 3 nodes han de ser d'accessibles via SSH sense contrasenya
- Tenir els mateixos directoris de Hadoop en cada node
- Tenir copiat el software de Hadoop en el mateix directori.
- Tenir creat el directori de dades en tots els nodes i els seus permisos
  - /dades/datanode

(no és obligatori però és recomanable, els servidors haurien de ser clons en l'àmbit de Hadoop no només en l'àmbit de software sinó també en l'àmbit de hardware per un millor rendiment de les màquines)

❑ Haurem de modificar el directori /dades, ja que anteriorment el node1 feia les funciones de "master" i "slave"

HOST (master) : node1

- En el directori /dades hi ha els directoris:
  - **namenode**[14]
  - **datanode** : Eliminem el directori i els arxius que hi hagi, ja que només pot estar en el node slave
    - [hadoop@node1 dades]# rm -rf datanode

HOST (slave) : node2

- En el directori /dades hi ha els directoris:
  - **namenode** : Eliminem el directori i els arxius que hi hagi, ja que només pot estar en el node master
    - [hadoop@node2 dades]# rm -rf namenode
  - **datanode** : En el datanode, com que és una còpia, eliminarem el directori current (faria referencia al node1)
    - [hadoop@node2 datanode]# rm -rf current

HOST (slave) : node3

- En el directori /dades hi ha els directoris:
  - **namenode** : Eliminem el directori i els arxius que hi hagi, ja que només pot estar en el node master
    - [hadoop@node3 dades]# rm -rf namenode
  - **datanode** : En el datanode, com que és una còpia, eliminarem el directori current (faria referencia al node1)
    - [hadoop@node3 datanode]# rm -rf current

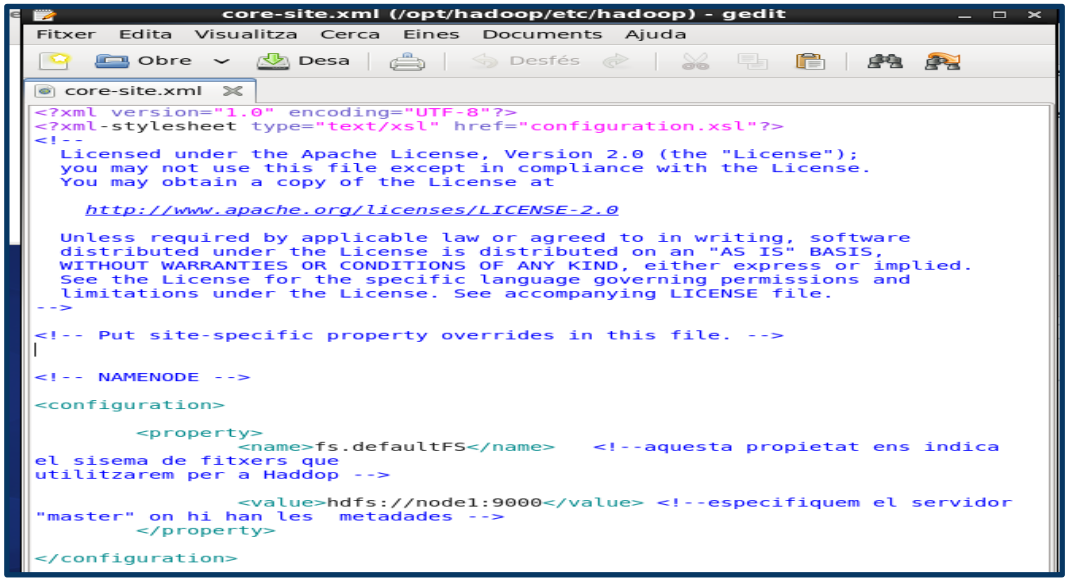
❑ Haurem d'accedir al directori /opt/hadoop/etc/hadoop per modificar els arxius de configuració.

[14] També podríem eliminar el contingut del namenode, però com que abans d'iniciar el clúster haurem de fer un format ja ho esborrarà.

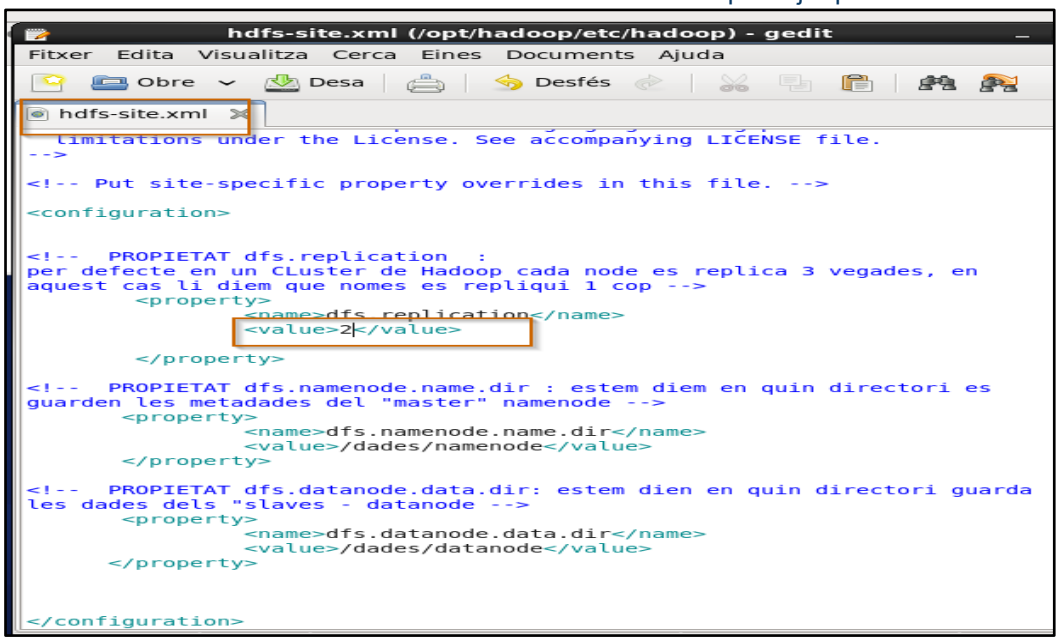
- core-site.xml
- hdfs-site.xml
- mapred-site.xml
- yarn-site.xml

HOST (master) : node1

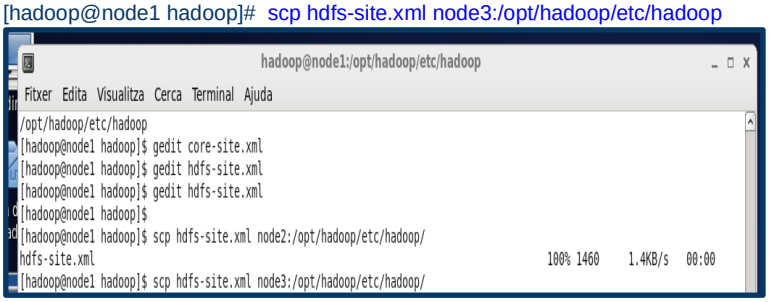
- l'arxiu core-site.xml (no el modifiquem)



- l'arxiu hdfs-site.xml s'haurà de modificar el número de rèplica ja que tindrem 2 "slaves"



- Un cop modificat l'arxiu hdfs-site.xml l'enviarem als altres nodes utilitzant el comando scp[15]
  - [hadoop@node1 hadoop]# scp hdfs-site.xml node2:/opt/hadoop/etc/hadoop
  - [hadoop@node1 hadoop]# scp hdfs-site.xml node3:/opt/hadoop/etc/hadoop



- l'arxiu mapred-site.xml (no el modifiquem)

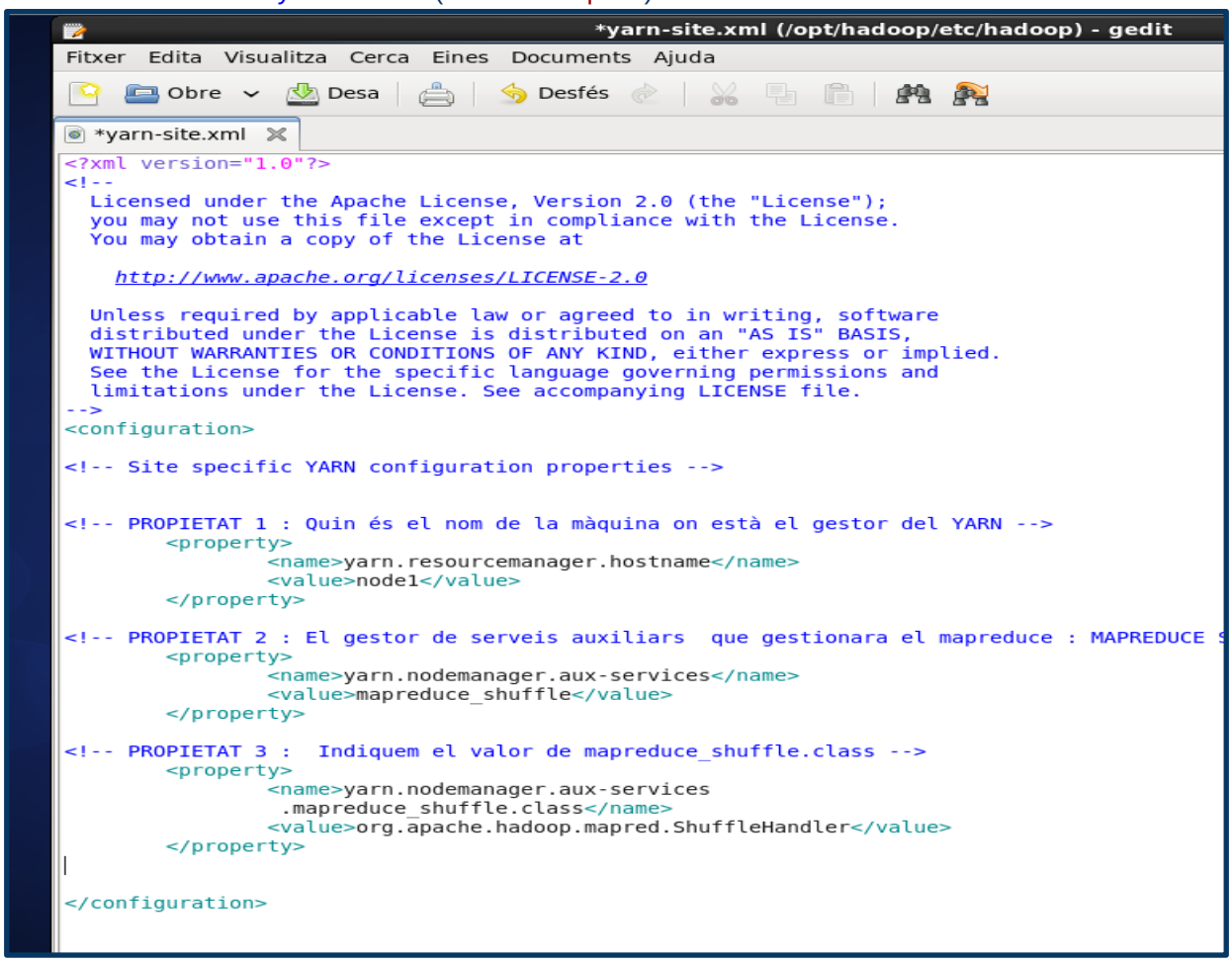
mapred-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!--
  Licensed under the Apache License, Version 2.0 (the "License");
  you may not use this file except in compliance with the License.
  You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License. See accompanying LICENSE file.
-->

<!-- Put site-specific property overrides in this file. -->

<configuration>


<!-- PROPIETAT mapreduce.framework.name li indiquem el motor sera de tipus
YARN -->
        <property>
                <name>mapreduce.framework.name</name>
                <value>yarn</value>
        </property>

</configuration>
```

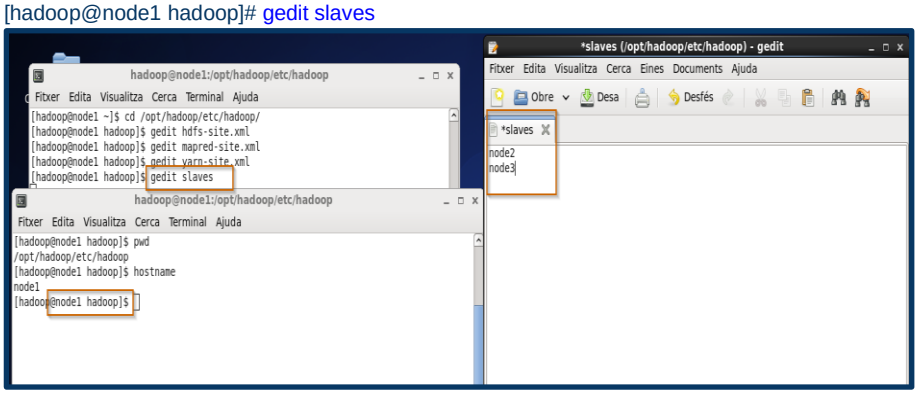
[15] Farem una còpia de l'arxiu via SSH

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- l'arxiu yarn-site.xml (no el modifiquem)

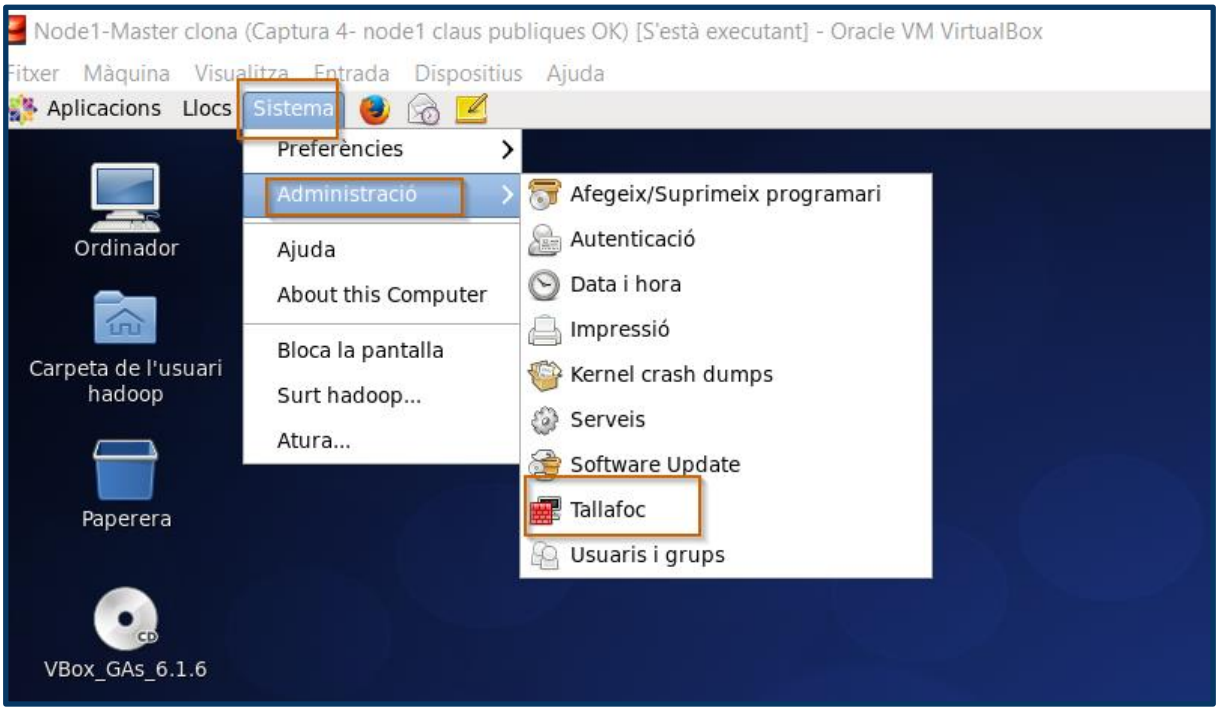


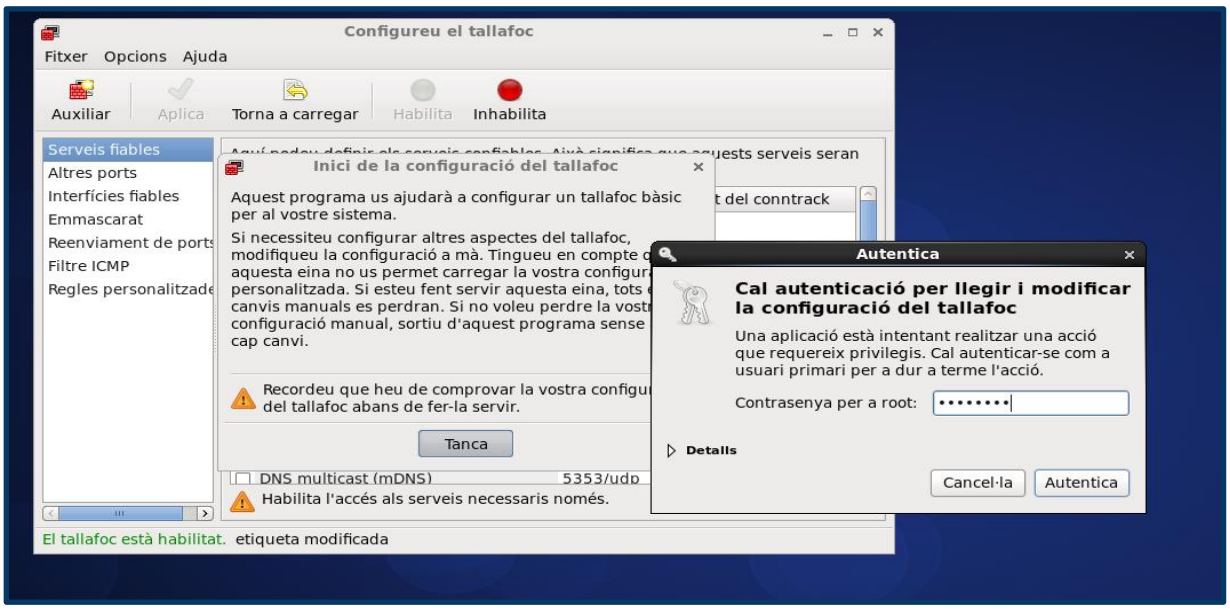
- Haurem de modificar l'arxiu "slaves" que per defecte només consta localhost
  - [hadoop@node1 hadoop]# gedit slaves



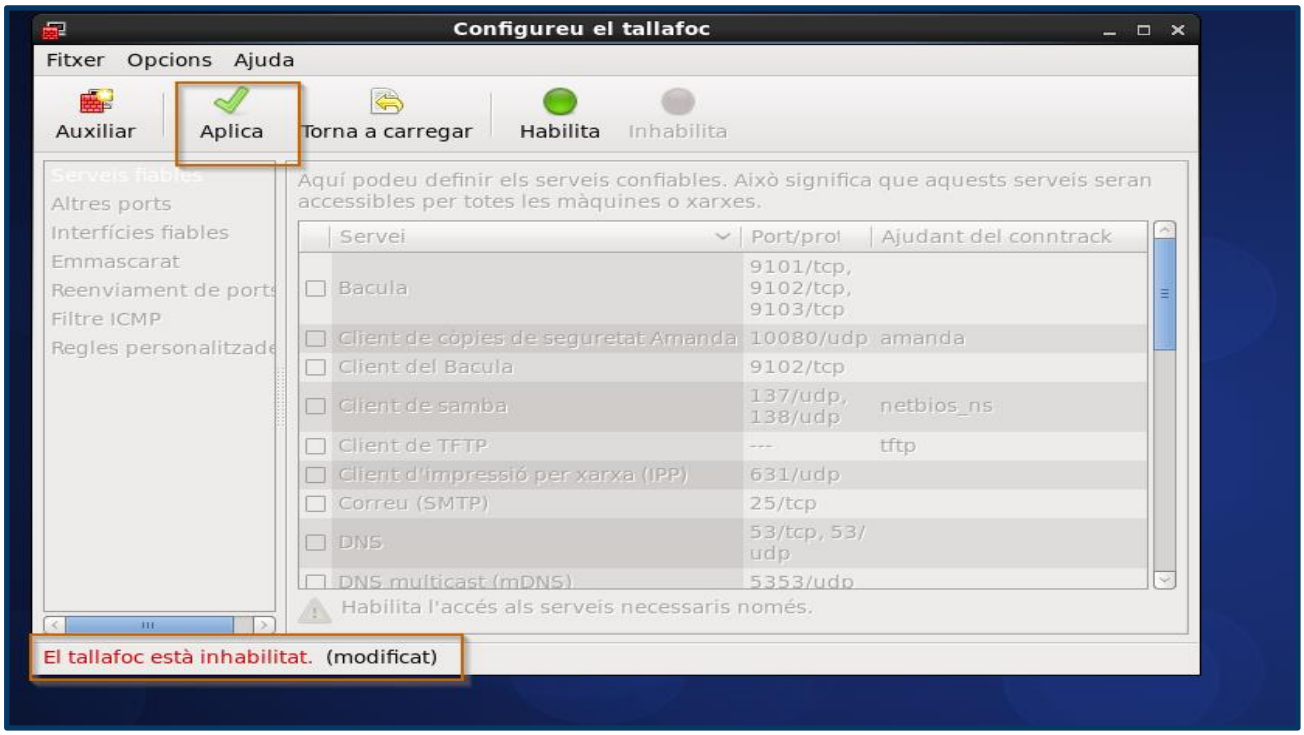
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Hem de tenir en compte que el sistema operatiu CentOS per defecte té el firewall actiu, i només permet les connexions al port 22 (SSH). Desactivarem els firewall per aquestes pràctiques tot i que l'aconsellable seria crear regles en les iptables obrint els ports que ens interessin.

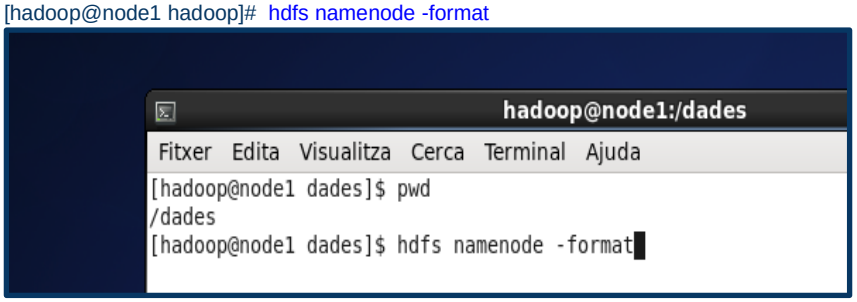


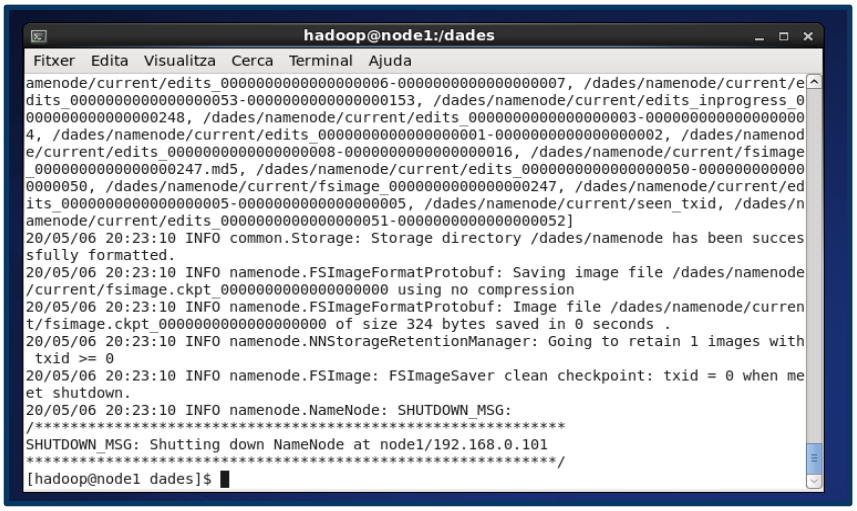


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



- ❑ Formatejar el **namenode**[16]
  - [hadoop@node1 hadoop]# hdfs namenode -format



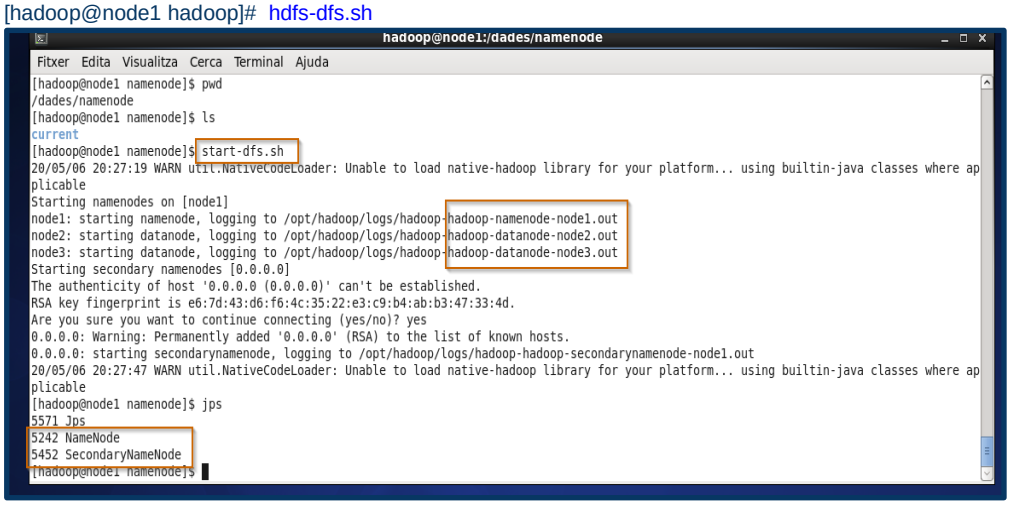


[16]Està en el node mater (node1)

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

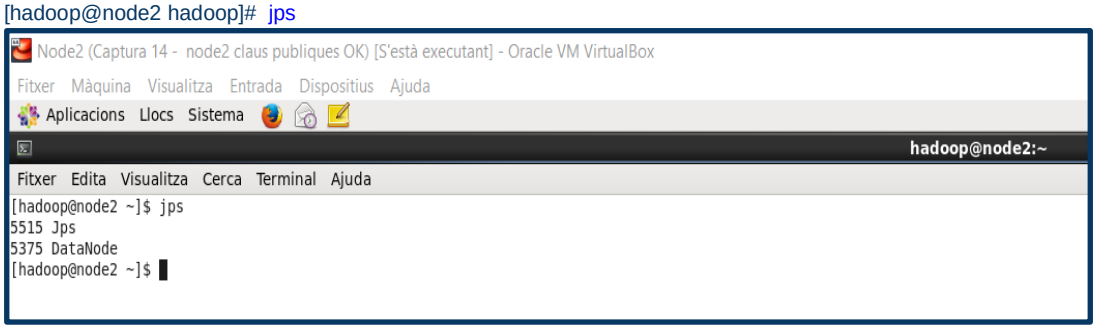
- Arrencar el **Namenode**

[hadoop@node1 hadoop]# hdfs-dfs.sh

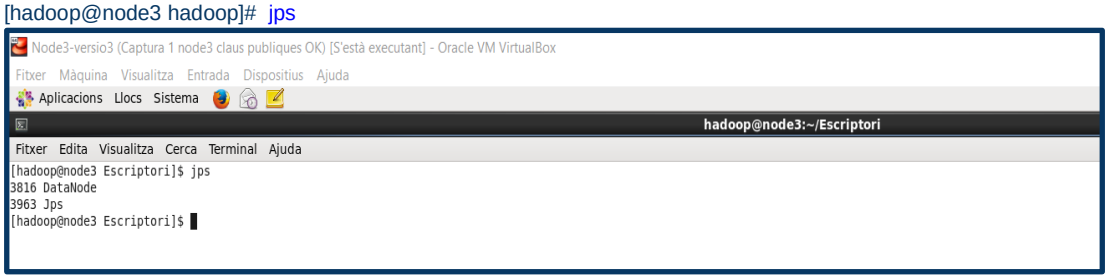


- Confirmem utilitzant el comando jps que tenim els processos en els nodes "slaves"

[hadoop@node2 hadoop]# jps



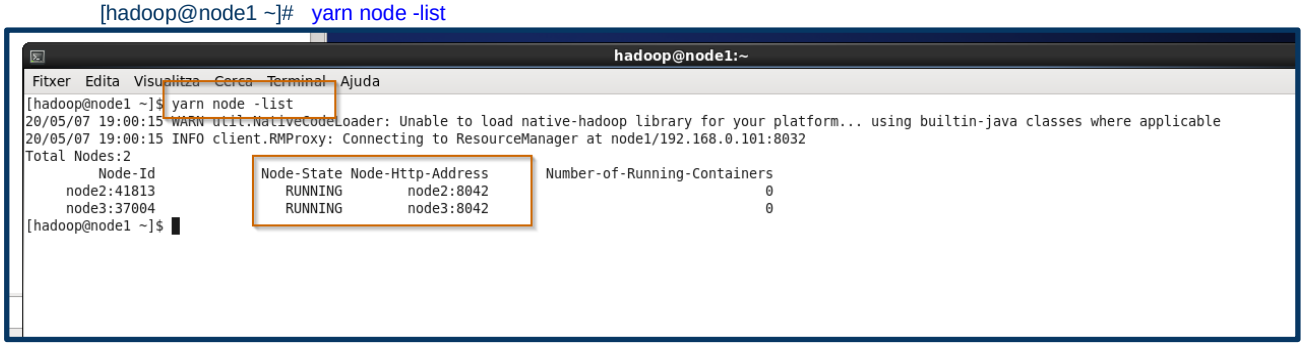
[hadoop@node3 hadoop]# jps

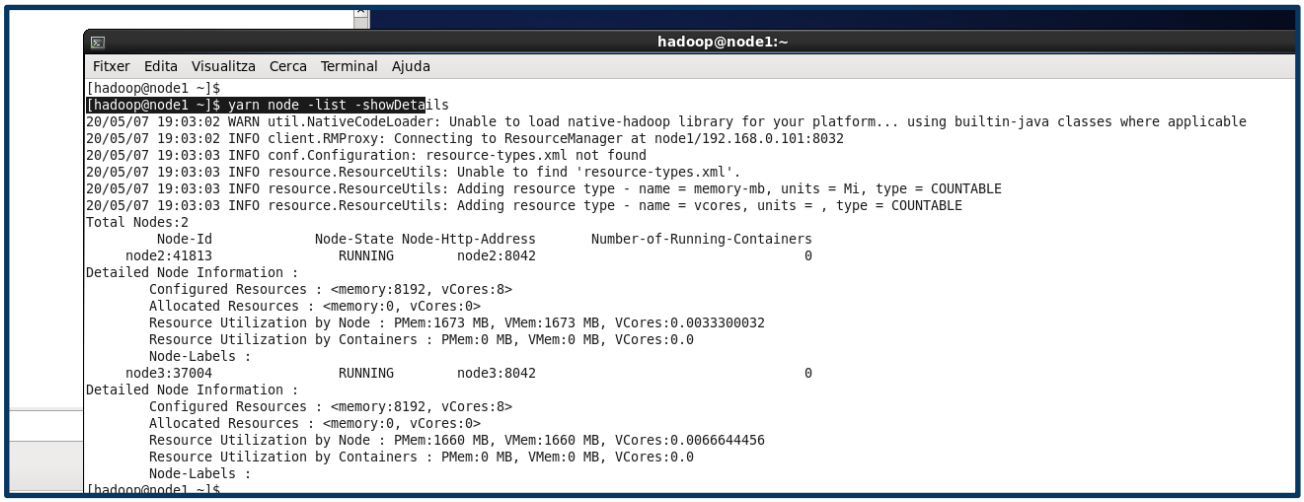


- Hi han comandes de "yarn" molt útils Per exemple saber tots els nodes del clúster

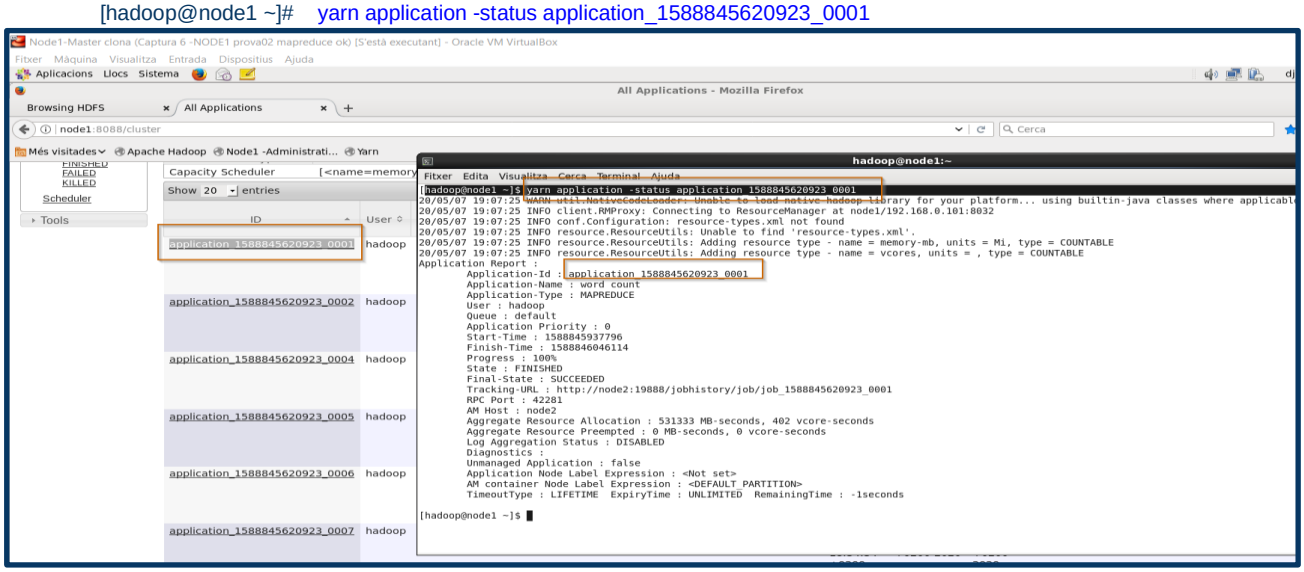
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

[hadoop@node1 ~]# yarn node -list



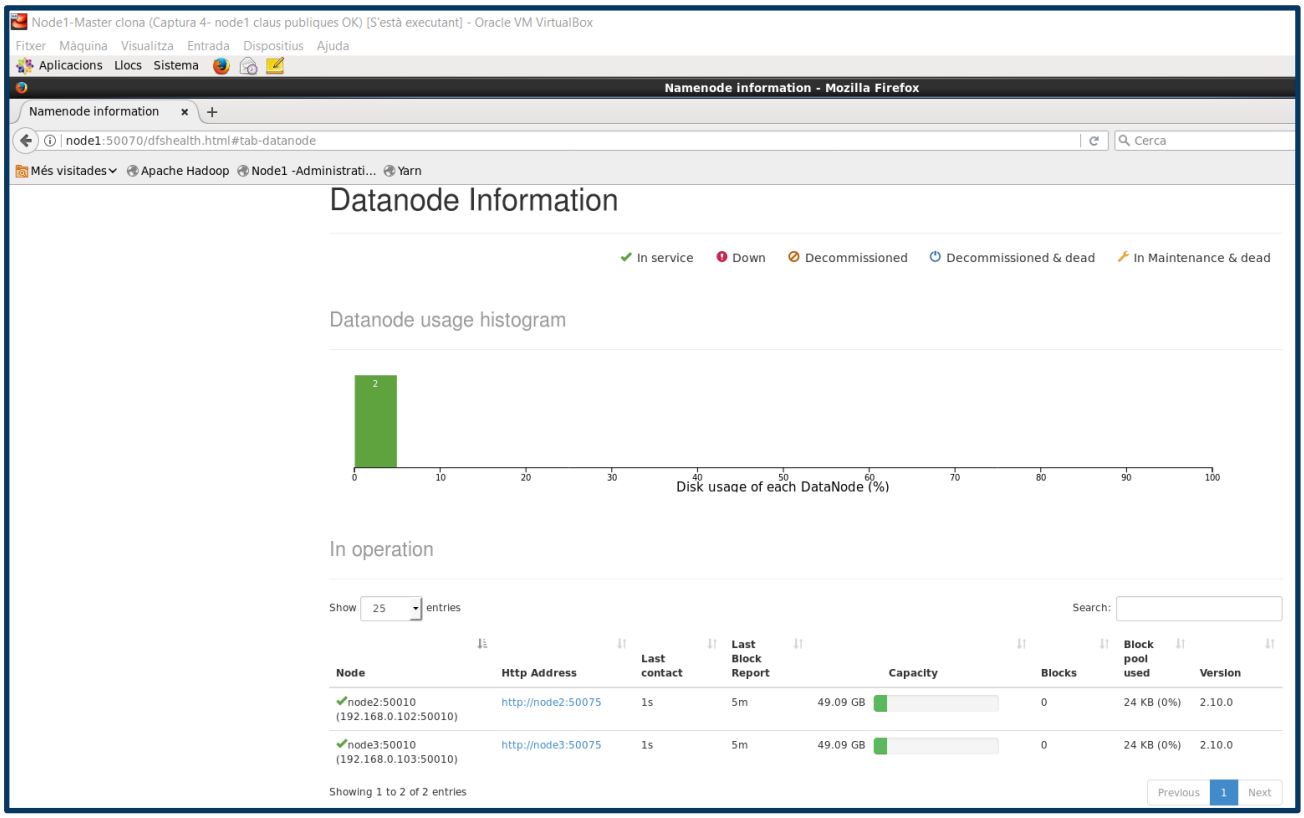


[hadoop@node1 ~]# yarn application -status application_1588845620923_0001



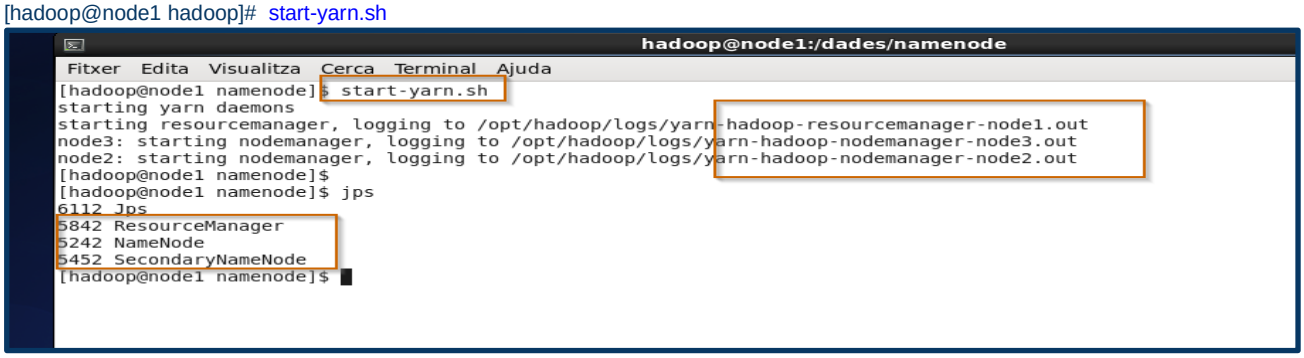
- Desde el node1 accedim a l'interfície web del Namenode (port 50070) i visualitzem la informació dels Datanodes.

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



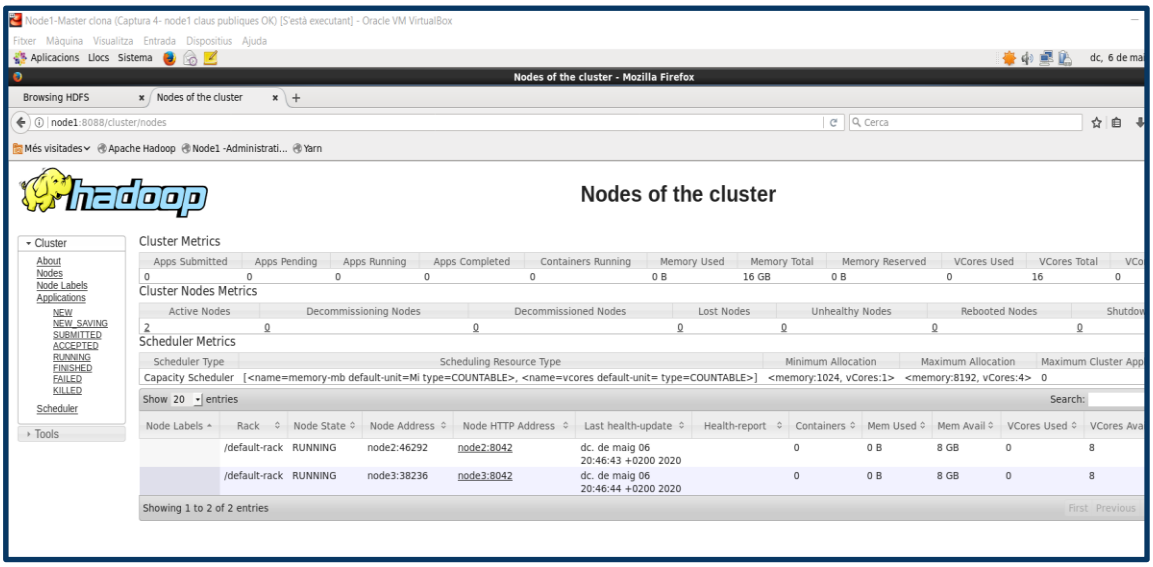
❑ Arrencar el **Yarn**

[hadoop@node1 hadoop]# start-yarn.sh



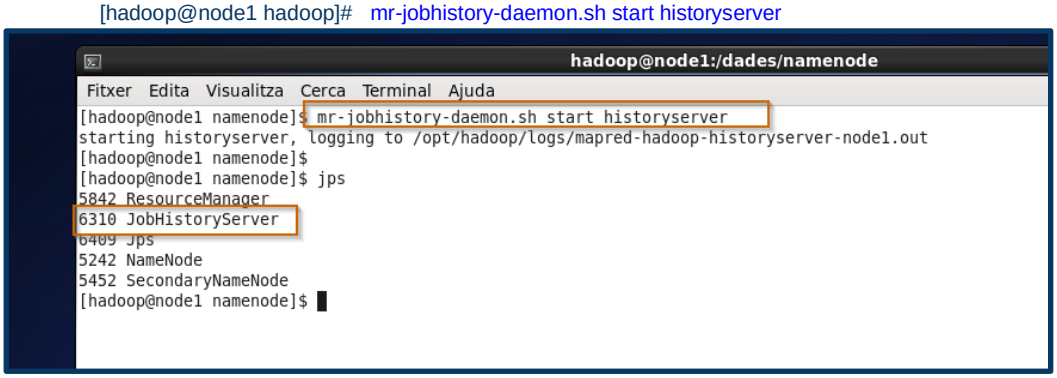
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Desde el node1 accedim a l'interfície web del Yarn (port 8088) i visualitzem la informació dels Datanodes.



- Abans de fer algun exemple d'un procés MapReduce contra el clúster haurem d'activar un procés per poder veure l'històric dels processos de cada aplicació executada

[hadoop@node1 hadoop]# mr-jobhistory-daemon.sh start historyserver

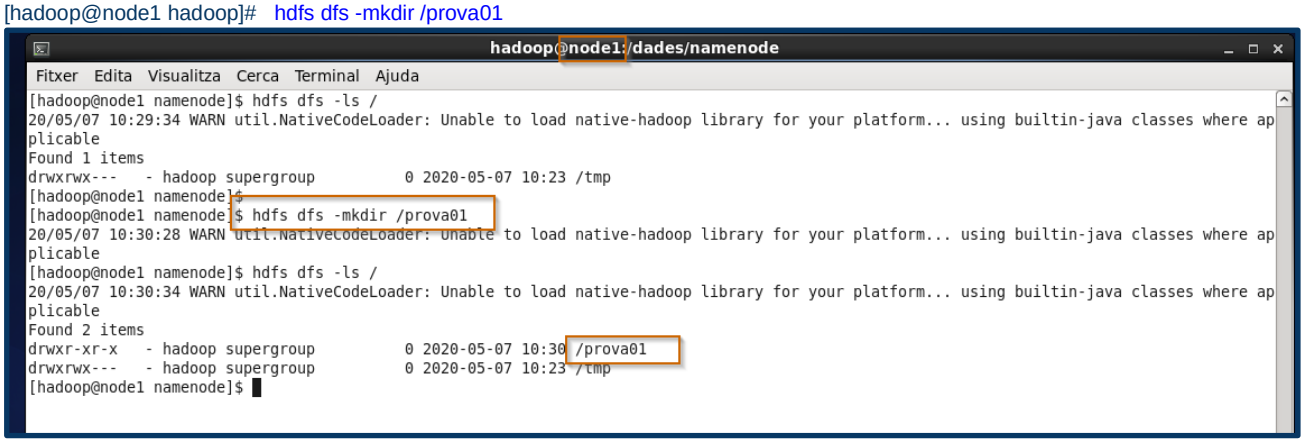


3.2.5. Exemples Mapreduce

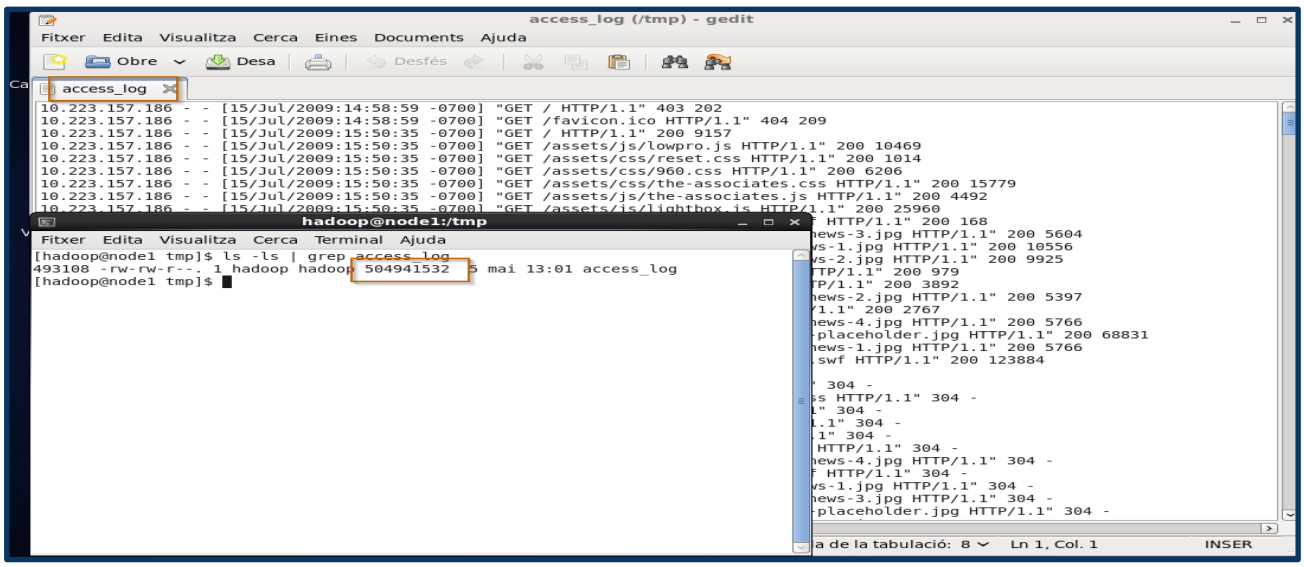
3.2.5.1. Exemple 1 - Llençar un procés Mapreduce contra el clúster

- Desde el node "master" crearem un directori HDFS anomenat "prova01"

[hadoop@node1 hadoop]# hdfs dfs -mkdir /prova01

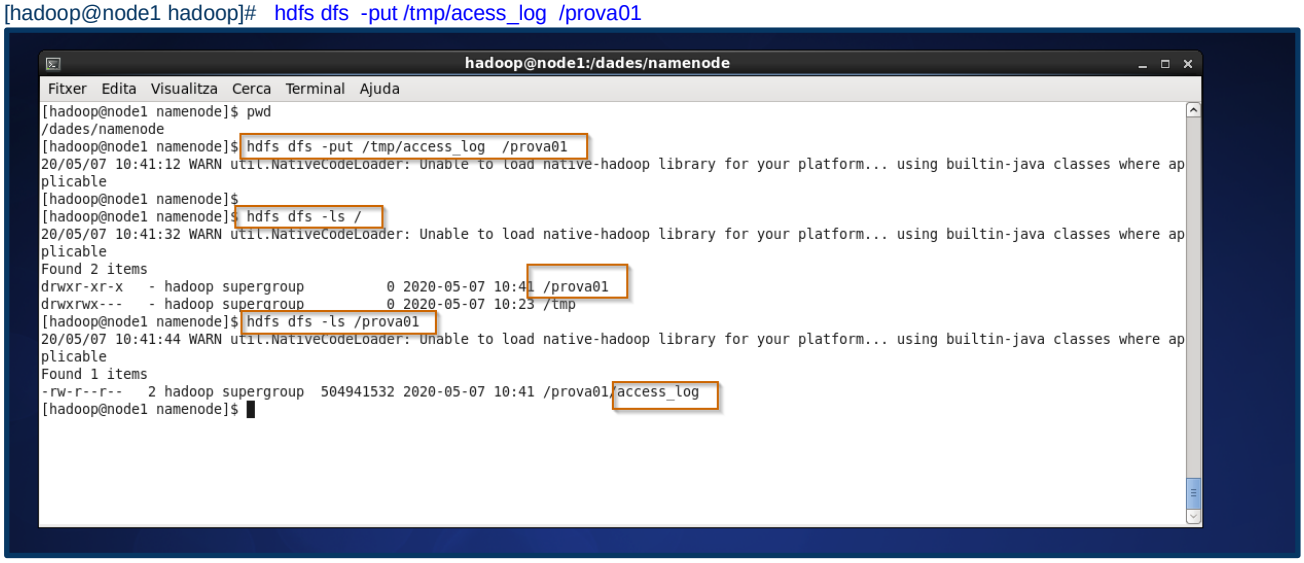


- Ens hem descarregat un fitxer d'exemple de logs de 504 MB i el dipositem a la carpeta /tmp.Posteriorment el pugem a HDFS

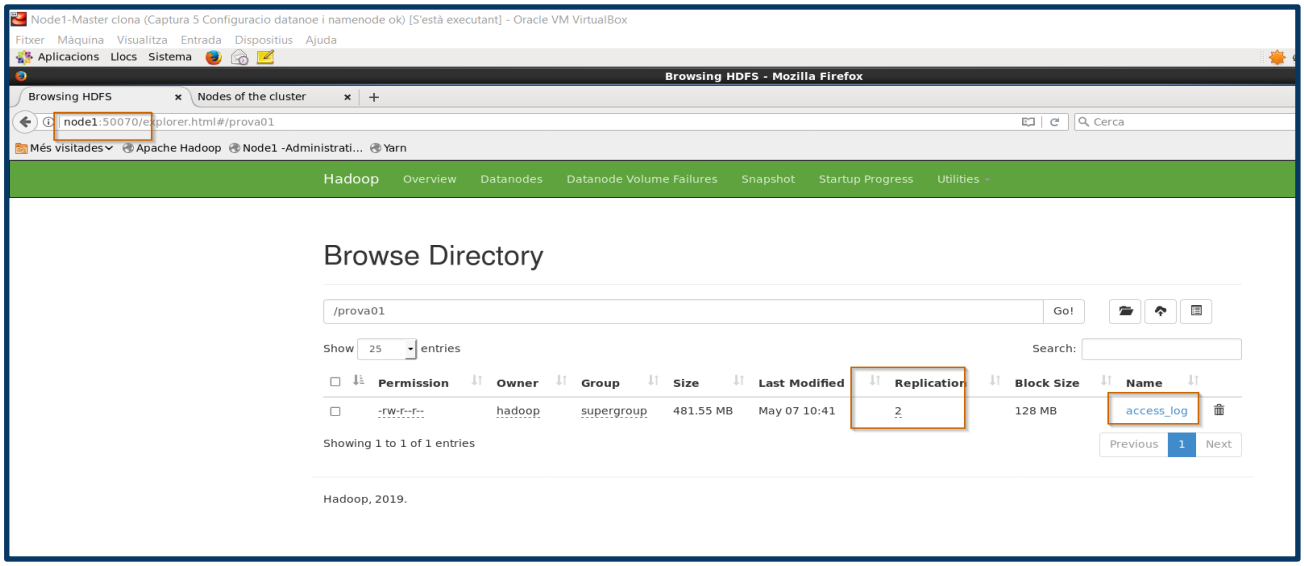


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

[hadoop@node1 hadoop]# hdfs dfs -put /tmp/acess_log /prova01

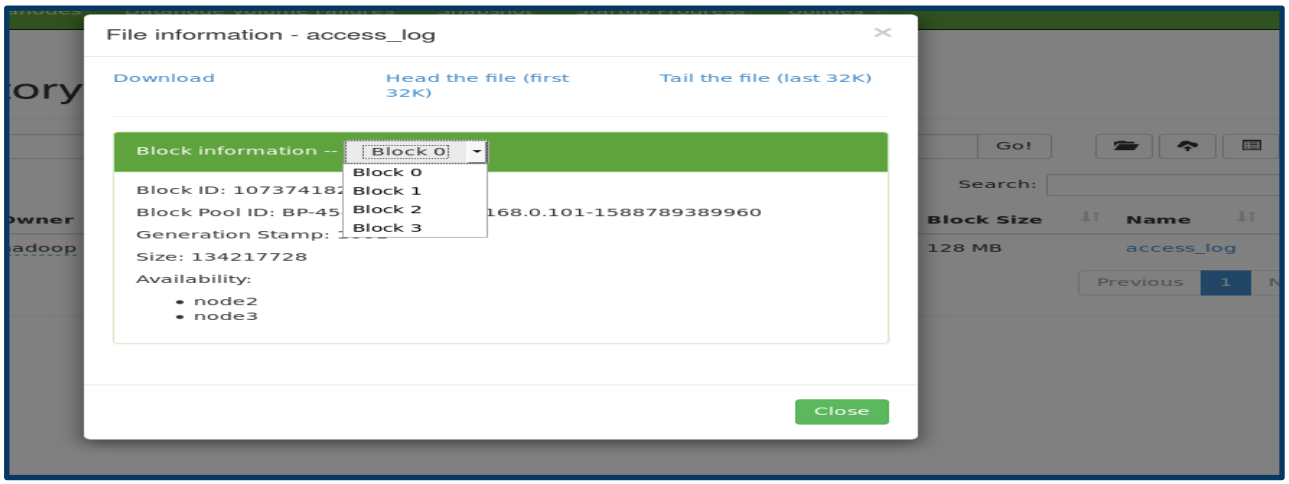


- Accedim a la interfície del Namenode i anem a la capeta a on hem pujat el fitxer. Podrem observar que el número de replicació és 2



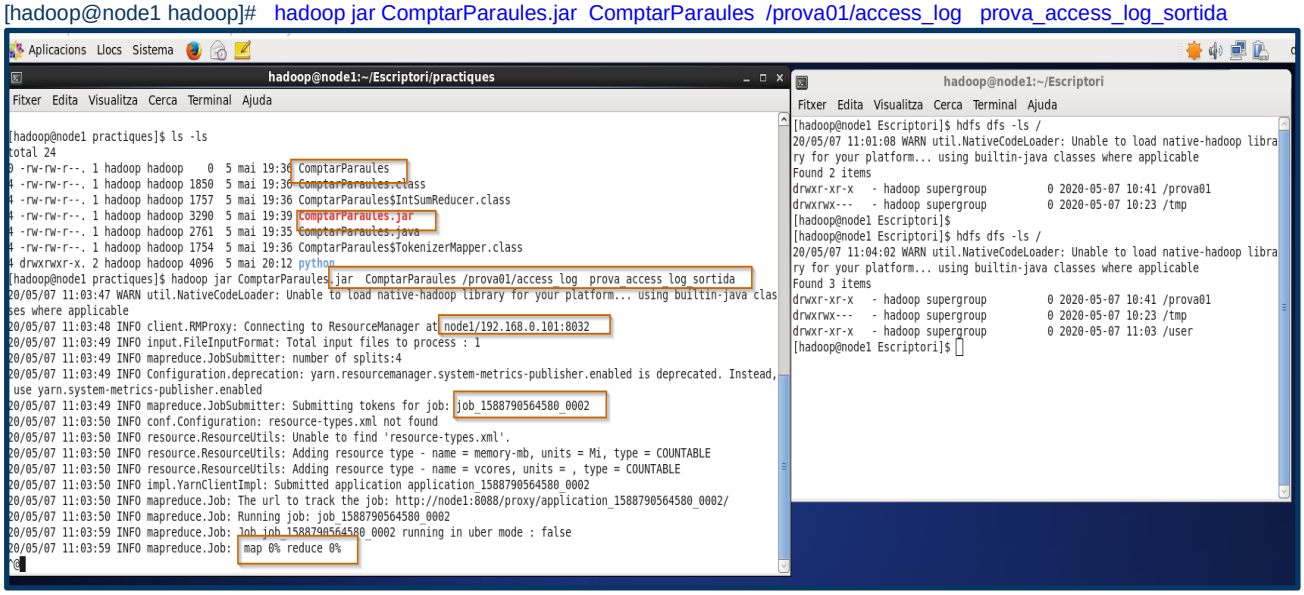
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Podrem observar que s'han creat 4 blocks, ja que la mida de l'arxiu és superior a 128 MB

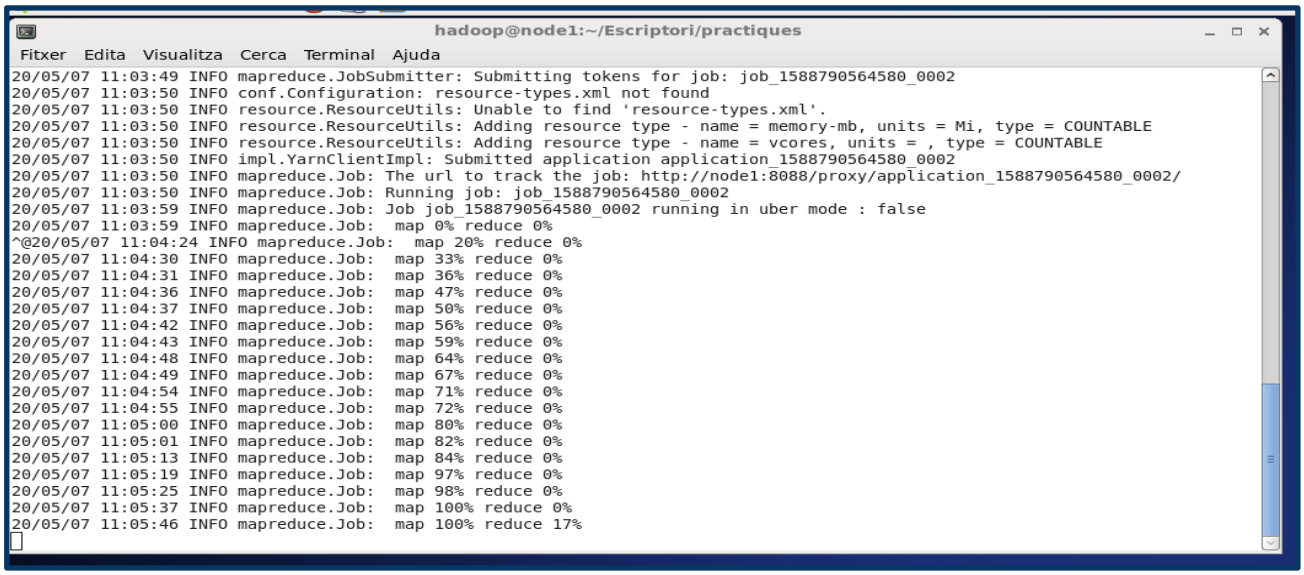


- Executem l'aplicació Java que vàrem utilitzar anteriorment "ComptarParaules" (en el clúster pseudodistribuït)

[hadoop@node1 hadoop]# hadoop jar ComptarParaules.jar ComptarParaules /prova01/access_log prova_access_log_sortida

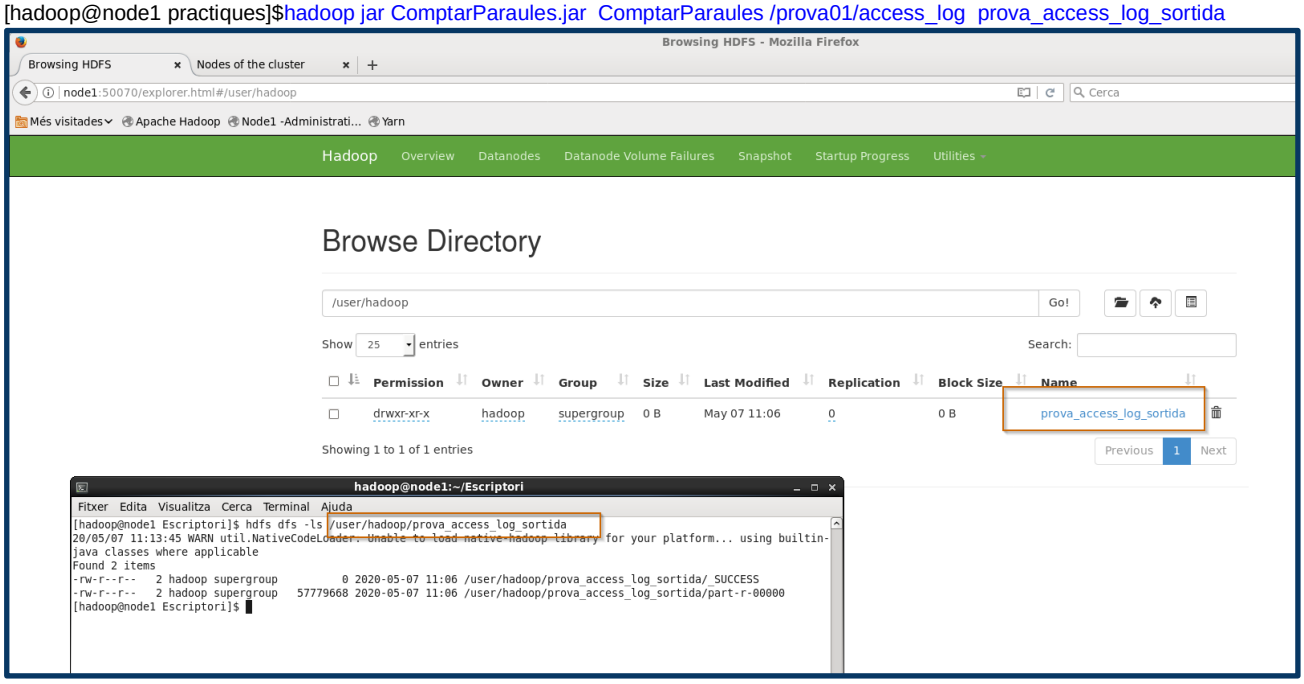


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



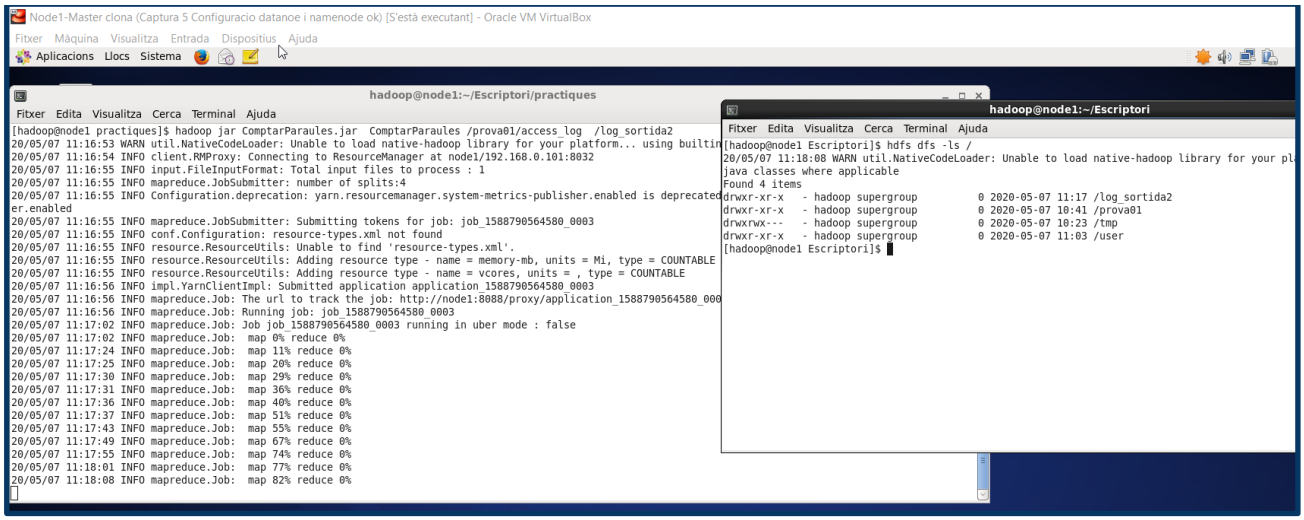
NOTA: com que en la comanda m'he oblidat l'arrel en el directori de sortida en la */*prova_access_log_sortida, es crearà a /users/haddop/prova_acces_log_sortida

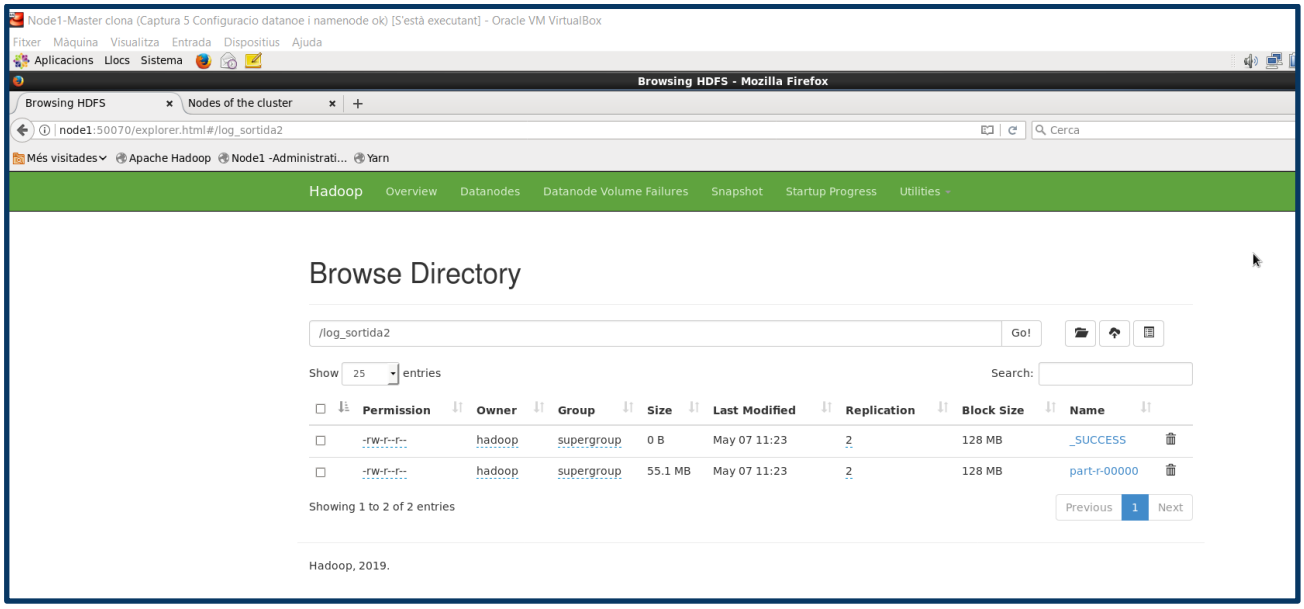
[hadoop@node1 practiques]$hadoop jar ComptarParaules.jar ComptarParaules /prova01/access_log prova_access_log_sortida



- Tornem a executar l'aplicació canviant el directori de sortida, li direm "log_sortida2"

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

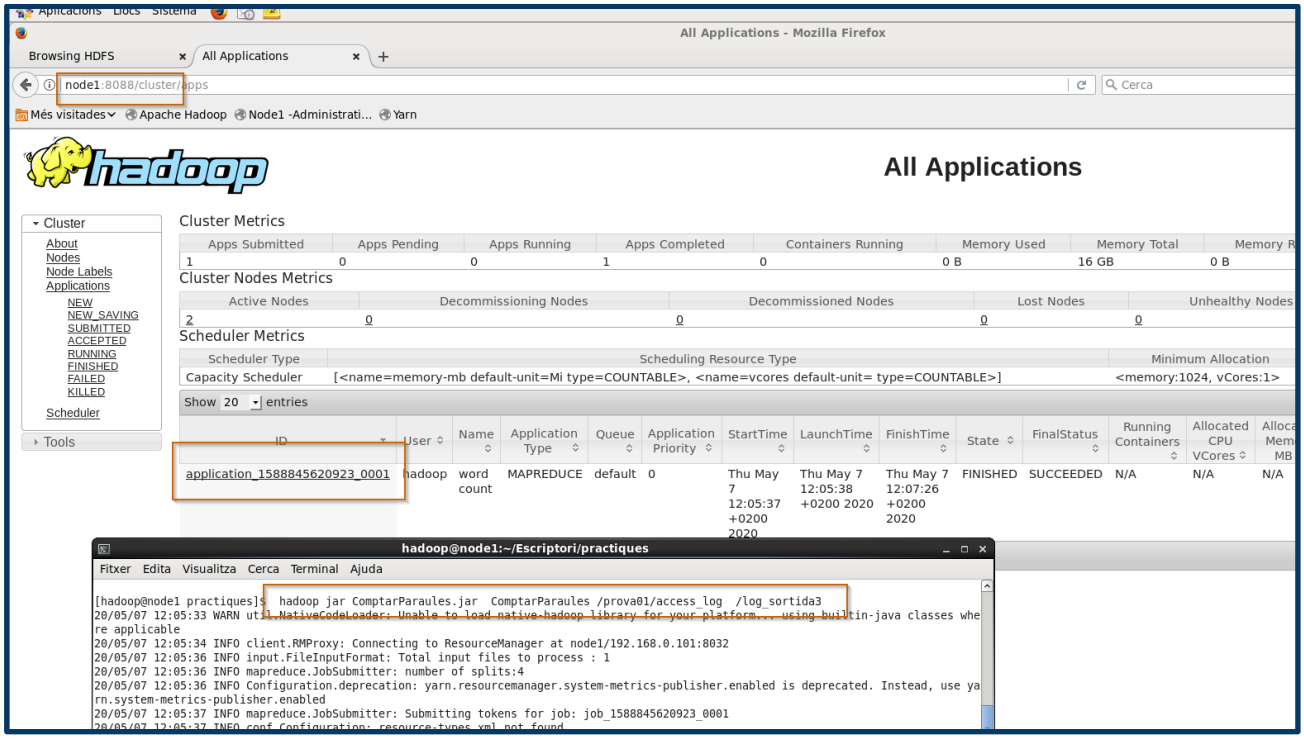


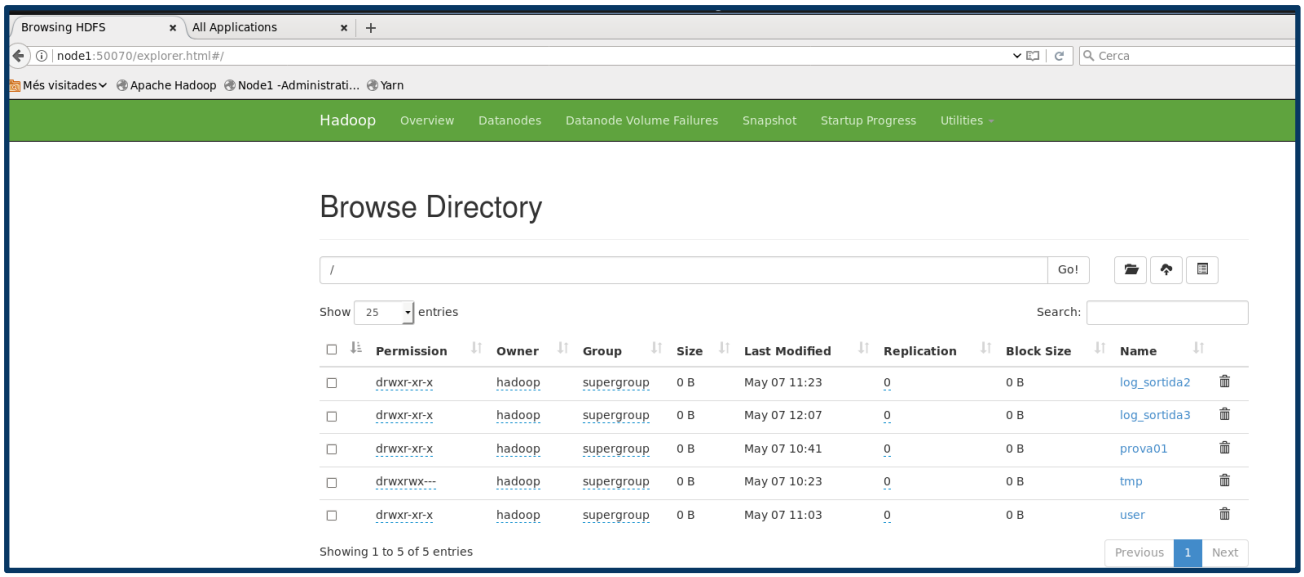


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

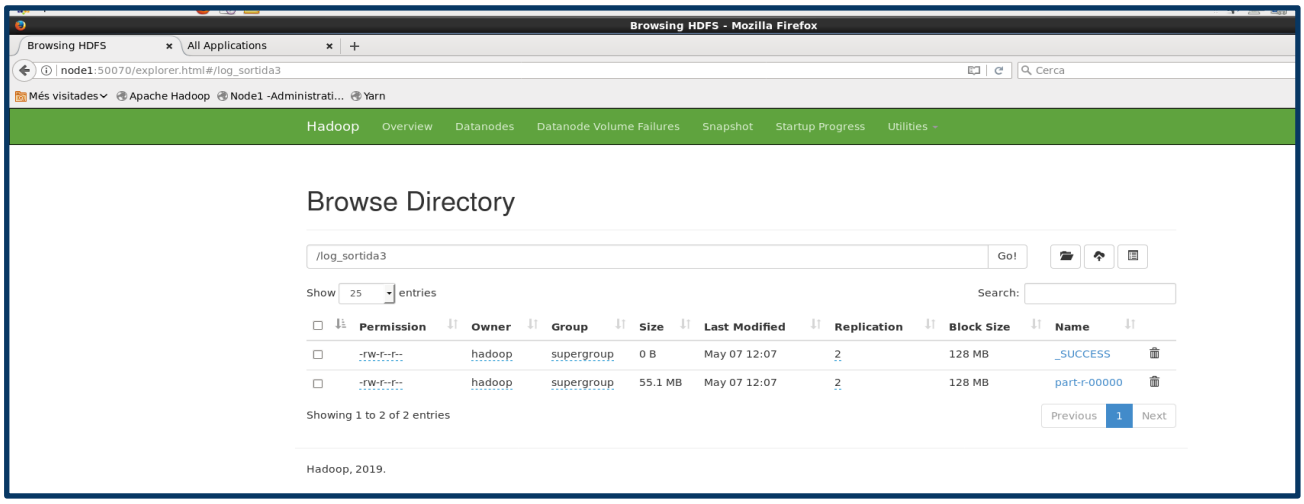
- Tornem a executar l'aplicació per comptar paraules canviant el directori de destí, li direm /log_sortida3, ja que l'ordinador s'ha bloquejat i s'ha hagut de reiniciar parant-se les màquines virtuals (i els processos que estaven actius). Hem hagut d'iniciar novament els serveis.

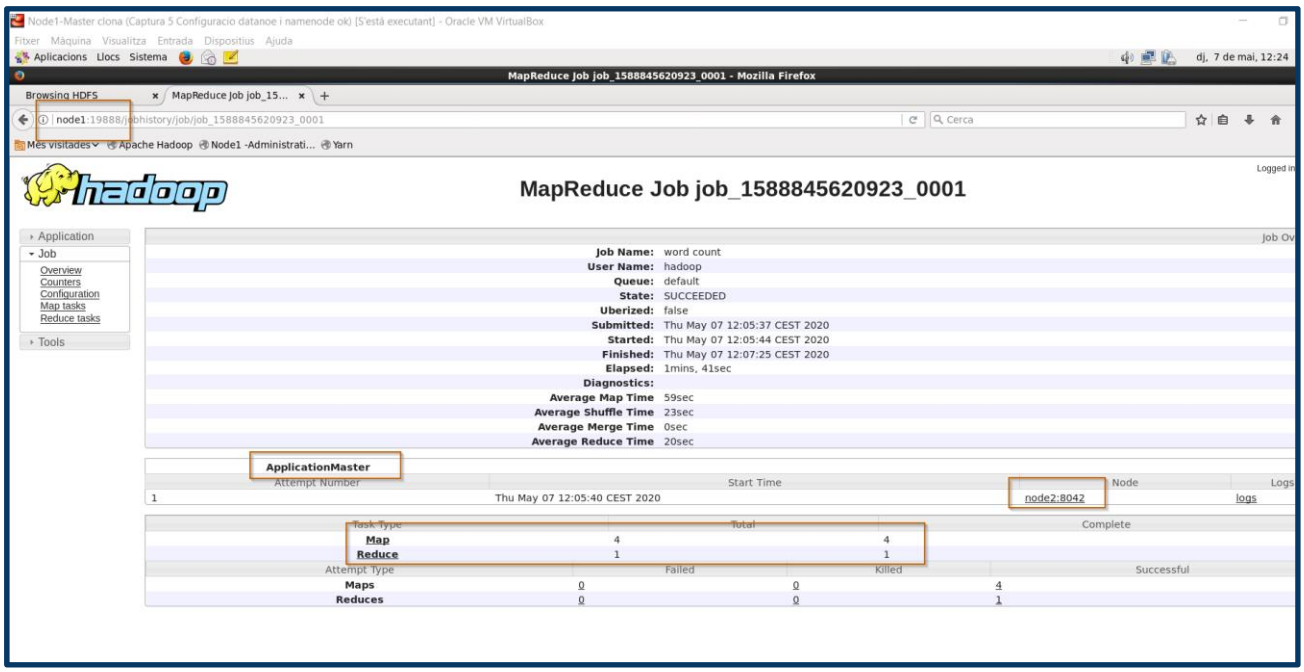
```
[hadoop@node1 practiques]$ start-dfs.sh
[hadoop@node1 practiques]$ start-yarn.sh
[hadoop@node1 practiques]$ mr-jobhistory-daemon.sh start historyserver
[hadoop@node1 practiques]$ hadoop jar ComptarParaules.jar  ComptarParaules /prova01/access_log  /log_sortida3
```





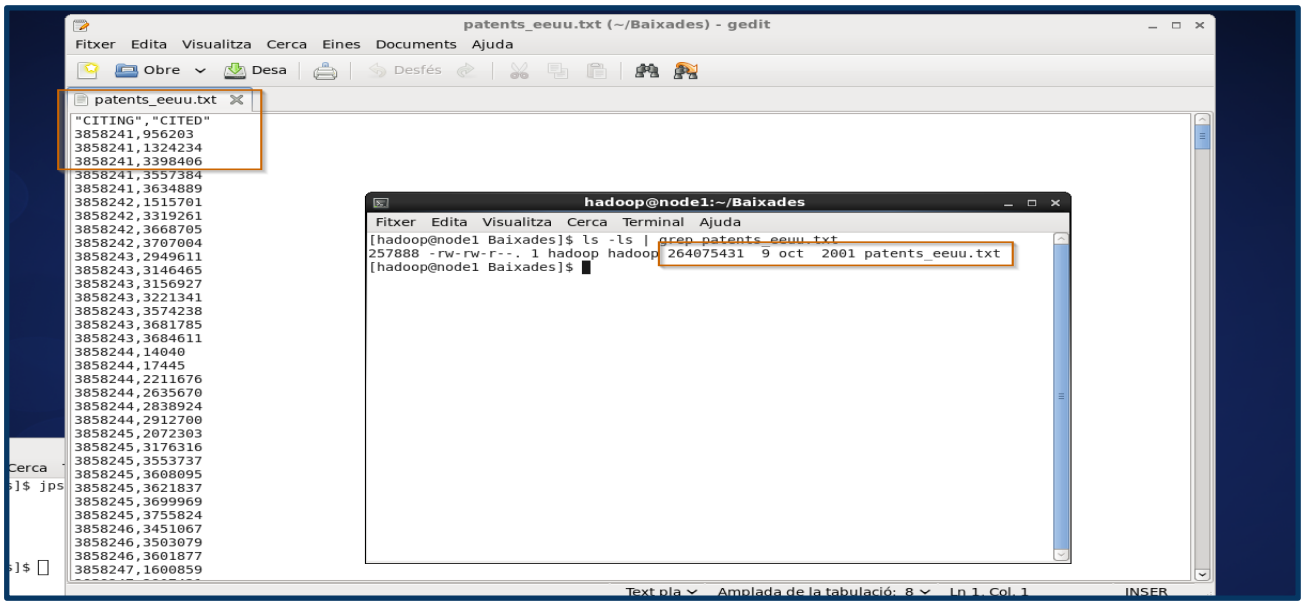
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





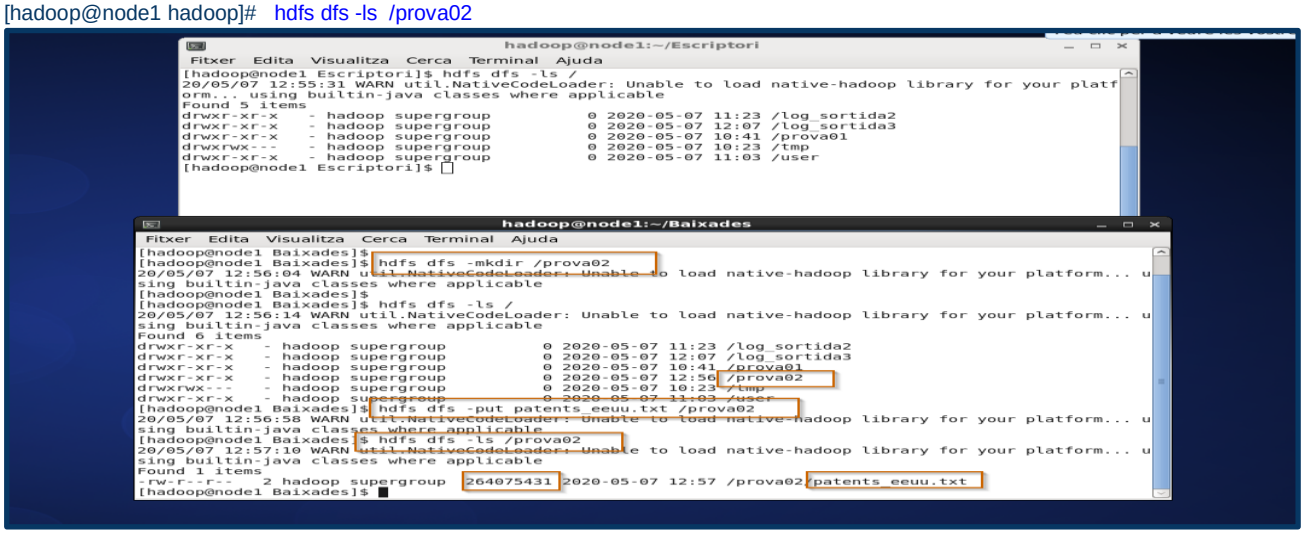
3.2.5.2. Exemple 2 - Llençar un procés Mapreduce contra el clúster

- Ens descarguem un arxiu d'exemple de patents dels E.E.U.U. anomenat "patents_eeuu.txt" de 264MB



- Desde el node "master" crearem un directori HDFS anomenat "prova02" i pujarem l'arxiu descarregat

```
[hadoop@node1 hadoop]#  hdfs dfs -mkdir /prova02
[hadoop@node1 hadoop]#  hdfs dfs -put patents_eeuu.txt /prova02
[hadoop@node1 hadoop]#  hdfs dfs -ls  /prova02
```



- Utilitzarem un programa Java anomenat "MyJob.java" per treballar amb aquest arxiu (ens agruparà les patents pel codi principal)

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

Fitxer Edita Visualitza Cerca Eines Documents Ajuda

Obre Desa Desfés

MyJob.java

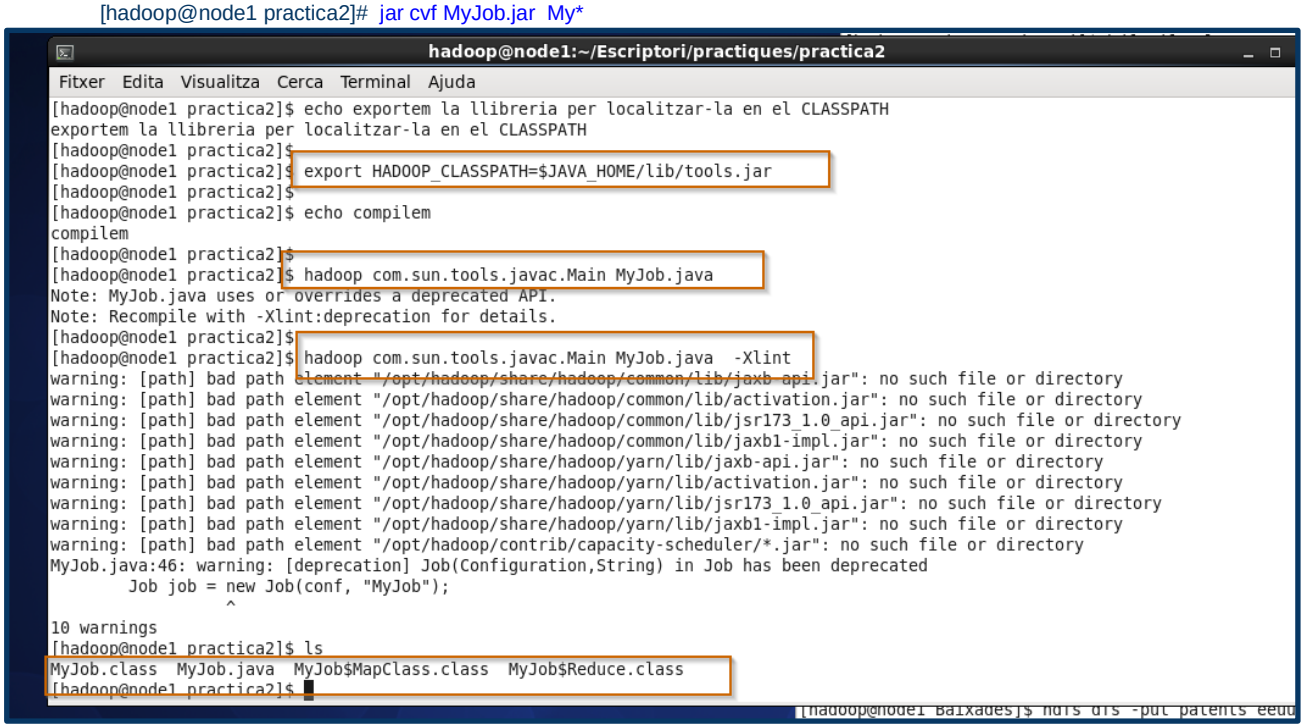
```java
public class MyJob extends Configured implements Tool {

    public static class MapClass extends Mapper<LongWritable, Text, Text, Text> {

        public void map(LongWritable key, Text value, Context context) throws IOException, InterruptedException {
            String[] citation = value.toString().split(",");
            context.write(new Text(citation[1]), new Text(citation[0]));
        }
    }

    public static class Reduce extends Reducer<Text, Text, Text, Text> {

        public void reduce(Text key, Iterable<Text> values, Context context) throws IOException, InterruptedException {
            String csv = "";
            for (Text val : values) {
                if (csv.length() > 0) {
                    csv += ",";
                }
                csv += val.toString();
            }
            context.write(key, new Text(csv));
        }
    }

    public int run(String[] args) throws Exception {
        Configuration conf = getConf();
        Job job = new Job(conf, "MyJob");
        job.setJarByClass(MyJob.class);
        Path in = new Path(args[0]);
        Path out = new Path(args[1]);
        FileInputFormat.setInputPaths(job, in);
        FileOutputFormat.setOutputPath(job, out);
        job.setMapperClass(MapClass.class);
        job.setReducerClass(Reduce.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(Text.class);
        System.exit(job.waitForCompletion(true) ? 0 : 1);
        return 0;

}
```

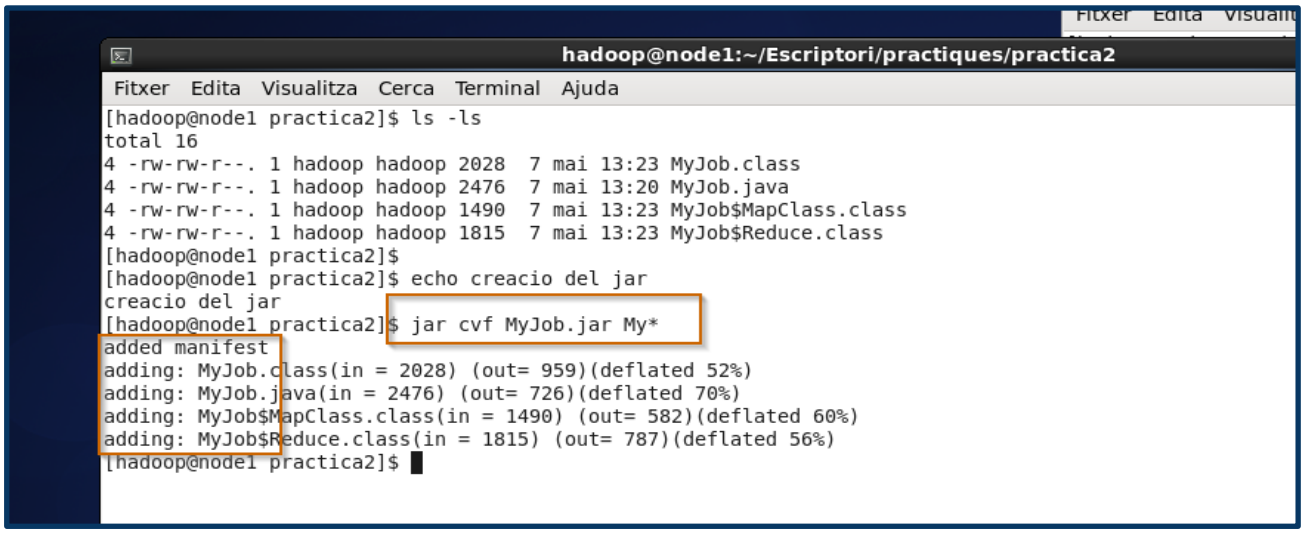
```java
}


  public static void main(String[] args) throws Exception {
        int res = ToolRunner.run(new Configuration(), new MyJob(), args);
                System.exit(res);
        }

}
```

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Exportem la llibreria per localitzar-la en el CLASSPATH i com que no tenim IDE per compilar aplicacions Java, utilitzarem hadoop per compila-ho

```
[hadoop@node1 practica2]#  export HADOOP_CLASSPATH=$JAVA_HOME/lib/tools.jar
[hadoop@node1 practica2]#  hadoop com.sun.tools.javac.Main  MyJob.java
[hadoop@node1 practica2]#  jar cvf MyJob.jar  My*
```

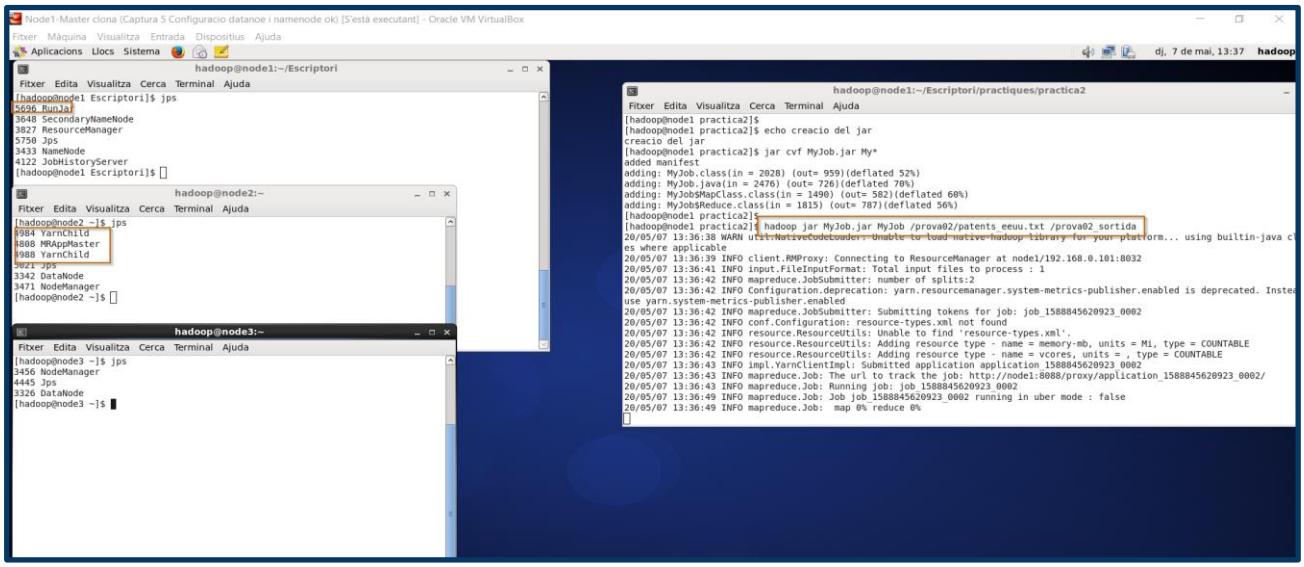




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

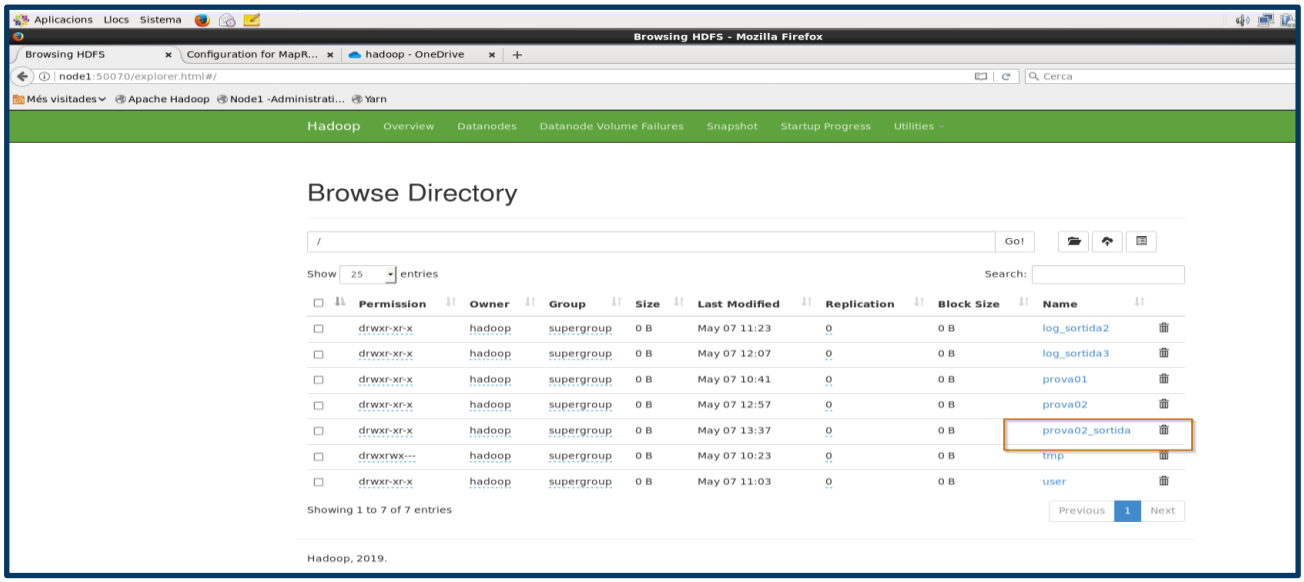
- Executem el jar que hem creat de MyJob on l'origen de les dades serà /prova02/patents_eeuu.txt i crearà en cas que no existeixi el directori destí anomenat prova02_sortida on és disposarà el resultat. Mentre s'està creant executem jps per veure els processos de tots els nodes.

```
[hadoop@node1 practica2]# hadoop jar MyJob.jar /prova02/patents_eeuu.txt /prova02_sortida
[hadoop@node1 practica2]# jps
[hadoop@node2 ~]# jps
[hadoop@node3 ~]# jps
```

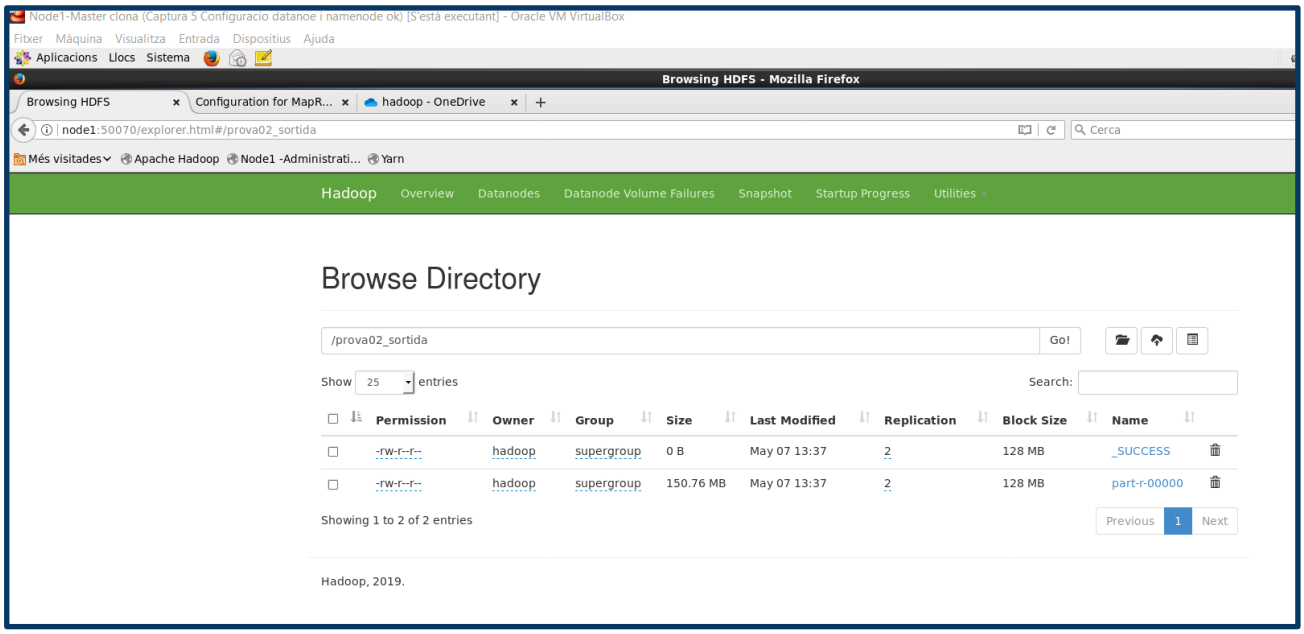


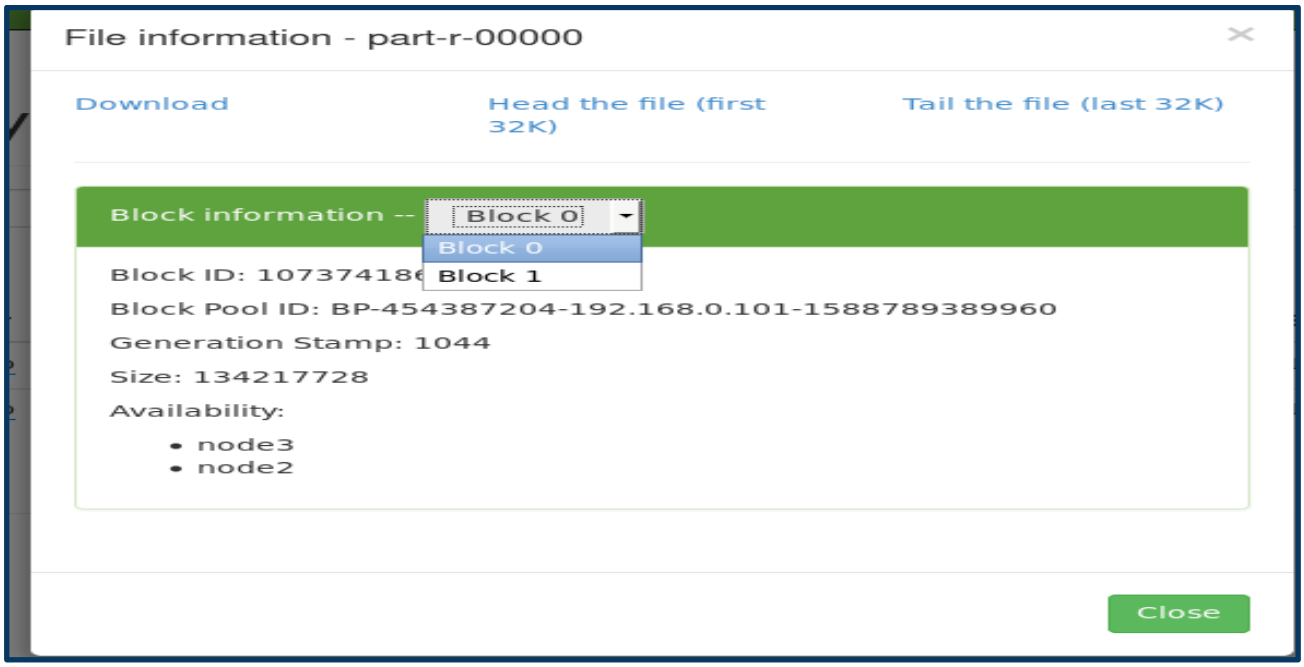
- Accedim a l'interfície web del Namenode i visualitzem el directori creat.

node1:50070



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

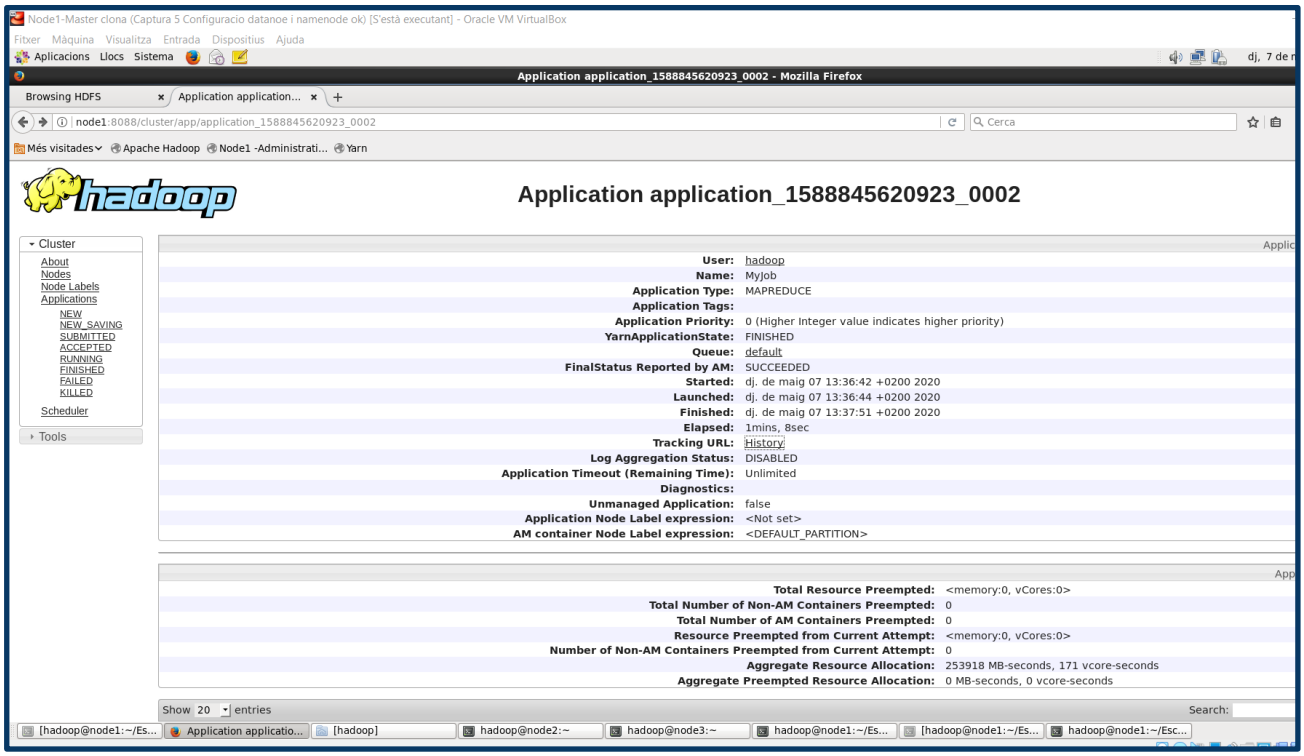


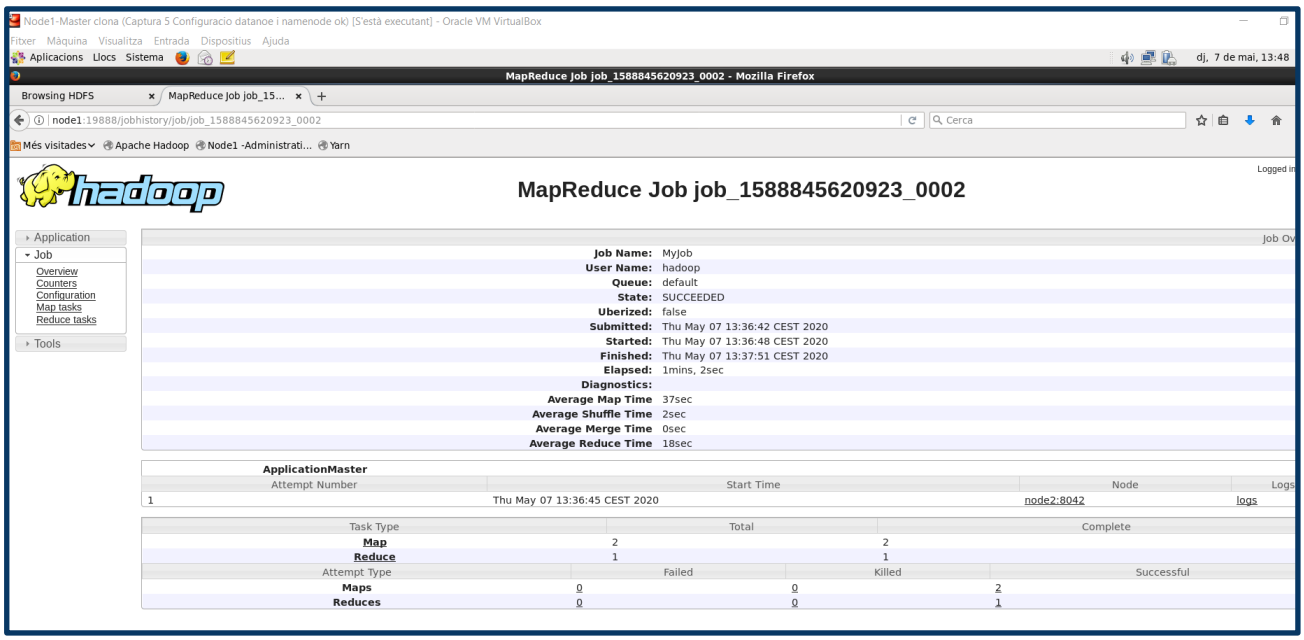


❑ Accedim a la interfície web del Yarn i visualitzem el directori creat.

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

node1:8088

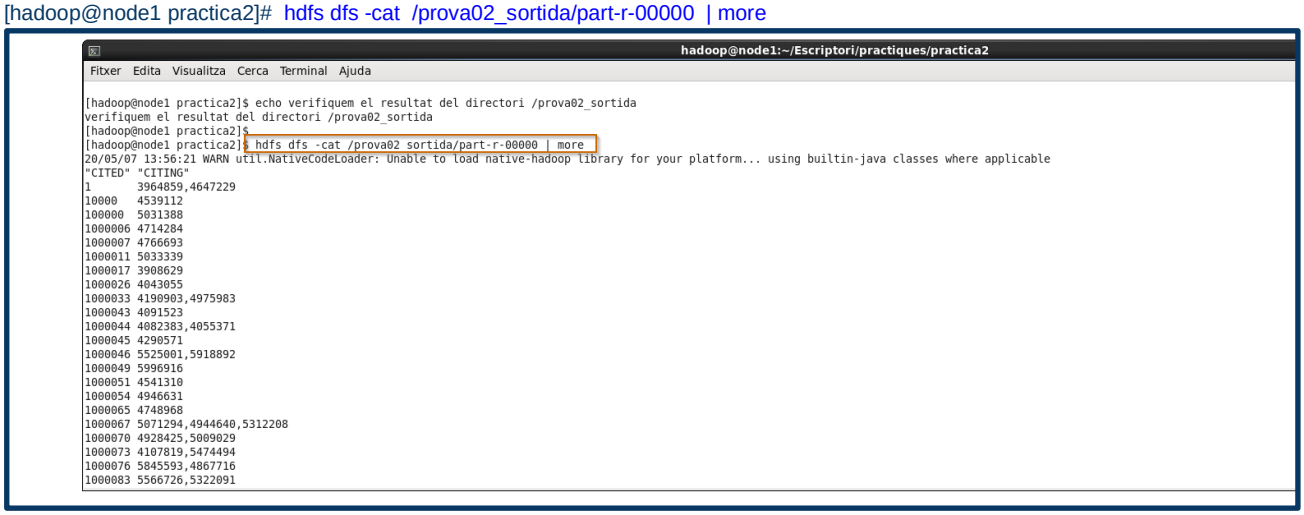




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- ❏ No ens fa falta descarregar el resultat per poder-ho llegir, ja que és un text pla

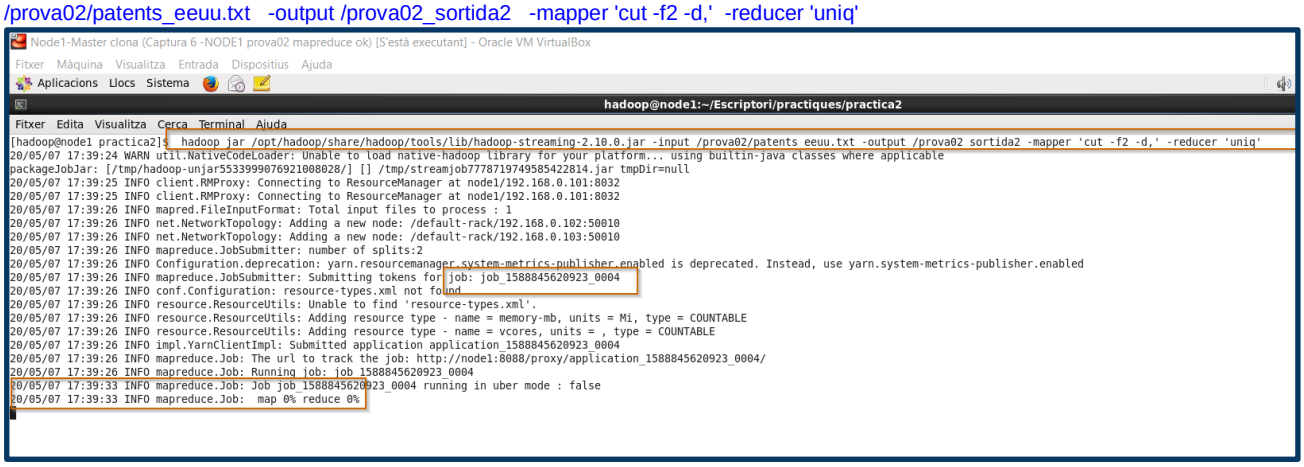
[hadoop@node1 practica2]# hdfs dfs -cat /prova02_sortida/part-r-00000 | more



3.2.5.3. Exemple 3 - Llençar un procés en streaming amb comandos de Shell de Linux.

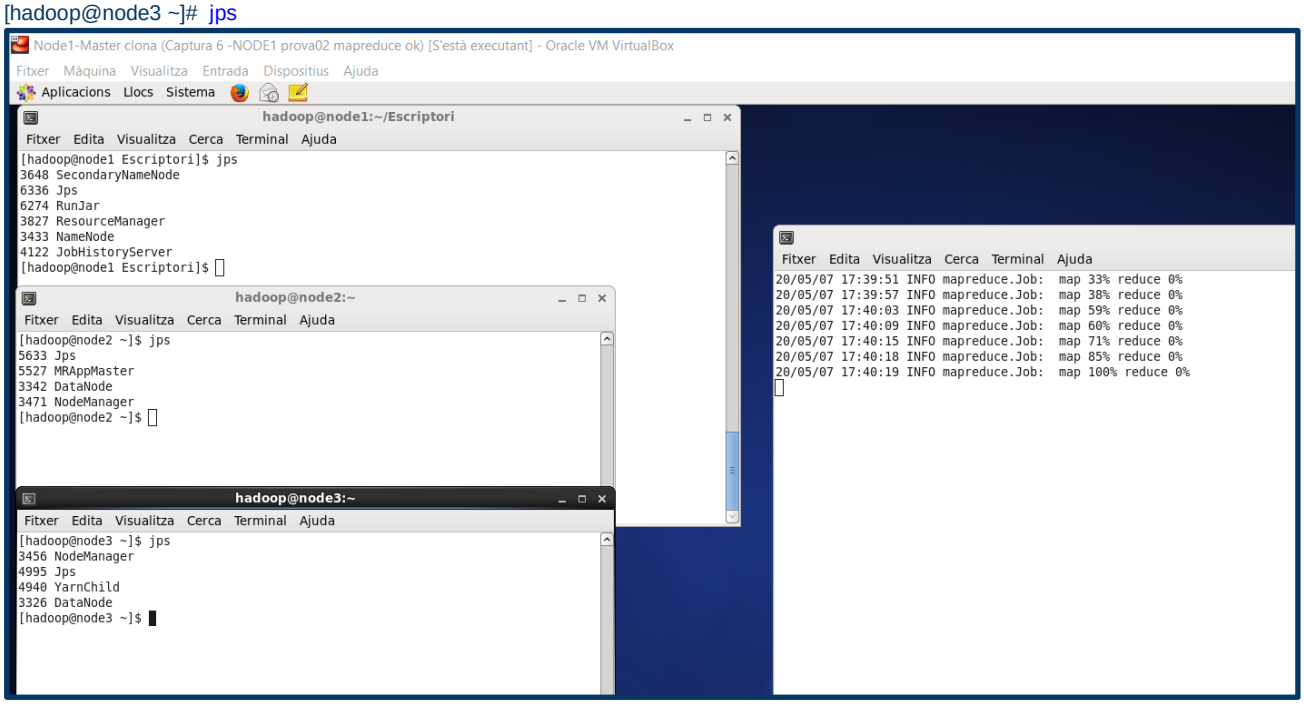
- ❏ En aquest exemple utilitzarem comandos de Shell de Linux per fer de Mapper i de Reducer, simulant el programa que hem fet en l'exemple 2.
- ❏ Utilitzarem la llibreria de streaming
- ❏ Li passarem el fitxer i en el Mapper extreurem les dades del camp2 amb el comando "cut" i eliminarem duplicats amb el "Reducer" i el comando "uniq"
- ❏ Mentre s'està creant execuem jps per veure els processos de tots els nodes.

[hadoop@node1 practica2]# hadoop jar /opt/hadoop/share/hadoop/tools/lib/hadoop-streaming-2.10.0.jar -input /prova02/patents_eeuu.txt -output /prova02_sortida2 -mapper 'cut -f2 -d,' -reducer 'uniq'



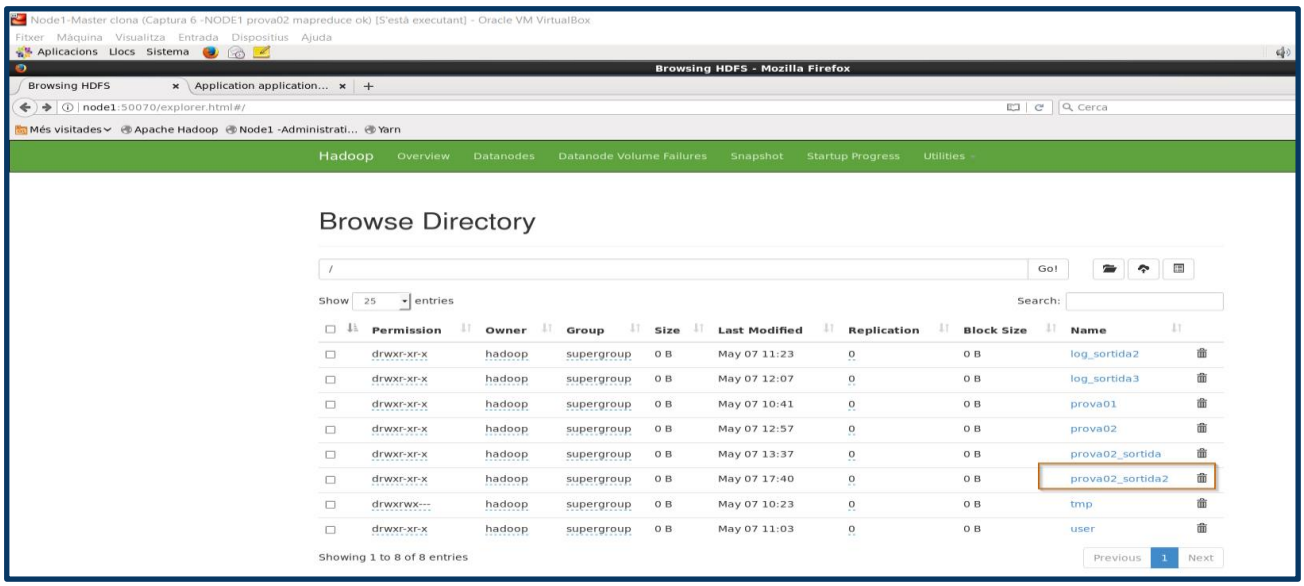
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

```
[hadoop@node1 practica2]#  jps
[hadoop@node2 ~]#  jps
[hadoop@node3 ~]#  jps
```

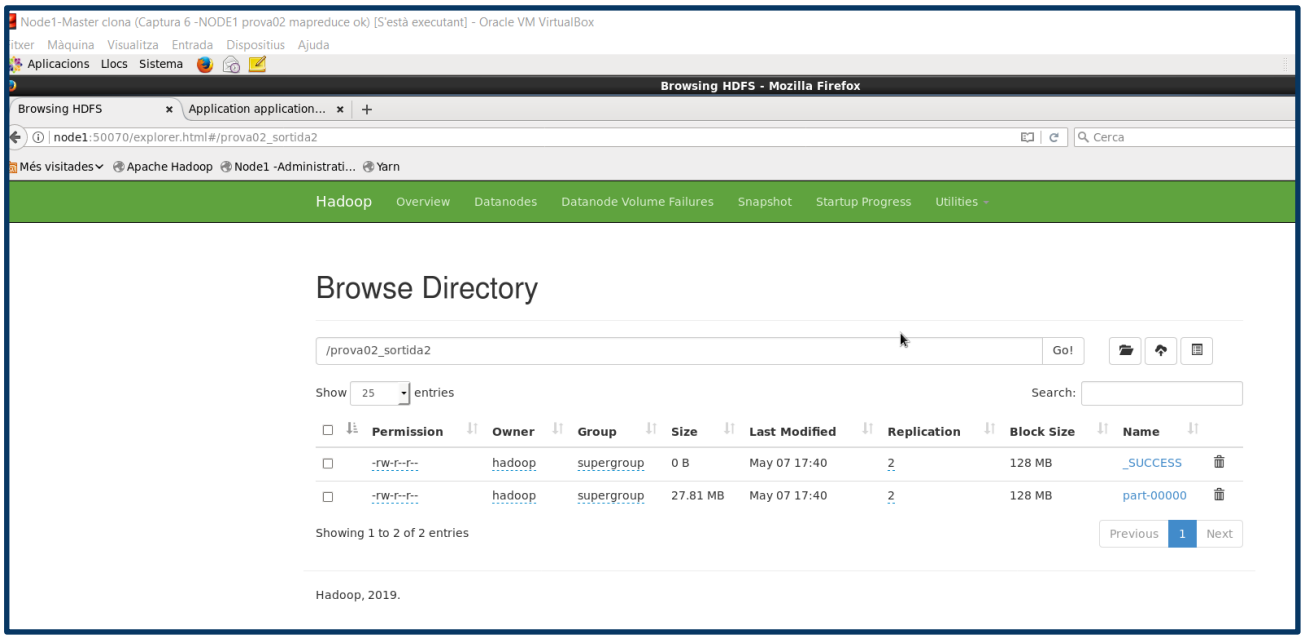


- ❑ Accedim a la interfície web del Namenode i visualitzem el directori creat.

node1:50070

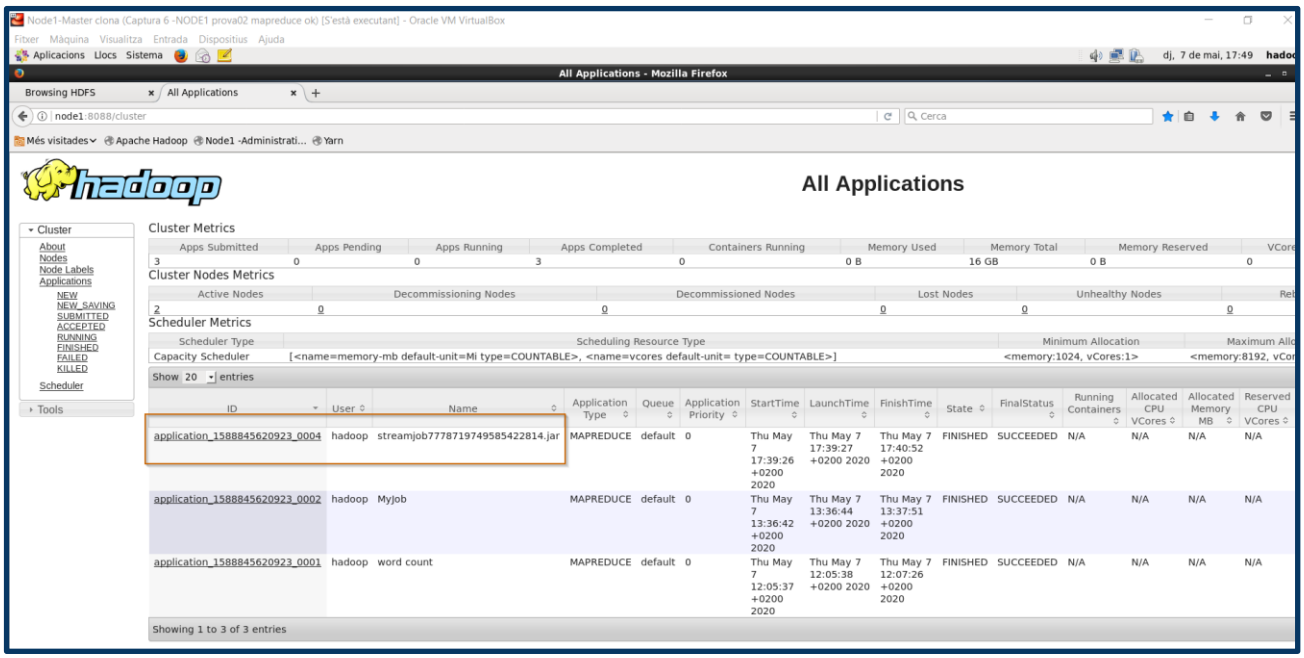


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

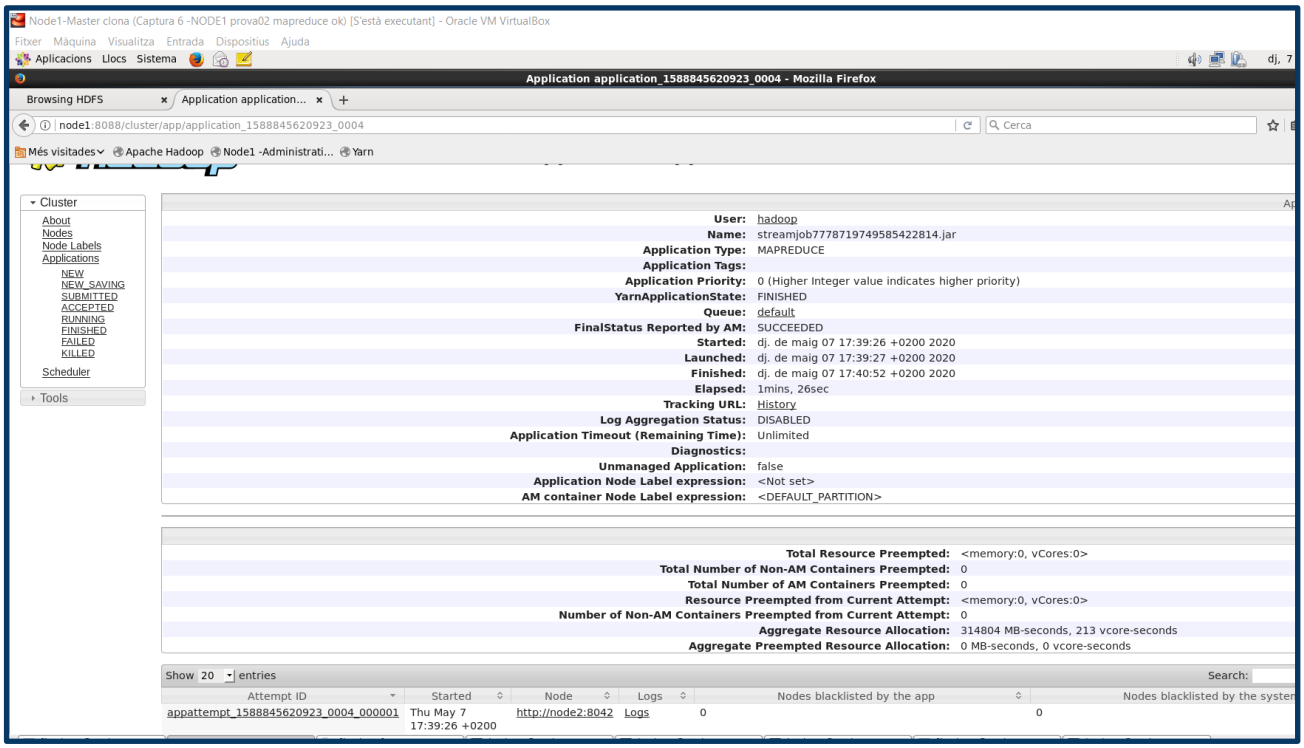


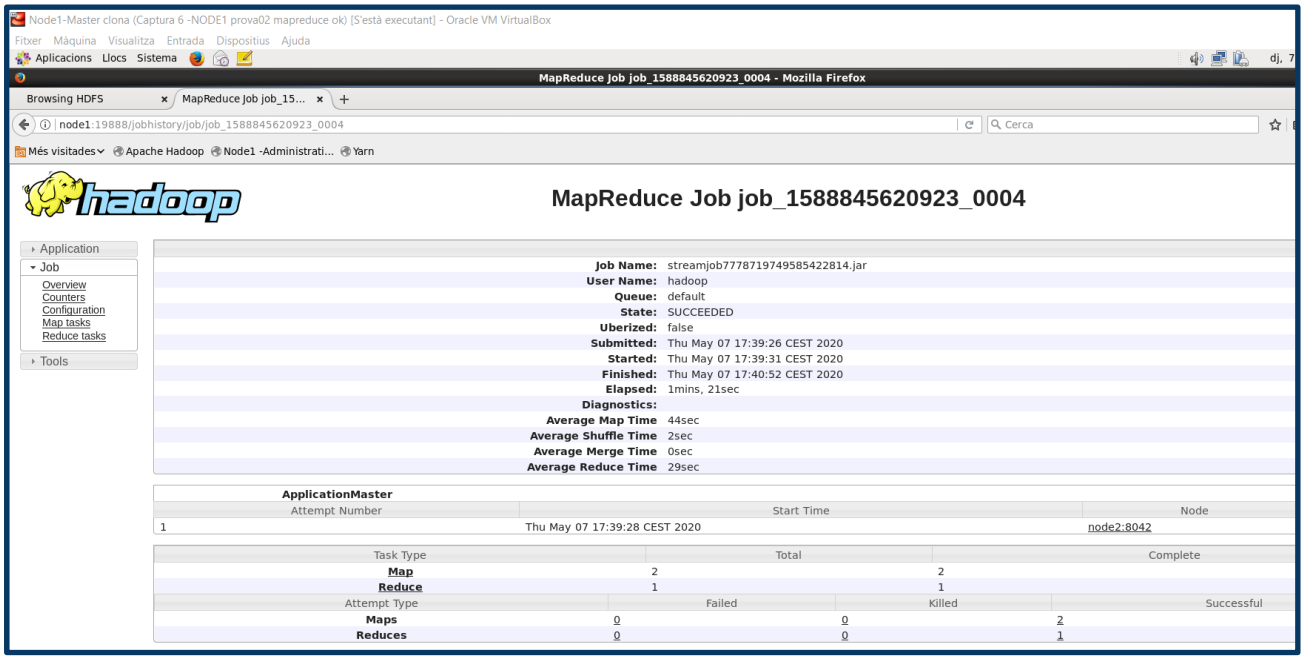
❑ Accedim a la interfície web del Yarn i visualitzem el directori creat.

node1:8088



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





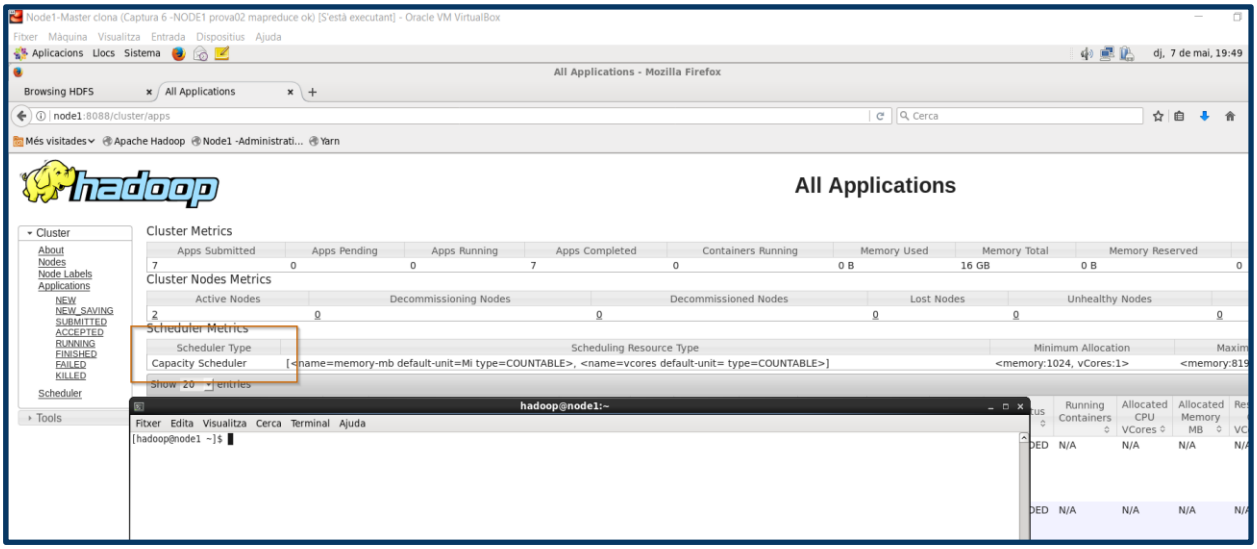
3.2.6. Yarn Schelduler

- El planificador determina els recursos que tenen cada un dels components del clúster. Per defecte si no està configurat, tots els components tenen els mateixos recursos. Hi ha 2 tipus de planificadors:

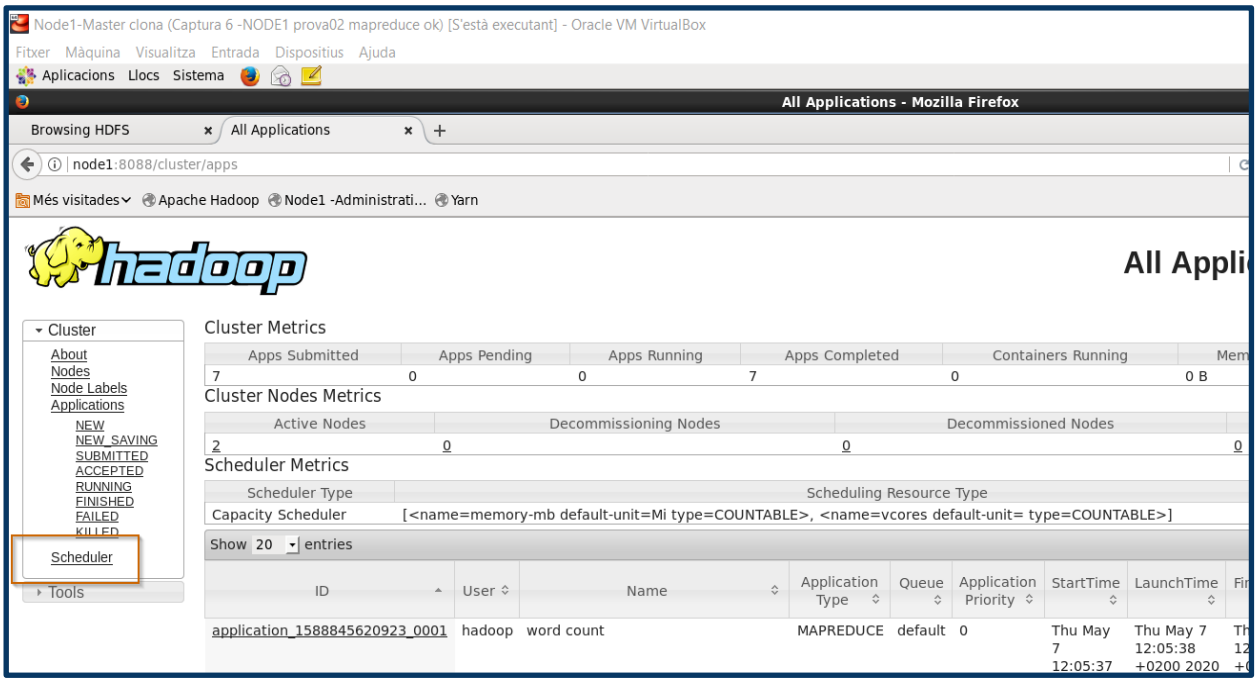
  - **Fair Schelduler** : Tots els processos s'assignen de manera homogènia els recursos dins el clúster
    - utilitzat en les versions de Hadoop 1.x
  - **Capacity Schelduler :**En canvi aquest planificador utilitza una estructura de jerarquia. Si no especifiquem res(no el tenim configurat) , després de root, tindrem una cua de recursos anomenada "default" que està associada al 100% dels recursos, seria com si estiguessin fem servir Fair Scheduler
    - utilitzat en les versions de Hadoop 2.x i 3.x

- Com que estem utilitzant una versió de Hadoop 2.x estem utilitzant Capacity Schelduler però com que no està configurat és com si estiguéssim utilitzant Fair Schelduler. Accedim a l'interfície web del Yarn per veure la configuració "default" del Yarn Schelduler

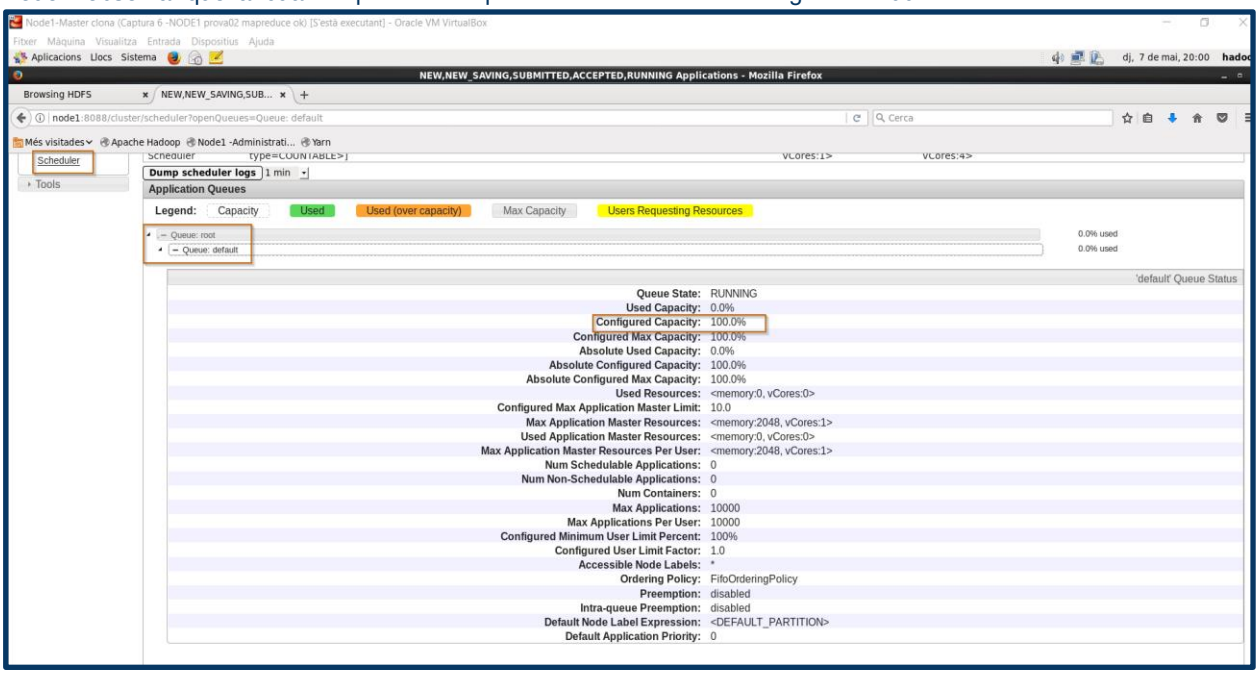
node1:8088



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

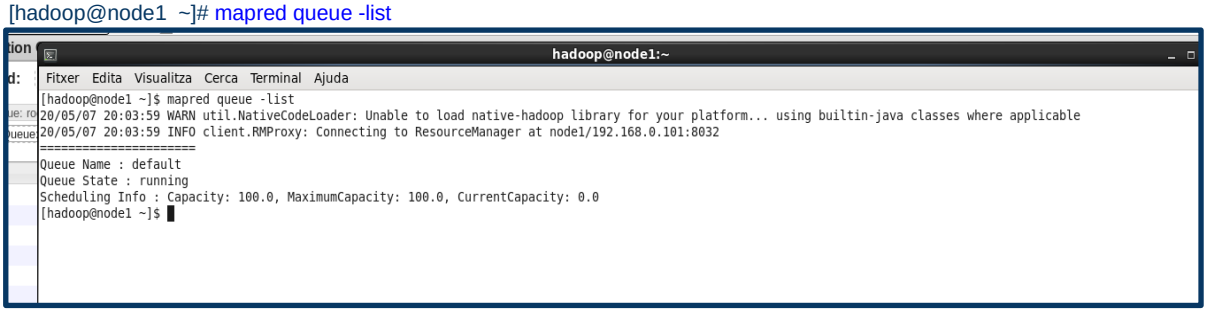


Podem observar que la cua del planificador per defecte es "default" configurat al 100%



- La informació de la cua també la podem visualitzar per terminal

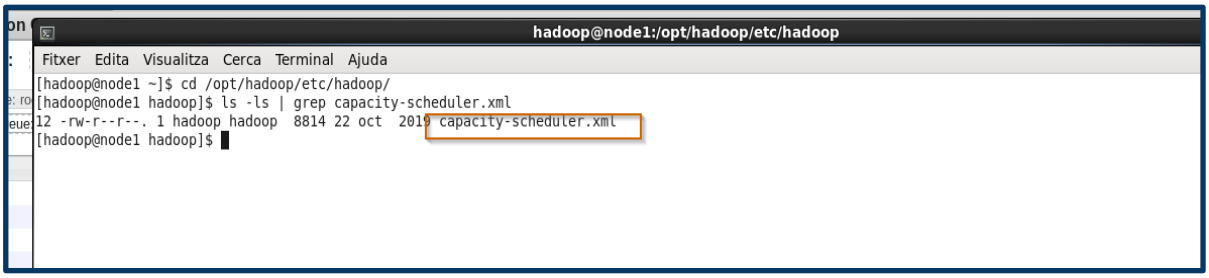
[hadoop@node1 ~]# mapred queue -list

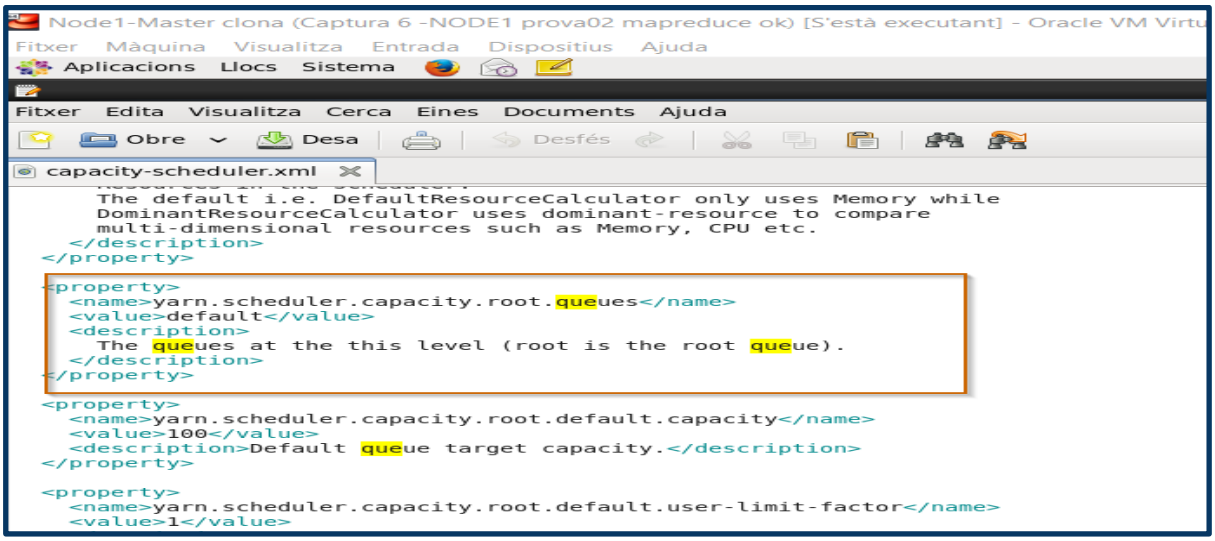


3.2.6.1. Configuració de Yarn Schelduler

- L'arxiu de configuració com ja hem vist amb altres està ubicat a /opt/hadoop/etc/haddop

[hadoop@node1 ~]# gedit capacity-schelduler.xml



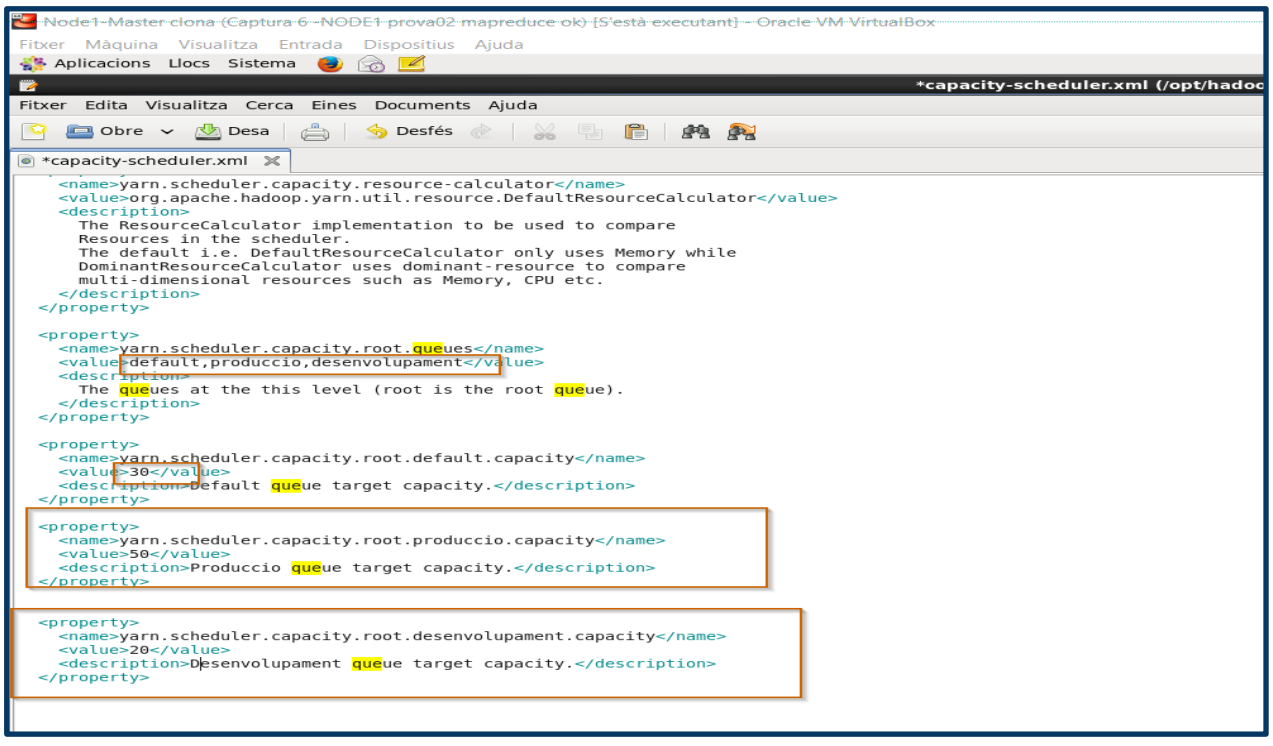


- Crearem noves cues : producció i desenvolupament. Les cues que no estiguin assignades aniran a default

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

```
<property>
  <name>yarn.scheduler.capacity.root.queues</name>
  <value>default,produccio,desenvolupament</value>
  <description>
    The queues at the this level (root is the root queue).
  </description>
</property>

<property>
  <name>yarn.scheduler.capacity.root.default.capacity</name>
  <value>100</value>
  <description>Default queue target capacity.</description>
</property>
```

- Després haurem de modificar la propietat de la capacitat, ja que per defecte el 100% està assignat a default. Li cambiarem els valors
  - queue default : 30%
  - queue produccio 50%
  - queue desenvolupament 20%

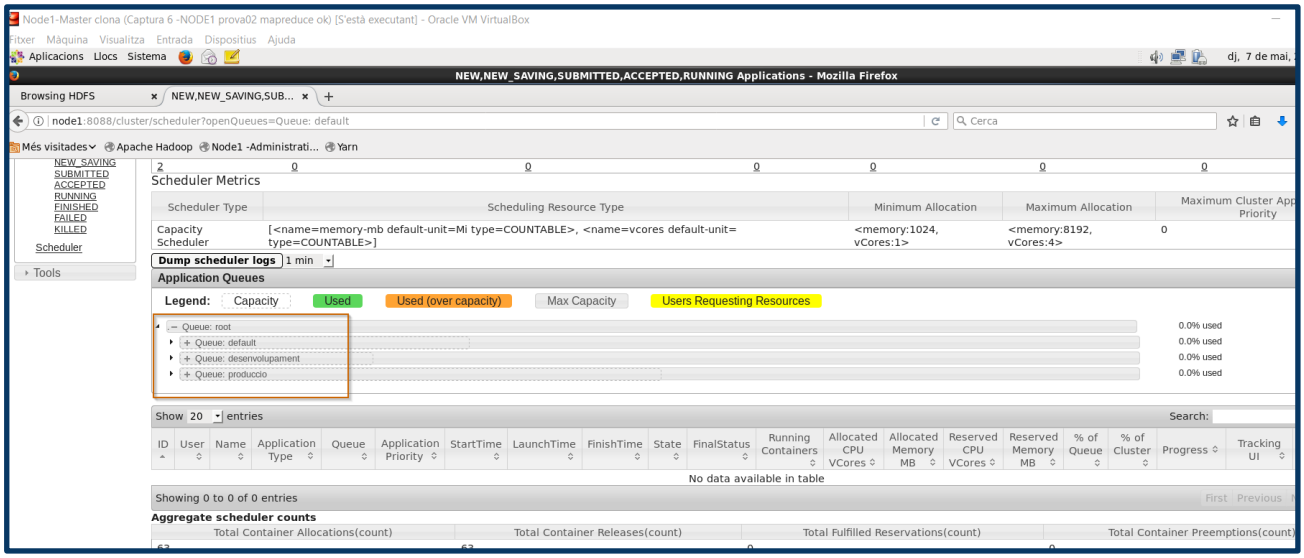


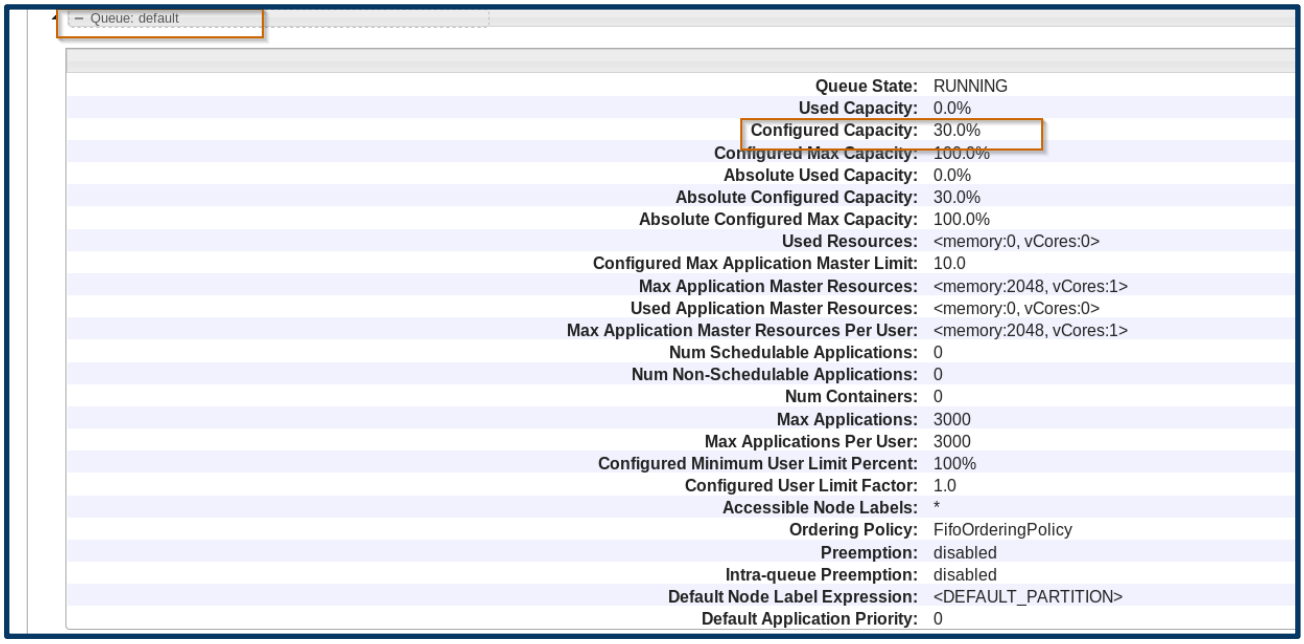
- Un cop hem modificat l'arxiu i el guardem ens fa falta activar els canvis.
  [hadoop@node1 ~]# yarn rmadmin -refreshQueues

- Accedim a la interfície web del Yarn per veure els canvis realitzats.

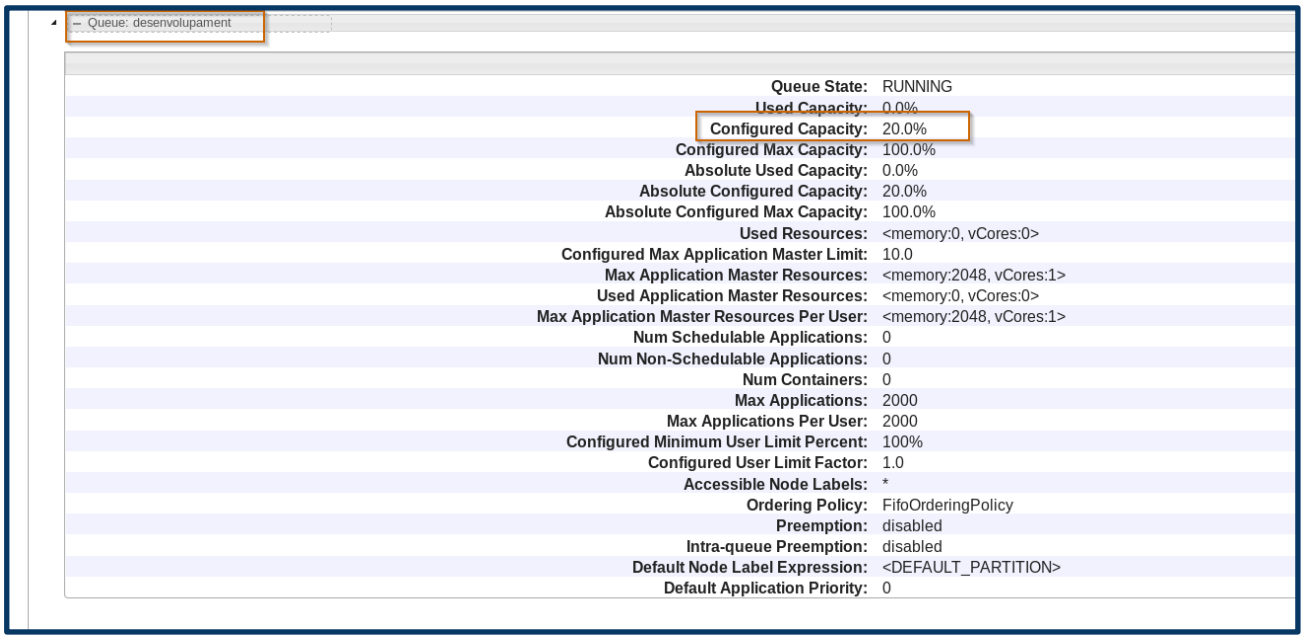
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

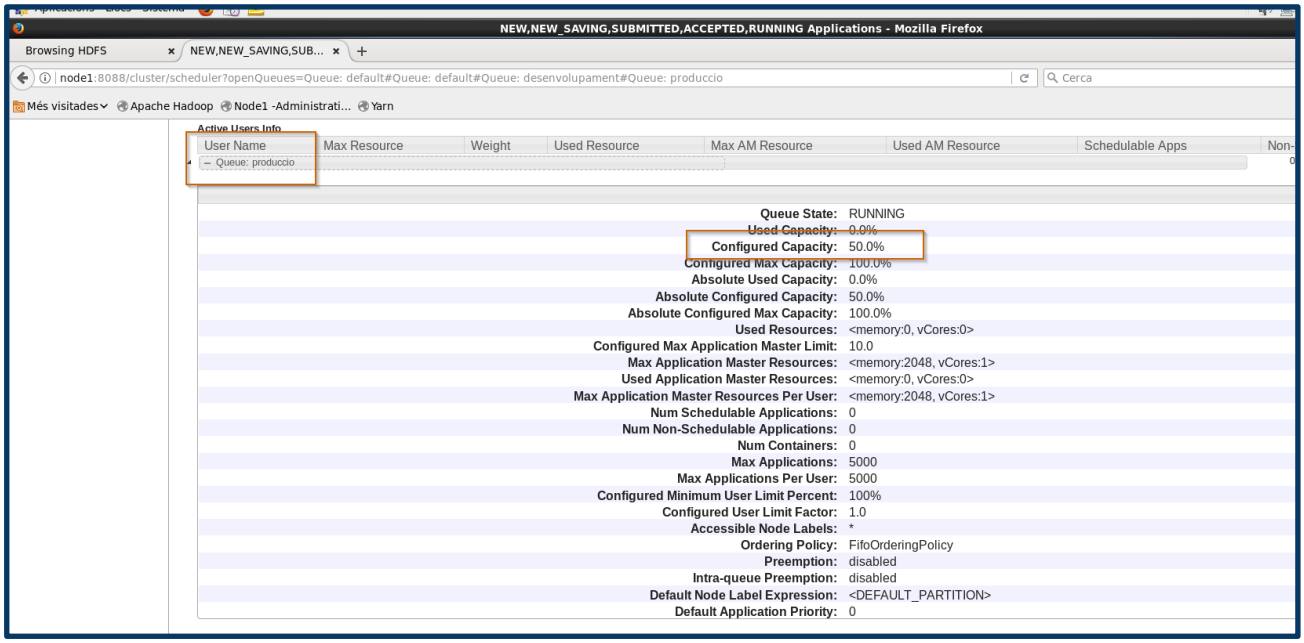
node1:8088





Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





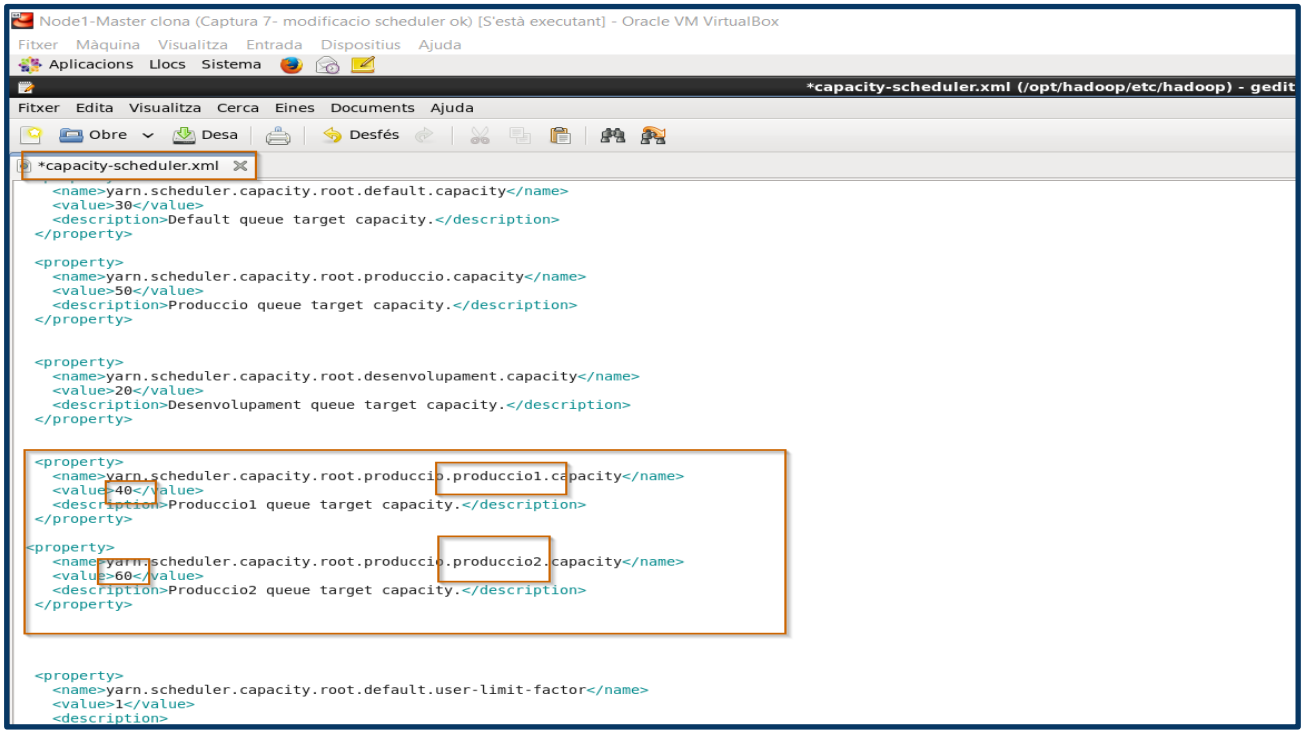
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Com que ens ha funcionat, crearem 2 cues on la cua arrel no serà root sinó producció i després farem un exemple pràctic . Els valors de les noves cues seran:
  - queue produccio 50%
    - producció1 40%
    - producció2 60%

[hadoop@node1 ~]# gedit capacity-schelduler.xml

*capacity-scheduler.xml

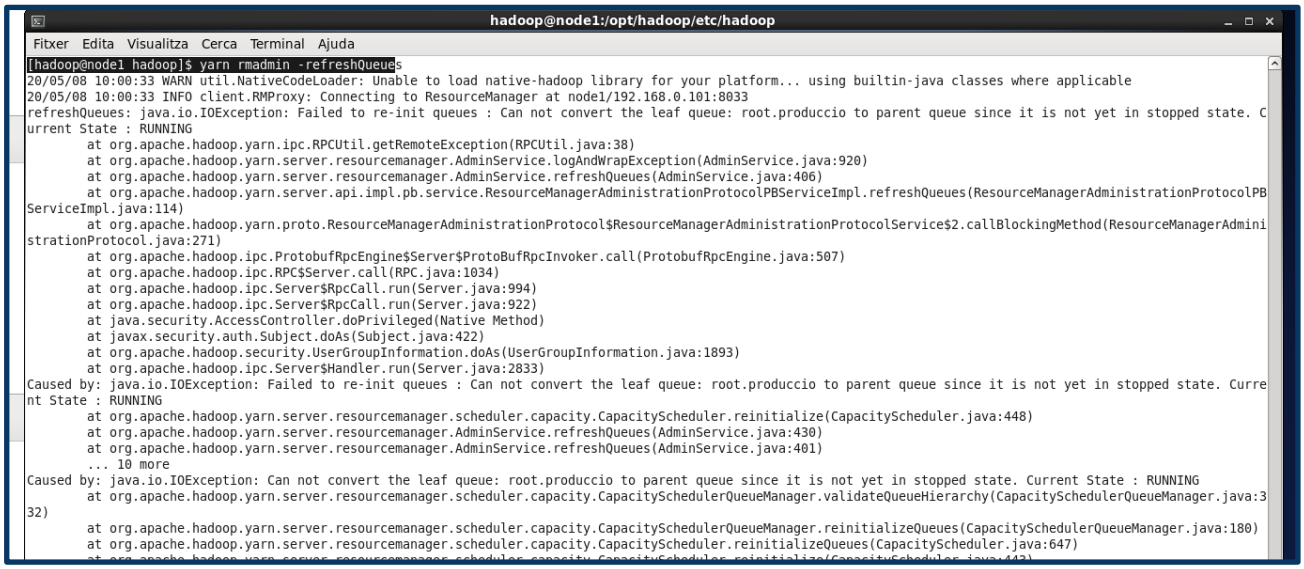
```
    <value>org.apache.hadoop.yarn.util.resource.DefaultResourceCalculator</value>
    <description>
      The ResourceCalculator implementation to be used to compare
      Resources in the scheduler.
      The default i.e. DefaultResourceCalculator only uses Memory while
      DominantResourceCalculator uses dominant-resource to compare
      multi-dimensional resources such as Memory, CPU etc.
    </description>
  </property>

  <property>
    <name>yarn.scheduler.capacity.root.queues</name>
    <value>default,produccio,desenvolupament</value>
    <description>
      The queues at the this level (root is the root queue).
    </description>
  </property>

  <property>
    <name>yarn.scheduler.capacity.root.produccio.queues</name>
    <value>produccio1,produccio2</value>
    <description>
      The queues at the this level (produccio is the root queue).
    </description>
  </property>
```



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

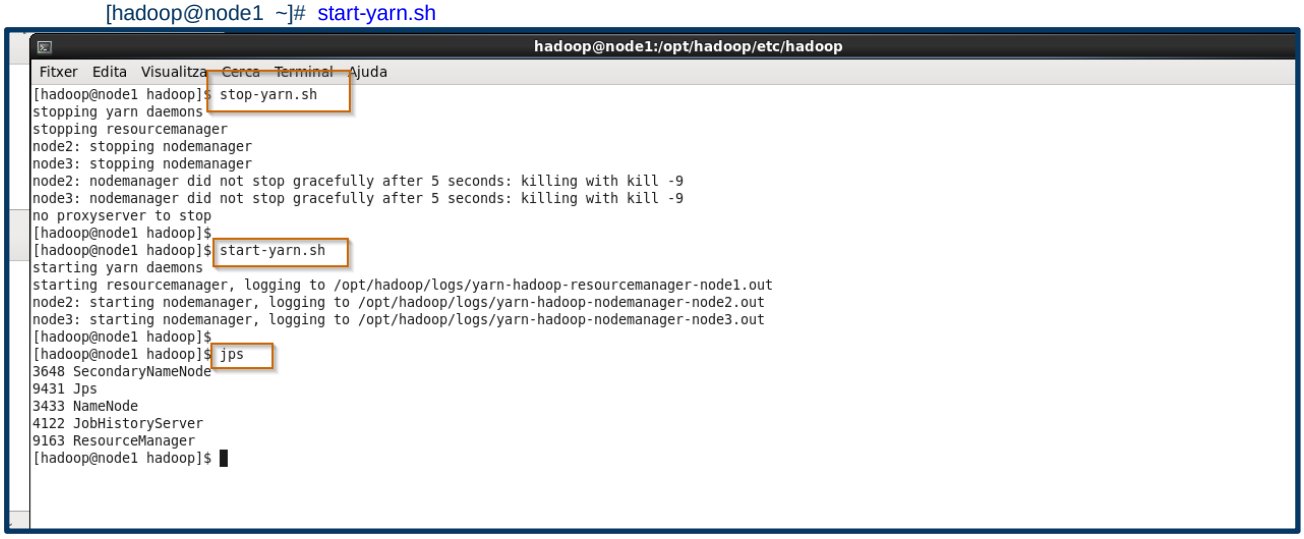
- Un cop hem modificat l'arxiu i el guardem ens fa falta activar els canvis.

[hadoop@node1 ~]# yarn rmadmin -refreshQueues



- Ens dóna error, ja que hem assignat 2 cues filla a cues que ja estan funcionant i per tant no puc refrescar. Per solucionar el problema, hauré de parar el clúster (només la part de processos) i tornar-lo a reiniciar.

[hadoop@node1 ~]# stop-yarn.sh
[hadoop@node1 ~]# start-yarn.sh

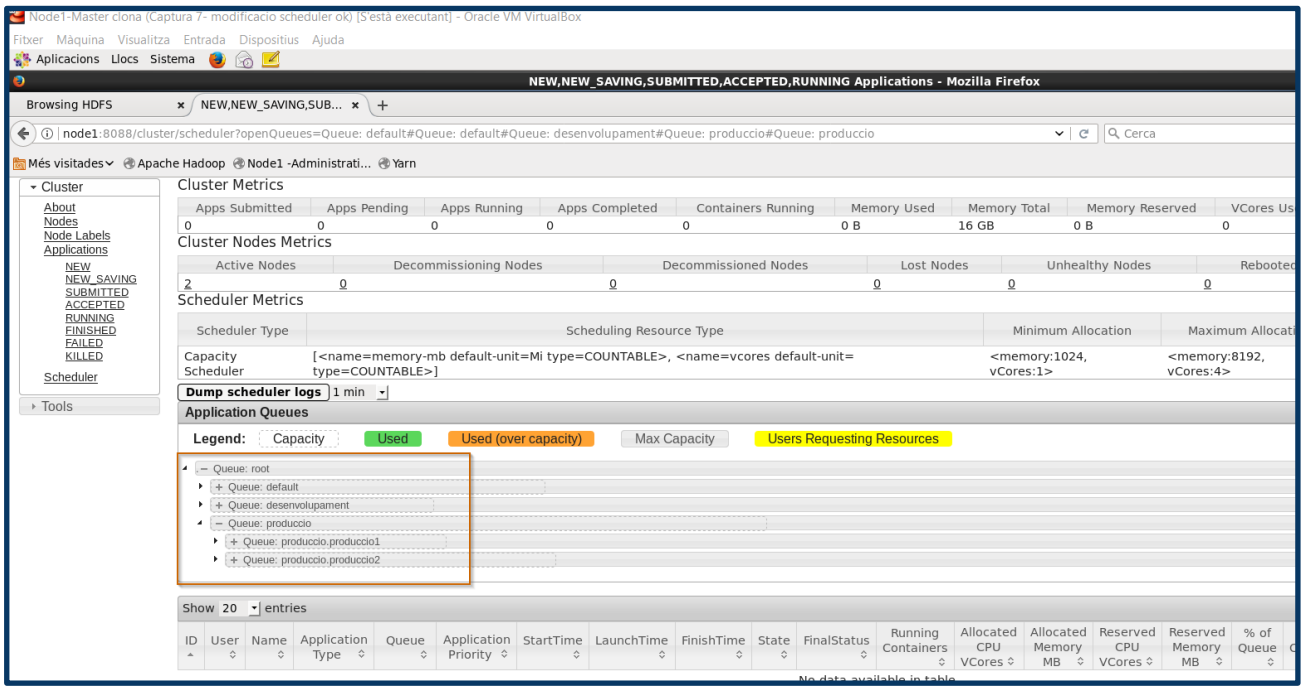


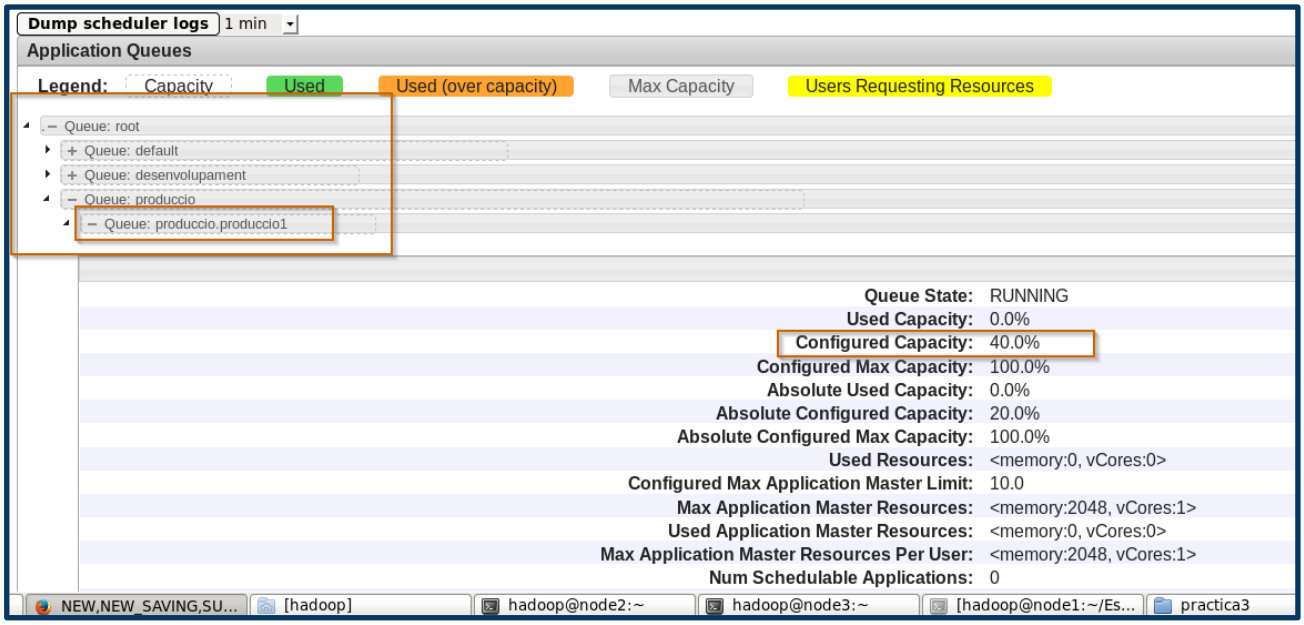
Nom i Cognoms | **Data**

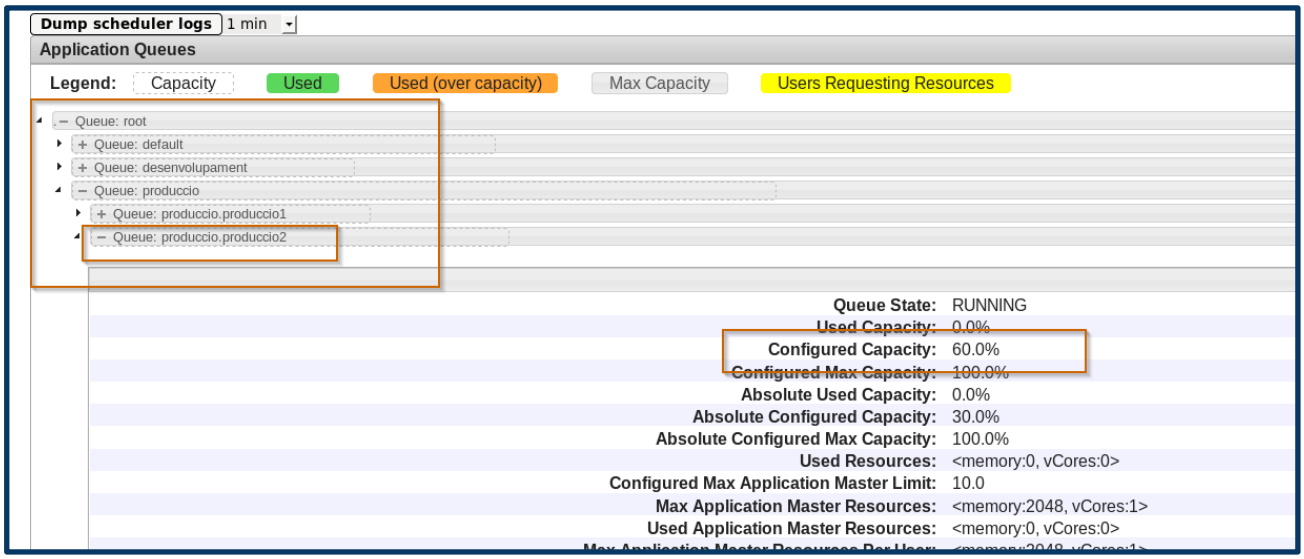
Arnau Subirós Puigarnau | 02-06-2020

- Accedim a la interfície web del Yarn per veure els canvis realitzats.

node1:8088

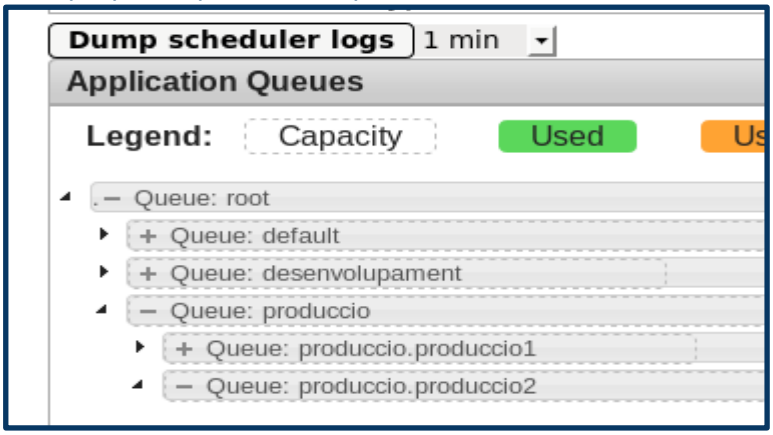






3.2.6.1.1. Exemple : llençar un job

- Un cop tenim configurat el Yarn Schelduler amb la jerarquia de cues i la seva capacitat, farem un exemple pràctic per verificar que funciona

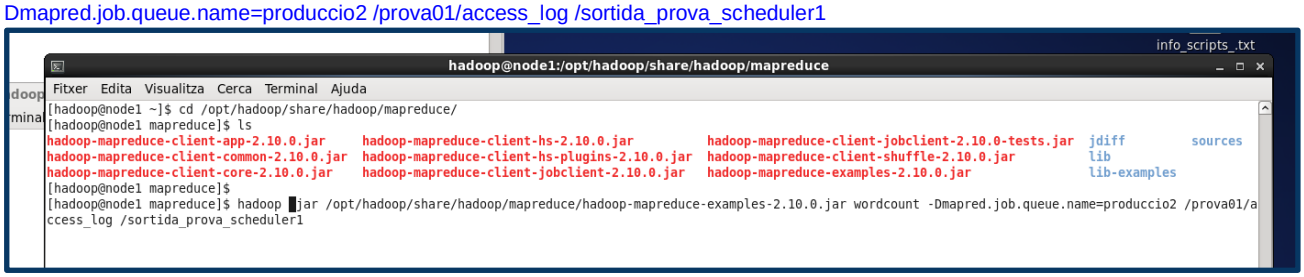


- Farem servir un exemple Mapreduce anomenat "wordcount".

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

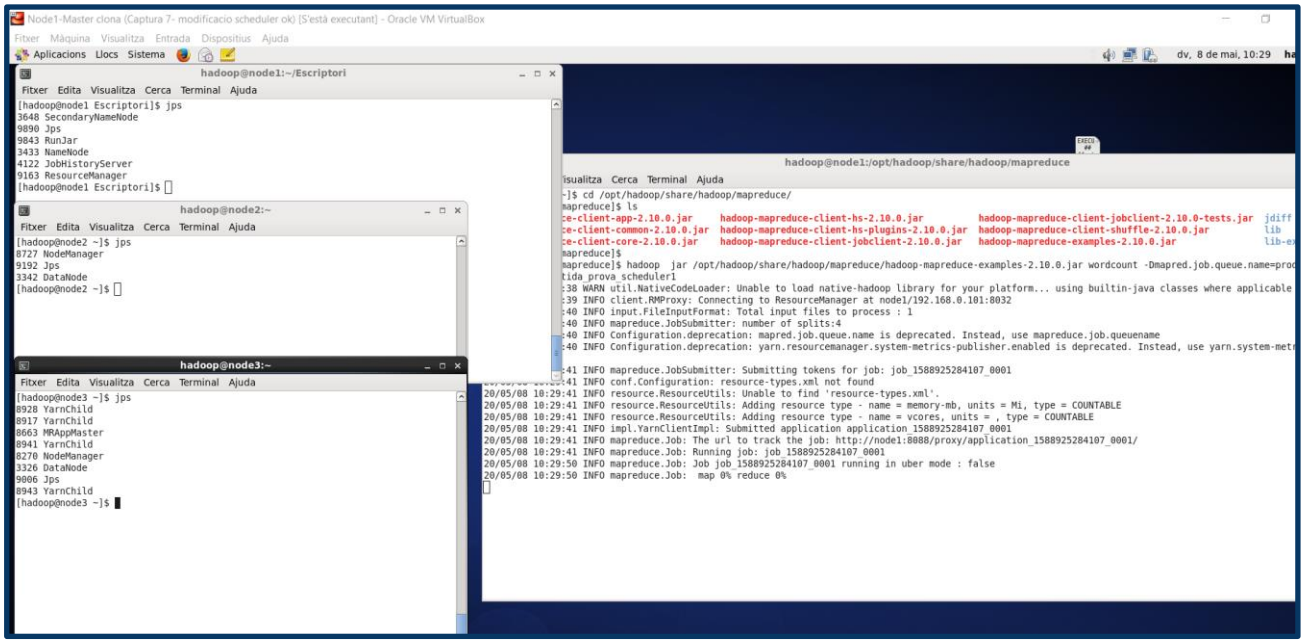
- -Dmapred.job.queue.name :Especifiquem la cua
- origen del fitxer : /prova01/access_log
- desti de resultat : /sortida_prova_schelduler1

[hadoop@node1 ~]# hadoop jar /opt/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.10.0.jar wordcount -Dmapred.job.queue.name=produccio2 /prova01/access_log /sortida_prova_scheduler1



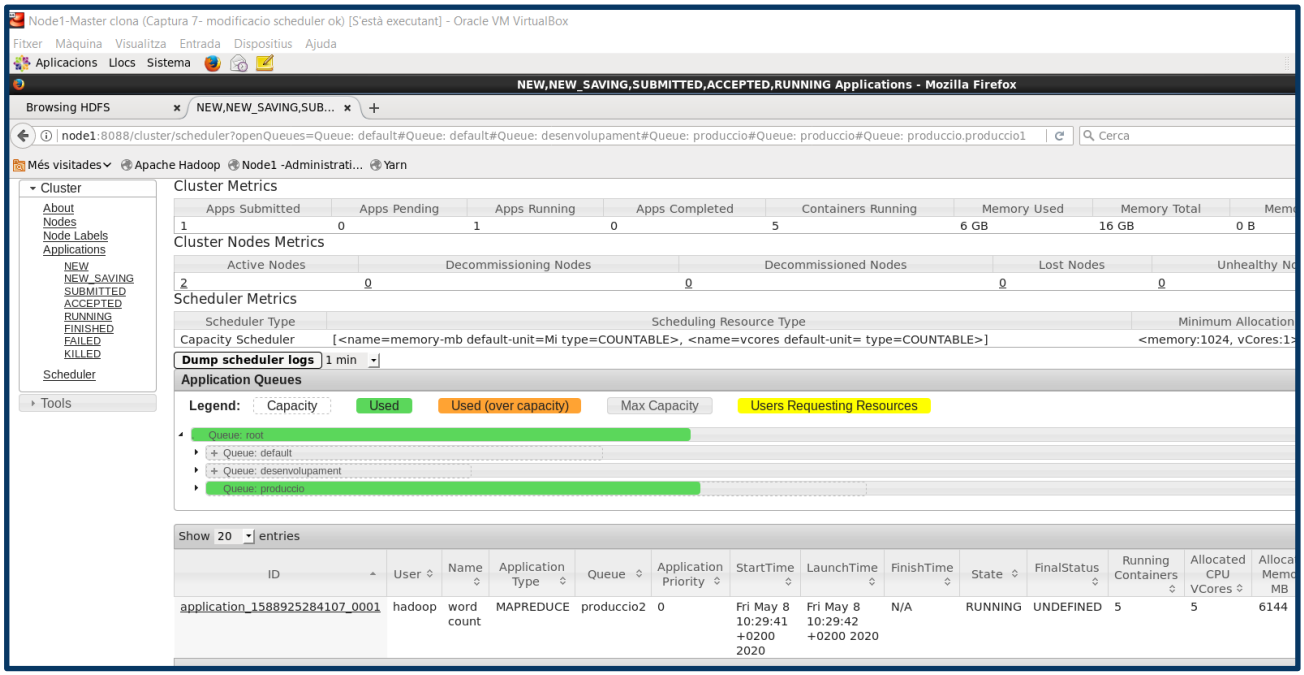
- ❑ Executem jps en tots els nodes per veure els processos.

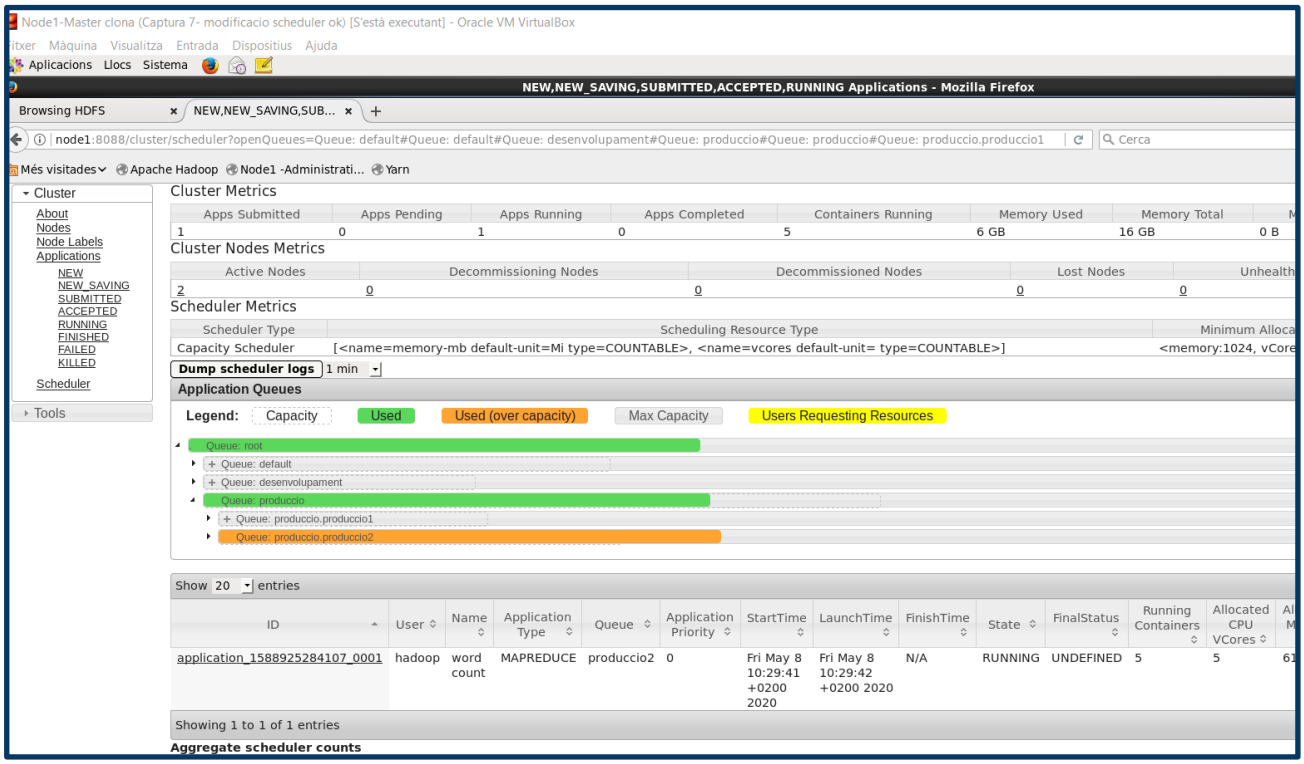
[hadoop@node1 ~]# jps
[hadoop@node2 ~]# jps
[hadoop@node3 ~]# jps



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

❑ Accedim a l'interficie web de Yarn i observem el Schelduler mentre els processos estan funcionant





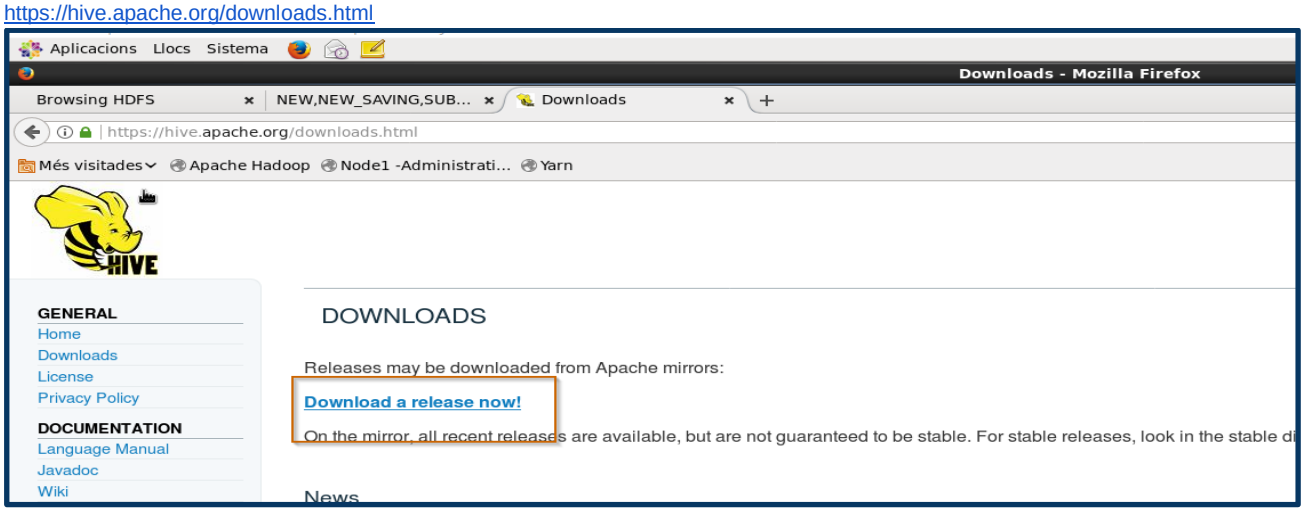
3.2.6.2. Ecosistema Hadoop

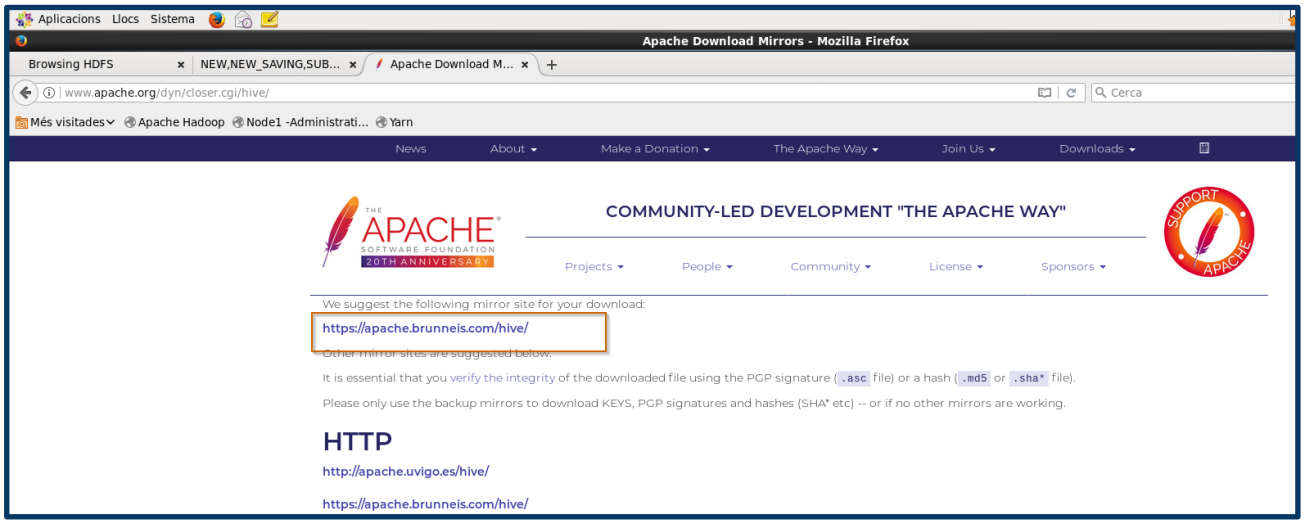
- A més a més del clúster Hadoop i els seus components bàsics. A continuació mostrem la instal·lació, configuració i algun exemple de Hive i Spark.
  Nota: Tot i que vàrem instal·lar Ambari no ho mostrarem, ja que té un apartat propi on es crearà un nou clúster de 0 utilitzant Ambari i 3 màquines CentOS (les mateixes màquines clonades sense Hadoop)

3.2.6.2.1. Instal.lació i configuracio de Hive

- Accedim a la pàgina oficial, per realitzar la descarrega

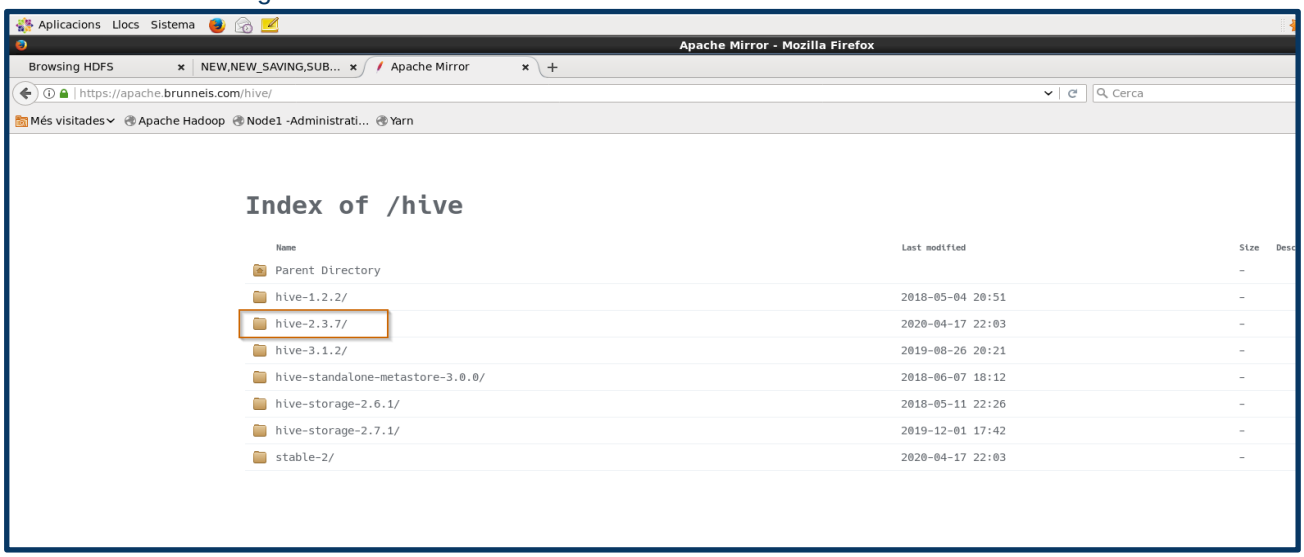
https://hive.apache.org/downloads.html

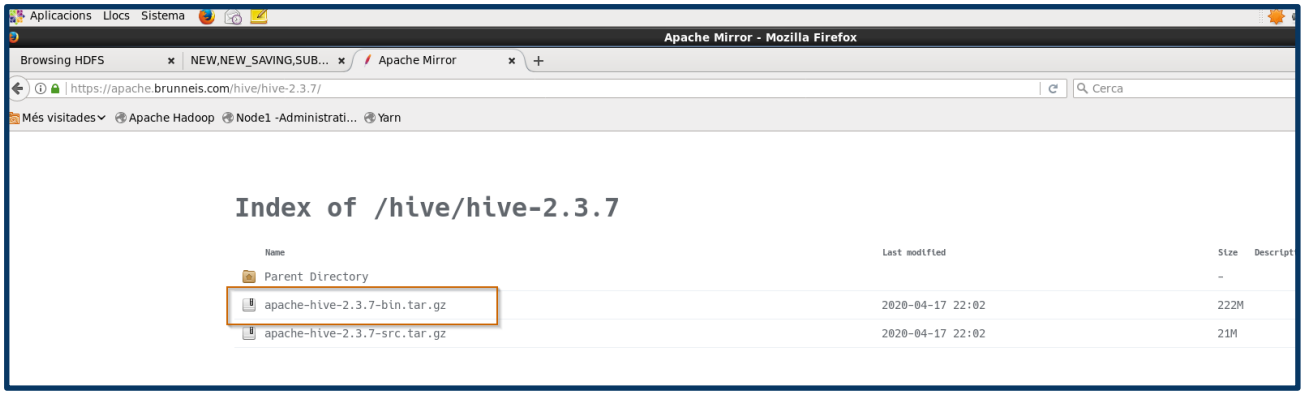




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

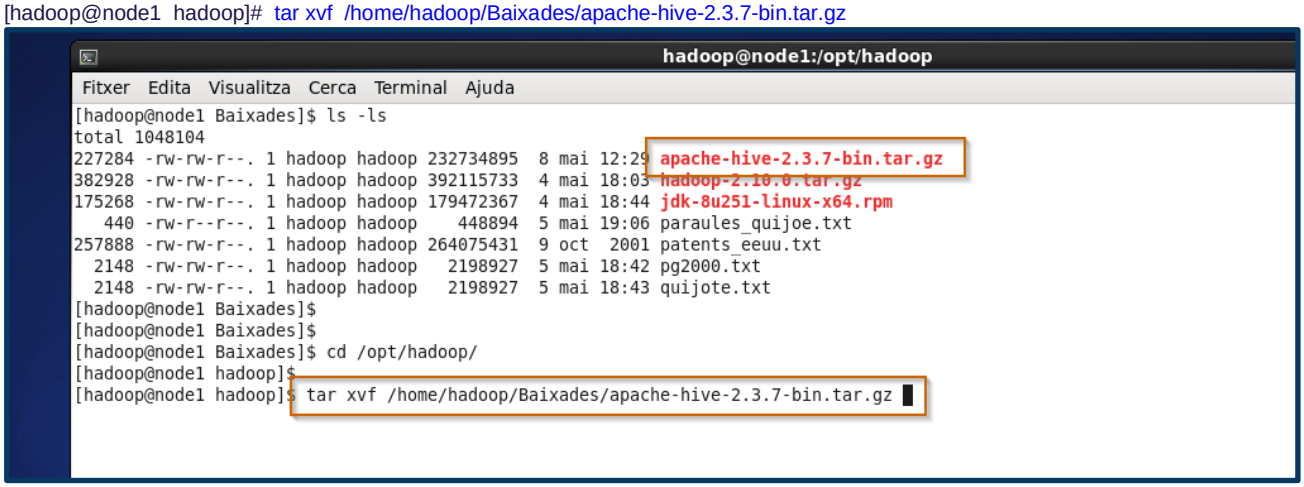
- Ens descarreguem la versió 2.3.7





- Descomprimim l'arxiu i l'ubiquem dins el directori /opt/hadoop

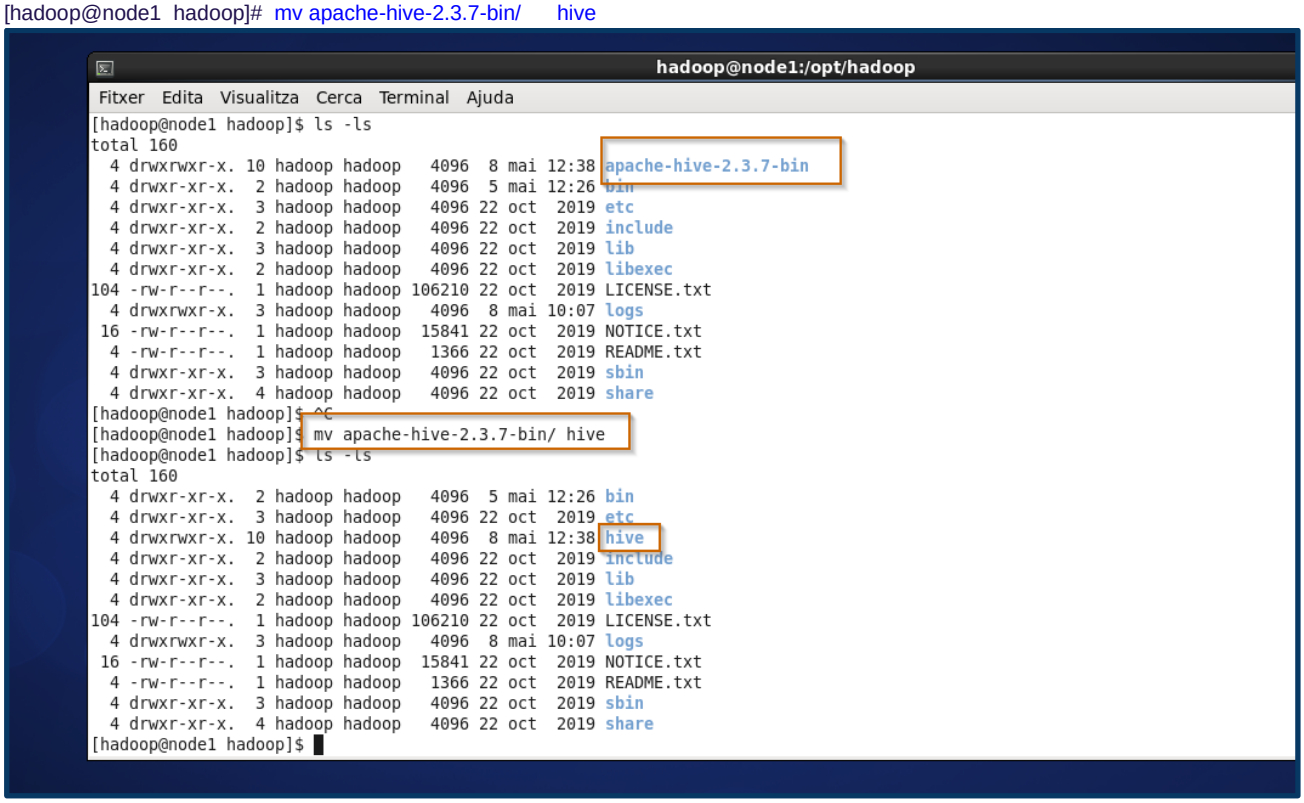
[hadoop@node1 hadoop]# tar xvf /home/hadoop/Baixades/apache-hive-2.3.7-bin.tar.gz



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

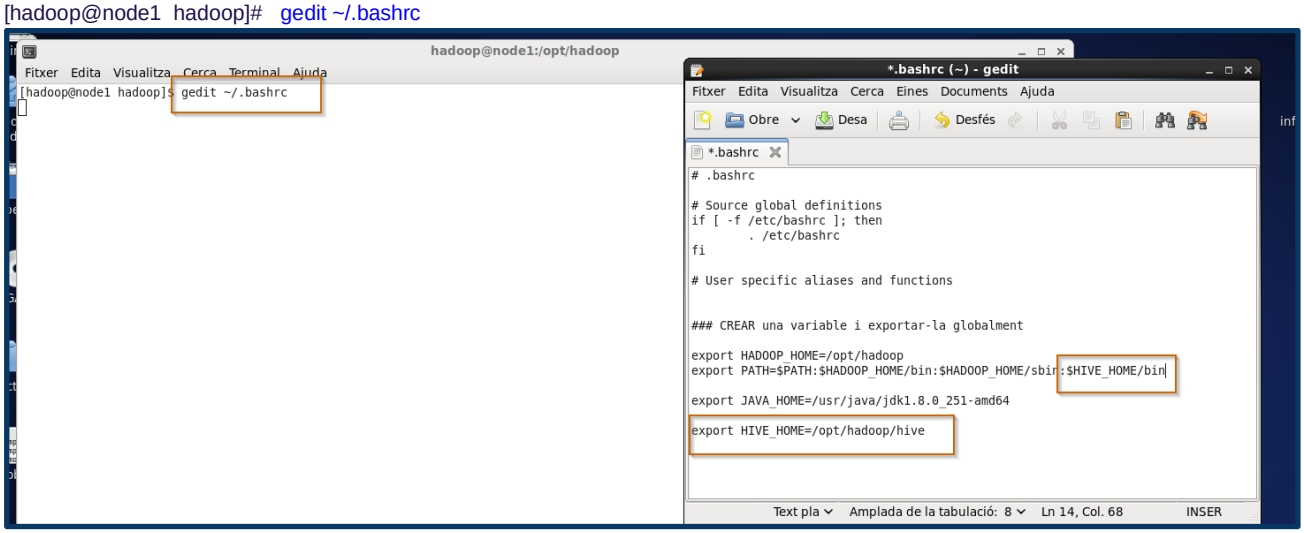
- Desplacem els arxius del directori creat, ja que es molt llarg a un nou directori anomenat hive

[hadoop@node1 hadoop]# mv apache-hive-2.3.7-bin/ hive



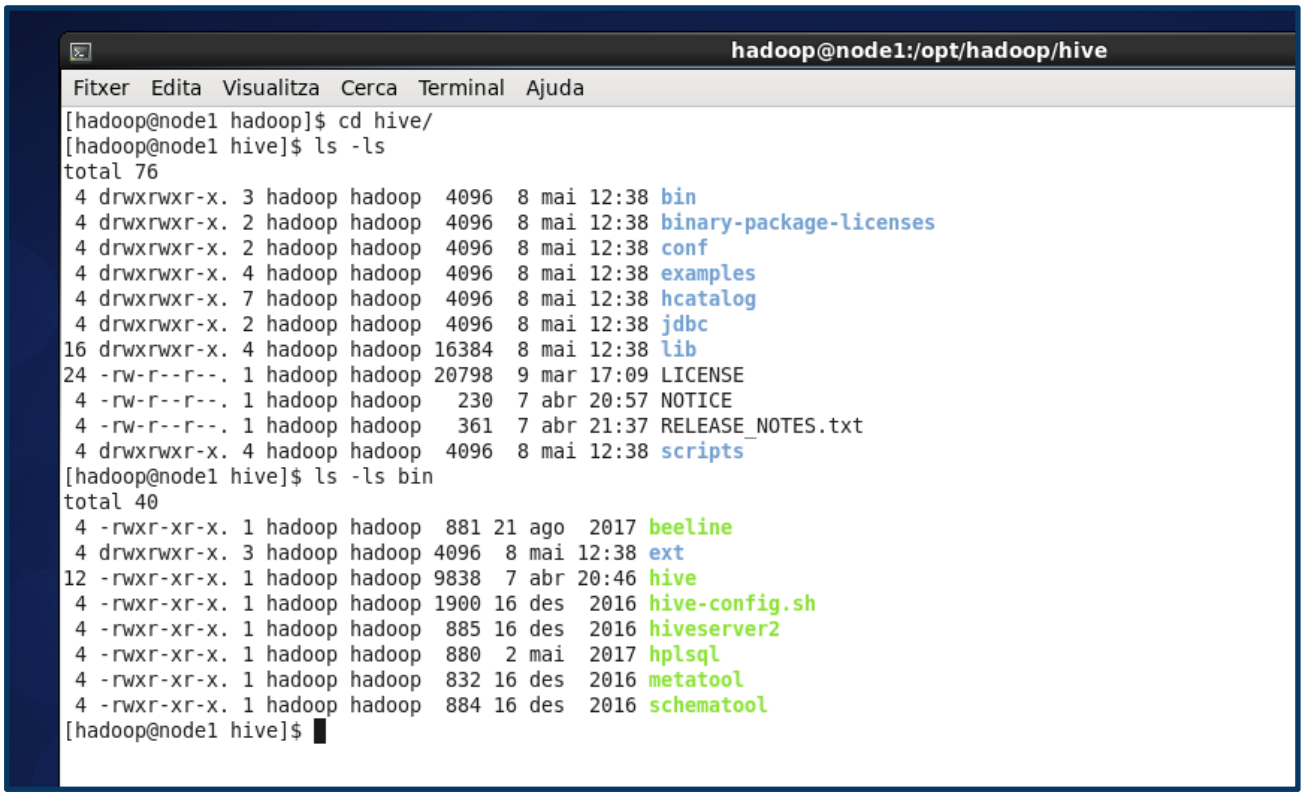
- Haurem d'afegir les variables d'entorn de l'arxiu ~/.bashrc

[hadoop@node1 hadoop]# gedit ~/.bashrc

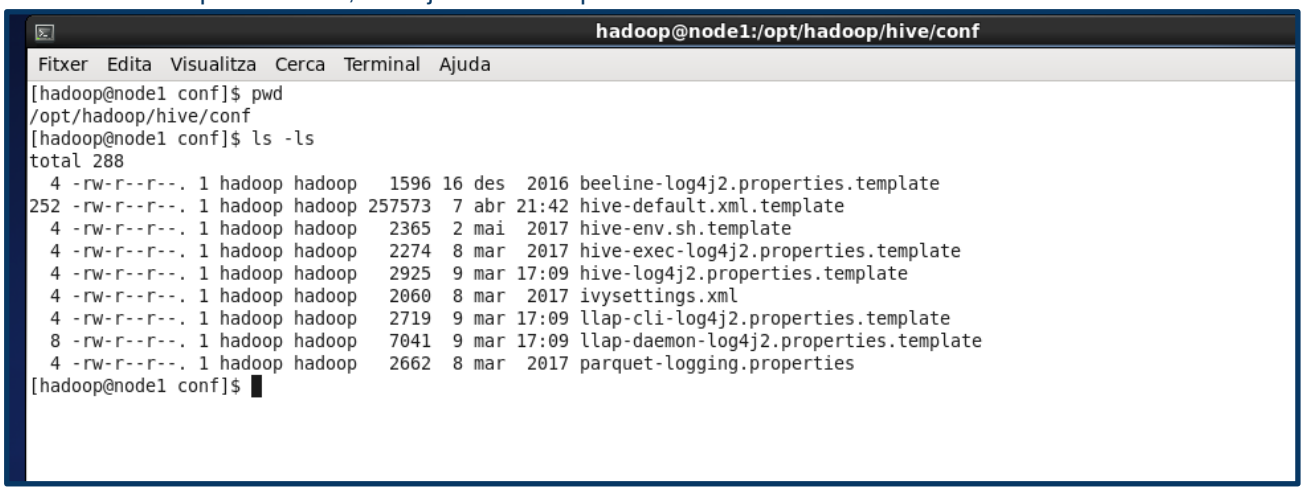


Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

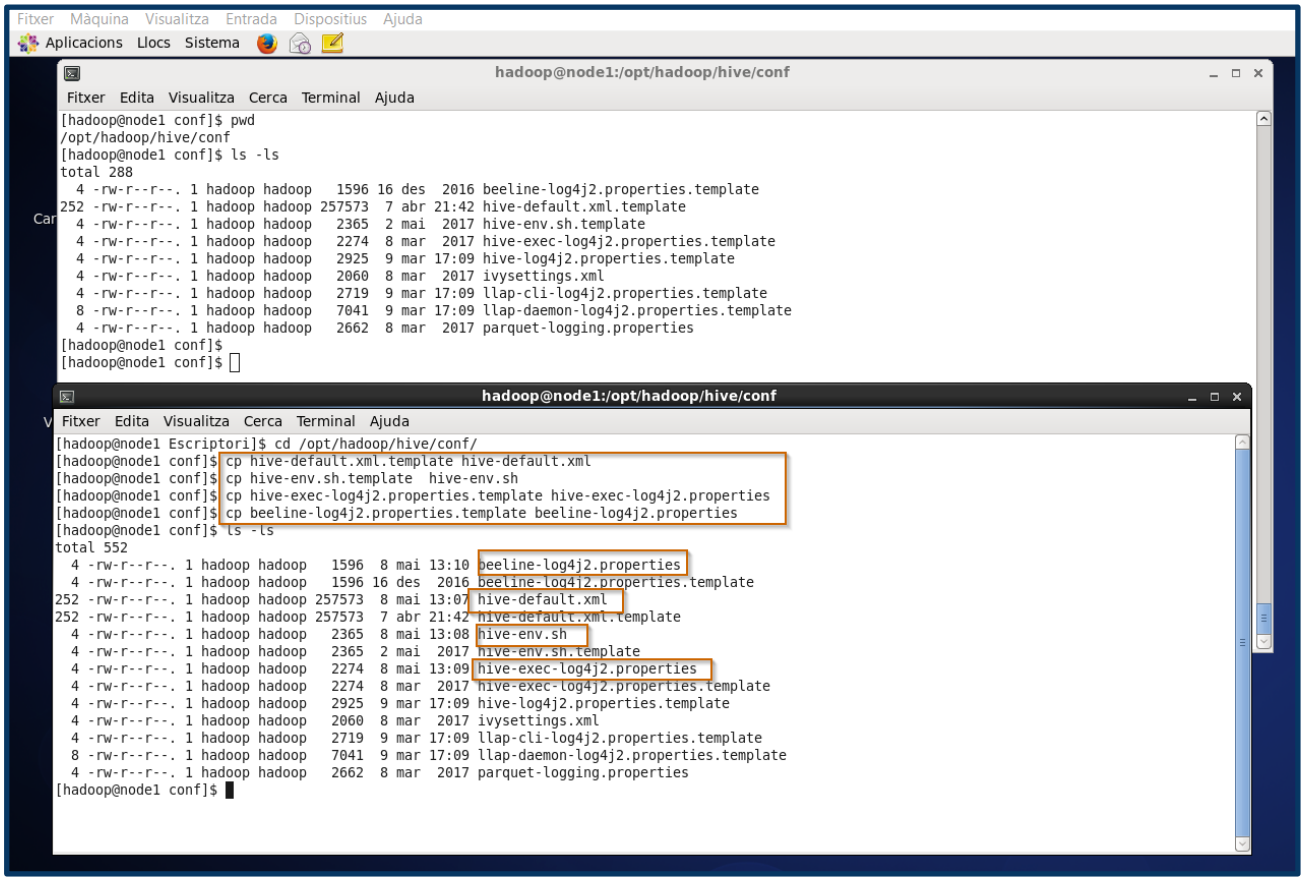


- Seguidament haurem d'editar l'arxiu de configuració de Hive ubicat al directori /opt/hadoop/hive/conf
  Com es pot observar, la majoria son templates



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

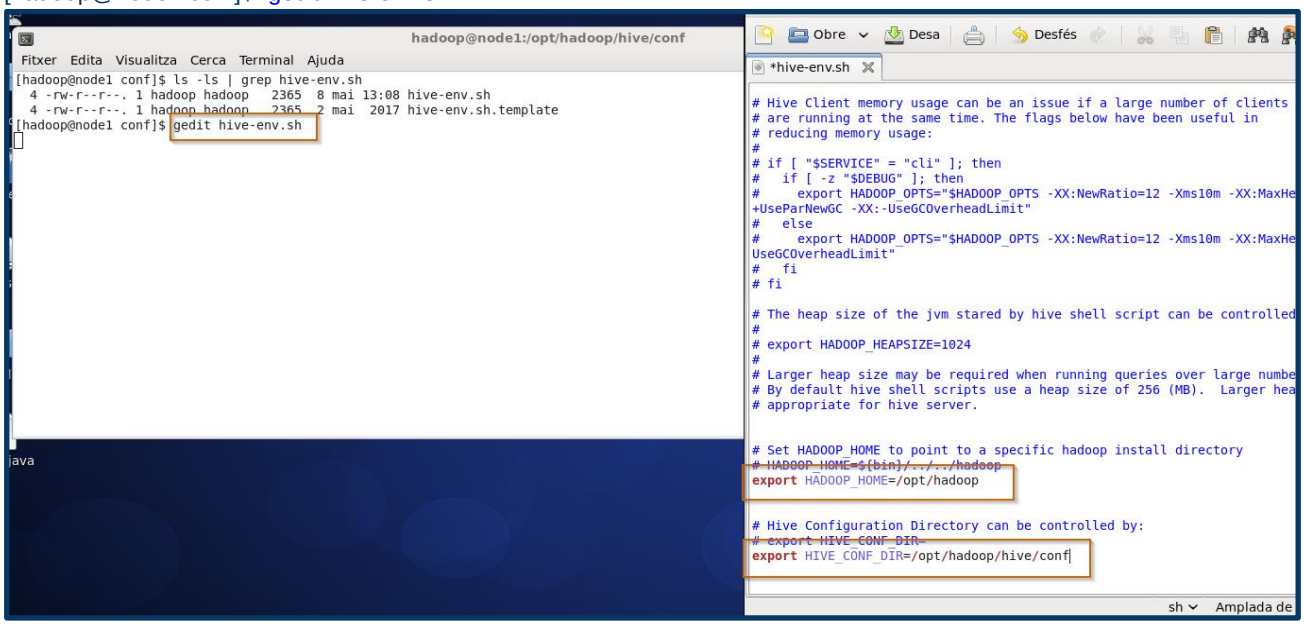
- Copiarem les templates que ens interessa modificar canviant el nom de l'arxiu (sense el template final)
  - [hadoop@node1 conf]$ cp hive-default.xml.template hive-site.xml
  - [hadoop@node1 conf]$ cp hive-env.sh.template hive-env.sh
  - [hadoop@node1 conf]$ cp hive-exec-log4j2.properties.template hive-exec-log4j2.properties
  - [hadoop@node1 conf]$ cp beeline-log4j2.properties.template beeline-log4j2.properties



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

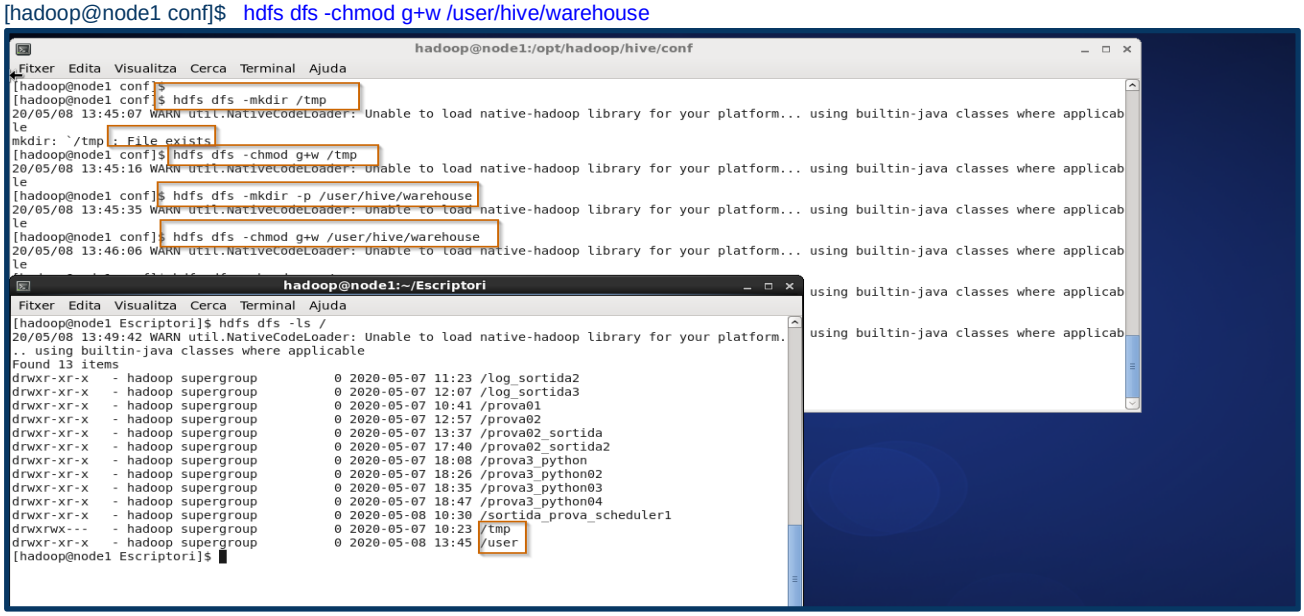
- ❑ Accedim a l'arxiu hive-env.sh per definir 2 variables:
  - HADOOP_HOME
  - HIVE_CONF_DIR

[hadoop@node1 conf]$ gedit hive-env.sh



- ❑ Posteriorment crearem 2 directoris HDFS (/tmp i /user/hive/warehouse) i li assignem permisos, ja que és el lloc (per defecte) treballa Hive. Si no els tenim, ens podria fallar.

[hadoop@node1 conf]$ hdfs dfs -mkdir **/tmp**
[hadoop@node1 conf]$ hdfs dfs -chmod g+w /tmp
[hadoop@node1 conf]$ hdfs dfs -mkdir -p **/user/hive/warehouse**
[hadoop@node1 conf]$ hdfs dfs -chmod g+w /user/hive/warehouse

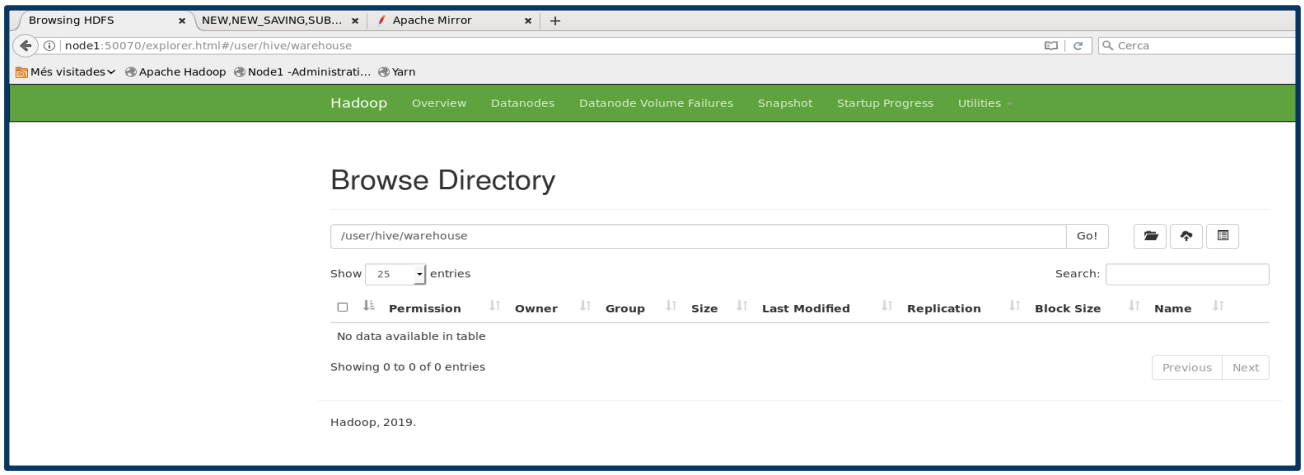


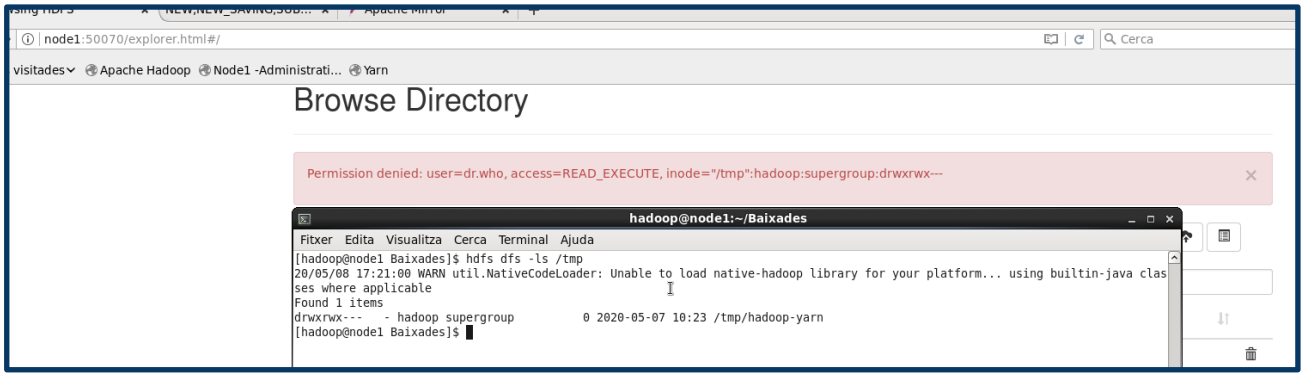
Nom i Cognoms

Arnau Subirós Puigarnau

Data

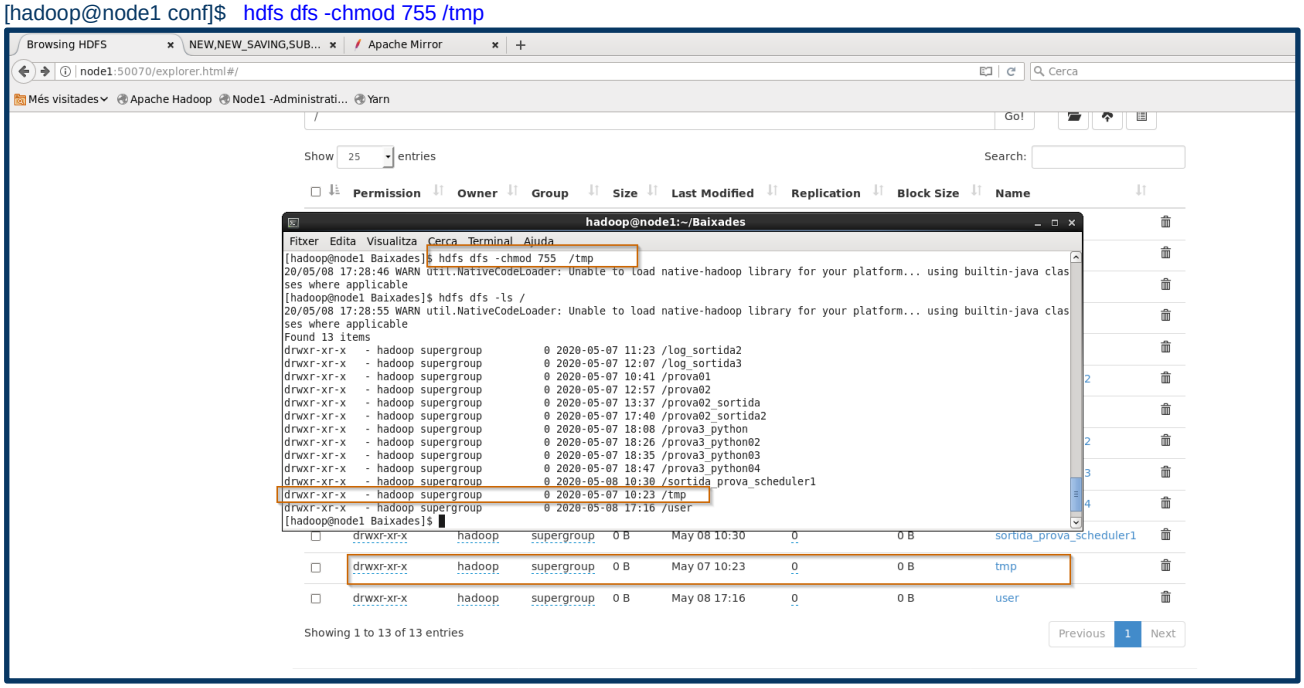
02-06-2020





- ❑ Com es pot veure no li havia donat tots els permisos. Modifico els permisos

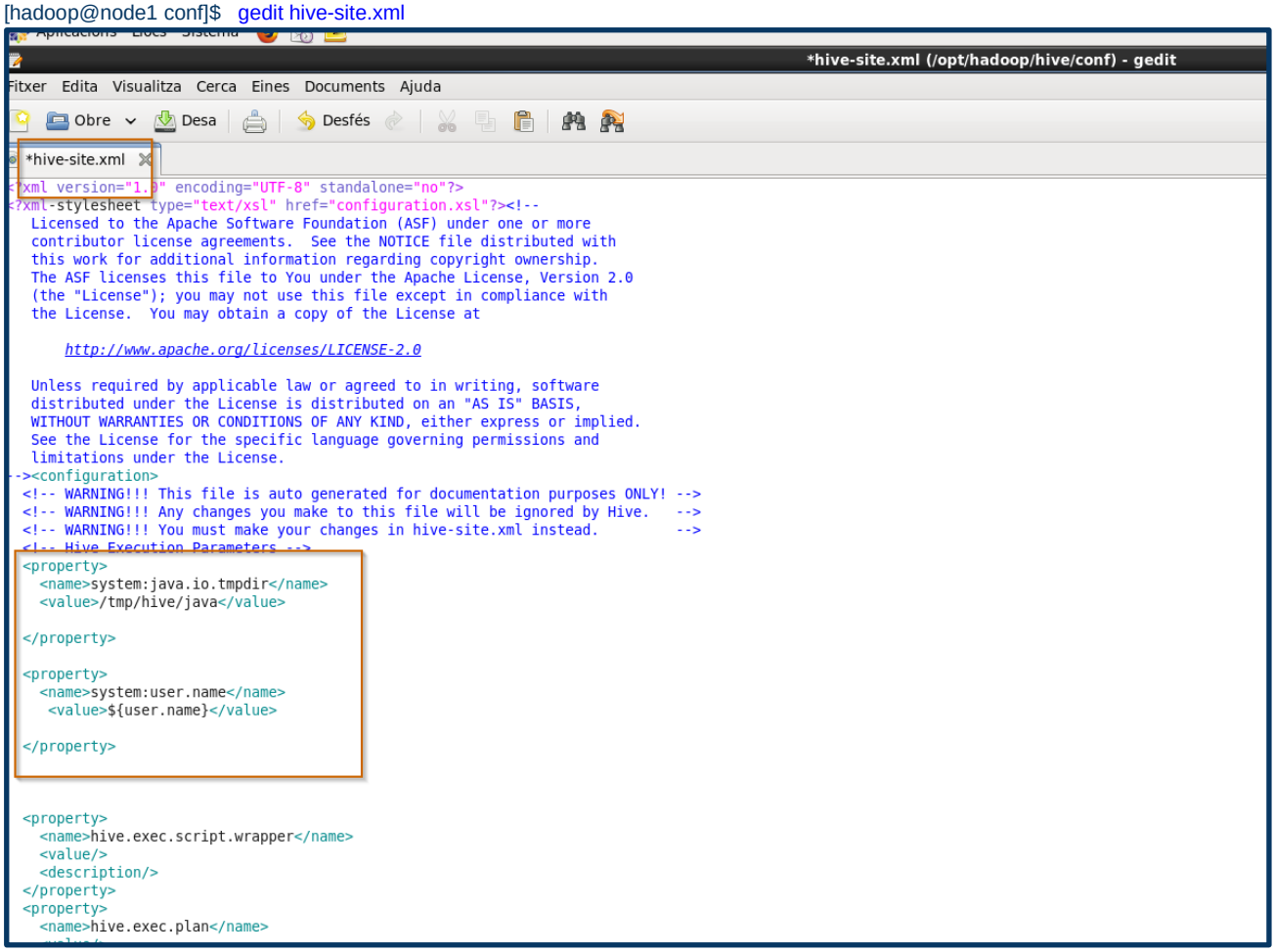
[hadoop@node1 conf]$ hdfs dfs -chmod 755 /tmp



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

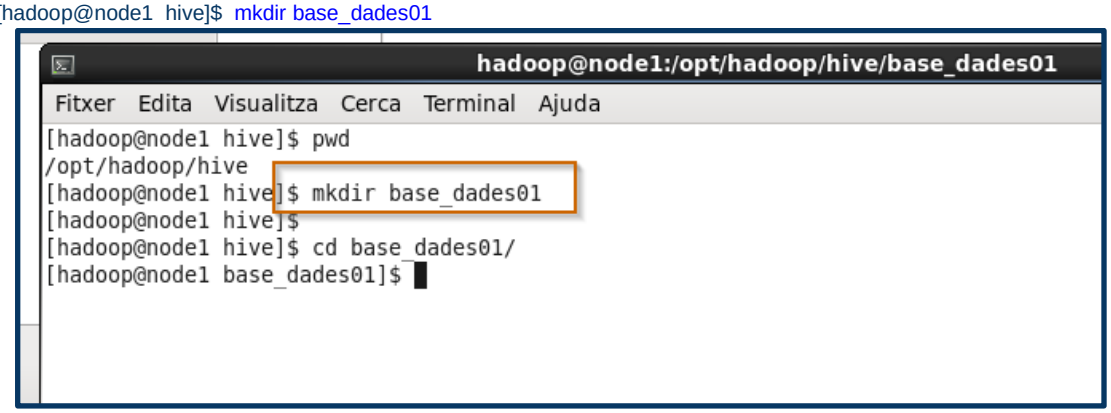
- Abans d'utilitzar un client Hive, haurem d'afegir unes propietats bàsiques en l'arxiu de configuració hive-site.xml

[hadoop@node1 conf]$ gedit hive-site.xml



- Anirem al directori principal de Hive i crearem un directori anomenat base_dades01 on treballarem

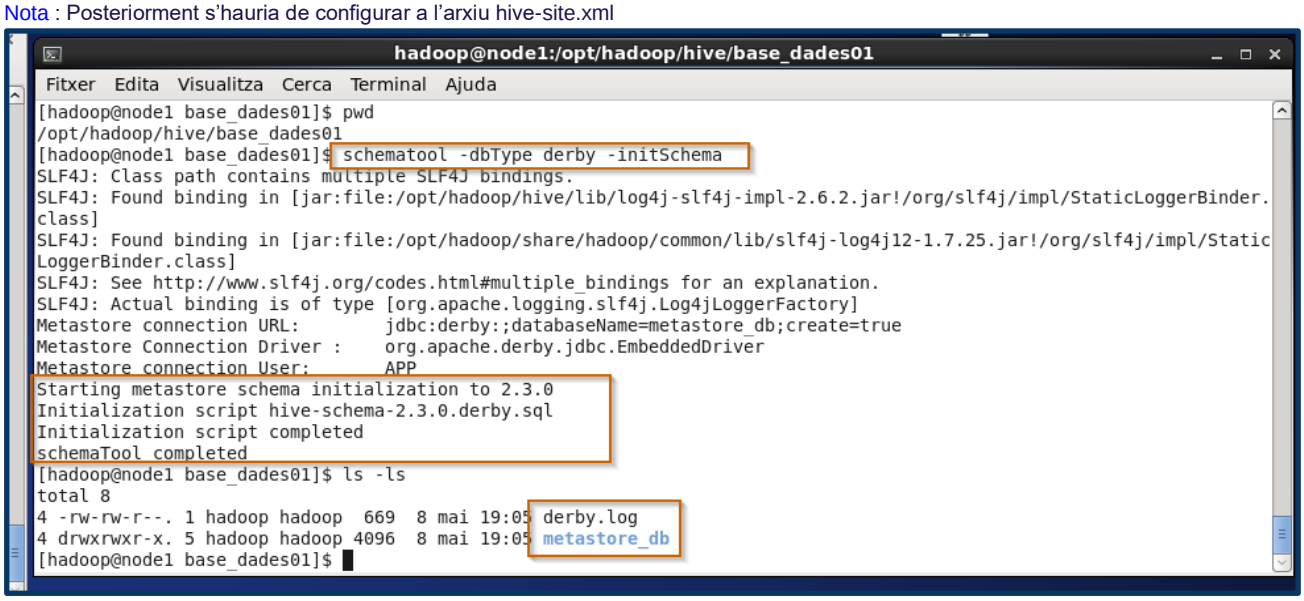
[hadoop@node1 hive]$ mkdir base_dades01



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

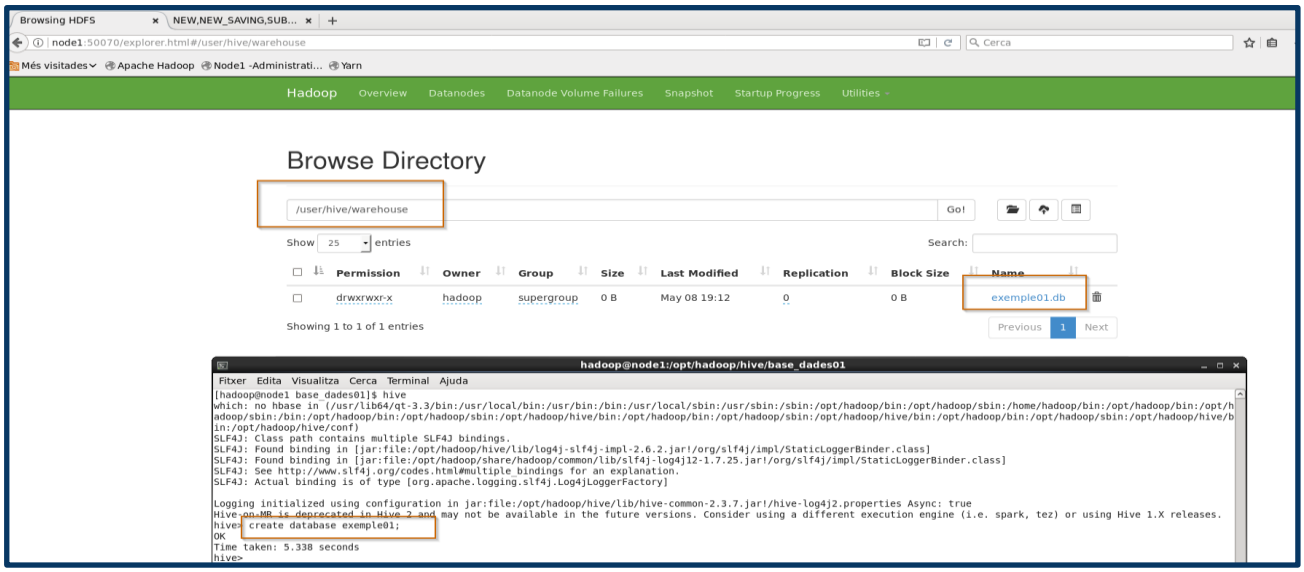
- El tipus de Hive que farem servir és "encastat" on per defecte utilitza la BBDD Derby. Hem d'inicialitzar la base de dades en mode local.

[hadoop@node1 hive]$ schematool -dbType derby -initSchema
Nota : Posteriorment s'hauria de configurar a l'arxiu hive-site.xml



- Creem una base de dades anomenada exemple01 i ho visualitzem per l'interfície web del Namenode

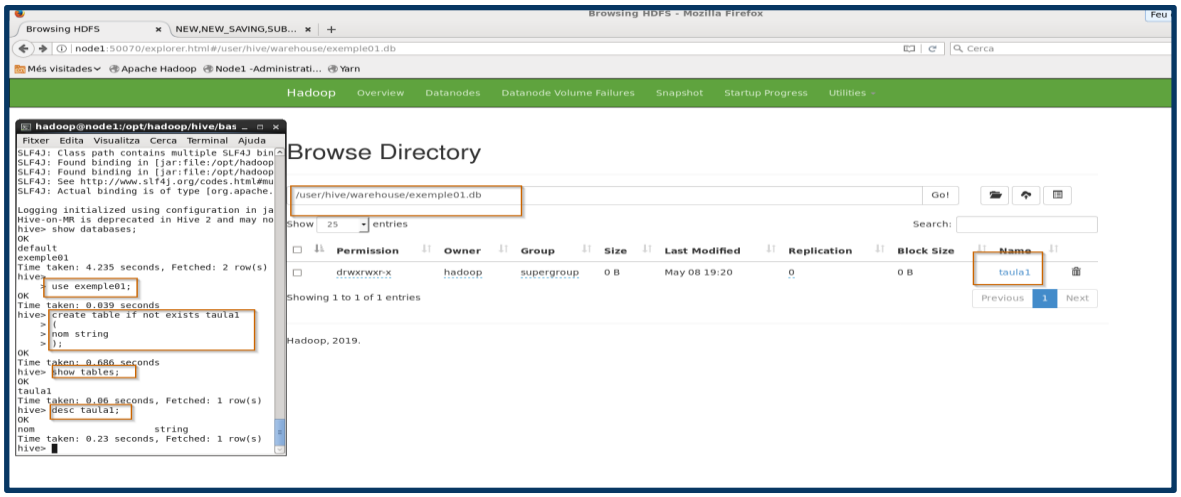
[hadoop@node1 hive]$ hive
hive> create database exemple01
node1:50070



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

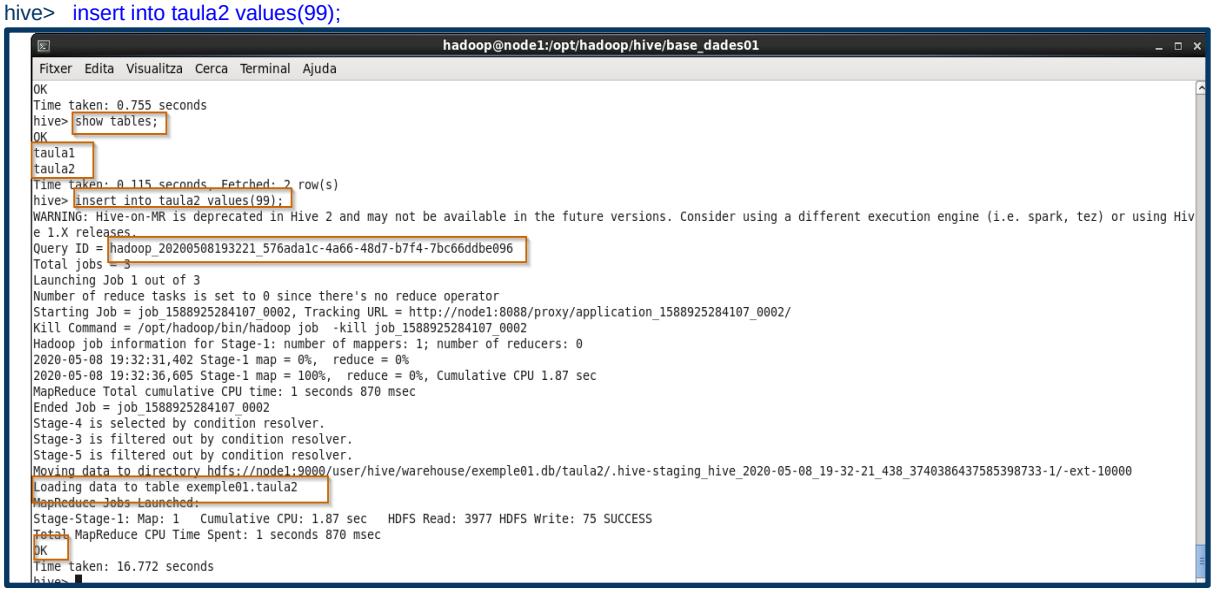
- Farem un exemple senzill, creant una taula i ho visualitzem per l'interfície web

```
[hadoop@node1  hive]$  hive
hive>   use  exemple01;
hive>   create table if not exists taula1;
hive>  desc taula1;
node1:50070
```



- Crearem una altra taula i farem un insert on veurem que llença un job Mapreduce

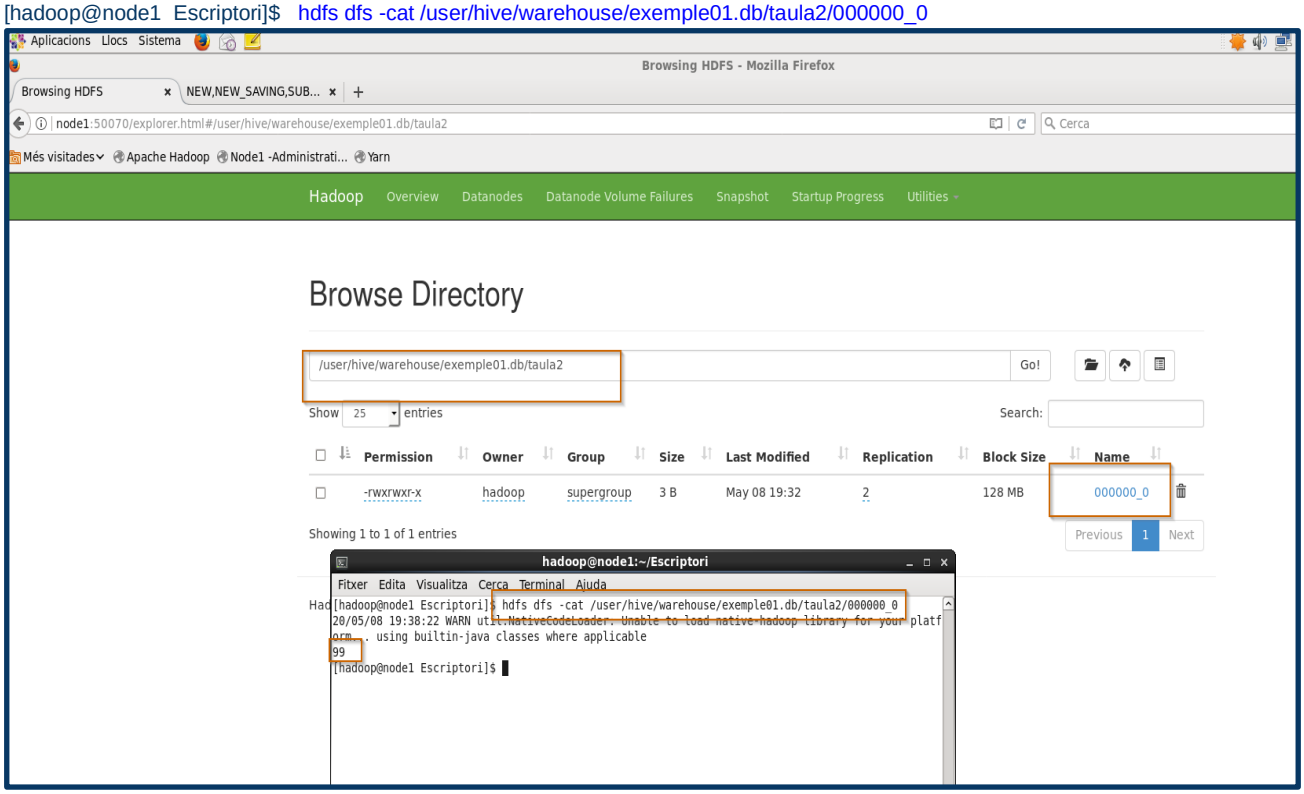
```
hive>   create table if not exists taula2;
hive>   show tables;
hive>   insert into taula2 values(99);
```



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Com que el resultat és text pla, desde HDFS ho podem visualitzar.

```
[hadoop@node1  Escriptori]$  hdfs dfs -cat /user/hive/warehouse/exemple01.db/taula2/000000_0
```



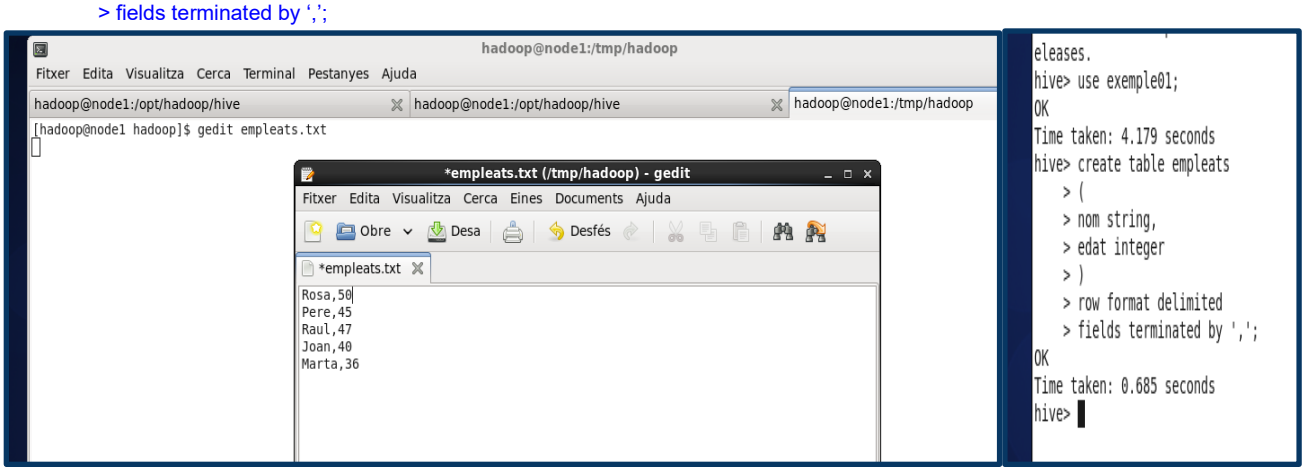
- Hem de diferenciar entre taules internes i taules externes.
  - Les **taules internes** tot el seu cicle de vida (informació que contenen, ho gestiona el mateix Hive. Si crees una taula amb dades, si elimino la taula , des de Hive, elimini els fitxers associats).
  - Si la **taula** és **externa,** segurament ja tinc unes dades que s'han carregat d'una alta forma i segurament estan essent utilitzats per altres productes i Hive també pot que utilitzar-les .En aquest cas, no és Hive que controla el cicle de vida de la taula, per tant si esborro la taula, no s'esborren les dades.

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

❑ Creació d'una taula interna

- Creació d'un arxiu de text (empleats.txt) pla on hi han els noms i les edats separats per comes
- Creació d'una taula anomenada "empleats" amb les variables "nom" i "edat"

```
[hadoop@node1  hadoop]$  gedit empleats.txt
[hadoop@node1  hadoop]$  hive
hive>    use exemple01;
hive>   create table empleats
      > (
      > nom string,
      > edat integer,
     > row format delimited
> fields terminated by ',';
```



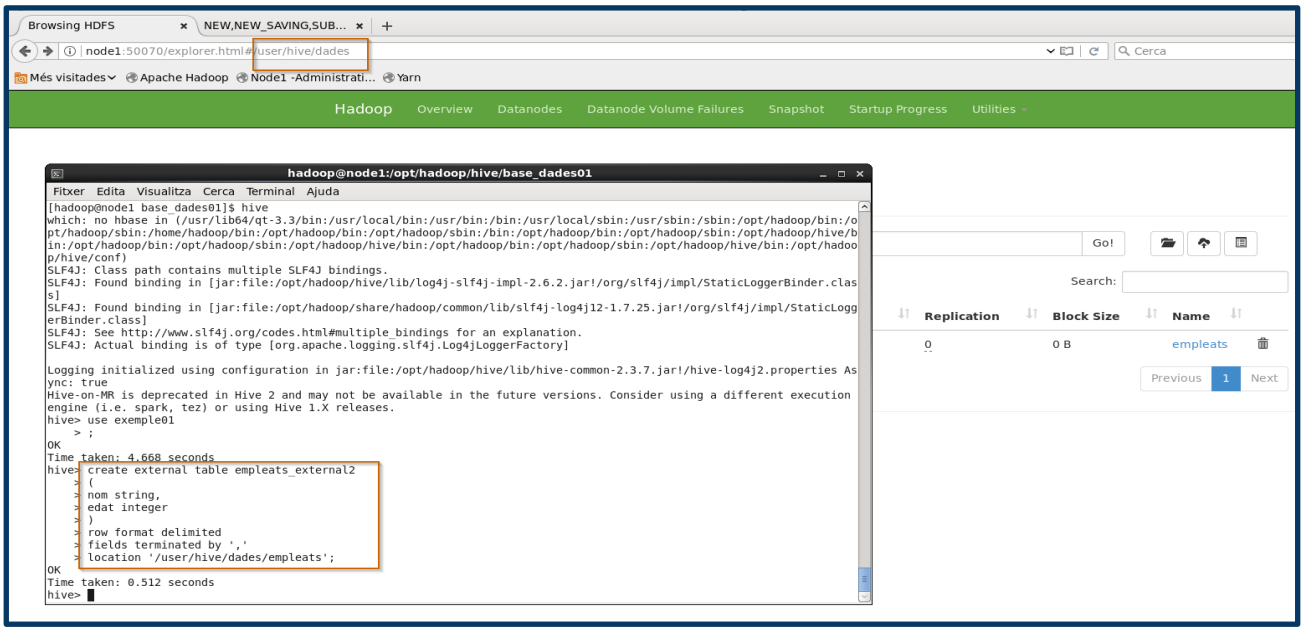
- Seguidament carreguem la taula empleats i l'associem a l'arxiu empleats.txt ubicat a /tmp/hadoop . Posteriorment ho visualitzem en la interfície web del Namenode.
- També visualitzem el resultat en HDFS ja que és un text pla.

NOTA: Com que és una taula interna, si fes un "drop" no només eliminaria la taula sinó els fitxers HDFS associats a ella.

```
hive>    load data local inpath '/tmp/hadoop/emplets.txt' into table empleats;
hive>  select  * from empleats;
[hadoop@node1  hadoop]$   hdfs dfs -cat /user/hive/warehouse/exemple01.db/empleats/empleats.txt
```

Nom i Cognoms | **Data**

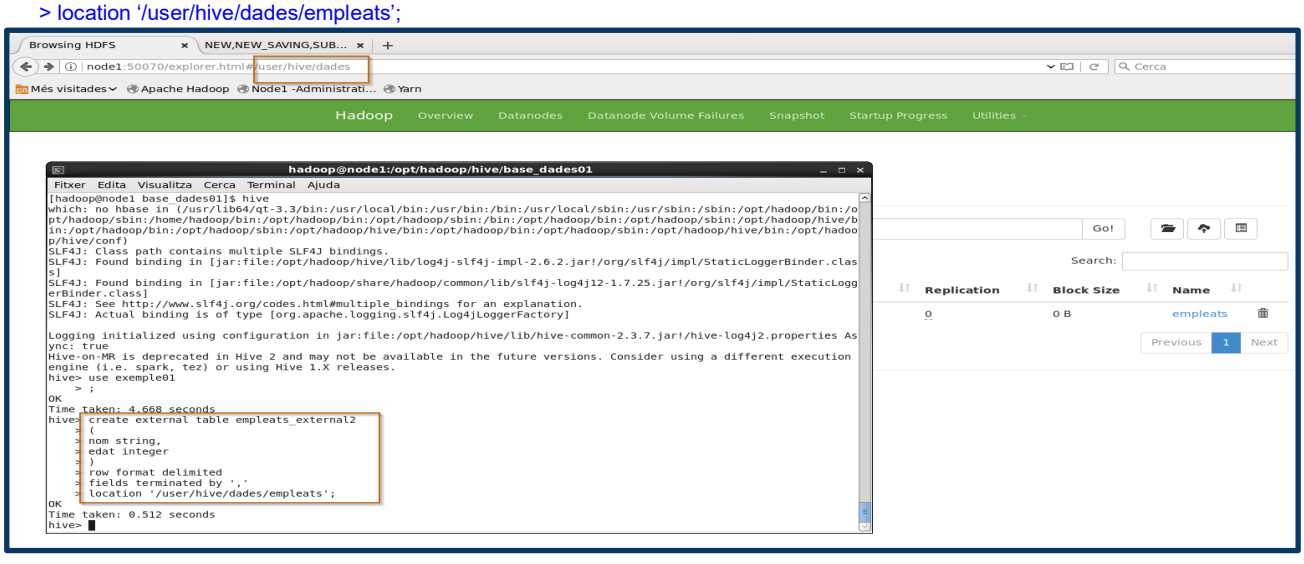
Arnau Subirós Puigarnau | 02-06-2020



❑ Creació d'una taula externa

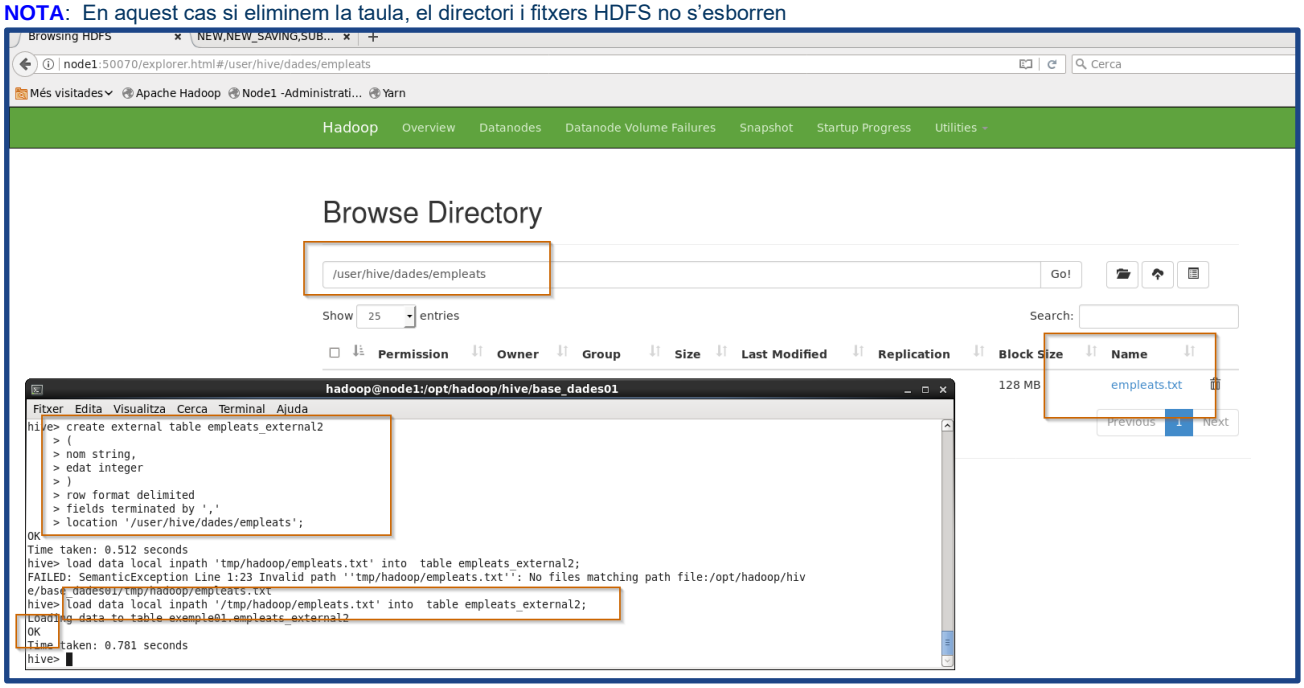
- Creació d'una taula externa ubicada a /user/hive/dades/empleats
- Farem servir el mateix arxiu "emplats.txt"

```
[hadoop@node1  hadoop]$  hive
hive>    use exemple01;
hive>   create  external table empleats_external2
      > (
      > nom string,
      > edat integer,
     > row format delimited
     > fields terminated by ','
    > location '/user/hive/dades/empleats';
```




```
[hadoop@node1  hadoop]$ load data local inpath '/tmp/hadoop/empleats.txt' into table empleats_external2;
```

NOTA: En aquest cas si eliminem la taula, el directori i fitxers HDFS no s'esborren

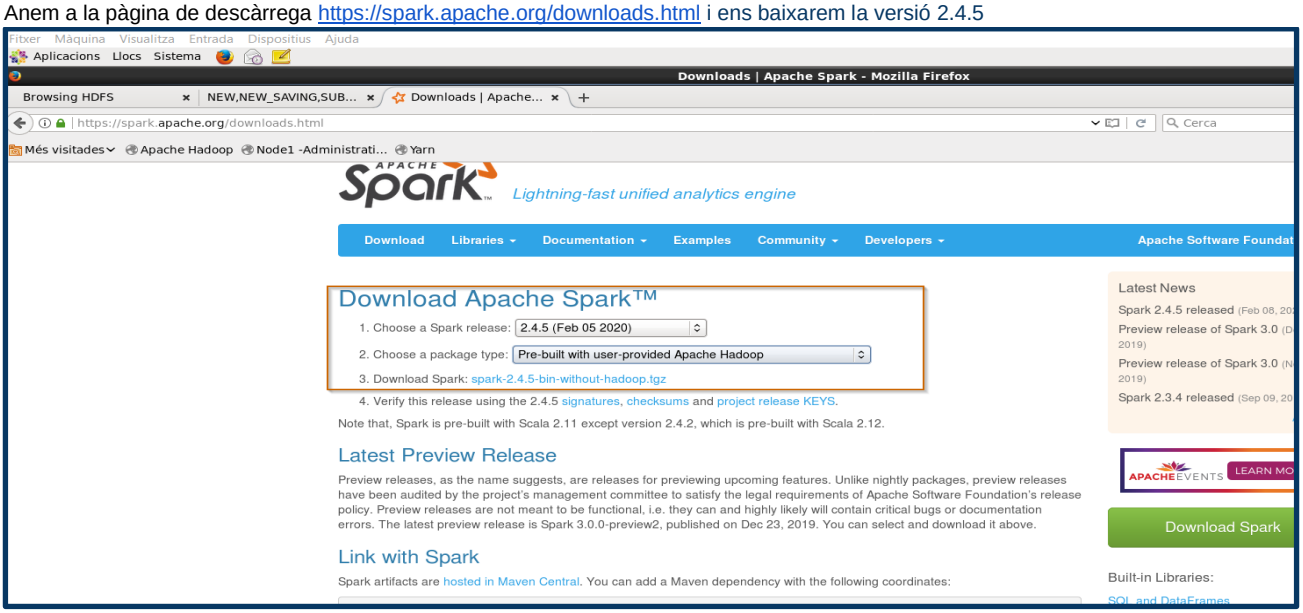


3.2.6.2.2. Instal.lació i configuracio de Spark

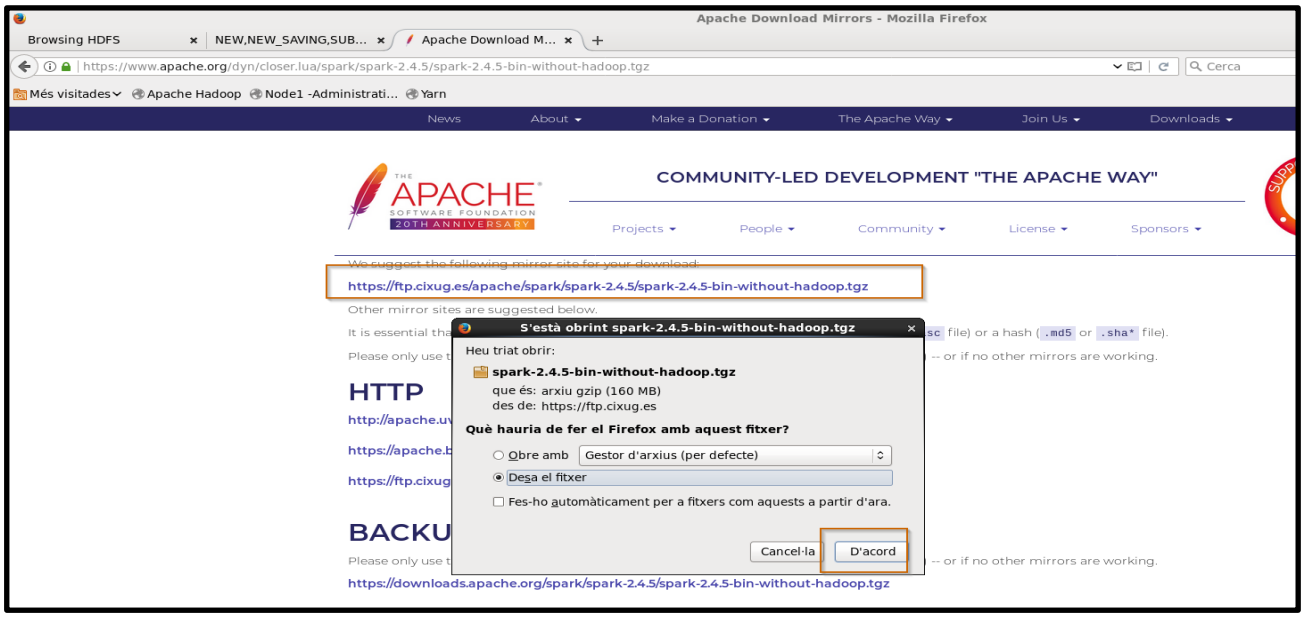
- ❏ En les opcions de descàrrega, com que ja tenim instal·lat Hadoop, seleccionarem el binari sense Hadoop per poder utilitzar-lo en el nostre clúster.

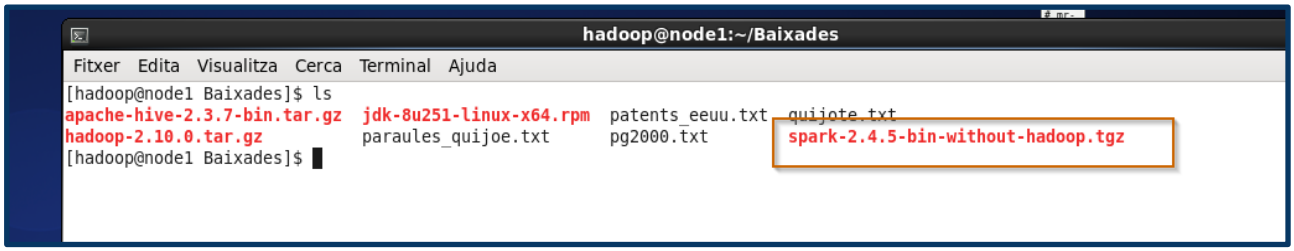
- ❏ Requisits mínims: Java 8 i Python 2.7

Anem a la pàgina de descàrrega https://spark.apache.org/downloads.html i ens baixarem la versió 2.4.5



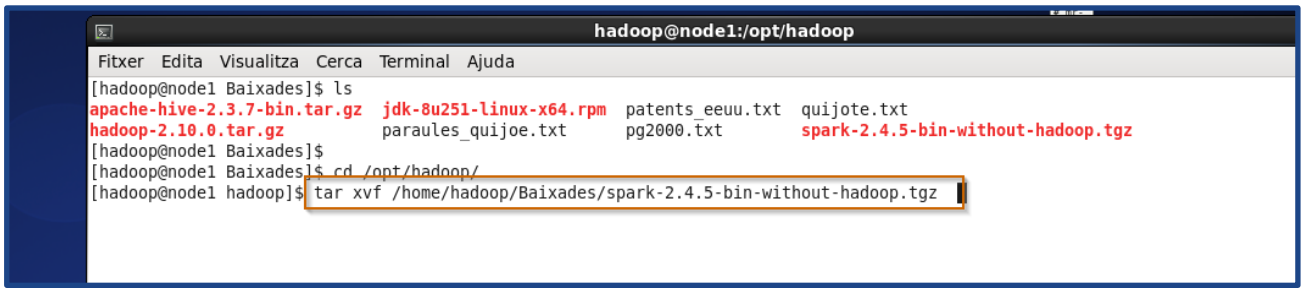
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

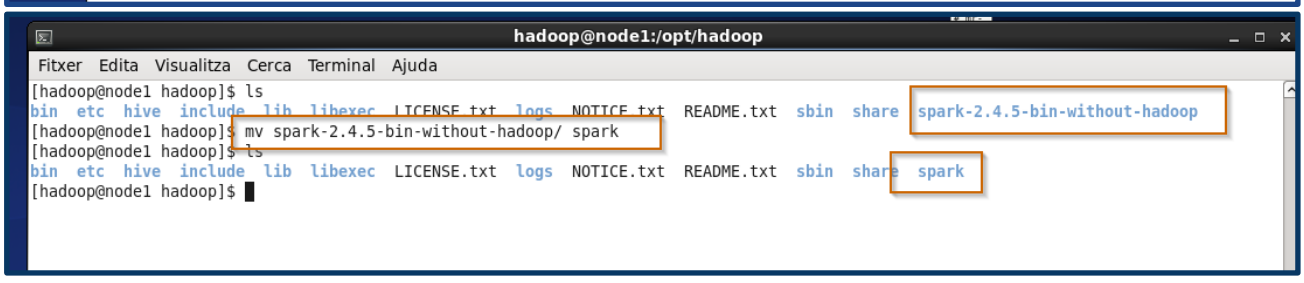




- ❑ Descomprimim l'arxiu a /opt/hadoop i posteriorment li canviarem el nom, ja que el nom del directori és molt llarg

```
[hadoop@node1  hadoop]$  tar xvf  /home/hadoop/Baixades/spark-2.4.5-bin-without-hadoop.tgz
[hadoop@node1  hadoop]$ mv spark-2.4.5-bin-without-hadoop/      spark
```

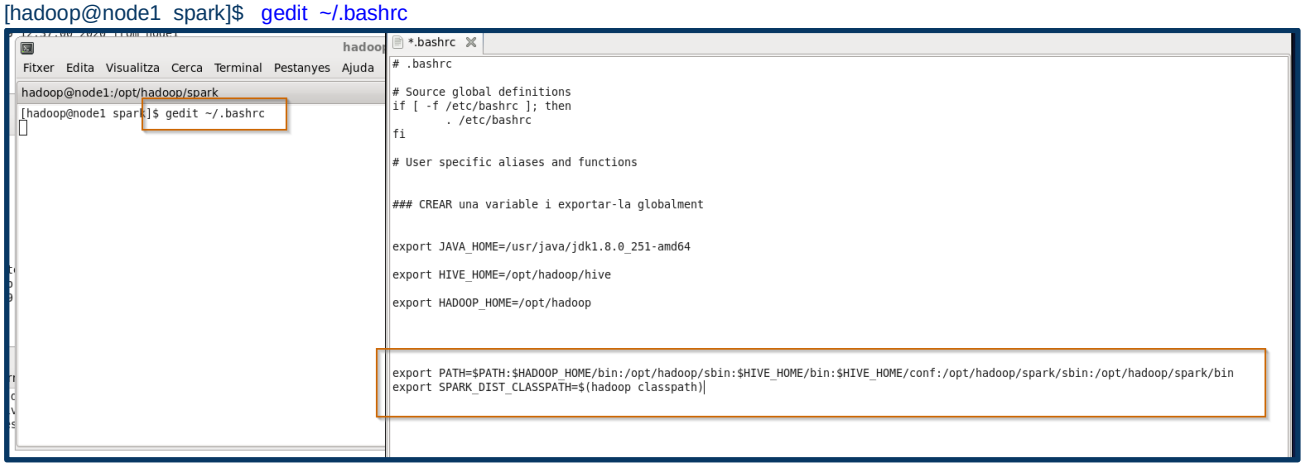




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- ❏ Haurem d'accedir a l'arxiu ~/.bashrc per afegir les variables d'entorn de Spark

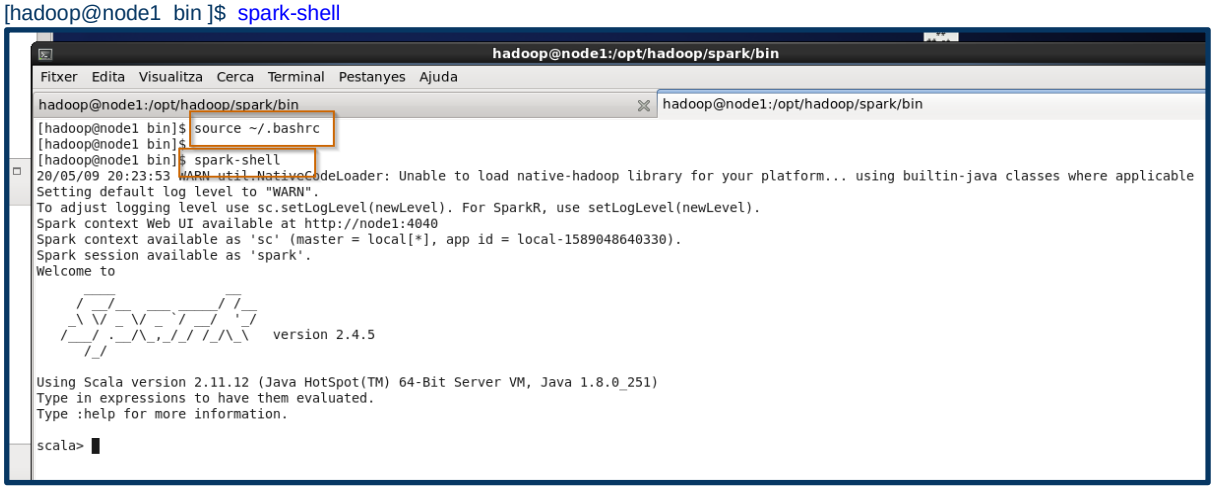
[hadoop@node1 spark]$ gedit ~/.bashrc



Exemple amb Spark Shell

- ❏ Carreguem les variables d'entorn i obrim un terminal Spark-Shell on crea un context Spark i després crida l'entorn de línia d'ordres de Scala

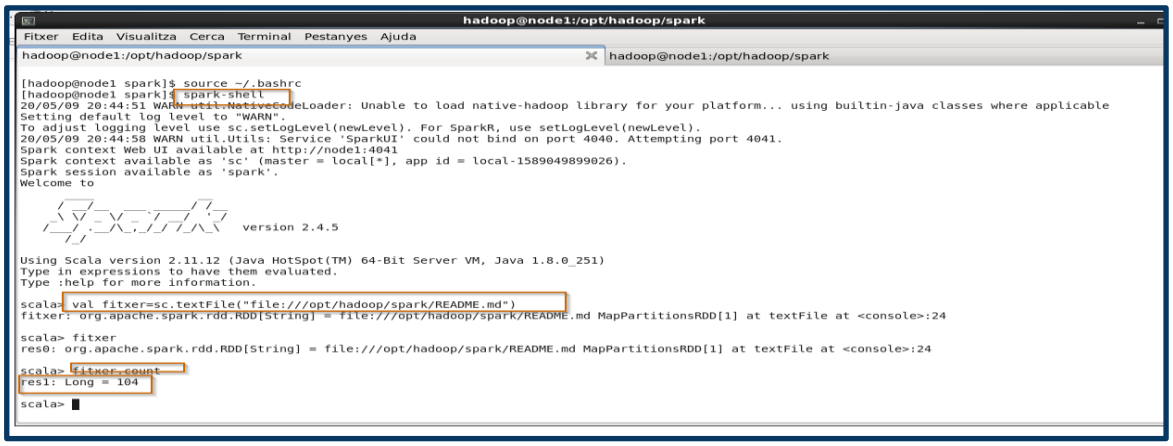
[hadoop@node1 bin]$ source ~/.bashrc
[hadoop@node1 bin]$ spark-shell



- ❏ Farem un exemple senzill, llegirem un fitxer de text readme que ja ve incorporat amb el mateix Spark i contarem el número de línies
  - crearem una variable Scala anomenada fitxer (val fitxer)
  - A traves del context(sc) afegirem un comando " textFile" per accedir a un fitxer de text
  - Com que ens interessa un fitxer local(per defecte ho busca a HDFS) hem d'indicar "file://+path complet"
  - Associem el fixer a la variable fitxer

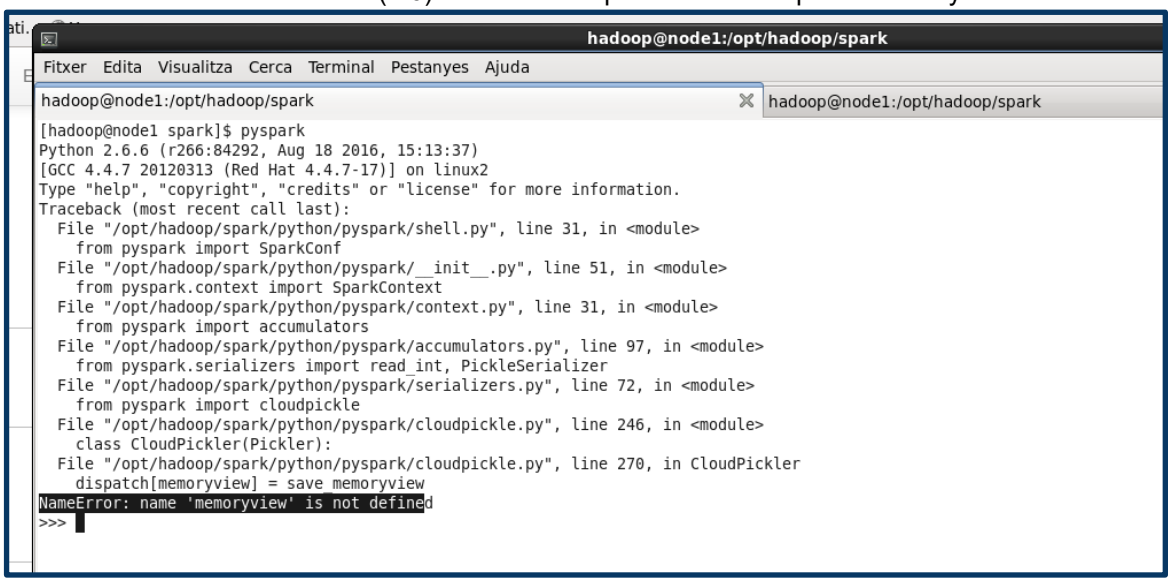
[hadoop@node1 bin]$ spark-shell
scala> val fitxer=sc.textFile("file:///opt/hadoop/spark/README.md")
scala> fitxer.count

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



Exemple amb Pyspark

- Volem fer el mateix però amb Pyspark, en lloc de Scala utilitzant Python. Però em dóna error, ja que tinc una versió inferior a 2.7 (2.6) i es la versió que utilitza els repositoris de "yum"

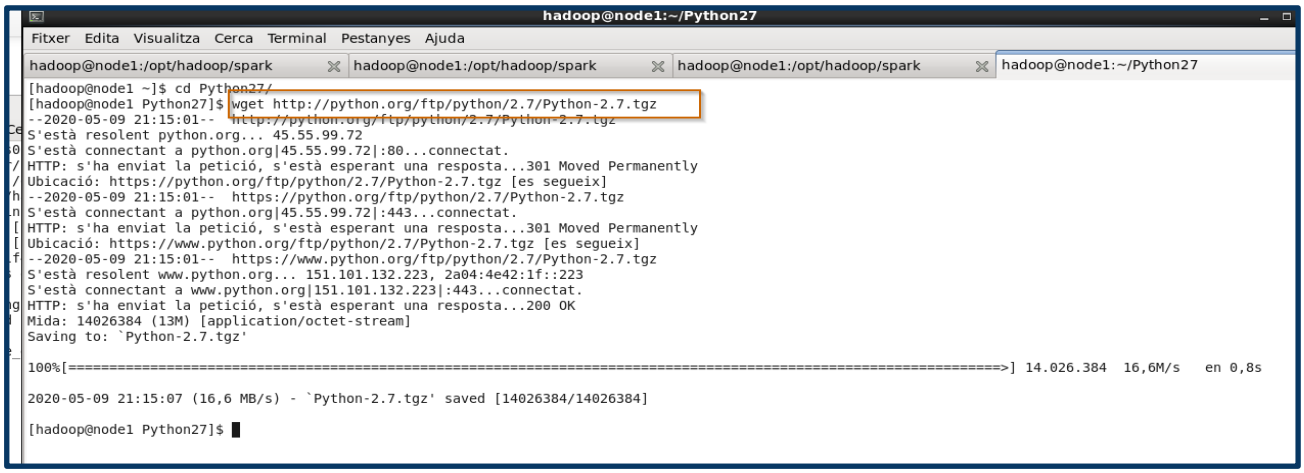


- Instal·lem Python 2.7 i els paquets necessaris. En acabar ens assegurem la versió de Python que estem utilitzant

```
[hadoop@node1  ~]$  yum -y groupinstall "Development Tools"
[hadoop@node1  ~]$  yum -y install openssl-dev* ncurses-devel* zlib*.x86_64
[hadoop@node1  ~]$  yum -y install  bzip2 bzip2-devel bzip2-libs
[hadoop@node1  ~]$   mkdir Python27
[hadoop@node1  ~]$   cd Python27/
[hadoop@node1   Python27]$   wget http://python.org/ftp/python/2.7/Python-2.7.tgz
[hadoop@node1  Python27]$    tar xvf  Python-2.7.tgz
[root @node1  ~]$  which python
```

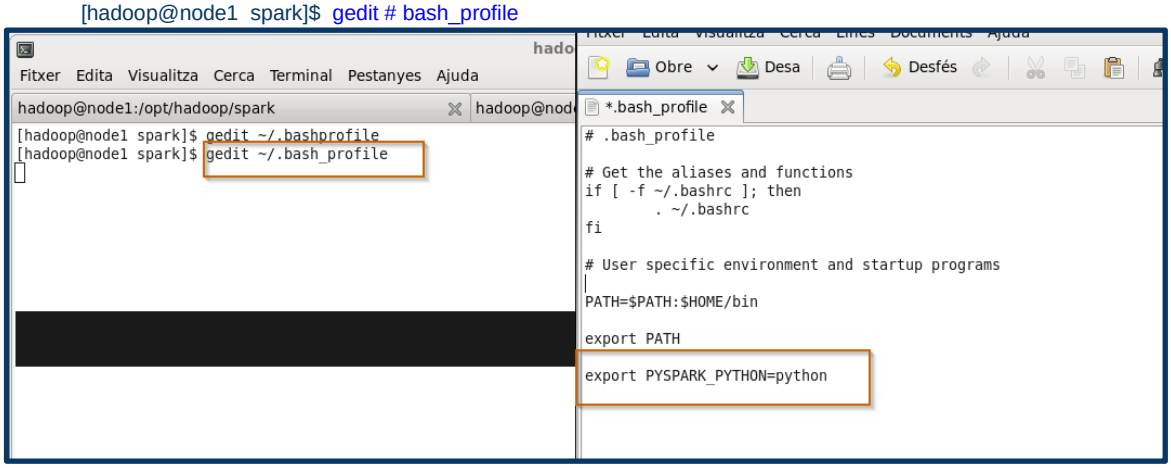
Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020



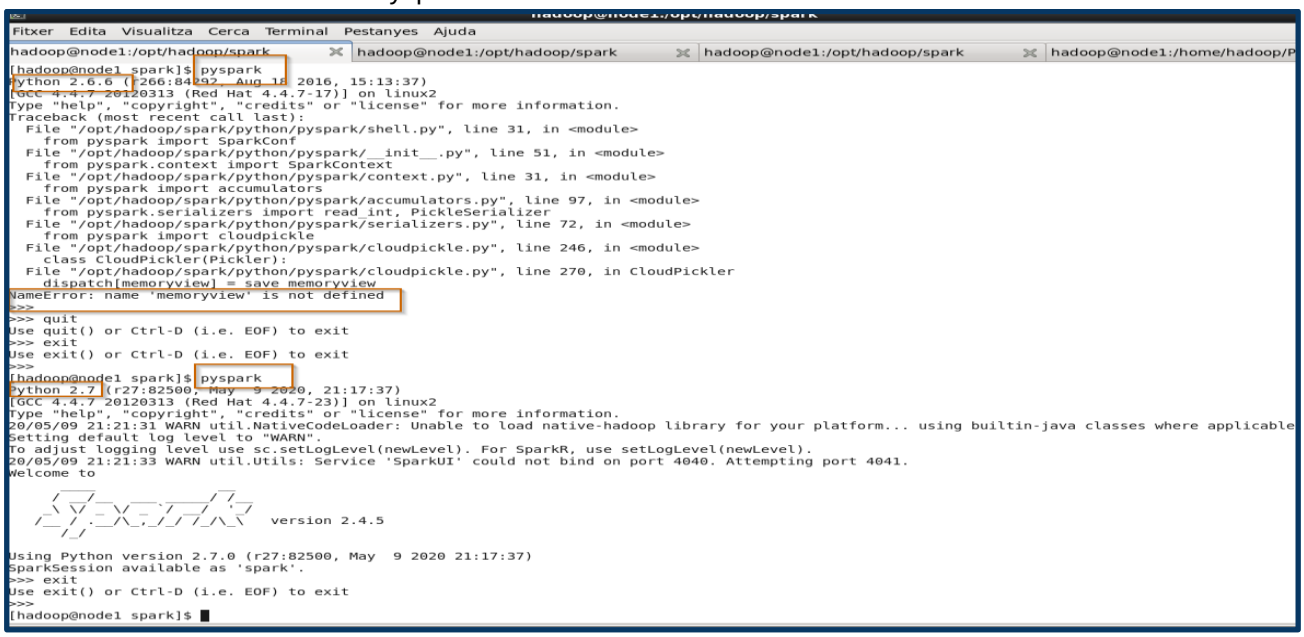
- Un cop instal.lat afegim una nova variable d'entorn però a ~/.bash_profile

[hadoop@node1 spark]$ gedit # bash_profile



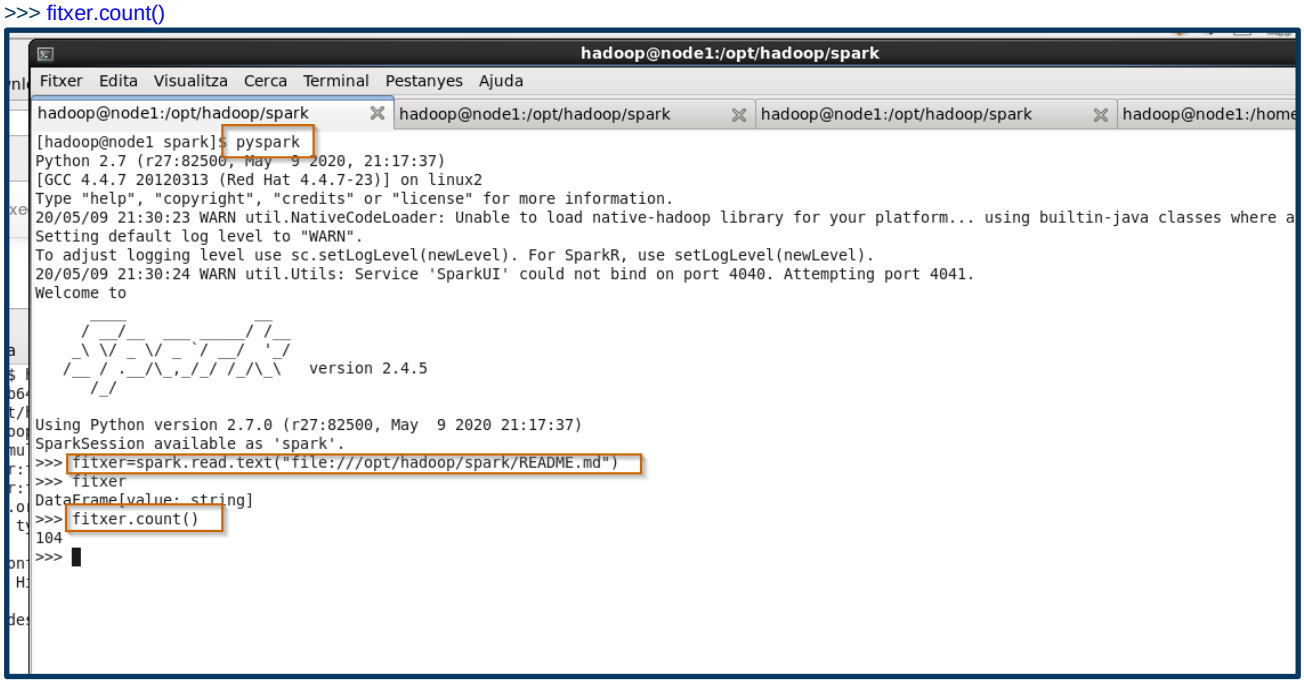
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- ❏ Ara tornem a executar Pyspark



- ❏ Farem un exemple de prova (el mateix que hem fet anteriorment amb Scala)

```
[hadoop@node1  bin ]$  pyspark
>>>  fitxer=spark.red.text("file:///opt/hadoop/spark/REARME.md")
>>> fitxer.count()
```



Nom i Cognoms | **Data**

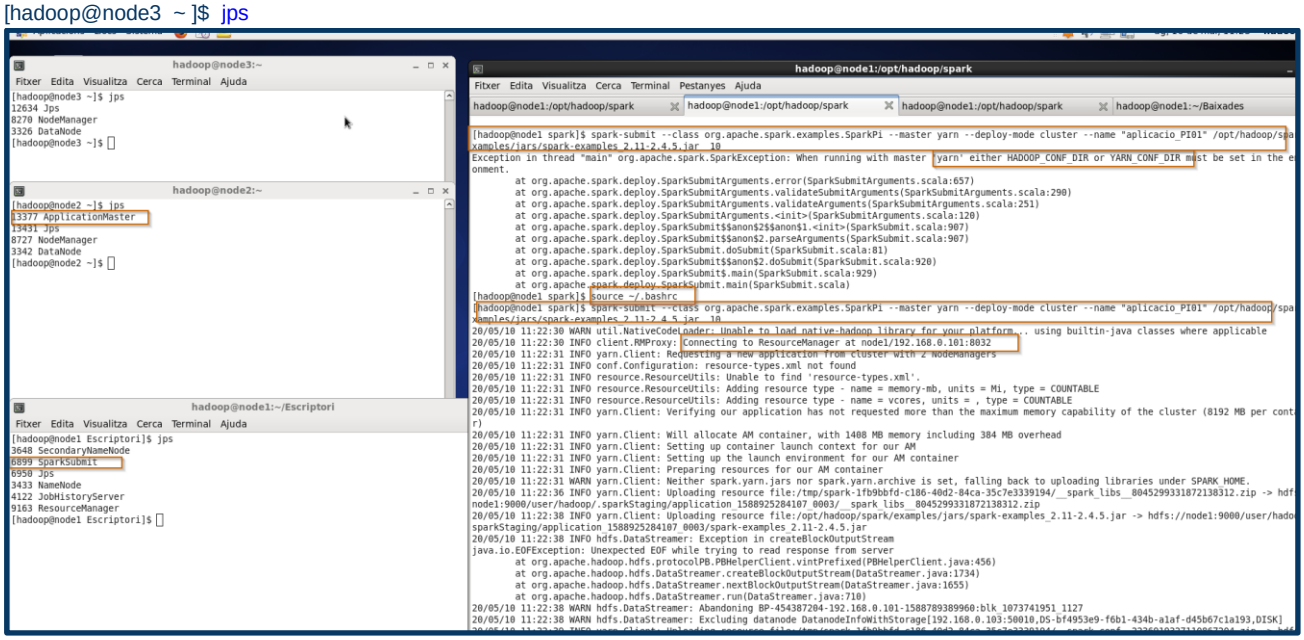
Arnau Subirós Puigarnau | 02-06-2020

Exemple amb Spark Shell interactuant amb HDFS

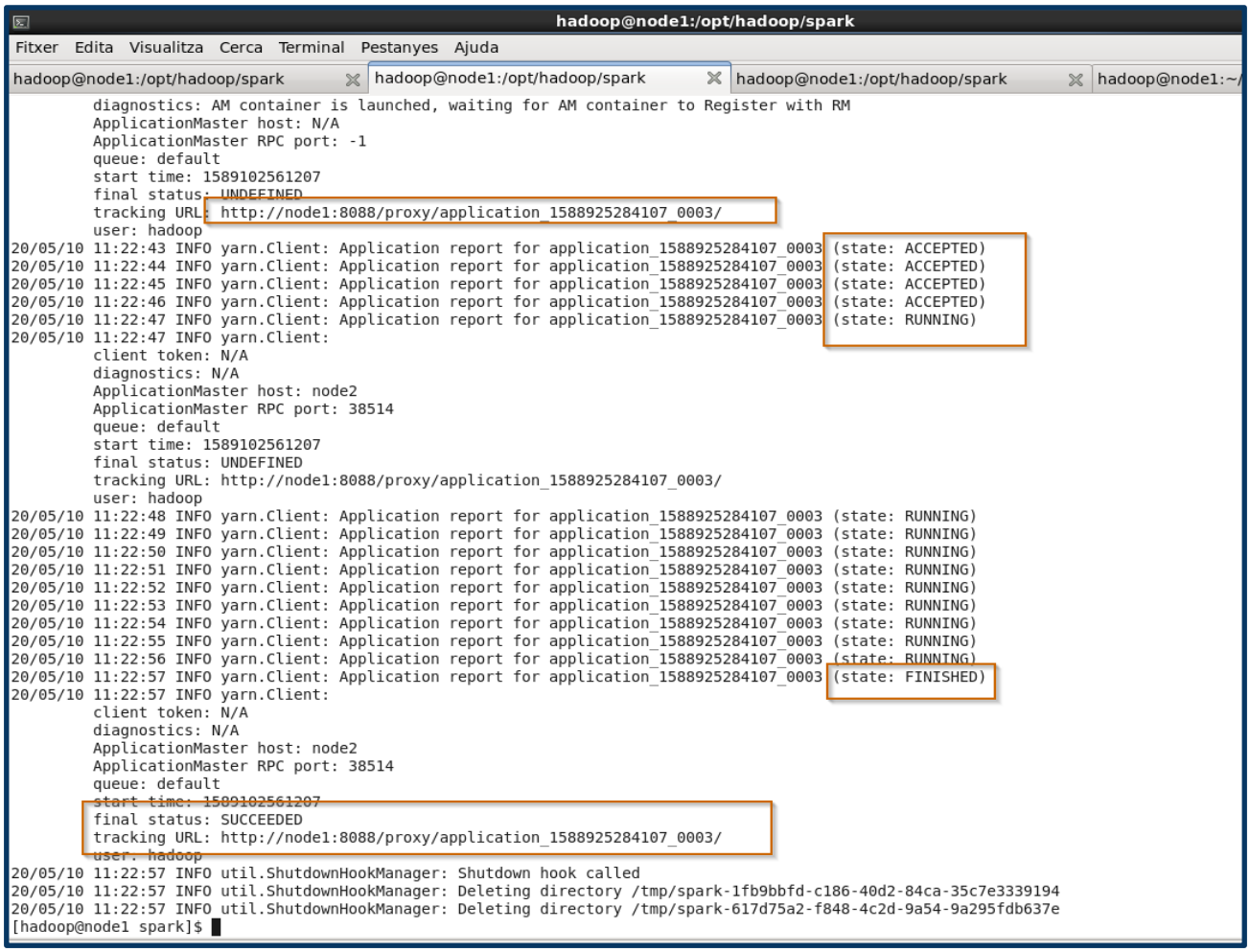
- Per llençar comandes de Spark contra el clúster haurem d'utilitzar la comanda spark-submit
- A continuació mostrem els arguments utilitzats:
  - -class : utilitzarem un exemple que serveix per calcular el numero Pi
  - --master yarn : Estem dient que executarem aquest comando spark contra el nostre clúster Hadoop (de tipus yarn)
  - --deploy-mode cluster : S'ha d'especificar ja que sino treballaria en mode local
  - --name : Per posar un nom a l'aplicació
  - després haurem d'especificar el path del fitxer + 1 un número (com més gran sigui, més operacions farà i més precís serà)

- Com que ens dóna error, ja que no troba el context, tornem a carregar les variables d'entorn. I executarem jps en tots els nodes per observar els processos que 'estan executant.

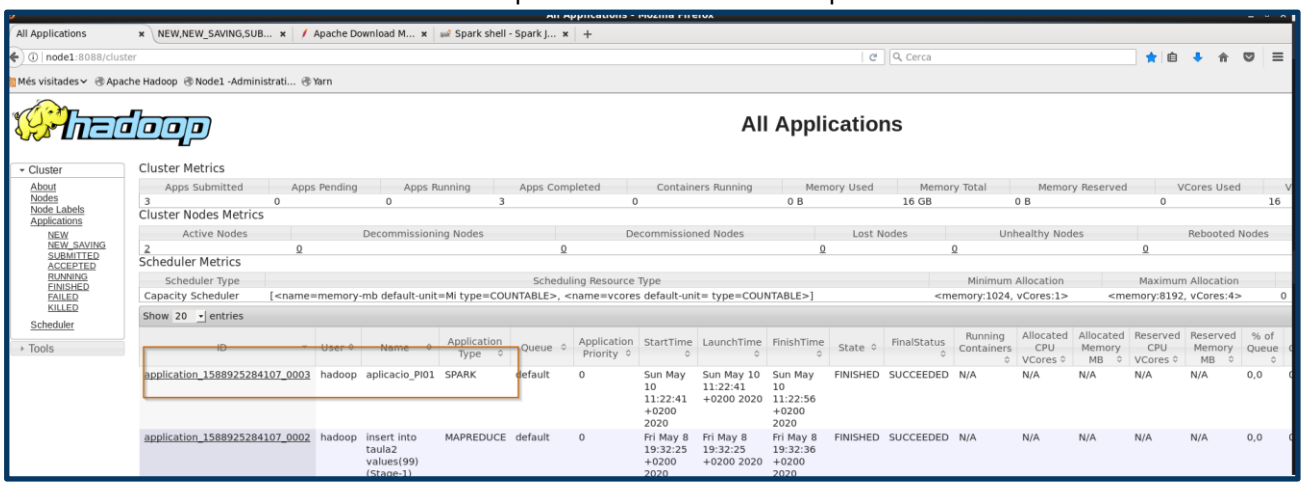
```
[hadoop@node1 spark ]$ source ~/.bashrc
[hadoop@node1 spark ]$ spark-submit --class org.apache.spark.exemplesSparkPi --master yarn --deploy-mode cluster --name "aplicacio_PI01" /opt/hadoop/spark/examples/jars/spark-examples_2.11-2.4.5.jar 10

[hadoop@node1 spark ]$ jps
[hadoop@node2 ~ ]$ jps
[hadoop@node3 ~ ]$ jps
```



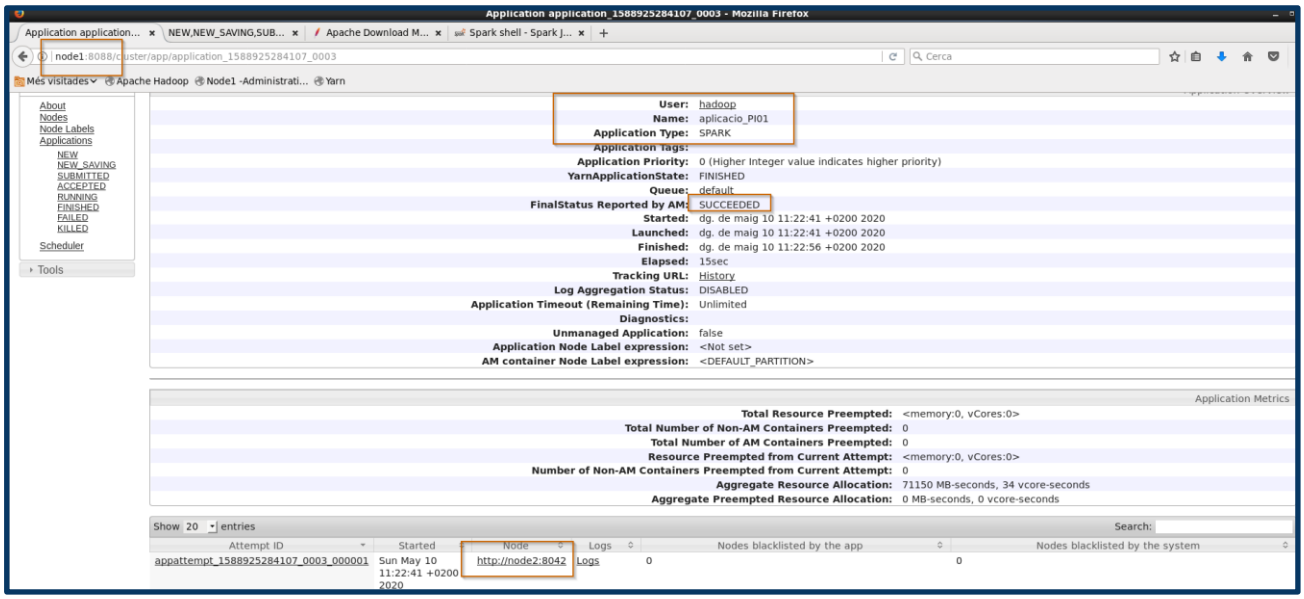
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

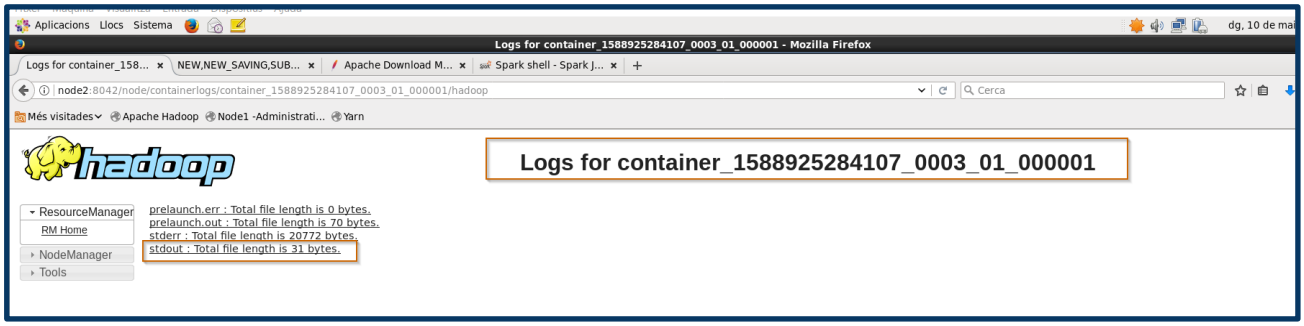


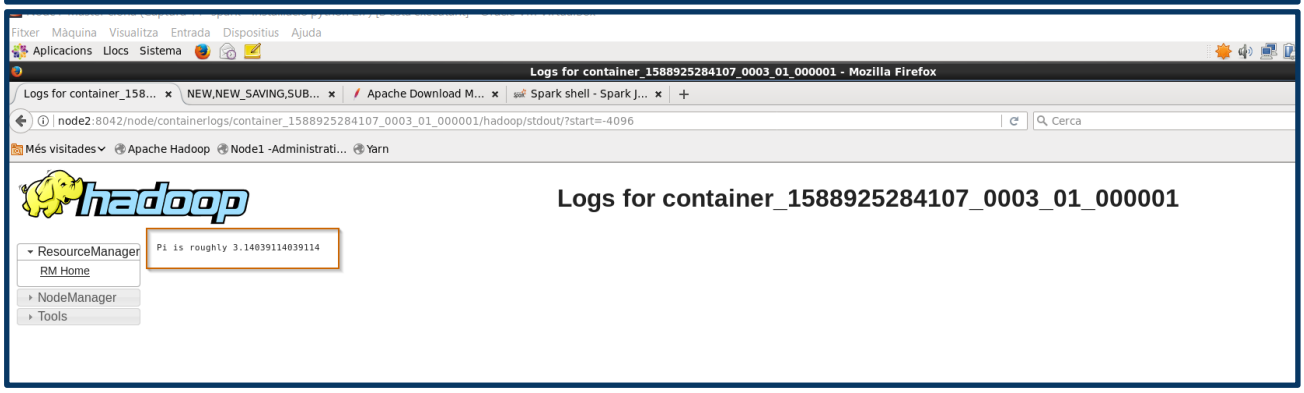
- ❑ Accedim a l'interfície web del Yarn per veure informació de l'aplicació



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020







Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

3.3. Clúster Hadoop amb Ambari

- En aquest clúster tornarem a fer servir la màquina virtual node1 on només té el sistema operatiu (CentOS 6 i les Guest Additions)
- Haurem de canviar els noms del host, configurar les xarxes i configura el SSH com ja havíem fet anteriorment
- Perquè funcioni, hem d'utilitzar l'usuari "root".
- Hem de tenir en compte que un cop estigui operatiu, si parem les màquines haurem d'encendre el servei (1 per un, tenint en compte les prioritats, he fet la prova i demora uns 20 min aprox.)

3.3.1. Connexions entre el clúster: Xarxa intern a

Node1-ambari

- **Modifiquem el nom del host**
  - [root@node1 Escriptori]# hostname node1-ambari
  - [root@node1-ambari Escriptori]# nano /etc/sysconfig/network

- **Configurant l'arxiu /etc/hosts**
  - [root@node1 Escriptori]# nano /etc/hosts
    - 192.168.30.201 node1-ambari
    - 192.168.30.202 node2-ambari
    - 192.168.30.203 node3-ambari
  - [root@node1 Escriptori]# ip a

Node2-ambari

- **Modifiquem el nom del host**
- [root@node1 Escriptori]# hostname node1-ambari
- [root@node2-ambari Escriptori]# nano /etc/sysconfig/network

- **Configurant l'arxiu /etc/hosts**
- [root@node2 Escriptori]# nano /etc/hosts
    - 192.168.30.201 node1-ambari
    - 192.168.30.202 node2-ambari
    - 192.168.30.203 node3-ambari
- [root@node2 Escriptori]# ip a

Node3-ambari

- **Modifiquem el nom del host**
- [root@node1 Escriptori]# hostname node1-ambari
- [root@node3-ambari Escriptori]# nano /etc/sysconfig/network

- **Configurant l'arxiu /etc/hosts**
- [root@node3 Escriptori]# nano /etc/hosts
    - 192.168.30.201 node1-ambari
    - 192.168.30.202 node2-ambari
    - 192.168.30.203 node3-ambari
- [root@node3 Escriptori]# ip a

- Fem ping entre les màquines per confirmar que es poden veure entre elles.

3.3.2. SSH :Claus públiques entre els nodes

- No mostrarem les captures, ja que es el mateix que abans. Així i tot mostrarem les comandes.

Node1-ambari

- [root@node1-ambari .ssh]# ssh-keygen
- [root@node1-ambari .ssh]# cp id_rsa.pub authorized_keys
- [root@node1-ambari .ssh]# cat authorized_keys
- [root@node1-ambari .ssh]# scp authorized_keys node2-ambari:/root/.ssh
-

Node2-ambari

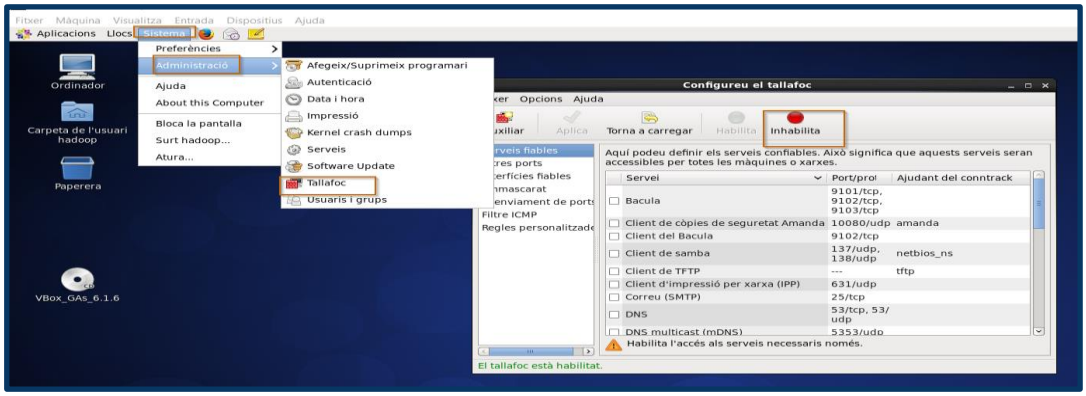
- [root@node1-ambari Escriptori]$ ssh node2-ambari
- [root@node1-ambari Escriptori]$ cd .ssh
- [root@node2-ambari .ssh]# ssh-keygen
- [root@node2-ambari .ssh]# cat id_rsa.pub >> authorized_keys
- [root@node2-ambari .ssh]# scp authorized_keys node3-ambari:/root/.ssh
-

Node3-ambari

- [root@node1-ambari Escriptori]$ ssh node3-ambari
- [root@node3-ambari Escriptori]$ cd .ssh
- [root@node3-ambari .ssh]# ssh-keygen
- [root@node3-ambari .ssh]# cat id_rsa.pub >> authorized_keys
- [root@node3-ambari .ssh]# scp authorized_keys node2-ambari:/root/.ssh
- [root@node3-ambari .ssh]# scp authorized_keys node1-ambari:/root/.ssh

3.3.3. Instal.lació i configuració del servidor Ambari

- Abans de continuar ens hem d'assegurar que el firewall estigui desactivat (o crear regles en les iptables de tots els ports que ens interessi que estiguin oberts, ja que per defecte només està obert el port 22 que hem utilitzat anteriorment amb el SSH).

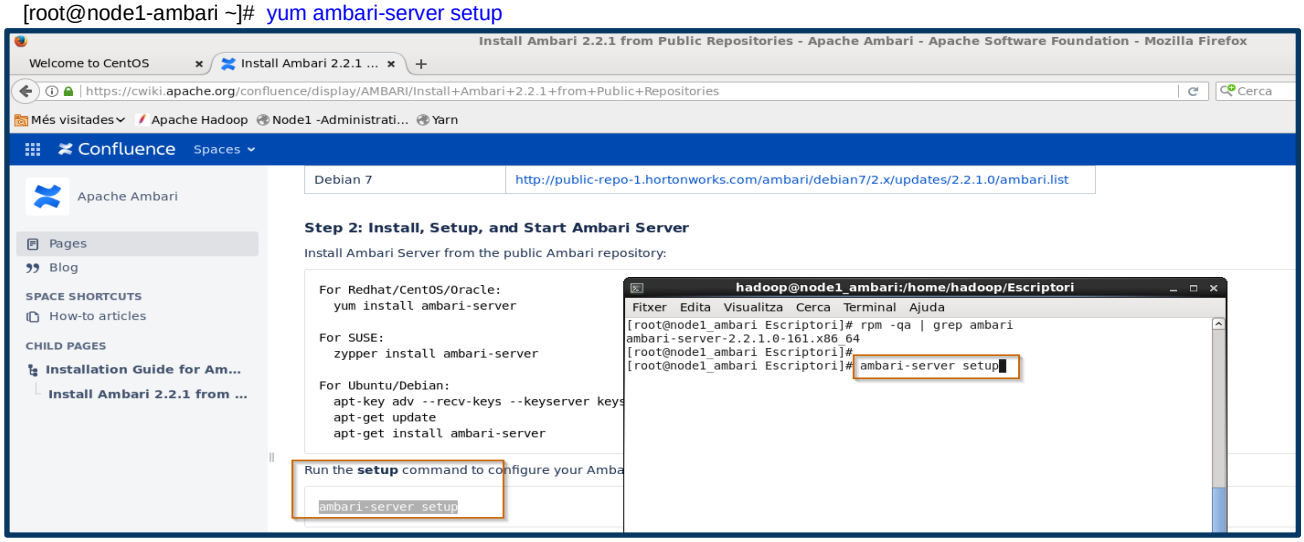


- Com que el node1-ambari farà de master instal·larem Ambari 2.2.1 des de repositoris públics seguint les pautes de la url oficial :
  - https://cwiki.apache.org/confluence/display/AMBARI/Install+Ambari+2.2.1+from+Public+Repositories

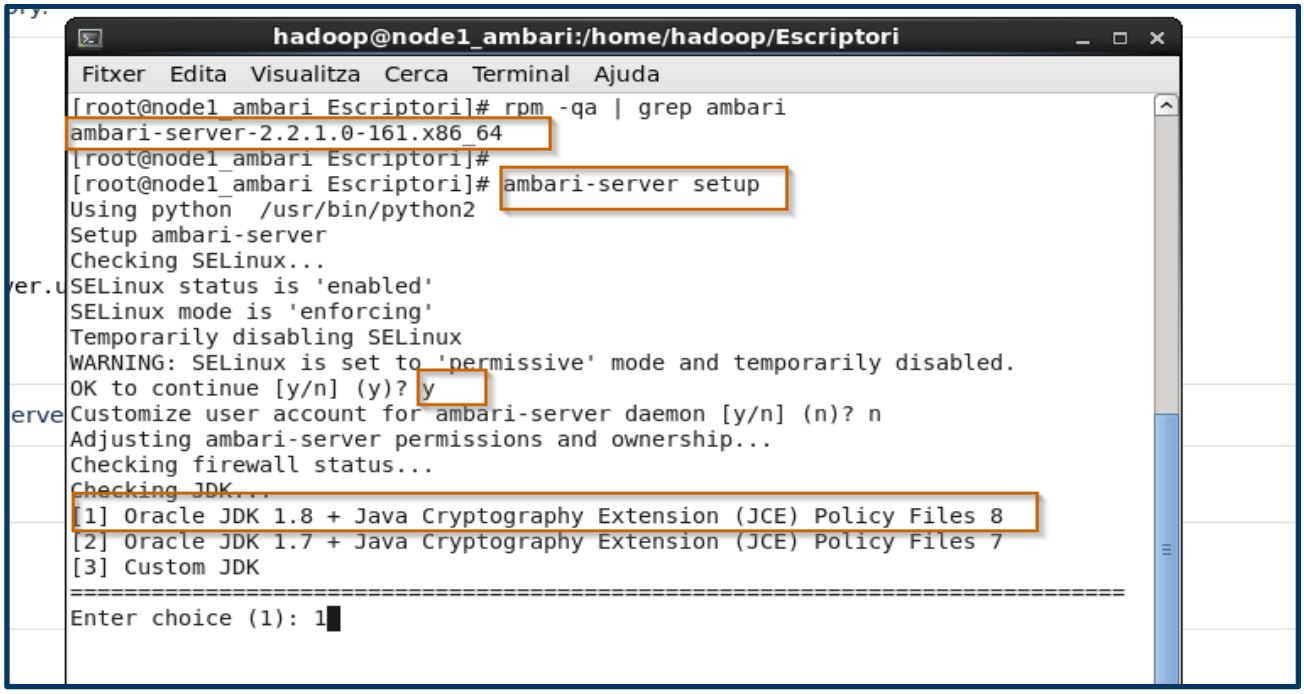
Node1-ambari

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

```
[root@node1-ambari ~]# cd /etc/yum.repos.d/
[root@node1-ambari ~]# wget http://public-repo-1.hortonworks.com/ambari/centos6/2.x/updates/2.2.1.0/ambari.repo
[root@node1-ambari ~]# yum install ambari-server
[root@node1-ambari ~]# yum ambari-server setup
```



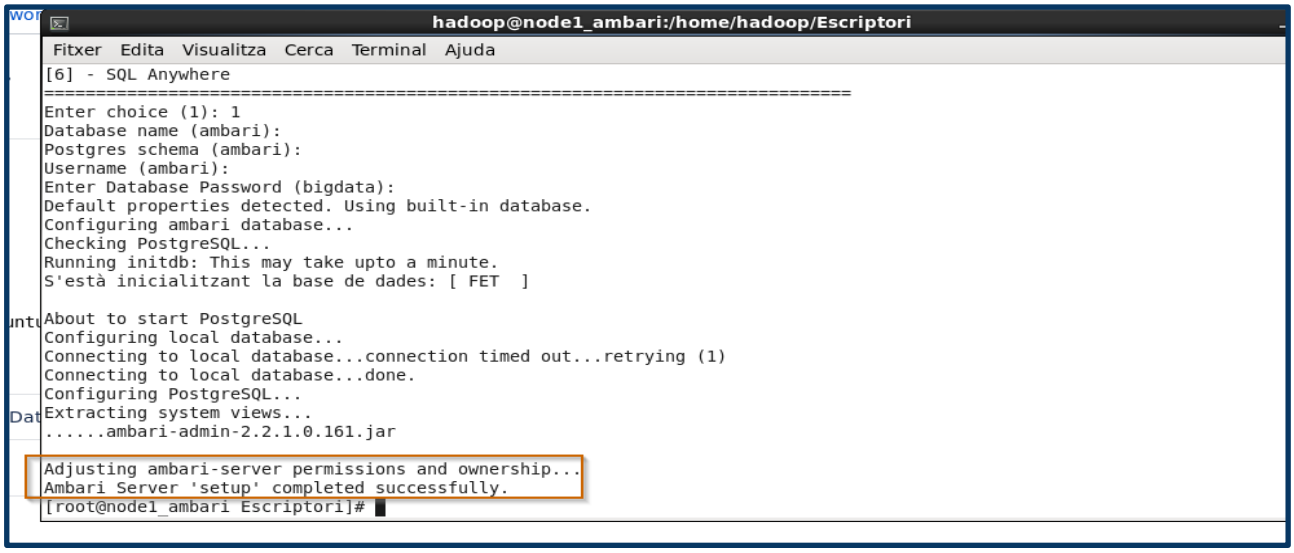
- ❑ Hem de desactivar SELinux[17]. Si no tenim JAVA i JCE ens ho instal.lara



- ❑ Si no tenim cap base de dades, una opció ràpida és la número 1 PostgreSQL ja està integrat

[17] És un mòdul de seguretat del kernel de Linux

Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

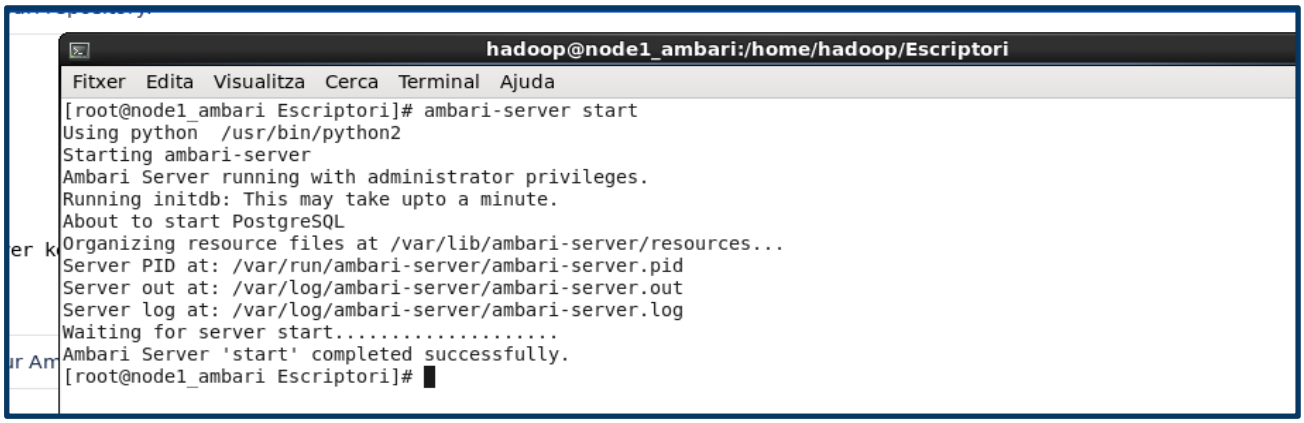


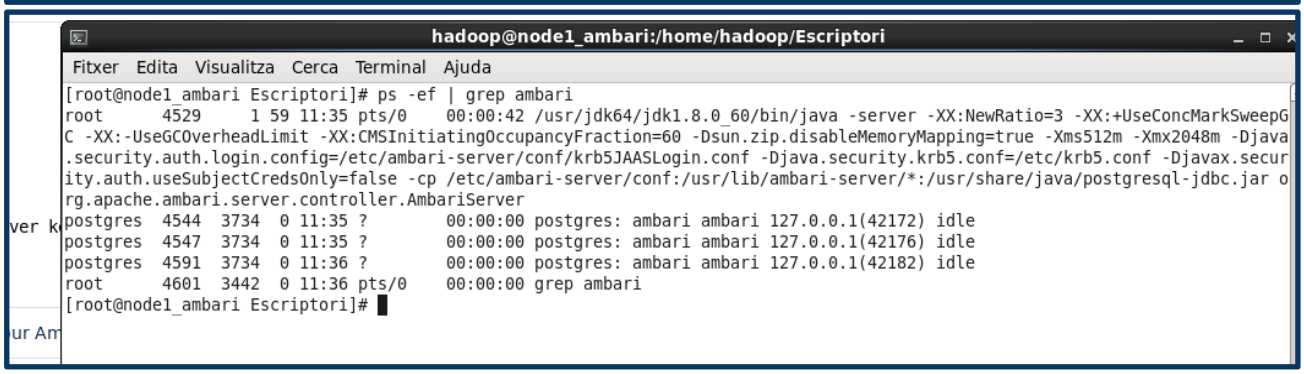
- Arrenquem el servei i revisem els processos

Node1-ambari

[root@node1-ambari ~]# yum ambari-server start
[root@node1-ambari ~]# ps -ef | grep ambari

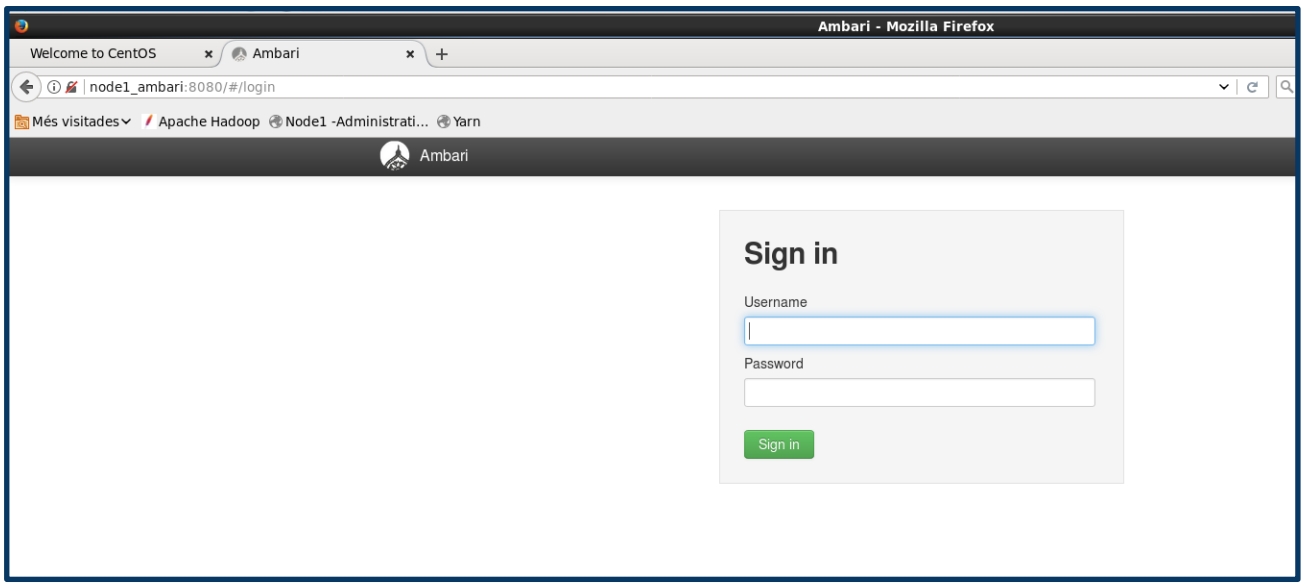
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



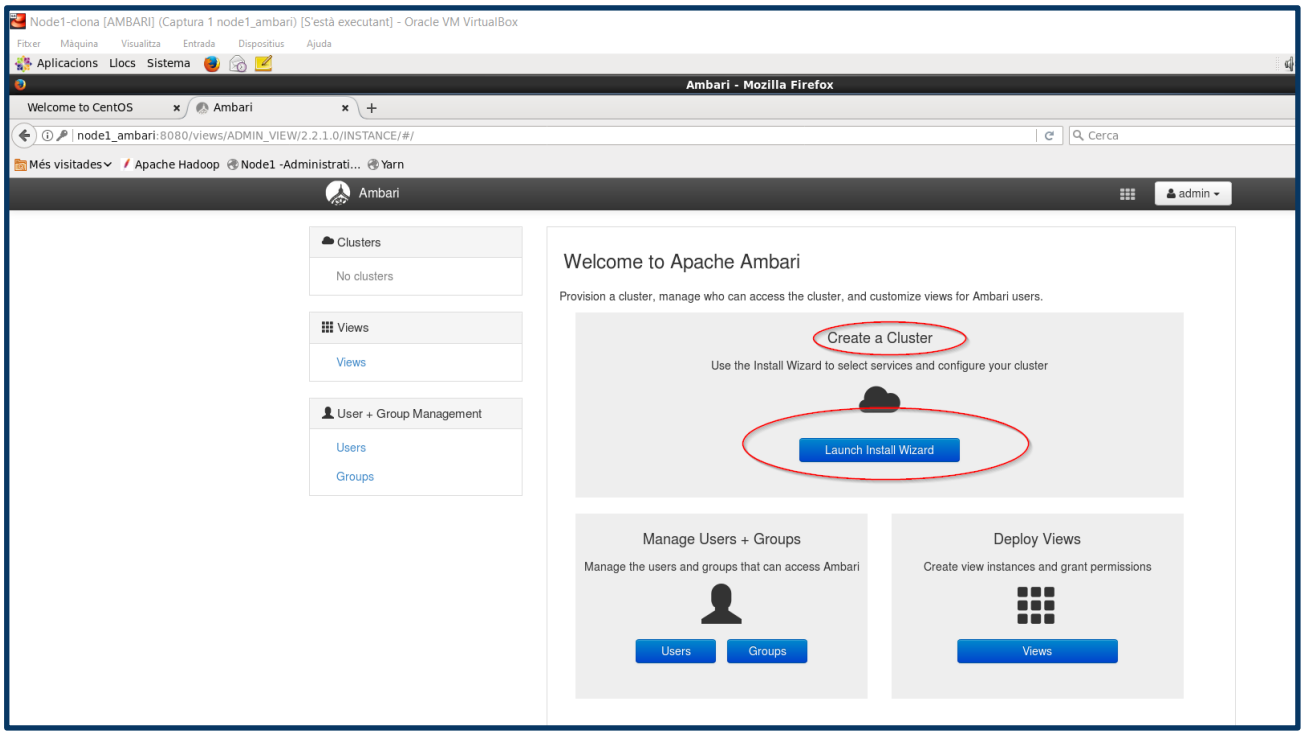


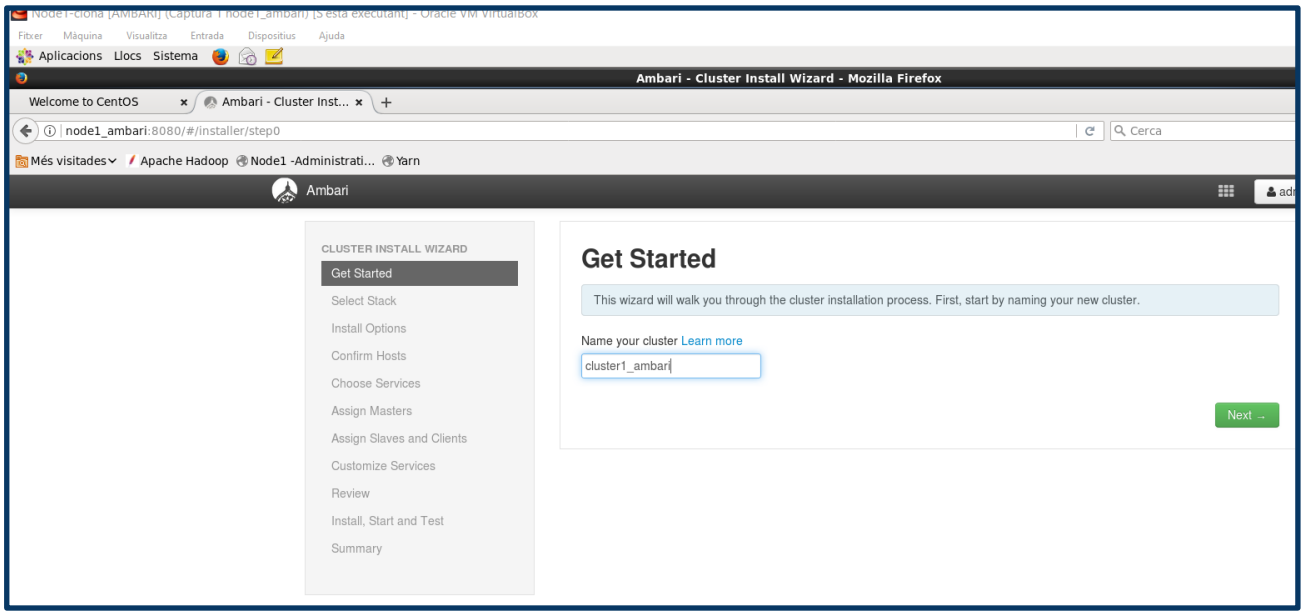
- ❑ Accedim a la interfície web del servidor Ambari (utilitza el port 8080) on haurem de fer la configuració del clúster.
  Nota: per defecte el username i password és admin

node1:8080



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





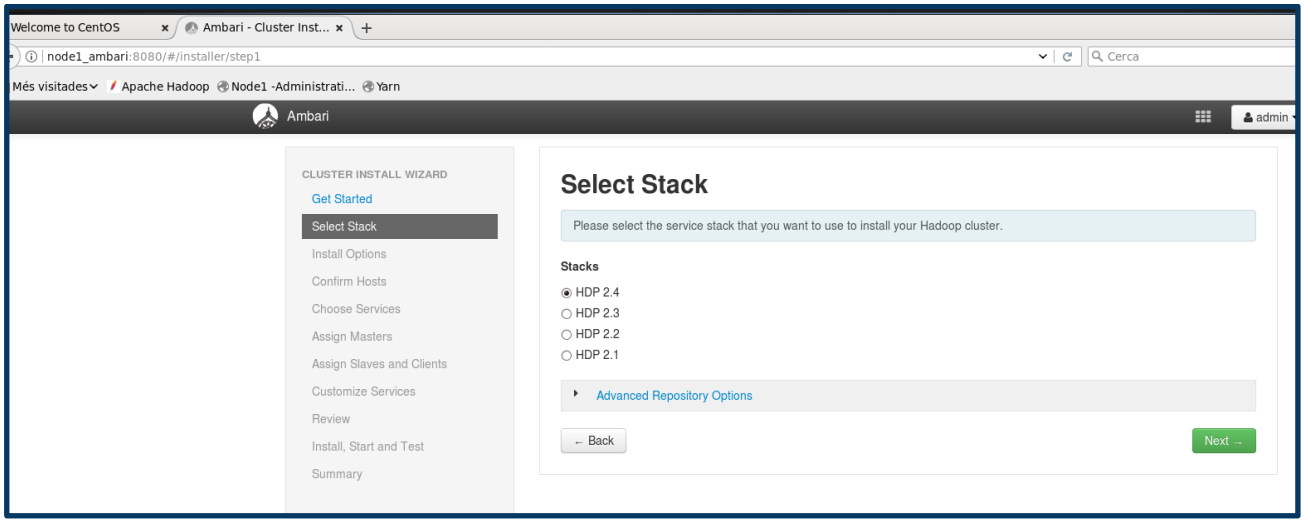
Nom i Cognoms

Arnau Subirós Puigarnau

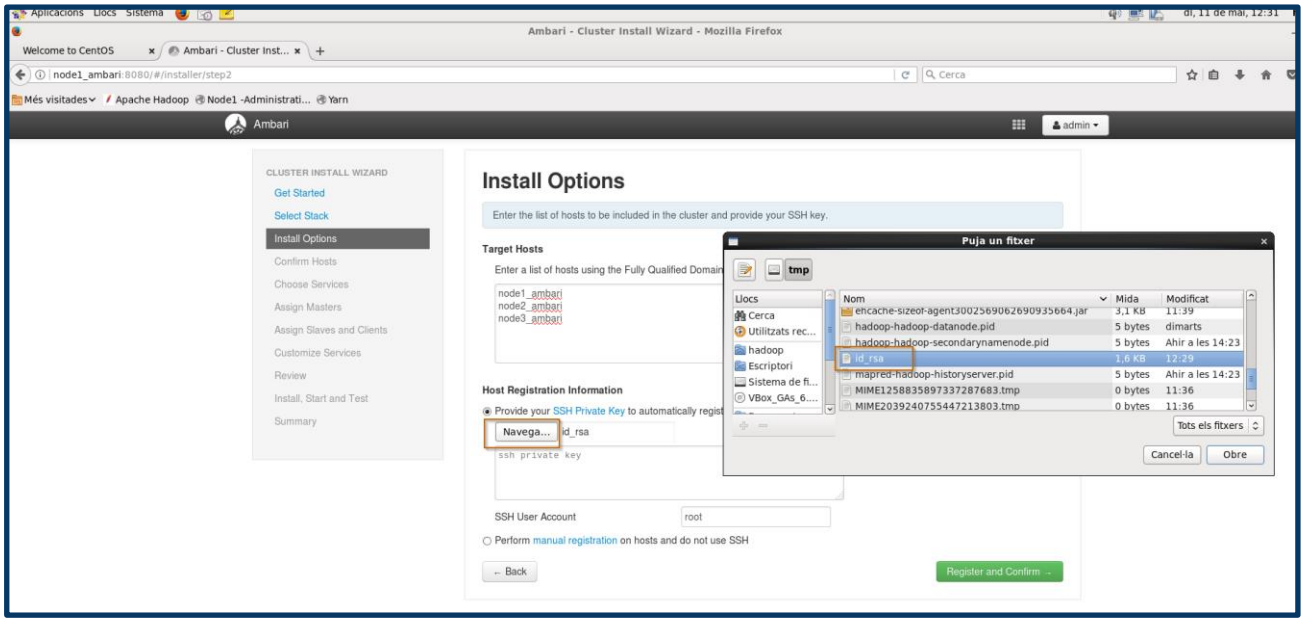
Data

02-06-2020

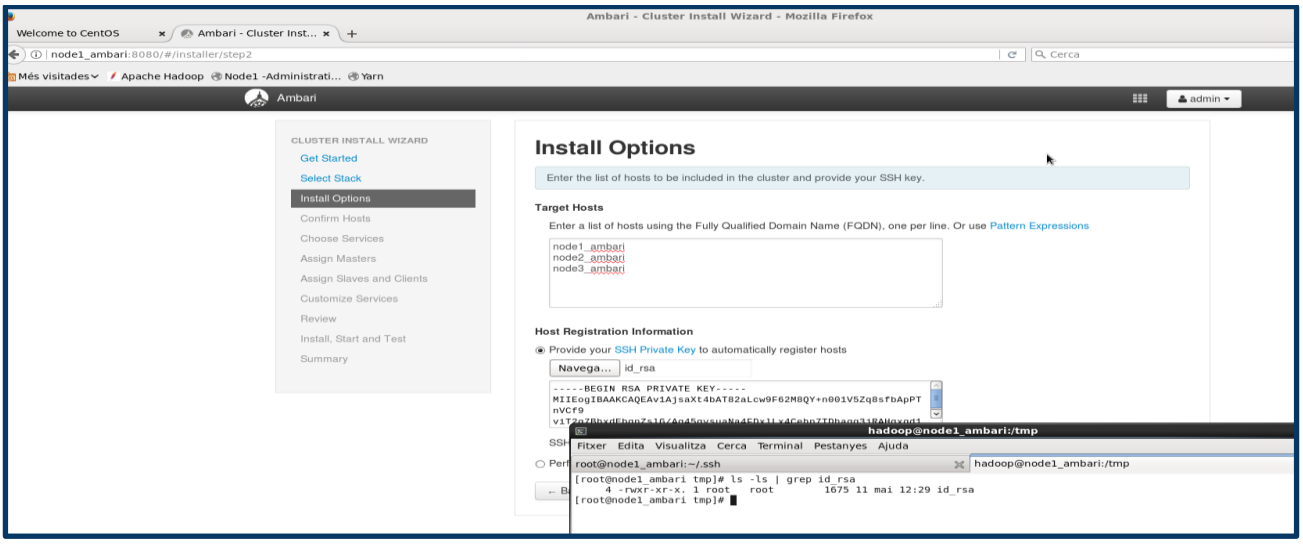
Seleccionem l'última versió dels repositoris de Hortonworks

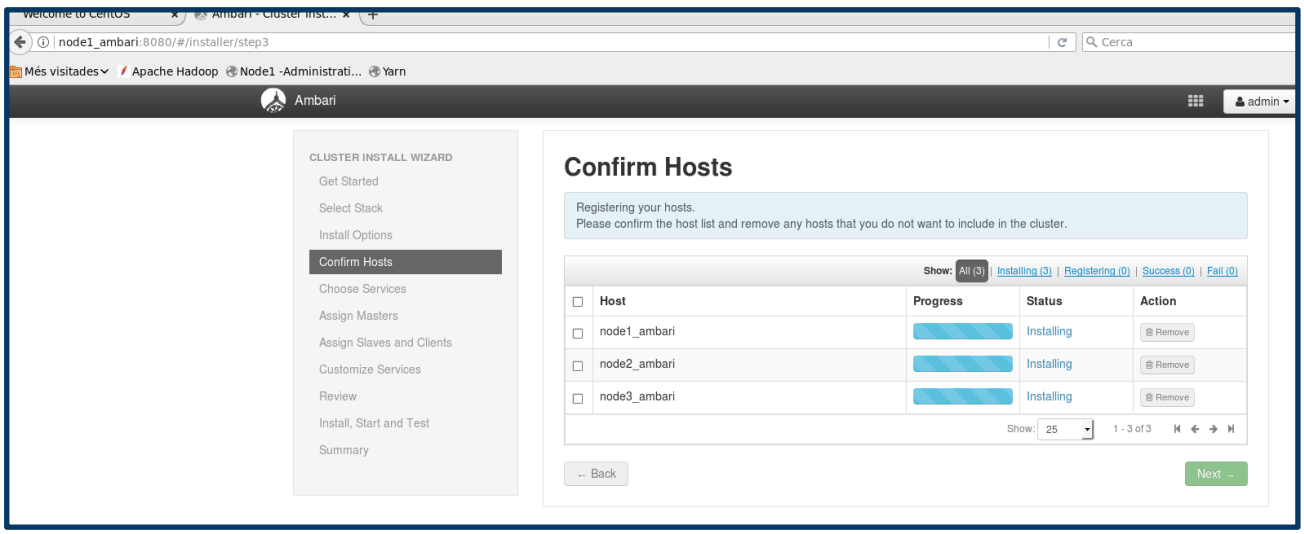


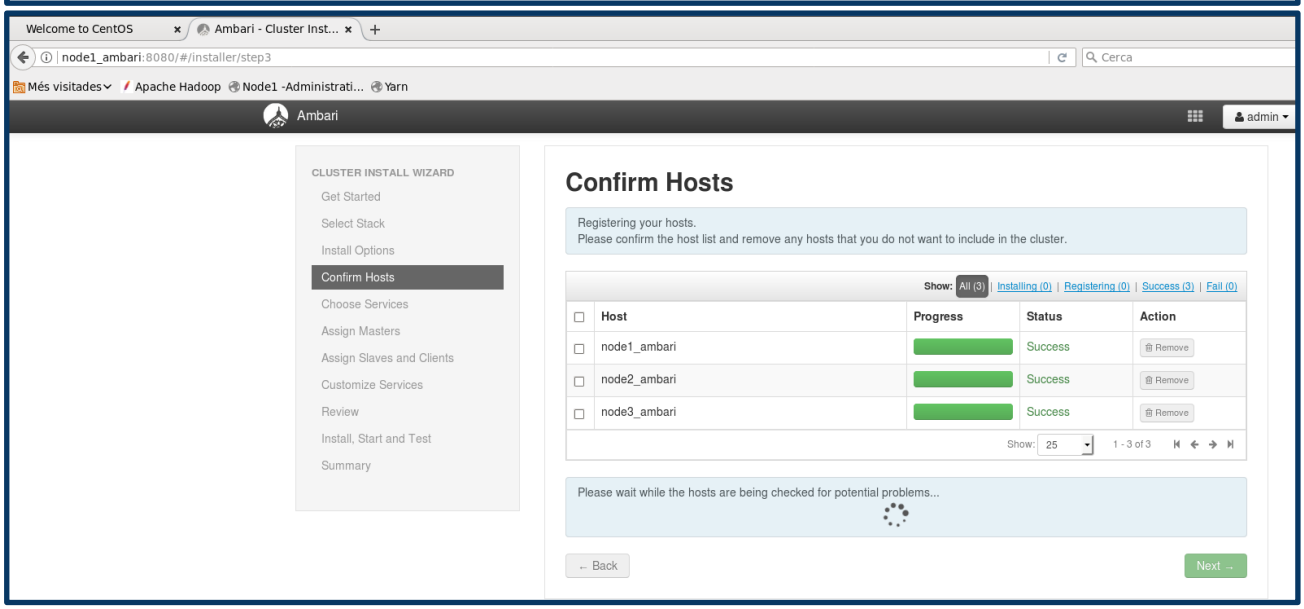
- La següent opció hem d'afegir els noms del host i la clau privada del host(id_rsa del node1-ambari). Com que la clau privada està en una carpeta oculta no la podrem insertar, per aixo l'haurem a una altre carpeta(/tmp) que tingui visibilitat i permisos.



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

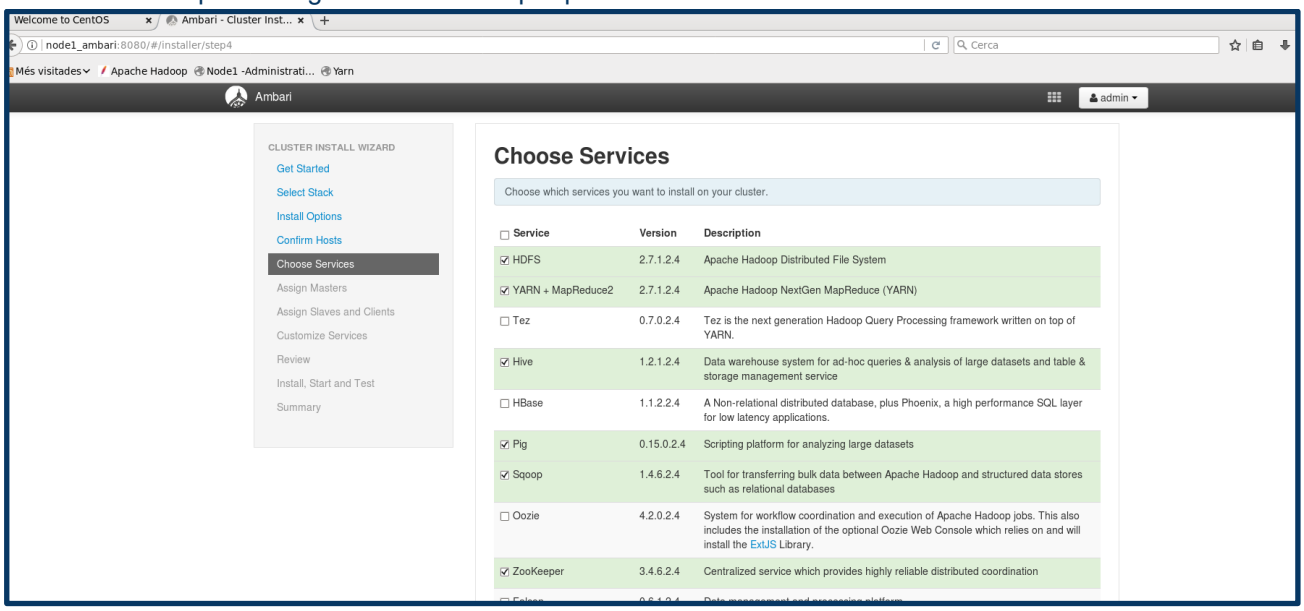


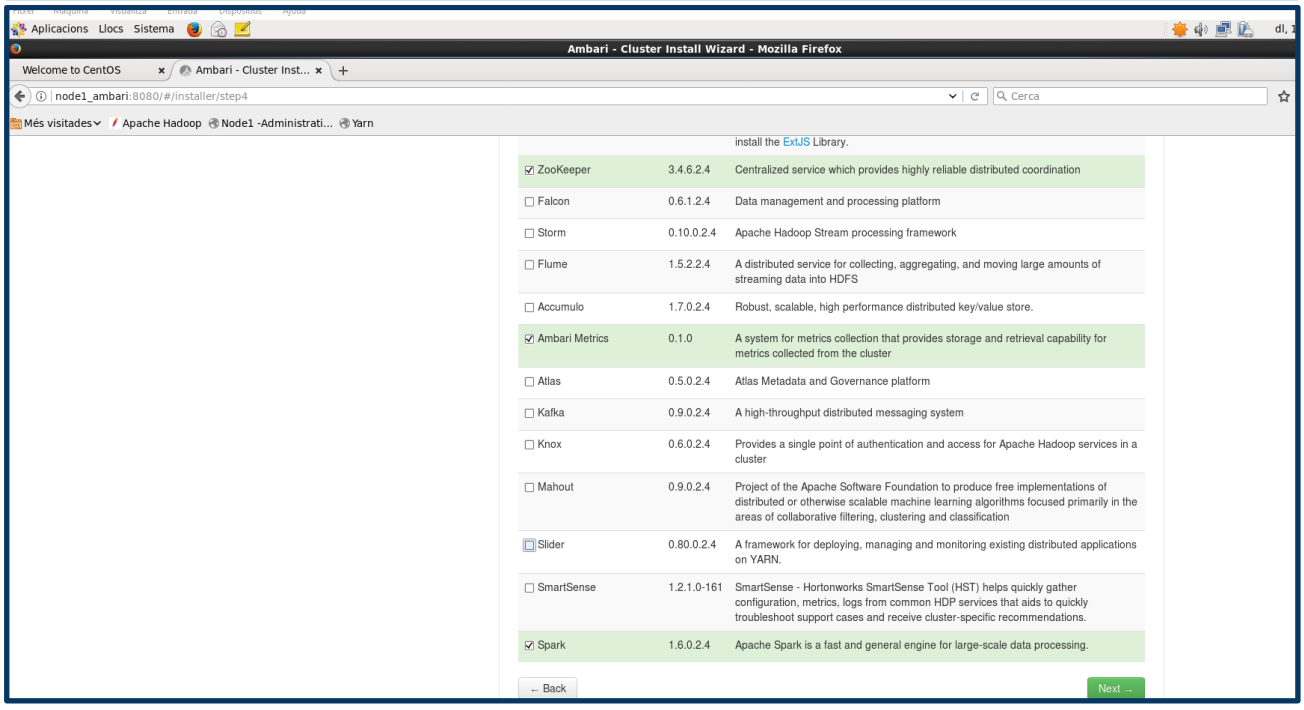




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

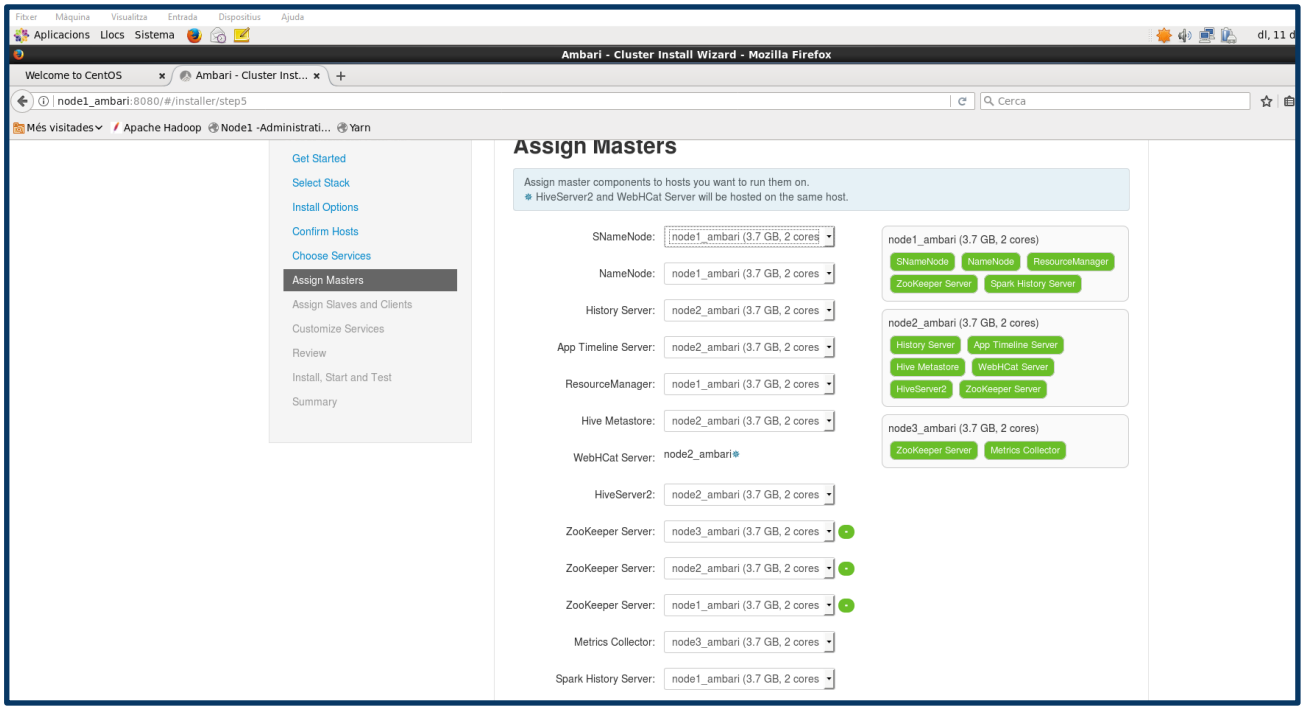
- Un cop instal·lat hem de seleccionar quins serveis volem afegir, tot i que n'hi ha alguns com "Zookeeper" t'obliga a instal·lar-los per poder continuar.

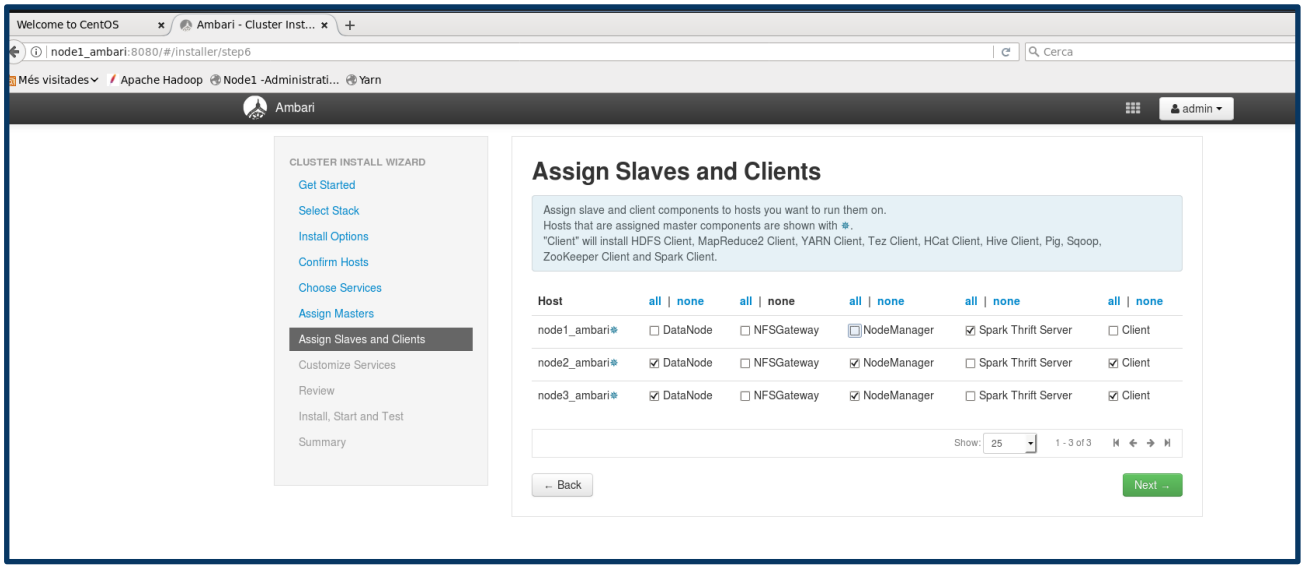




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

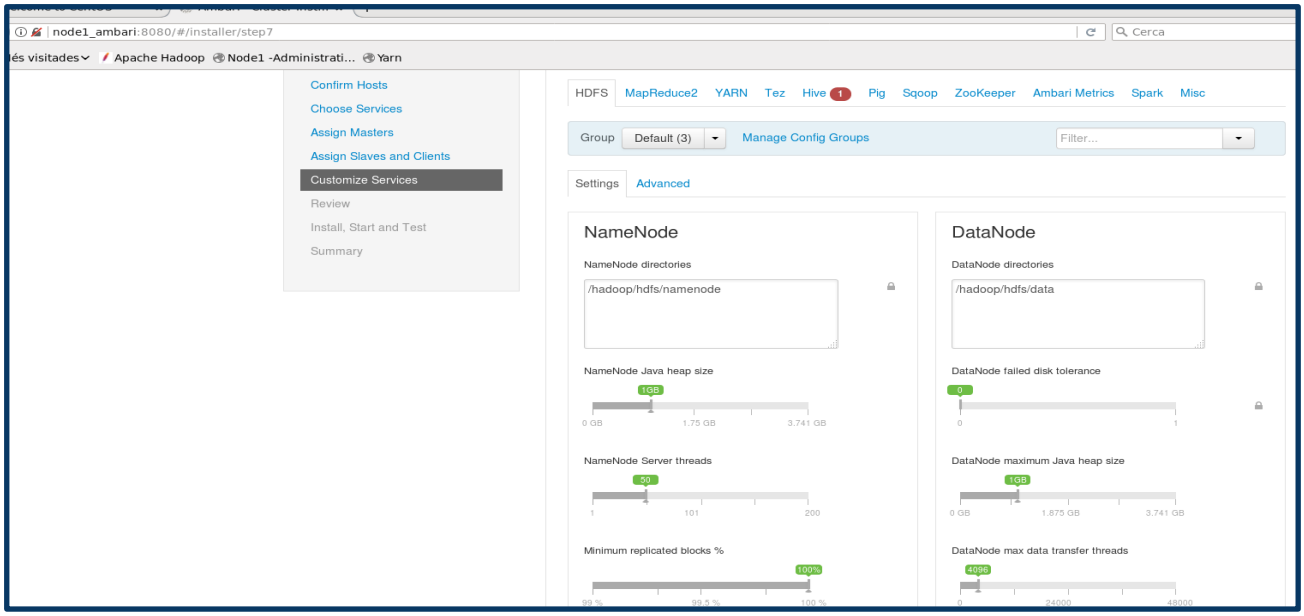
- Després hem d'especificar a quin node volem instal·lar els servidors

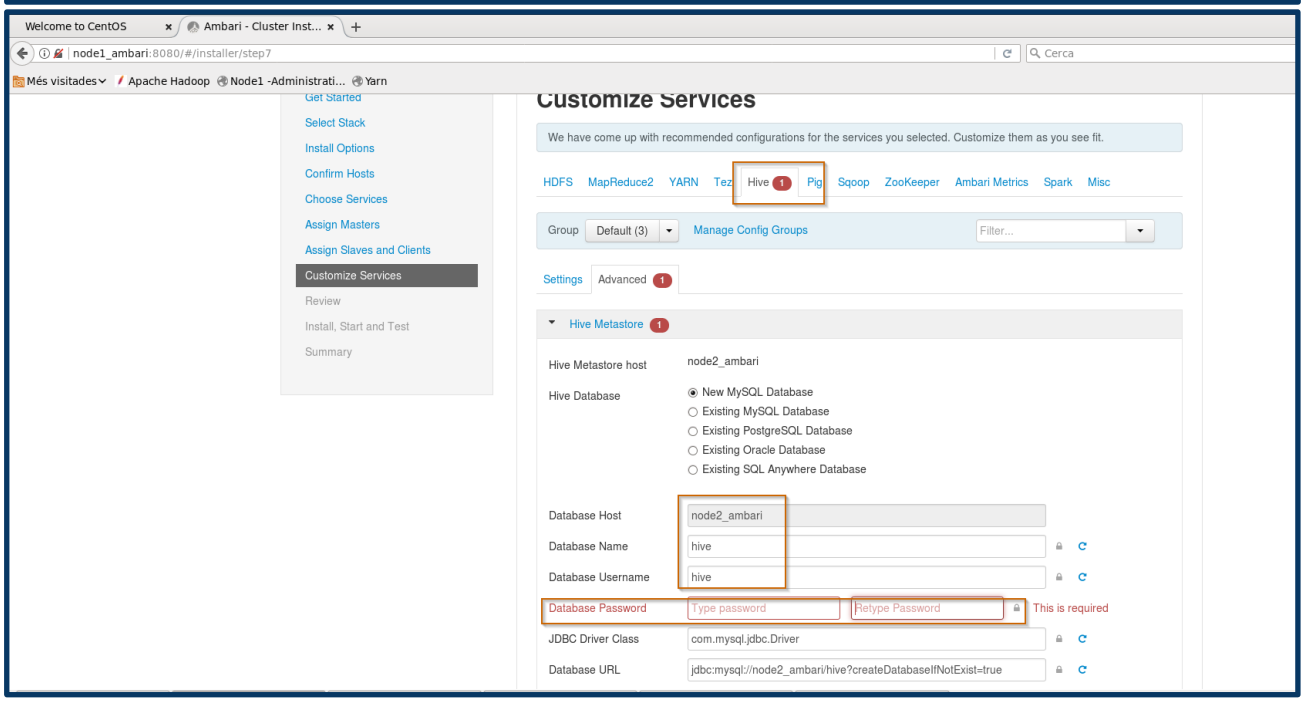




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Indica un error en el Hive, ja que la base de dades i usuari hive que s'ha generat hem de escriure el password



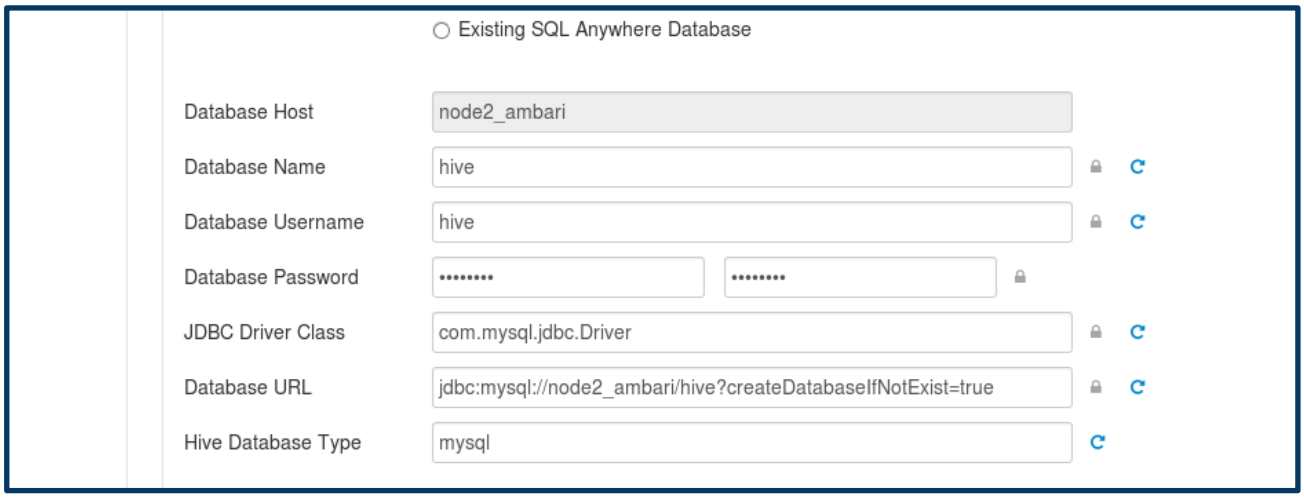


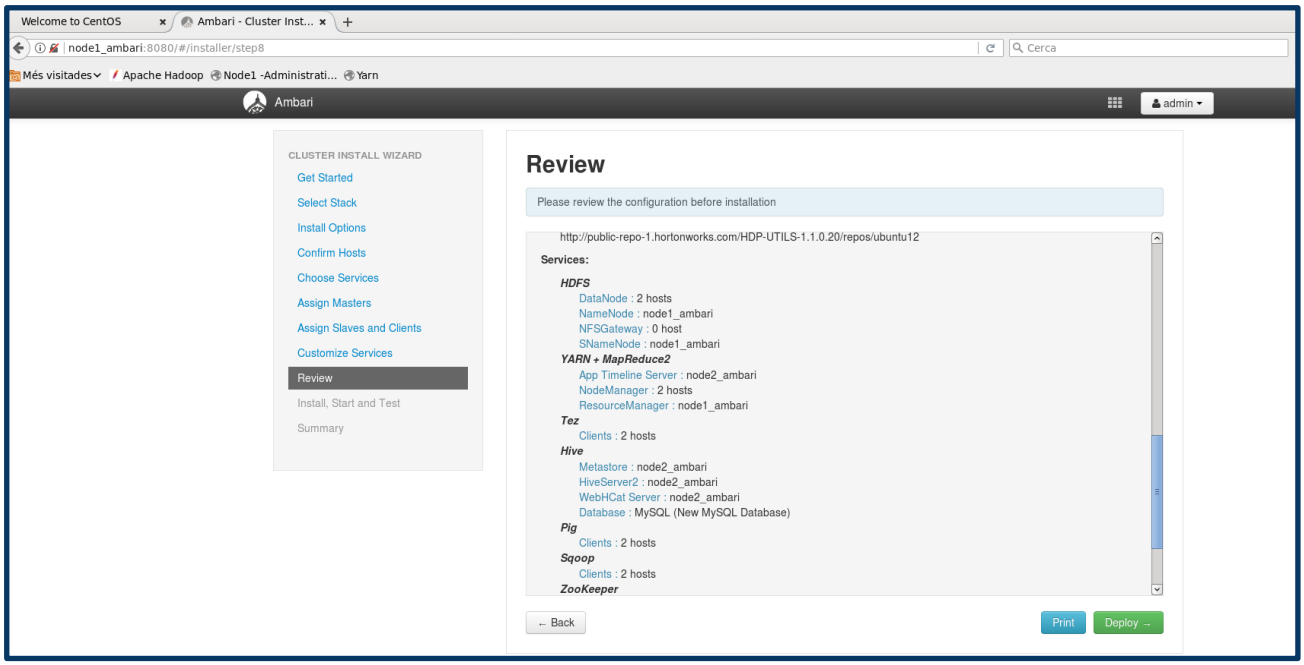
Nom i Cognoms

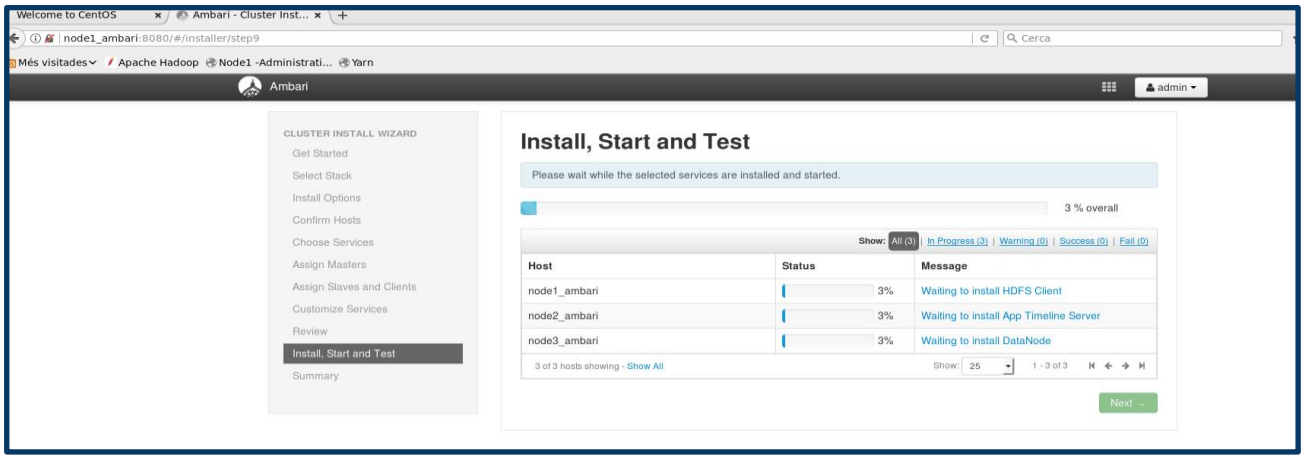
Arnau Subirós Puigarnau

Data

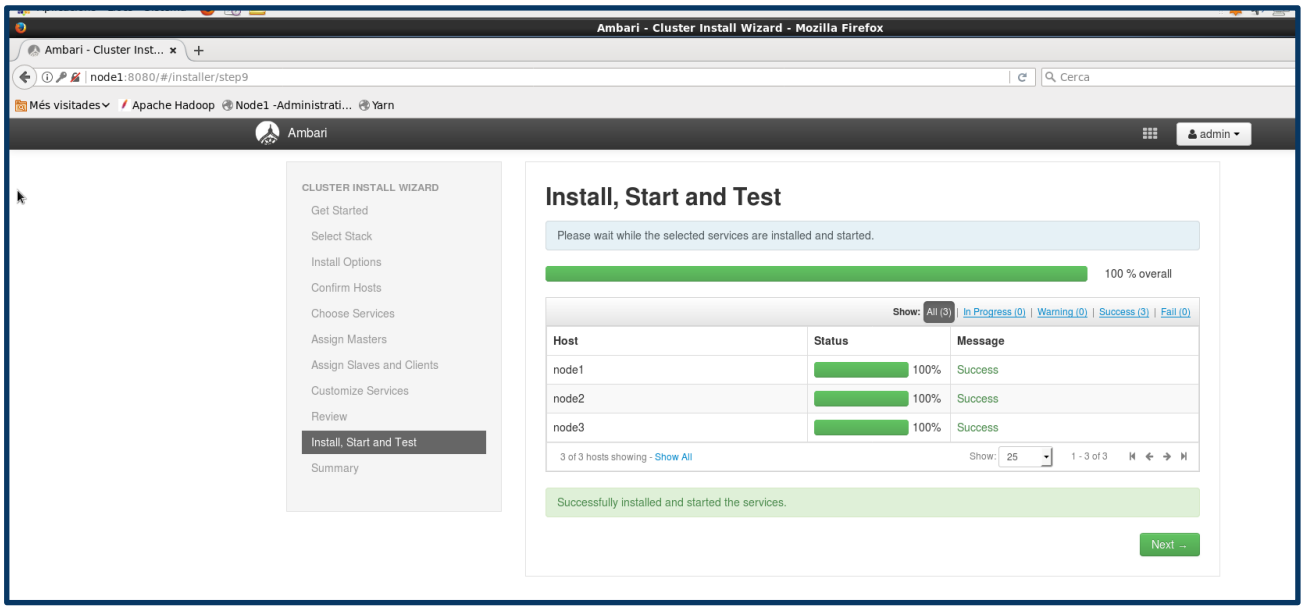
02-06-2020

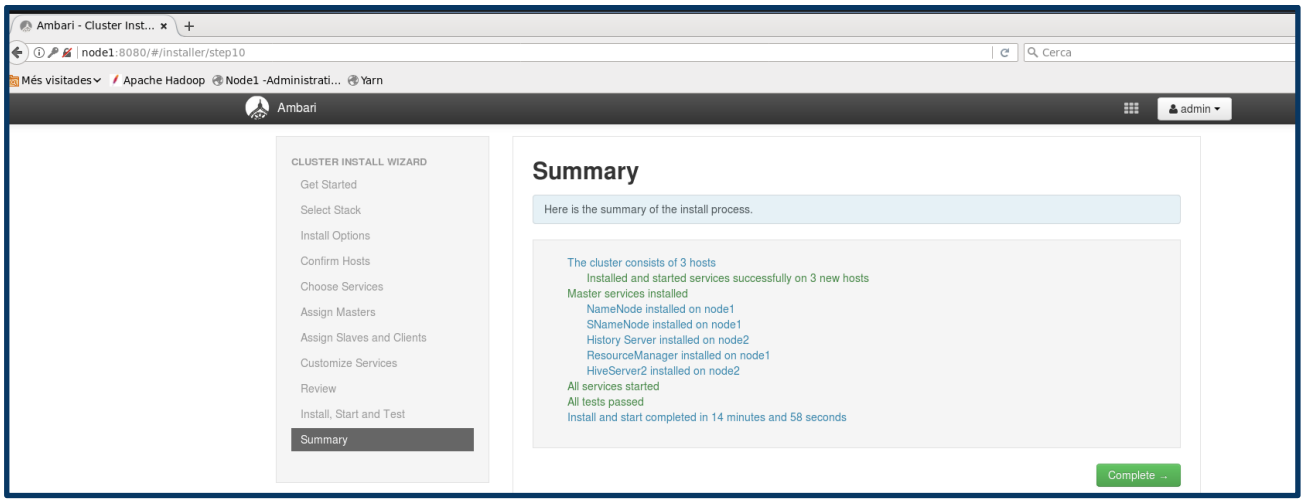




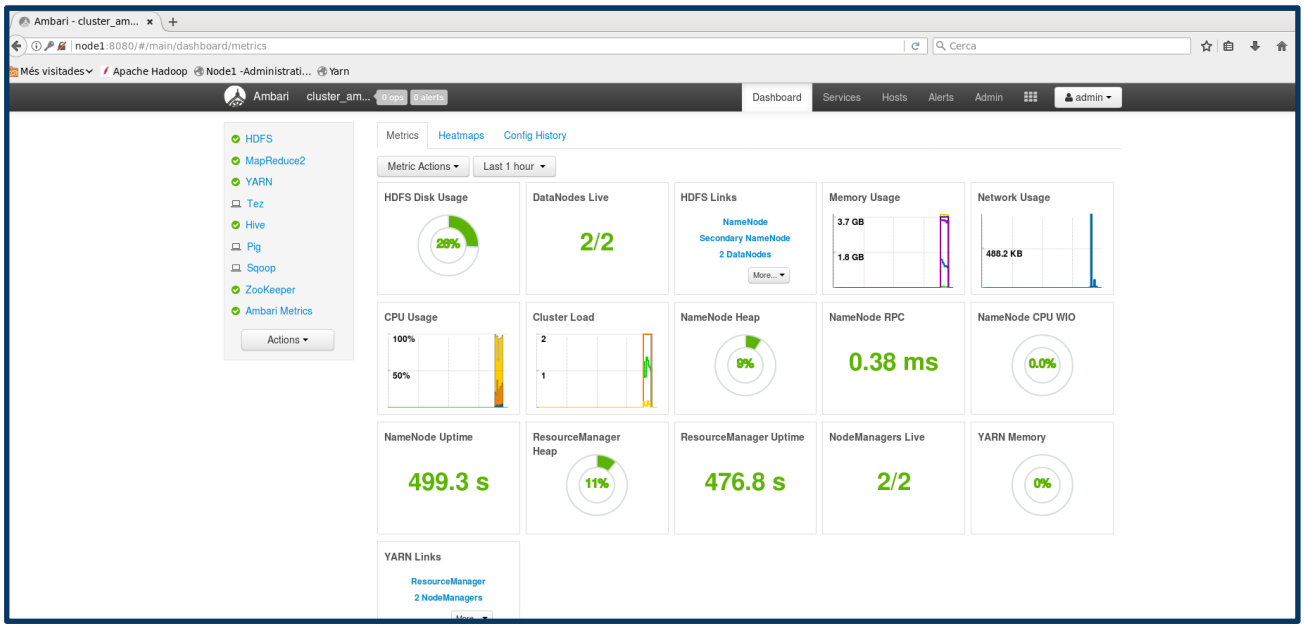


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020





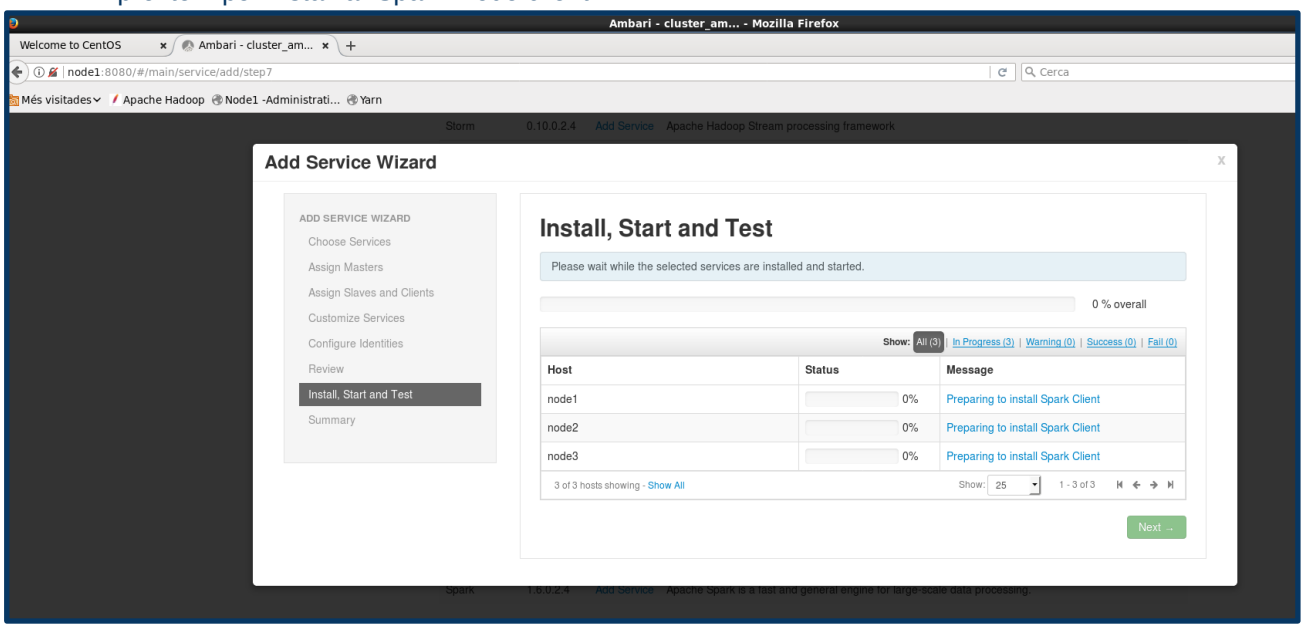
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

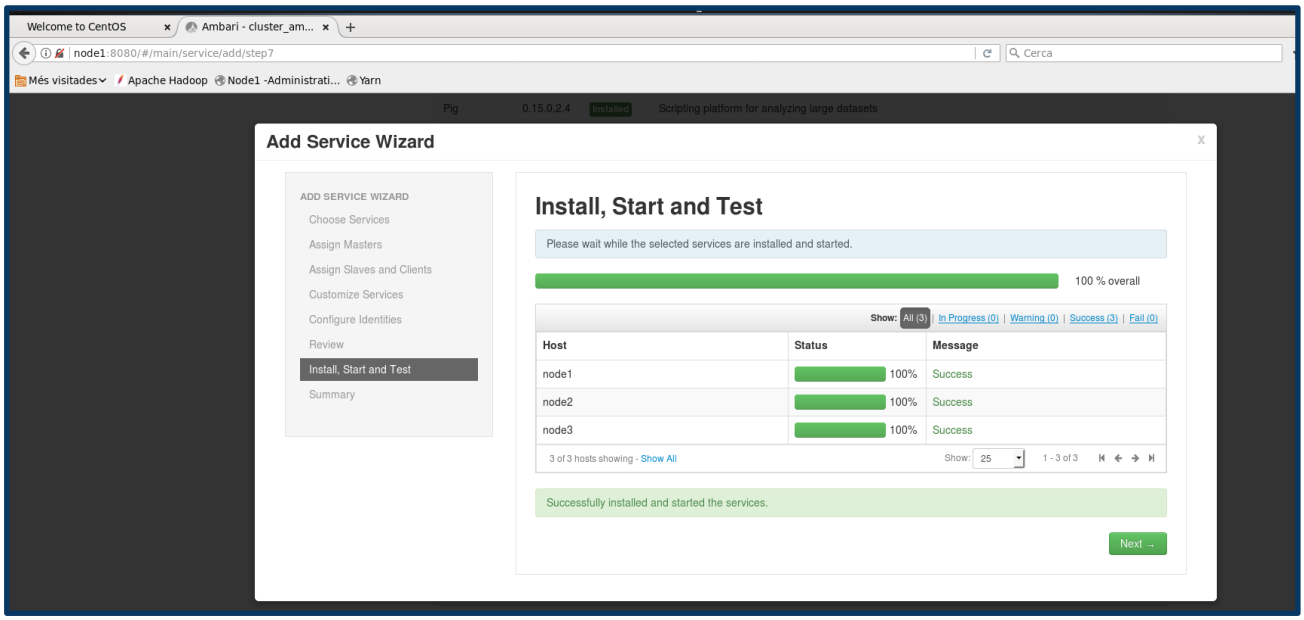


- ❏ Després de fer un backup de les màquines a un disc extern. En iniciar el servidor Ambari he tingut problemes, ja que no s'iniciaven els serveis. Per solucionar-ho desde el terminal com usuari hdfs hem tingut que desactivar el safe mode

[root@node1-ambari ~]#su - hdfs
[hdfs@node1-ambari ~]# *hdfs dfsadmin -safemode leave*

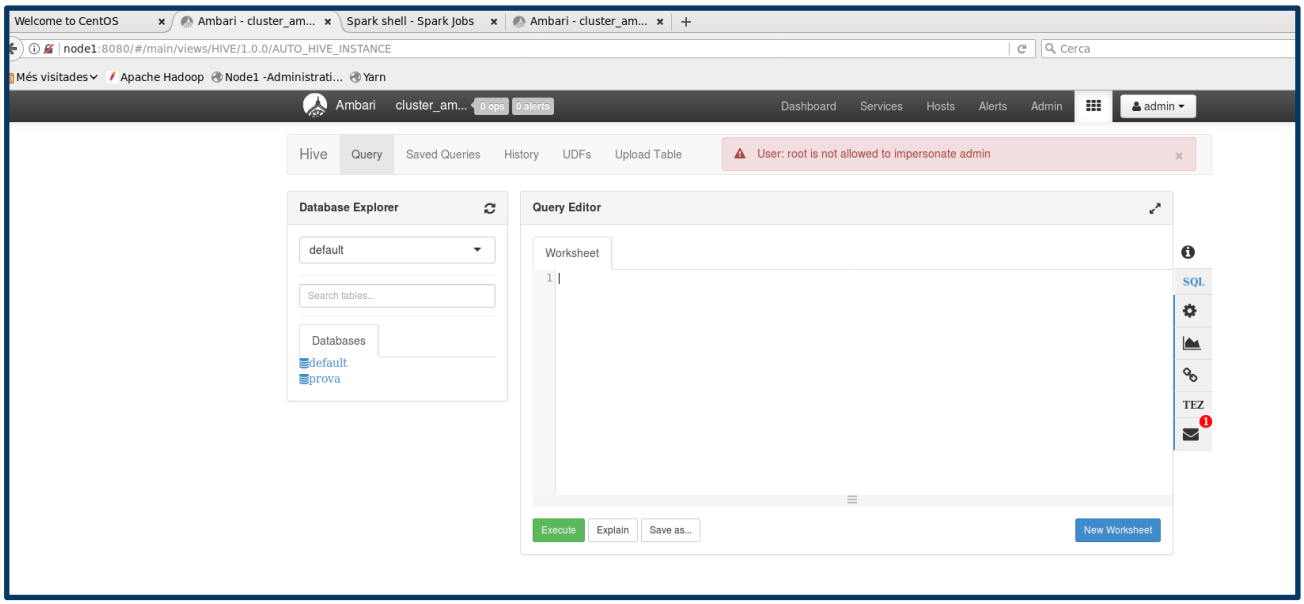
- ❏ Aprofitem per instal·lar Spark mode client





3.3.3.1. Configuració de Hive

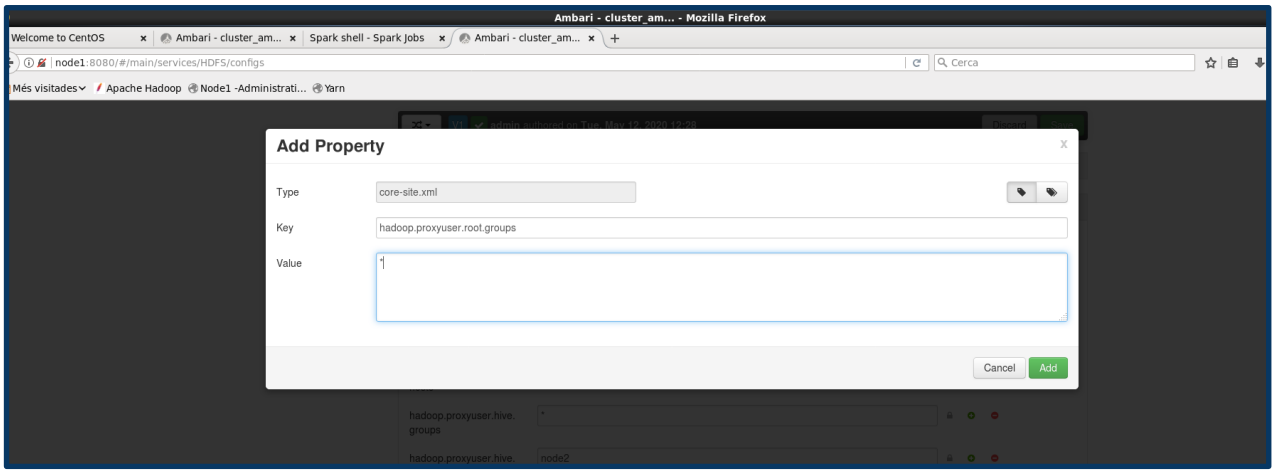
- ❏ En accedir a l'editor de Hive, tinc problemes amb els permisos, ja que per defecte l'usuari és hive i no admin. Des del terminal fem la prova de crear una base de dades anomenada prova i l'intentem obrir sense èxit

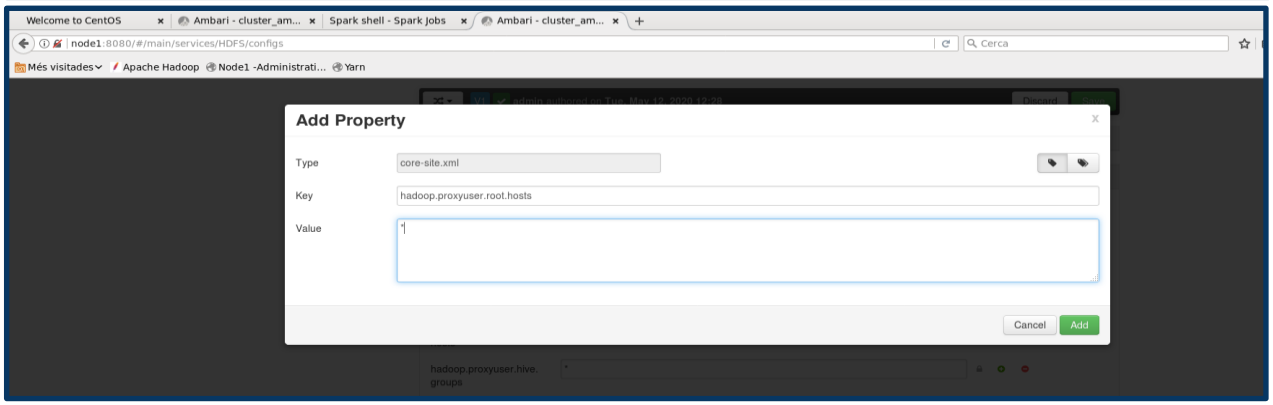


Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

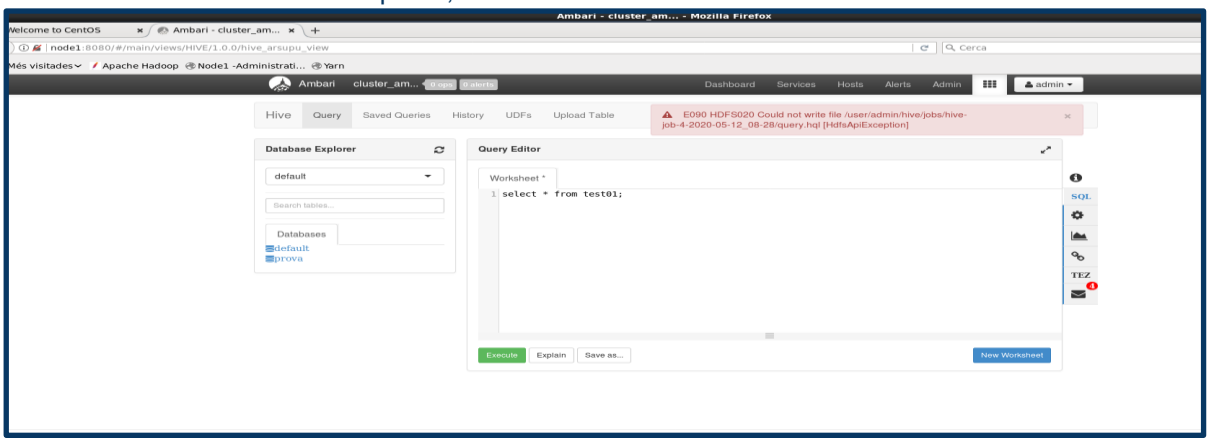
- ❑ Hem d'anar a les opcions avançades de HDFS i afegir unes propietats a custom core-site.
- ❑ Totes les modificacions s'han de fer des de la interfície web, ja que de l'altra manera podria haver-hi errors

  - hadoop.proxyuser.root.groups=*
  - hadoop.proxyuser.root.hosts=*



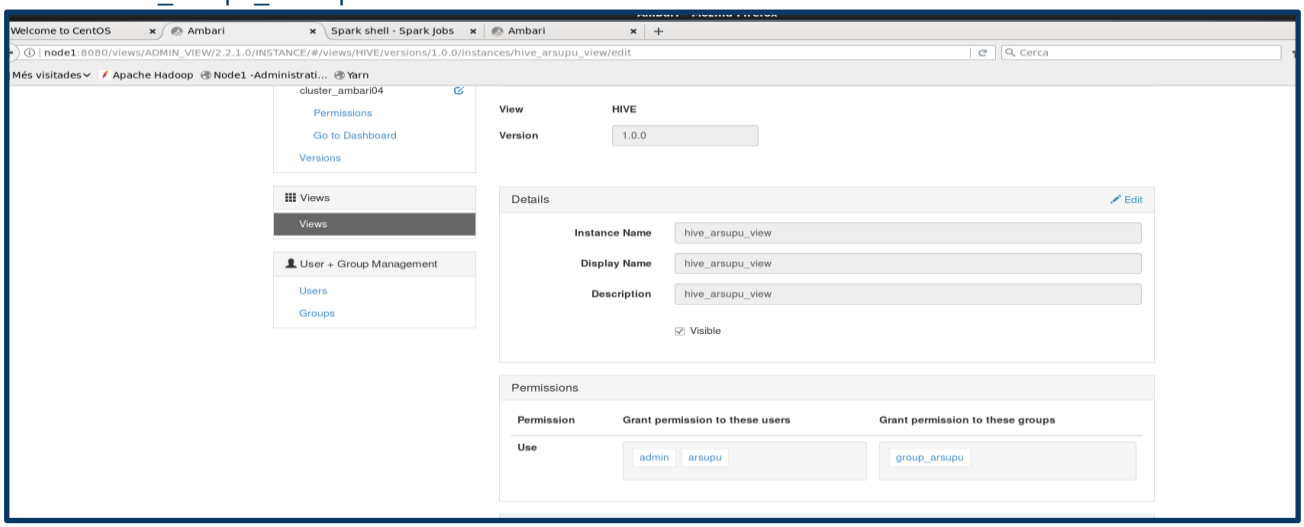


- ❑ Després de fer les modificacions, podré accedir però no podré crear res. Al intentar fer un select de la taula01 de la bse de dades prova, no troba l'usuari admin



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

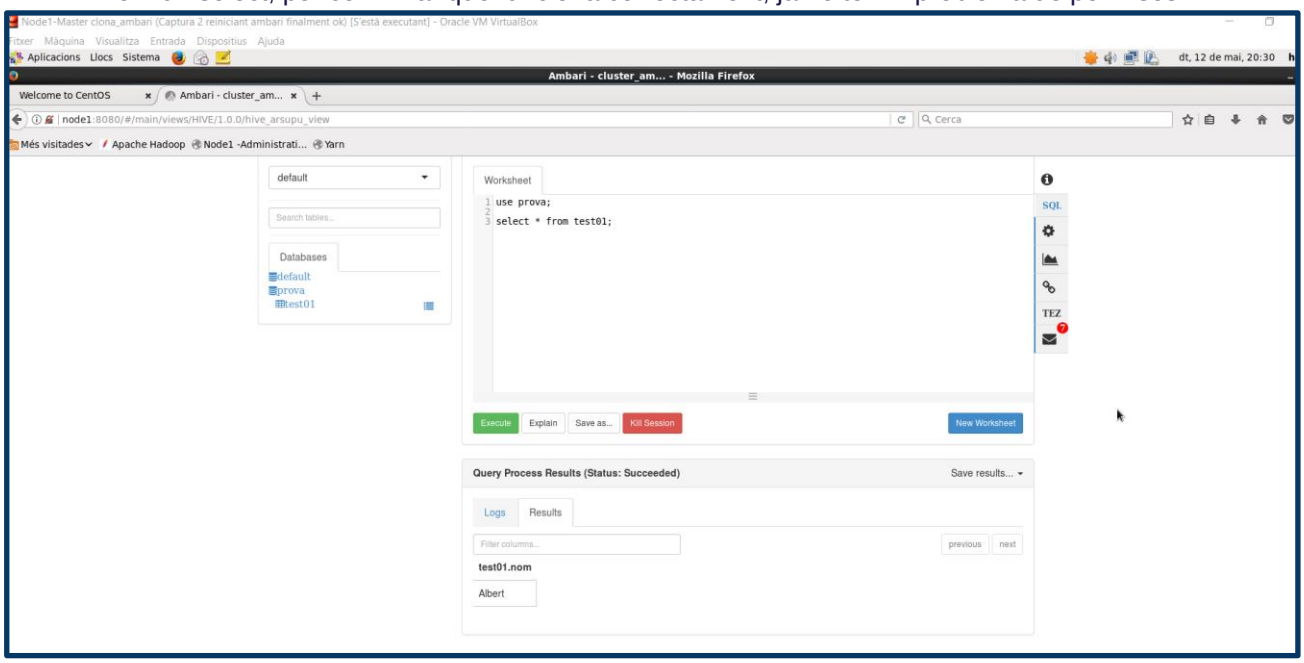
- Creem l'usuari "arsupu i el grup :" group_arsupu" i una instancia de HIVE anomenada "hive_arsupu_view" per no només tenir l'usuari admin.



- IMPORTANT: Creem u directori HDFS pels 2 usuaris i li donem permisos.

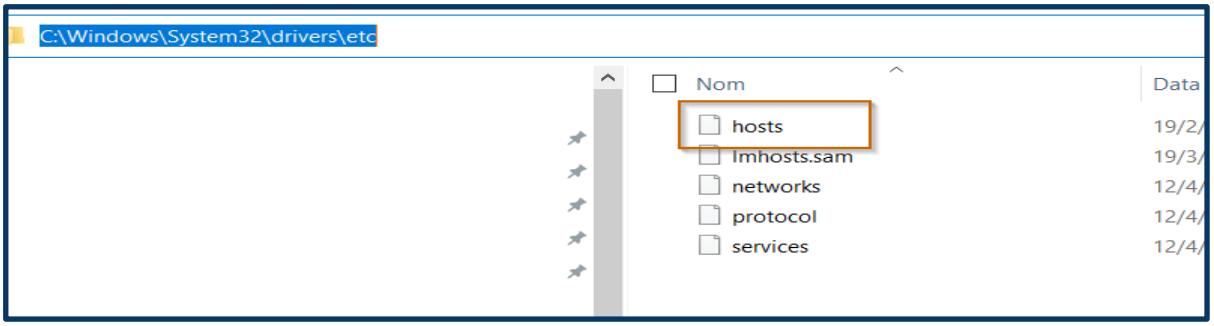
```
[root@node1-ambari ~]# hdfs dfs -ls /user/hive
[root@node1-ambari ~]# hadoop fs -mkdir /user/admin
[root@node1-ambari ~]# hadoop fs -mkdir /user/arsupu
[root@node1-ambari ~]# hadoop fs  -chown admin:hadoop  /user/admin
[root@node1-ambari ~]# hadoop fs  -chown arsupu:hadoop  /user/arsupu
```

- ❏ Fem un select, per confirmar que funciona correctament, ja no tenim problema de permisos

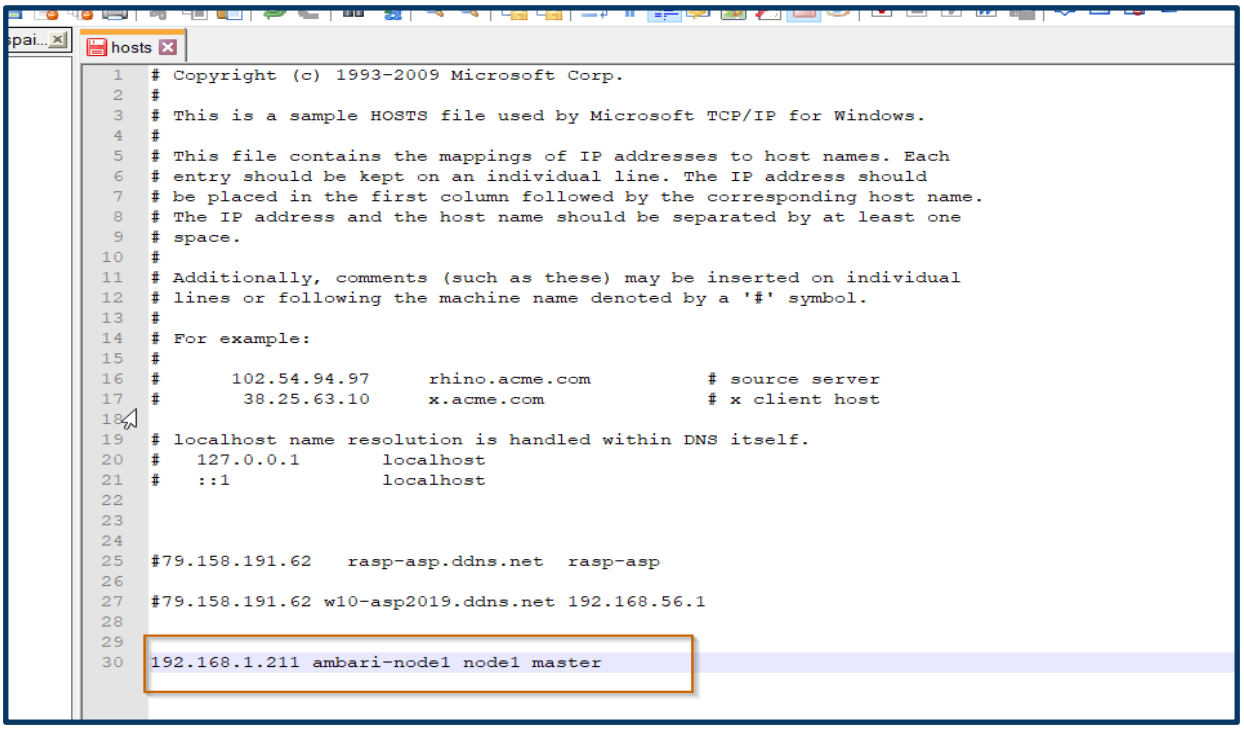


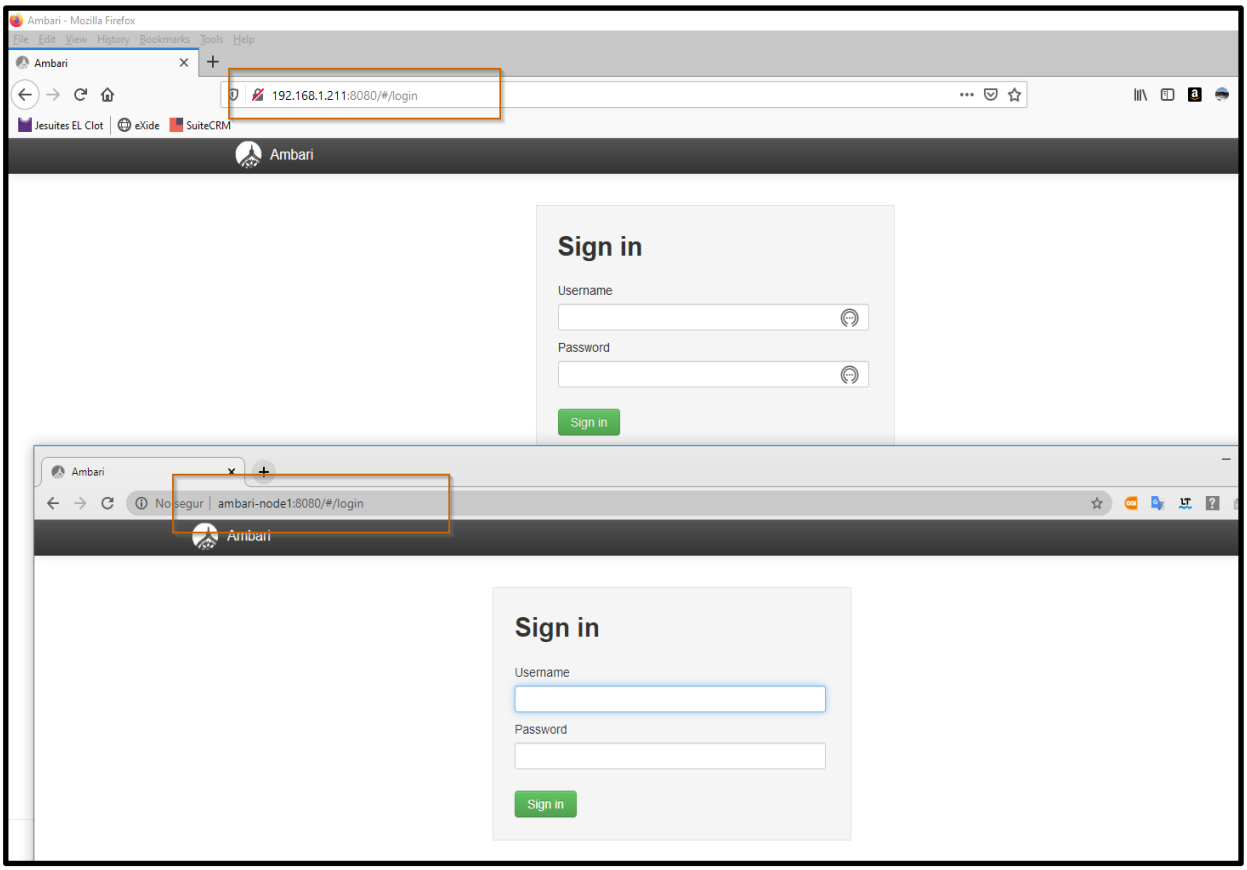
3.3.3.2. Visualització des de Windows 10

- ❏ Hem canviat el tipus d'adaptador pont de les màquines virtuals , no farem servir xarxa interna, sinó adaptador pont (Ethernet) ja que ens interessa fer servir la mateixa xarxa que utilitza el meu host amfitrió (192.168.1.0).
- ❏ Hem configurat les IP estàtiques en un rang que no doni problemes
- ❏ S'ha afegit les IP i el nom de les màquines en l'arxiu host (ja que no tenim servidor DNS)



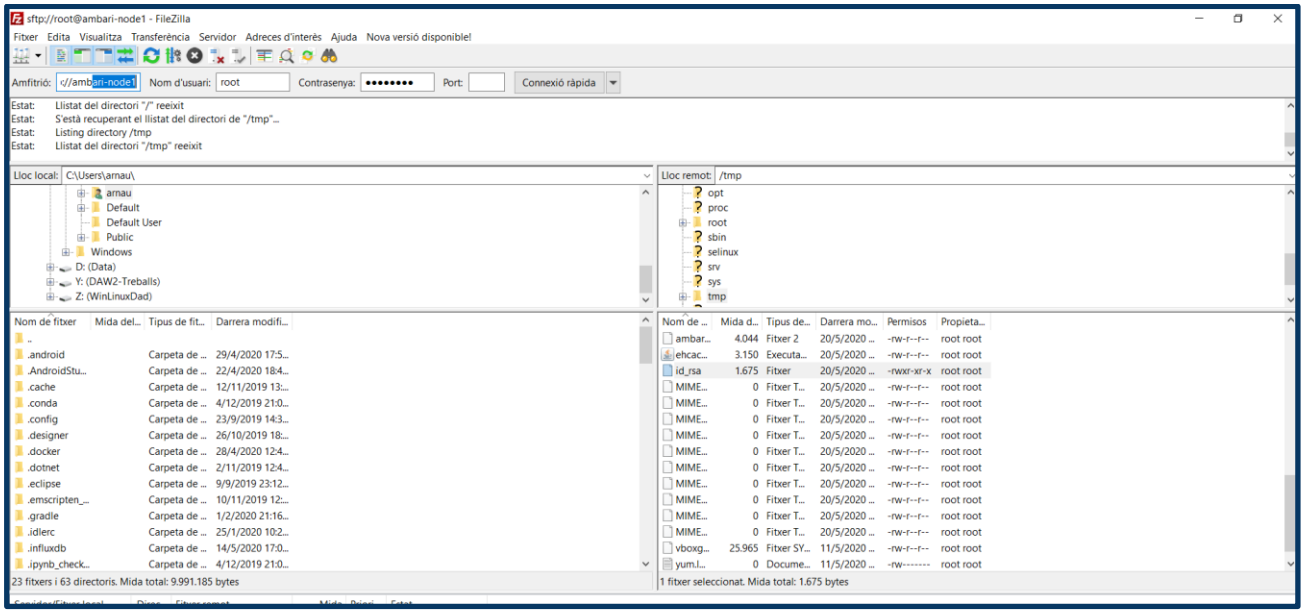
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

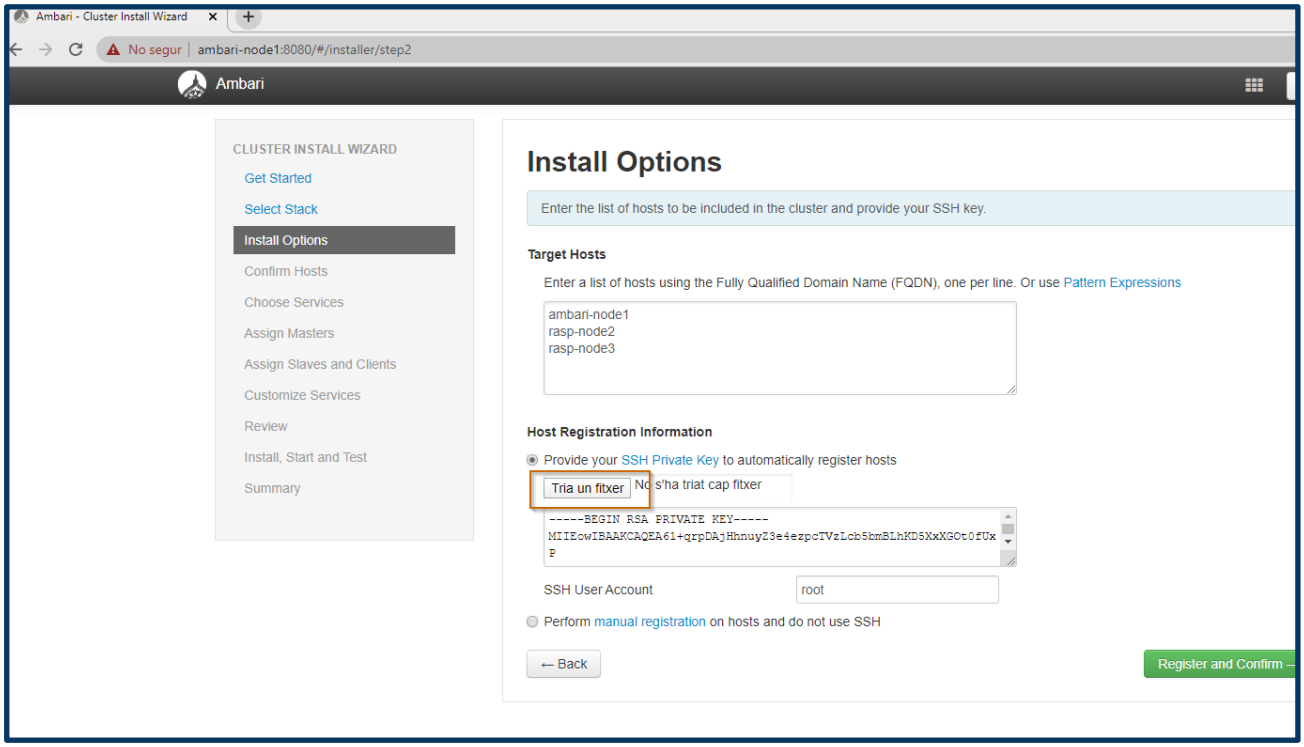




Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

NOTA: Però si volguéssim configurar-ho el clúster des de Windows 10 a més a més hauríem descarregar la clau privada (id_rsa) del server Ambari utilitzant Filezilla, ja que desde el navegador no podriem.





Nom i Cognoms | **Data**

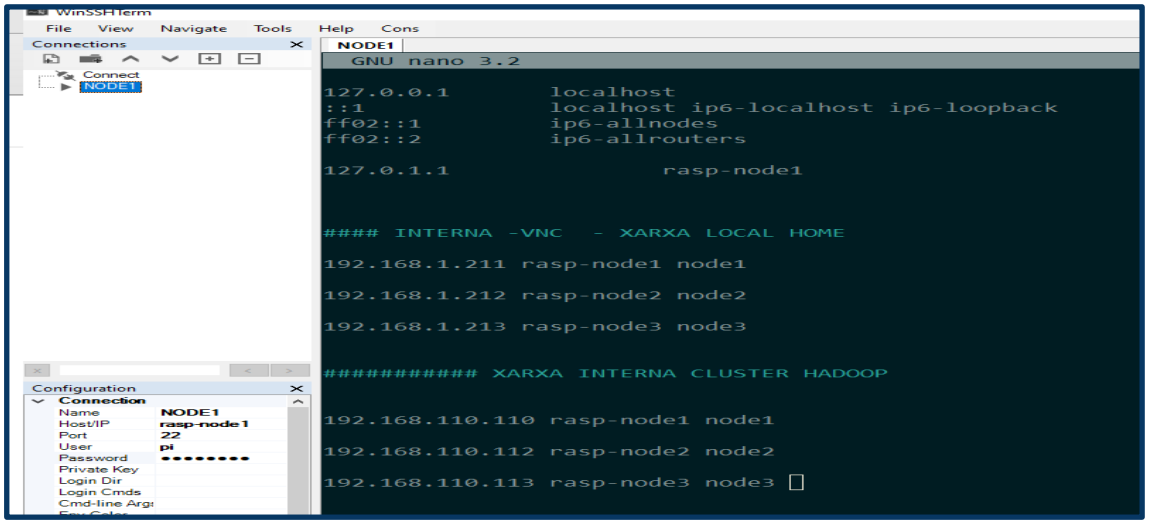
Arnau Subirós Puigarnau | 02-06-2020

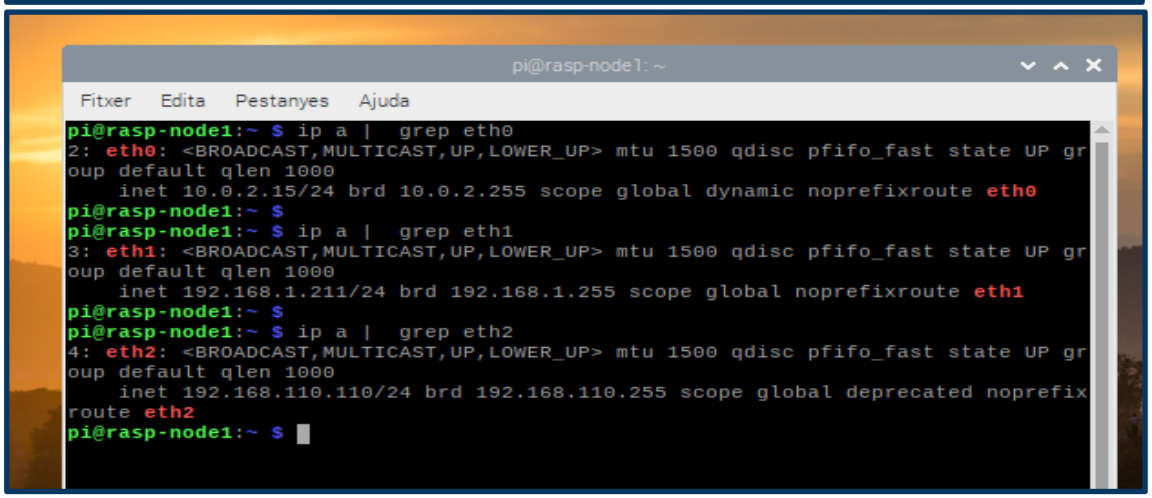
4. Annexe: Raspberry Pi Desktop (testing)

4.1. Intent amb Raspberri Pi Desktop (Debian 9 i Debian 10)

- Com que la idea original era fer un clúster d' Hadoop amb 3 Raspberry Pi, però a causa de la COVID-19 no s'ha pogut fer per falta de dispositius, s'ha intentat fer-ho al VirtualBox. S'ha de dir que VirtualBox no permet l'arquitectura ARM , però hi han isos de Raspberry Pi Desktop(Debian 9) i Raspberry Pi Desktop(Debian 10) que són un Debian amb la interfície gràfica de la Raspberry Pi (amb algunes limitacions).
    - Es volia fer un clúster de Hadoop utilitzant Ambari i un nou sistema operatiu.

- Com s'ha fet anteriorment. Després d'instal·lar una màquina, s'ha clonat diversos cops.
- S'ha configurat el hostname, la xarxa i el SSH.

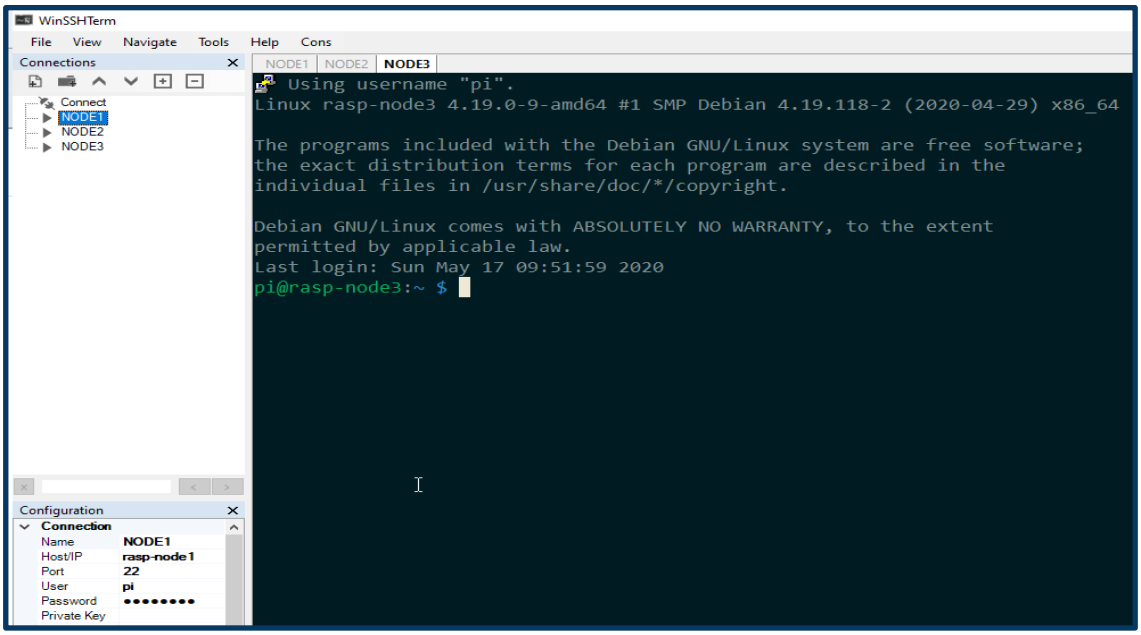




- Com que les Guest Additions tot i estar instal.lades , no canviada la resolució de pantalla, es va instal.lar VNC SERVER al node master per establir connexió remota desde el host amfitrió (Windows 10)
- Després s'ha aconseguit amb la comanda "xrand -s + tamany (sudo xrand -s 1680x1050)

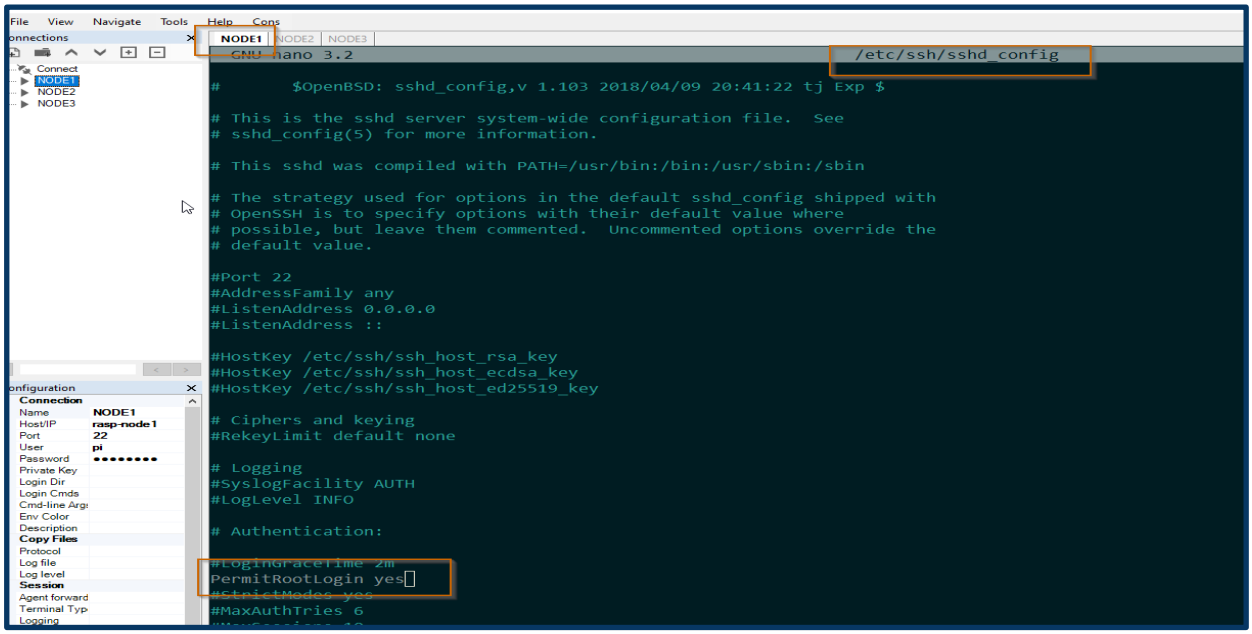
Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020

- Hem afegit les 3 màquines a WinSSHTerm per podent-se connectar via SSH



- Respecte a SSH, en les distribucions Debian per defecte si volem accedir SSH amb l'usuari "root" està deshabilitat. Després de la momificació, s'ha de reiniciar el servei SSH.

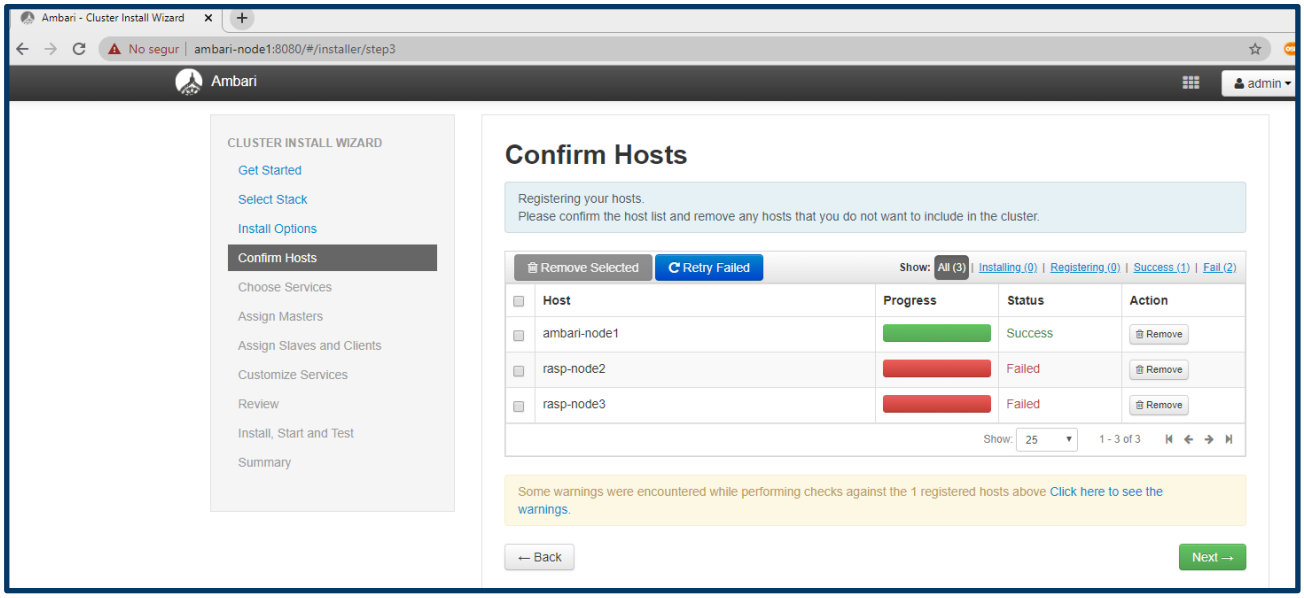
root@rasp-node1 : ~/.ssh# nano /etc/ssh/sshd_config



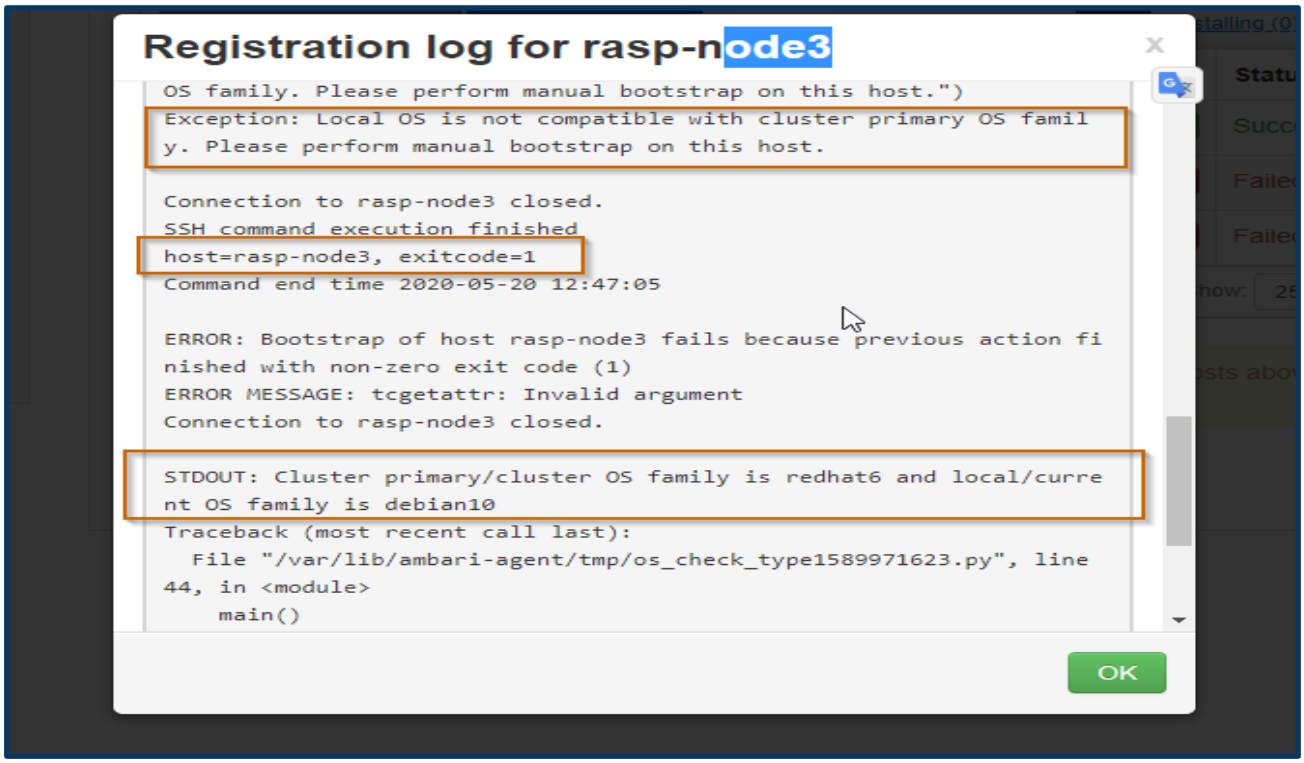
- Un cop ja hem creat les claus públiques de cada node com ja hem explicat anteriorment. Ens hem de descarregar els repositoris de Ambari 2.2.1
    - Només està disponible per Debian 7(s'ha intentat amb aquesta i versions d'Ambari superior sense èxit, donava error.)

4.2. Intent de S.O. hibrid (master :CentOS 6 i 2 slaves: Raspberri Pi Desktop)

- S'ha provat una altra alternativa, utilitzar el node master on hi ha el servidor d' Ambari que anteriorment s'havia configurat amb un CentOS 6 i fer que els slaves siguin Raspberri Pi Desktop. Però al moment de confirmar els hosts, ha donat error, ja que troba sistema operatius diferents.
- S'ha intentat sense èxit instal·lar els agents d'Ambari de forma manual



Nom i Cognoms	Data
Arnau Subirós Puigarnau	02-06-2020



Nom i Cognoms **Data**

Arnau Subirós Puigarnau 02-06-2020

5. Conclusions finals

Per finalitzar aquest projecte, podem confirmar que tot i les alternatives del confinament del Covid-19 per poder fer viable el projecte s'han pogut assolir els objectius de crear un clúster Hadoop de diversos nodes. S'ha de reconèixer que ha sigut un tema molt apassionant i enriquidor.

- Hem començat amb el clúster pseudodistribuit (1 node) per començar a practicar i tenir una mínima configuració, ja que després faríem clonacions. S'han fet diversos exemples explicatius:
  - HDFS
  - HDFS Snapshot
  - Diversos processos Mapreduce

- S'ha creat un cluster Hadoop (1 master i 2 slaves) .S'han fet diversos exemples explicatius:
  - Llençar diversos processos Mapreduce contra el clúster
  - Llençar un job amb una nova configuracio de Yarn Schelduler
  - Un exemple de Spark-Shell
  - Un exemple de Pyspark
  - Un exemple de Spark-Shell interactuant amb HDFS

- S'ha creat de 0 un altre cluster utilitzant Ambari
  - S'ha configurat Hive pel seu funcionament
  - Visualitzacio de Ambari sobre Windows 10

- Finalment s'ha volgut fer proves amb Raspberri Pi Desktop amb la finalitat de crear un clúster Hadoop amb Ambari però ha donat molts problemes de repositoris, versions i falta de dependències que per falta de temps s'ha volgut deixar-ho.

Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

6. Bibliografia

- **Clúster Hadoop en Centos**
  - https://www.tecmint.com/install-hadoop-multinode-cluster-in-centos/
  - https://tecadmin.net/set-up-hadoop-multi-node-cluster-on-centos-redhat/
  - https://jmchung.github.io/post/running-hadoop-on-centos-6-multi-node-cluster/
  - https://medium.com/@tiago.slucas/set-up-hadoop-cluster-on-virtualbox-machines-running-centos-7-df9121b1eab0

- **Clúster Hadoop utilitzant Apache Ambari**
  - https://www.alibabacloud.com/blog/set-up-a-hadoop-cluster-with-apache-ambari_595722
  - https://hadoopjournal.com/2015/08/09/hortonworks-hadoop-installation-using-apache-ambari-on-centos6/
  - https://community.cloudera.com/t5/Support-Questions/App-Timeline-Server-not-start/td-p/230460
  - https://community.cloudera.com/t5/Support-Questions/HDFS020-Could-not-write-file-user-admin-hive-jobs-hive-job/td-p/153109
  - https://docs.cloudera.com/HDPDocuments/Ambari-2.2.0.0/bk_ambari_views_guide/content/_configuring_your_cluster_for_files_view.html

- **Instal.lar Hadoop a Ubuntu 18.04 Bionic Beaver**
  - https://linuxconfig.org/how-to-install-hadoop-on-ubuntu-18-04-bionic-beaver-linux

- **Editor de Text (Vi, Gedit)**
  - https://docs.oracle.com/cd/E19620-01/805-7644/x-5lei2/index.html
  - https://askubuntu.com/questions/52523/errors-shown-in-terminal-when-editing-files-with-gedit
  - https://ajpdsoft.com/modules.php?name=Foros&file=viewtopic&t=666

- **CentOs6- canvi de versió de Python**
  - http://www.ghanshammahajan.com/how-to-upgrade-python-2-6-to-2-7-on-centos-6/

- **Raspberry Pi Desktop**
  - http://www.aoakley.com/articles/2017-07-04-raspbian-x86-virtualbox.php
  - https://github.com/superjamie/lazyweb/wiki/Raspberry-Pi-Debian-Backports
  - https://www.raspberrypi.org/documentation/remote-access/vnc/
  - https://www.osboxes.org/raspbian/

Nom i Cognoms | **Data**

Arnau Subirós Puigarnau | 02-06-2020

Agraïments:

M'agradaria expressar tot el meu agraïment al meu professor responsable, el Sr. Sergi Grau per tot l'acompanyament i suport rebut durant el transcurs del projecte.