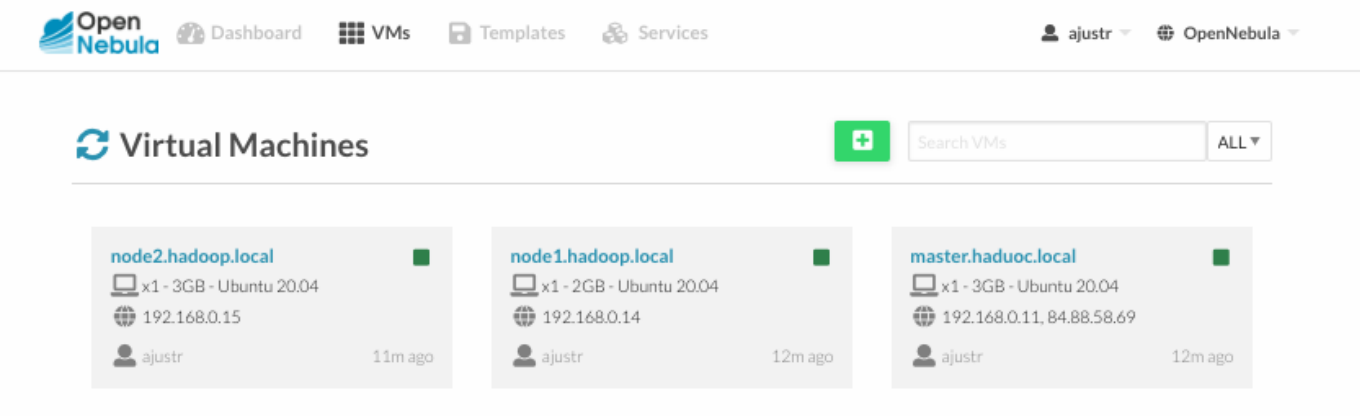
Despliegue de un cluster de Hadoop.

Se va a construir un clúster virtualizado sobre OpenNebula de 3 MV para desplegar y utilizar Hadoop.

> Desplegamos un clúster Hadoop con un Master (master) con una IP privada y una pública y dos Workers (node1 y node2) con IP privadas. Se usa el template de Ubuntu pero es equivalente en Debian11 o en los otros templates.



> **a.** Agrego el /etc/hosts las IP privadas asociadas a los nombres de las MV.

```
➜  ~ ssh -i .ssh/id_rsa root@84.88.58.69 -p 55000

        The authenticity of host [84.88.58.69]:55000 ([84.88.58.69]:55000)
cant be established.
        ED25519 key fingerprint is
SHA256:Tt3z4goUvS/eBUazUe9Ri848bNv3Y/Mv/HX9+13bMAU.
        This key is not known by any other names.
        Are you sure you want to continue connecting
(yes/no/[fingerprint])? yes
        Warning: Permanently added [84.88.58.69]:55000 (ED25519) to the
list of known hosts.
        Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@localhost:~#

# cambio nombre
root@localhost:~# hostnamectl set-hostname master.hadoop.local
root@localhost:~# hostname
    master.hadoop.local

root@localhost:~# sudo reboot
```

Tranfiriendo Clave privada al servidor de salto

```
➜  ~ cat .ssh/id_rsa.pub
    ssh-rsa AA...

# transfiriendo clave privada al Server de salto
➜  ~ scp -P 55000 .ssh/id_rsa root@84.88.58.69:/root/.ssh
    id_rsa

# accedo al Server de salto
➜  ~ ssh -i .ssh/id_rsa root@84.88.58.69 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@master:~# ls -la /root/.ssh
    total 20
    drwx------ 2 root root 4096 May 13 10:55 .
    drwx------ 5 root root 4096 May 13 10:02 ..
    -rw------- 1 root root  582 May 13 09:39 authorized_keys
    -rw------- 1 root root 2622 May 13 10:55 id_rsa
    -rw-r--r-- 1 root root  444 May 13 10:46 known_hosts
```

Habilitar IP Forwarding

```
# Habilitar IP Forwarding:
root@localhost:~#  echo 1 > /proc/sys/net/ipv4/ip_forward

# Habilita IP forwarding en el servidor
root@localhost:~# nano /etc/sysctl.conf

        # Uncomment the next line to enable packet forwarding for IPv4
        net.ipv4.ip_forward=1

# Aplicando cambios
root@localhost:~# sysctl -p
net.ipv4.ip_forward = 1
```

Configurar NAT

```
# identificar el nombre de mi interfaz (en este caso es eth0)
root@master:~# ip addr
    1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
group default qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
    2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
        link/ether 02:00:c0:a8:00:0b brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.11/24 brd 192.168.0.255 scope global eth0
```

```
        valid_lft forever preferred_lft forever
        inet6 fe80::c0ff:fea8:b/64 scope link
        valid_lft forever preferred_lft forever
    3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel
state UP group default qlen 1000
        link/ether 02:00:54:58:3a:45 brd ff:ff:ff:ff:ff:ff
        inet 84.88.58.69/26 brd 84.88.58.127 scope global eth1
        valid_lft forever preferred_lft forever
        inet6 fe80::54ff:fe58:3a45/64 scope link
        valid_lft forever preferred_lft forever

# Este comando configura iptables para hacer NAT de los paquetes que salen
del servidor.
root@localhost:~# iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

# añado reglas de tráfico según enunciado ejercicio 2
root@localhost:~# iptables -A FORWARD -i eth1 -o eth0 -m state --state
RELATED,ESTABLISHED -j ACCEPT
root@localhost:~# iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT

# Hacemos las reglas persistentes (si a todo)
root@localhost:~# sudo apt-get update
root@localhost:~# apt-get install iptables-persistent
```

```
root@master:~# ip route add default via 192.168.0.11
    RTNETLINK answers: File exists

root@master:~# ip route show
    default via 84.88.58.65 dev eth1 onlink
    84.88.58.64/26 dev eth1 proto kernel scope link src 84.88.58.69
    192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.11
```

Verifico la configuración de NAT con master

```
# Verificar la configuración de iptables
root@master:~# ping -c 3 google.com
    PING google.com (142.250.200.110) 56(84) bytes of data.
    64 bytes from mad41s13-in-f14.1e100.net (142.250.200.110): icmp_seq=1
ttl=119 time=14.0 ms
    64 bytes from mad41s13-in-f14.1e100.net (142.250.200.110): icmp_seq=2
ttl=119 time=14.0 ms
    64 bytes from mad41s13-in-f14.1e100.net (142.250.200.110): icmp_seq=3
ttl=119 time=14.0 ms

    --- google.com ping statistics ---
    3 packets transmitted, 3 received, 0% packet loss, time 2004ms
    rtt min/avg/max/mdev = 13.983/13.996/14.020/0.017 ms

root@master:~# iptables -t nat -L -v -n
```

```
    Chain PREROUTING (policy ACCEPT 508 packets, 34374 bytes)
    pkts bytes target     prot opt in     out     source
destination

    Chain INPUT (policy ACCEPT 508 packets, 34374 bytes)
    pkts bytes target     prot opt in     out     source
destination

    Chain OUTPUT (policy ACCEPT 10 packets, 730 bytes)
    pkts bytes target     prot opt in     out     source
destination

    Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
    pkts bytes target     prot opt in     out     source
destination
    10   730 MASQUERADE  all  --  *      eth1    0.0.0.0/0
0.0.0.0/0

root@master:~# iptables -L -v -n
    Chain INPUT (policy ACCEPT 4694 packets, 684K bytes)
    pkts bytes target     prot opt in     out     source
destination

    Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
    pkts bytes target     prot opt in     out     source
destination
        0     0 ACCEPT     all  --  eth1   eth0    0.0.0.0/0
0.0.0.0/0            state RELATED,ESTABLISHED
        0     0 ACCEPT     all  --  eth0   eth1    0.0.0.0/0
0.0.0.0/0

    Chain OUTPUT (policy ACCEPT 4946 packets, 884K bytes)
    pkts bytes target     prot opt in     out     source
destination
```

Doy DNS a nodos y agregamos la puerta de enlace predeterminada 192.168.0.11

Estos pasos permiten el tráfico de Internet hacia y desde los nodos a través del servidor

y **agrego el /etc/hosts las IP privadas asociadas a los nombres de las MV**

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

# el cambio el nombre
root@localhost:~# hostnamectl set-hostname node1
root@localhost:~# sudo reboot
    sudo: unable to resolve host nodo1: Temporary failure in name
resolution

# conectando a nodo1
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 55000
```

```
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@nodo1:~#
root@nodo1:~# cat /etc/resolv.conf
    nameserver 8.8.8.8
    nameserver 8.8.4.4

# Agrego el /etc/hosts las IP privadas asociadas
root@nodo1:~# sudo nano /etc/hosts
    127.0.0.1 localhost
    192.168.0.11     master.hadoop.local master
    192.168.0.14     nodo1.hadoop.local. nodo1
    192.168.0.15     nodo2.hadoop.local nodo2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts

root@nodo1:~# sudo ip route add default via 192.168.0.11
```

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@nodo2:~# cat /etc/resolv.conf
    nameserver 8.8.8.8
    nameserver 8.8.4.4

root@nodo2:~# sudo nano /etc/hosts
root@nodo2:~# cat /etc/hosts
    127.0.0.1 localhost
    192.168.0.11     master.hadoop.local master
    192.168.0.14     nodo1.hadoop.local. nodo1
    192.168.0.15     nodo2.hadoop.local nodo2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts

root@nodo2:~# sudo ip route add default via 192.168.0.11
    RTNETLINK answers: File exists

root@nodo2:~# ip route show
    default via 192.168.0.11 dev eth0
    192.168.0.0/24 dev eth0 proto kernel scope link src 192.168.0.15
```

```
➜  ~ ssh -i .ssh/id_rsa root@84.88.58.69 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

# Agrego el /etc/hosts las IP privadas asociada
root@master:~# sudo nano /etc/hosts
    127.0.0.1 localhost
    192.168.0.11     master.hadoop.local master
    192.168.0.14     node1.hadoop.local. node1
    192.168.0.15     node2.hadoop.local node2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts
```

```
# accedo a nodo 1 desde master
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 55000
    The authenticity of host [192.168.0.15]:55000 ([192.168.0.15]:55000)
cant be established.

# el cambio el nombre
root@localhost:~# hostnamectl set-hostname node2
root@localhost:~# sudo reboot

# conectando a nodo2
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 55000
root@nodo2:~#

# Agrego el /etc/hosts las IP privadas asociadas
root@nodo2:~# sudo nano /etc/hosts
    127.0.0.1 localhost
    192.168.0.11     master.hadoop.local master
    192.168.0.14     nodo1.hadoop.local. nodo1
    192.168.0.15     nodo2.hadoop.local nodo2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts
```

Verificamos conexiones

```
root@master:~# ping 192.168.0.14
    PING 192.168.0.14 (192.168.0.14) 56(84) bytes of data.
```

```
    64 bytes from 192.168.0.14: icmp_seq=1 ttl=64 time=1.37 ms
    64 bytes from 192.168.0.14: icmp_seq=2 ttl=64 time=0.908 ms
    64 bytes from 192.168.0.14: icmp_seq=3 ttl=64 time=0.777 ms
    64 bytes from 192.168.0.14: icmp_seq=4 ttl=64 time=0.801 ms
    --- 192.168.0.14 ping statistics ---
    4 packets transmitted, 4 received, 0% packet loss, time 3015ms
    rtt min/avg/max/mdev = 0.777/0.964/1.371/0.239 ms

root@master:~# ping 192.168.0.15
    PING 192.168.0.15 (192.168.0.15) 56(84) bytes of data.
    64 bytes from 192.168.0.15: icmp_seq=1 ttl=64 time=1.00 ms
    64 bytes from 192.168.0.15: icmp_seq=2 ttl=64 time=0.857 ms
    64 bytes from 192.168.0.15: icmp_seq=3 ttl=64 time=0.664 ms
    --- 192.168.0.15 ping statistics ---
    3 packets transmitted, 3 received, 0% packet loss, time 2003ms
    rtt min/avg/max/mdev = 0.664/0.840/1.000/0.137 ms

root@master:~# ping 8.8.8.8
    PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
    64 bytes from 8.8.8.8: icmp_seq=1 ttl=119 time=18.0 ms
    64 bytes from 8.8.8.8: icmp_seq=2 ttl=119 time=18.2 ms
    64 bytes from 8.8.8.8: icmp_seq=3 ttl=119 time=18.1 ms
    --- 8.8.8.8 ping statistics ---
    4 packets transmitted, 3 received, 25% packet loss, time 3006ms
    rtt min/avg/max/mdev = 18.038/18.120/18.198/0.065 ms
```

```
root@nodo1:~# ping 192.168.0.15
    PING 192.168.0.15 (192.168.0.15) 56(84) bytes of data.
    64 bytes from 192.168.0.15: icmp_seq=1 ttl=64 time=2.64 ms
    64 bytes from 192.168.0.15: icmp_seq=2 ttl=64 time=0.707 ms
    ^C
    --- 192.168.0.15 ping statistics ---
    2 packets transmitted, 2 received, 0% packet loss, time 1002ms
    rtt min/avg/max/mdev = 0.707/1.675/2.644/0.968 ms

root@nodo1:~# ping 8.8.8.8
    PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
    64 bytes from 8.8.8.8: icmp_seq=1 ttl=118 time=18.7 ms
    64 bytes from 8.8.8.8: icmp_seq=2 ttl=118 time=18.7 ms
    64 bytes from 8.8.8.8: icmp_seq=3 ttl=118 time=18.8 ms
    ^C
    --- 8.8.8.8 ping statistics ---
    3 packets transmitted, 3 received, 0% packet loss, time 2003ms
    rtt min/avg/max/mdev = 18.730/18.763/18.827/0.045 ms
```

```
root@nodo2:~# ping 192.168.0.14
    PING 192.168.0.14 (192.168.0.14) 56(84) bytes of data.
    64 bytes from 192.168.0.14: icmp_seq=1 ttl=64 time=0.991 ms
    64 bytes from 192.168.0.14: icmp_seq=2 ttl=64 time=0.818 ms
```

```
    --- 192.168.0.14 ping statistics ---
    2 packets transmitted, 2 received, 0% packet loss, time 1001ms
    rtt min/avg/max/mdev = 0.818/0.904/0.991/0.086 ms

root@nodo2:~# ping 8.8.8.8
    PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
    64 bytes from 8.8.8.8: icmp_seq=1 ttl=118 time=18.6 ms
    64 bytes from 8.8.8.8: icmp_seq=2 ttl=118 time=18.8 ms
    --- 8.8.8.8 ping statistics ---
    2 packets transmitted, 2 received, 0% packet loss, time 1001ms
    rtt min/avg/max/mdev = 18.618/18.717/18.817/0.099 ms
```

> **b)** creo en todas las MV un usuario hadoop e intalo el JDK/JRE en todas las MV: apt install default-jdk default-jre y verificar con un java -version

Master

Ejecuto el siguiente comando para:

- crear un nuevo **usuario llamado hadoop**.
- Este comando también **creará un directorio home** para el usuario y
- le **asignará una shell predeterminada**.

```
root@master:~# sudo adduser hadoop
    Adding user `hadoop' ...
    Adding new group `hadoop' (1000) ...
    Adding new user `hadoop' (1000) with group `hadoop' ...
    Creating home directory `/home/hadoop' ...
    Copying files from `/etc/skel' ...
    New password: hadoop1234
    Retype new password: hadoop1234
    passwd: password updated successfully
    Changing the user information for hadoop
    Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
    Is the information correct? [Y/n] Y
root@master:~#
```

Instalo el JDK en master. Instalo el paquete default-jdk, que es suficiente para satisfacer los requisitos de Java para Hadoop.

```
# actualizar paquetes
root@master:~# sudo apt update && sudo apt upgrade -y
```

```
# Instalar el JDK
# paquete default-jdk
root@master:~# sudo apt install default-jdk -y
    Reading package lists... Done
    done.
    done.
    Processing triggers for mime-support (3.64ubuntu1) ...
    Processing triggers for libc-bin (2.31-0ubuntu9.15) ...

root@master:~# java -version
    openjdk version "11.0.22" 2024-01-16
    OpenJDK Runtime Environment (build 11.0.22+7-post-Ubuntu-
0ubuntu220.04.1)
    OpenJDK 64-Bit Server VM (build 11.0.22+7-post-Ubuntu-0ubuntu220.04.1,
mixed mode, sharing)
```

Nodo 1

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@nodo1:~# sudo adduser hadoop
    sudo: unable to resolve host nodo1: Temporary failure in name
resolution
    Adding user `hadoop' ...
    Adding new group `hadoop' (1000) ...
    Adding new user `hadoop' (1000) with group `hadoop' ...
    Creating home directory `/home/hadoop' ...
    Copying files from `/etc/skel' ...
    New password:
    Retype new password:
    passwd: password updated successfully
    Changing the user information for hadoop
    Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
    Is the information correct? [Y/n] Y


root@nodo1:~#  sudo apt update && sudo apt upgrade -y

# Instalar el JDK
root@nodo1:~#  sudo apt install default-jdk -y
    Reading package lists... Done
    done.
    done.
    Processing triggers for mime-support (3.64ubuntu1) ...
    Processing triggers for libc-bin (2.31-0ubuntu9.15) ...
```

```
root@nodo1:~# java -version
    openjdk version "11.0.22" 2024-01-16
    OpenJDK Runtime Environment (build 11.0.22+7-post-Ubuntu-
0ubuntu220.04.1)
    OpenJDK 64-Bit Server VM (build 11.0.22+7-post-Ubuntu-0ubuntu220.04.1,
mixed mode, sharing)
```

Nodo 2

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@nodo2:~# sudo adduser hadoop
    Adding user `hadoop' ...
    Adding new group `hadoop' (1000) ...
    Adding new user `hadoop' (1000) with group `hadoop' ...
    Creating home directory `/home/hadoop' ...
    Copying files from `/etc/skel' ...
    New password:
    Retype new password:
    passwd: password updated successfully
    Changing the user information for hadoop
    Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
    Is the information correct? [Y/n] Y

# Instalar el JDK
root@nodo2:~#  sudo apt install default-jdk -y
    Reading package lists... Done
    done.
    done.
    Processing triggers for mime-support (3.64ubuntu1) ...
    Processing triggers for libc-bin (2.31-0ubuntu9.15) ...

root@nodo2:~# java -version
    openjdk version "11.0.22" 2024-01-16
    OpenJDK Runtime Environment (build 11.0.22+7-post-Ubuntu-
0ubuntu220.04.1)
    OpenJDK 64-Bit Server VM (build 11.0.22+7-post-Ubuntu-0ubuntu220.04.1,
mixed mode, sharing)
```

c) los comandos se ejecutarán como non-root user y a través de sudo por lo cual en cada MV entrar como root y ejecutar visudo agregando el usuario hadoop en una línea abajo de root como: hadoop ALL=(ALL:ALL) ALL

Configurar sudo para el Usuario hadoop

Para permitir que el usuario `hadoop` ejecute comandos con privilegios elevados, necesitas añadirlo al archivo de configuración de sudoers. Esto se hace de manera segura editando el archivo con `visudo`, que verifica la sintaxis antes de guardar cambios que podrían bloquear el acceso sudo.

```
# Añade la siguiente línea al final del archivo:
root@master:~# sudo visudo
    hadoop ALL=(ALL:ALL) ALL
# Verificar la Config
root@master:~# sudo -u hadoop sudo whoami
    [sudo] password for hadoop:
    root

# nodo1
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)
# Añade la siguiente línea al final del archivo:
root@nodo1:~# sudo visudo
    hadoop ALL=(ALL:ALL) ALL
# Verificar la Config
root@nodo1:~# sudo -u hadoop sudo whoami
    [sudo] password for hadoop:
    root

# nodo2
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 55000
    Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-52-generic x86_64)

# Añade la siguiente línea al final del archivo:
root@nodo2:~# sudo visudo
    hadoop ALL=(ALL:ALL) ALL

# Verificar la Config
root@nodo2:~# sudo -u hadoop sudo whoami
    [sudo] password for hadoop:
    root
```

> **d)** /etc/hosts

```
root@nodo2:~# cat /etc/hosts
    127.0.0.1 localhost
    192.168.0.11     master.hadoop.local master
    192.168.0.14     nodo1.hadoop.local. nodo1
    192.168.0.15     nodo2.hadoop.local nodo2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
```

```
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts

root@nodo1:~# cat /etc/hosts
    127.0.0.1 localhost
    192.168.0.11 master.hadoop.local master
    192.168.0.14 nodo1.hadoop.local nodo1
    192.168.0.15 nodo2.hadoop.local nodo2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts

root@master:~# cat /etc/hosts
    127.0.0.1 localhost
    192.168.0.11     master.hadoop.local master
    192.168.0.14     nodo1.hadoop.local. nodo1
    192.168.0.15     nodo2.hadoop.local nodo2
    # The following lines are desirable for IPv6 capable hosts
    ::1 ip6-localhost ip6-loopback
    fe00::0 ip6-localnet
    ff00::0 ip6-mcastprefix
    ff02::1 ip6-allnodes
    ff02::2 ip6-allrouters
    ff02::3 ip6-allhosts
```

> **e) Arquitectura del Hadoop Cluster**: antes de configurar los nodos master & worker es importante definir los elementos de la arquitectura de un Hadoop cluster: el master mantiene el conocimiento sobre el HDFS y planifica los recursos. Este nodo tendrá dos daemons: NameNode que maneja el DFS y el ResourceManager que maneja los trabajos YARN (jobs) y ejecuta los procesos en los worker nodes.

Los Worker almacenan los datos y proveen la potencia para ejecutar los trabajos (ellos serán node1 & node2) y tendrán dos daemons: DataNode que maneja los datos sobre el nodo y se llama NameNode (igual que en el master) y NodeManager que maneja la ejecución de tareas sobre el nodo.

Antes de configurar los nodos maestro y trabajadores, es importante definir los elementos de la arquitectura de un clúster Hadoop. El clúster se compone de nodos maestro y nodos trabajadores, cada uno desempeñando roles específicos.

Nodos Maestro (Master)

- `NameNode`: Este daemon gestiona el sistema de archivos distribuido de Hadoop (HDFS). Mantiene el árbol de directorios del sistema de archivos y la información de los bloques de datos almacenados en los DataNodes. El NameNode no almacena los datos del usuario, sino que maneja las tablas de metadatos que permiten localizar los datos distribuidos en los DataNodes.

- `ResourceManager`: Es el componente principal de YARN que gestiona los recursos del clúster y la planificación de las tareas. El ResourceManager coordina la asignación de recursos a las aplicaciones en el clúster.

Nodos Trabajadores (Workers)

- `DataNode`: Almacena y recupera los bloques de datos según las solicitudes de los clientes o del NameNode. Realiza operaciones como creación, eliminación y replicación de bloques bajo la instrucción del NameNode.
- `NodeManager`: Este daemon es responsable de la gestión de los recursos en su nodo. Se comunica con el ResourceManager para iniciar y monitorear la ejecución de contenedores (tareas) en el nodo.

Esta arquitectura asegura que el nodo maestro tenga el conocimiento y control sobre el sistema de archivos distribuido y la planificación de recursos, mientras que los nodos trabajadores se encargan del almacenamiento de datos y la ejecución de tareas.

> **f)** El master utilizará SSH para conectarse a través de PKI. En el master y como usuario hadoop crear las SSH key: ssh-keygen (introducir un a cada pregunta para no indicarle ninguna opción ni passwd). Visualizar la llave pública desde la consola cat /home/hadoop/.ssh/id_rsa.pub, copiar con el mouse, editar en node1/node2 el archivo /home/hadoop/.ssh/authorized_keys. pegarla y salvar (se podría copiar con scp o ssh-copy-id pero el ssh de las MV del OpenNebula solo deja conectarse por PKI y no por usuario/passwd, se podría configurar pero por política de seguridad de la UOC solo admite PKI). Cambiar las protecciones chmod 640 /home/hadoop/.ssh/authorized_keys

Configuración de Autenticación SSH mediante PKI

```
➜  ~ ssh -i .ssh/id_rsa root@84.88.58.69 -p 55000
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-52-generic x86_64)

# Cambiar al usuario hadoop
root@master:~# su - hadoop

# Genero el par de llaves SSH
hadoop@master:~$ ssh-keygen
    Generating public/private rsa key pair.
    Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
    Created directory '/home/hadoop/.ssh'.
    Enter passphrase (empty for no passphrase):
    Enter same passphrase again:
    Your identification has been saved in /home/hadoop/.ssh/id_rsa
    Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub
    The key fingerprint is:
    SHA256:WmiZaegfb2g7hRAz32XL+XS8uN8QHbcqxnXrqHNjKA4
hadoop@master.hadoop.local
    The key's randomart image is:
    +---[RSA 3072]----+
    |                 |
    |    +     o      |
    |     = . + o . ..|
    |    ...=. + . o.+|
```

```
    |    ..B.S  o +.+.|
    |   . o.o. . + +..|
    |    . oE   +.o.. |
    |     .+oo..oo+oo |
    |     .o+o...=oo..|
    +----[SHA256]-----+

# copio clave publica
hadoop@master:~$ cat /home/hadoop/.ssh/id_rsa.pub
    ssh-rsa AAAAB3NzaC1...6nxMz7M= hadoop@master.hadoop.local

hadoop@master:~$ exit
```

Agregar la Llave Pública a los Nodos Trabajadores: - A cada nodo trabajador como usuario `root`. - Edito el archivo `authorized_keys` del usuario `hadoop` y pego la llave pública copiada. - Me aseguro de que el directorio `.ssh` y el archivo `authorized_keys` existan.

En node1:

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 55000
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-52-generic x86_64)

# accediendo a hadoop
root@nodo1:~# su - hadoop

# creando carpeta
hadoop@nodo1:~$ ls -a
    .  ..  .bash_logout  .bashrc  .profile

hadoop@nodo1:~$ mkdir -p /home/hadoop/.ssh
hadoop@nodo1:~$ ls -a
    .  ..  .bash_logout  .bashrc  .profile  .ssh

# añadiendo codigo key de master hadoop
hadoop@nodo1:~$ nano /home/hadoop/.ssh/authorized_keys

# generando permisos
hadoop@nodo1:~$ chmod 640 /home/hadoop/.ssh/authorized_keys
hadoop@nodo1:~$ exit
    logout
root@nodo1:~# exit
    logout
```

En node2:

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 55000
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-52-generic x86_64)

root@nodo2:~# su - hadoop
```

```
hadoop@nodo2:~$ ls -a
.  ..  .bash_logout  .bashrc  .profile
hadoop@nodo2:~$ mkdir -p /home/hadoop/.ssh
hadoop@nodo2:~$ nano /home/hadoop/.ssh/authorized_keys

# generando permisos
hadoop@nodo2:~$ chmod 640 /home/hadoop/.ssh/authorized_keys
```

Con estos pasos, la autenticación SSH mediante PKI está configurada correctamente entre el nodo maestro y los nodos trabajadores.

> **g)** Descargar los binarios de Hadoop. Sobre master y como usuario hadoop descargar (si bien está la version 3.4 se ha utilizado la 3.3.5)

wget https://downloads.apache.org/hadoop/common/hadoop-3.3.5/hadoop-3.3.5.tar.gz tar -xzf hadoop-3.3.5.tar.gz mv hadoop-3.3.5 hadoop

```
# Descargar los Binarios de Hadoop
hadoop@master:~$ wget https://downloads.apache.org/hadoop/common/hadoop-
3.3.5/hadoop-3.3.5.tar.gz
   Saving to: ‘hadoop-3.3.5.tar.gz’
   hadoop-3.3.5.tar.gz            100%
[================================================>] 673.80M  2.94MB/s
in 3m 49s
   2024-05-14 16:40:04 (2.95 MB/s) - ‘hadoop-3.3.5.tar.gz’ saved
[706533213/706533213]

# Extraer el Archivo Tar
hadoop@master:~$ tar -xzf hadoop-3.3.5.tar.gz

# Mover el Directorio Extraído
hadoop@master:~$ mv hadoop-3.3.5 hadoop
```

> **h) Set Environment Variables**

```
# Agregar las Variables de Entorno al Archivo .profile
hadoop@master:~$ nano /home/hadoop/.profile
    PATH=/home/hadoop/hadoop/bin:/home/hadoop/hadoop/sbin:$PATH

# Agregar las Siguientes Líneas al Archivo .bashrc
hadoop@master:~$ nano /home/hadoop/.bashrc
    export HADOOP_HOME=/home/hadoop/hadoop
    export PATH=${PATH}:${HADOOP_HOME}/bin:${HADOOP_HOME}/sbin

# aplicar cambios
source /home/hadoop/.profile
source /home/hadoop/.bashrc
```

i) Configurar

```
# Encontrar la Ruta de JAVA_HOME
hadoop@master:~$ update-alternatives --display java
    java - auto mode
    link best version is /usr/lib/jvm/java-11-openjdk-amd64/bin/java
    link currently points to /usr/lib/jvm/java-11-openjdk-amd64/bin/java
    link java is /usr/bin/java
    slave java.1.gz is /usr/share/man/man1/java.1.gz
    /usr/lib/jvm/java-11-openjdk-amd64/bin/java - priority 1111
    slave java.1.gz: /usr/lib/jvm/java-11-openjdk-amd64/man/man1/java.1.gz

# Agregar la Variable JAVA_HOME al Archivo hadoop-env.sh
hadoop@master:~$ nano /home/hadoop/hadoop/etc/hadoop/hadoop-env.sh
    export JAVA_HOME=/usr/lib/jvm/java-11-openjdk-amd64
```

j) Configurar NameNode Location

```
# Agregar la Configuración de fs.default.name
hadoop@master:~$ nano /home/hadoop/hadoop/etc/hadoop/core-site.xml
    <configuration>
        <property>
            <name>fs.default.name</name>
            <value>hdfs://master:9000</value>
        </property>
    </configuration>
```

k) Set path for HDFS

La propiedad dfs.replication indica el número de replicas, si se pone 2 los datos estarán duplicados en 2 nodos (este valor no puede ser nunca superior al número de worker nodes).

```
# Agregar la Configuración de la Ruta de HDFS:
nano /home/hadoop/hadoop/etc/hadoop/hdfs-site.xml

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

    <configuration>
        <property>
            <name>dfs.namenode.name.dir</name>
            <value>/home/hadoop/data/nameNode</value>
        </property>
        <property>
```

```
            <name>dfs.datanode.data.dir</name>
            <value>/home/hadoop/data/dataNode</value>
        </property>
        <property>
            <name>dfs.replication</name>
            <value>2</value>
        </property>
    </configuration>
```

l) Set YARN as Job Scheduler

```
# Editar el Archivo mapred-site.xml
hadoop@master:~$ nano /home/hadoop/hadoop/etc/hadoop/mapred-site.xml

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

    <configuration>
        <property>
            <name>mapreduce.framework.name</name>
            <value>yarn</value>
        </property>
        <property>
            <name>yarn.app.mapreduce.am.env</name>
            <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
        </property>
        <property>
            <name>mapreduce.map.env</name>
            <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
        </property>
        <property>
            <name>mapreduce.reduce.env</name>
            <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
        </property>
    </configuration>
```

m) Configuro YARN:

```
# Configure YARN:
hadoop@master:~$ nano /home/hadoop/hadoop/etc/hadoop/yarn-site.xml
    <configuration>
    <!-- Site specific YARN configuration properties -->
        <property>
            <name>yarn.acl.enable</name>
            <value>0</value>
        </property>
        <property>
            <name>yarn.resourcemanager.hostname</name>
```

```
            <value>192.168.0.11</value>
        </property>
        <property>
            <name>yarn.nodemanager.aux-services</name>
            <value>mapreduce_shuffle</value>
        </property>
    </configuration>
```

n) Configure Workers

```
# Editar el Archivo workers
hadoop@master:~$ nano /home/hadoop/hadoop/etc/hadoop/workers
    nodo1
    nodo2
```

o) Duplicar los archivos de configuración sobre cada nodo: para ello editar en cada node /etc/ssh/sshd_config y hacia el final cambiar la segunda línea de Port 55000 por Port 22 salvar y reiniciar ssh con systemctl restart ssh. Se debe copiar hadoop en cada nodo worker.

Editar el Archivo de Configuración SSH en Cada Nodo

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 55000
root@nodo1:~# nano /etc/ssh/sshd_config
    # Example of overriding settings on a per-user basis
    #Match User anoncvs
    #       X11Forwarding no
    #       AllowTcpForwarding no
    #       PermitTTY no
    #       ForceCommand cvs server
    PasswordAuthentication no
    PermitRootLogin without-password
    UseDNS no
    Port 22
root@nodo1:~# systemctl restart ssh
```

```
root@nodo2:~# nano /etc/ssh/sshd_config
    # Example of overriding settings on a per-user basis
    #Match User anoncvs
    #       X11Forwarding no
    #       AllowTcpForwarding no
    #       PermitTTY no
    #       ForceCommand cvs server
    PasswordAuthentication no
    PermitRootLogin without-password
    UseDNS no
```

```
    Port 22
root@nodo2:~# systemctl restart ssh
```

```
root@master:~# sudo nano /etc/ssh/sshd_config
    # Example of overriding settings on a per-user basis
    #Match User anoncvs
    #       X11Forwarding no
    #       AllowTcpForwarding no
    #       PermitTTY no
    #       ForceCommand cvs server
    PasswordAuthentication no
    PermitRootLogin without-password
    UseDNS no
    Port 22
```

Copiar los Binarios de Hadoop a los Nodos Trabajadores

```
root@master:~# cd /home/hadoop/

root@master:/home/hadoop# ls -l
    total 689984
    drwxr-xr-x 10 hadoop hadoop      4096 Mar 15  2023 hadoop
    -rw-rw-r--  1 hadoop hadoop 706533213 Mar 15  2023 hadoop-3.3.5.tar.gz

root@master:/home/hadoop# scp hadoop-*.tar.gz nodo1:/home/hadoop
    The authenticity of host 'node1 (192.168.0.14)' can't be established.
    ECDSA key fingerprint is
SHA256:vp+hCP6fCLL8QkUQkOBkc4SvepgLa8d0gdS/5lfAxXw.
    Are you sure you want to continue connecting (yes/no/[fingerprint])?
yes
    Warning: Permanently added 'node1,192.168.0.14' (ECDSA) to the list of
known hosts.
    hadoop-3.3.5.tar.gz
99%  673MB  22.5MB/s   00:00 ETAscp: /home/hadoop/hadoop-3.3.5.tar.gz: No
space left on device
    hadoop-3.3.5.tar.gz
100%  674MB  24.0MB/s   00:28

root@master:/home/hadoop# scp hadoop-*.tar.gz nodo2:/home/hadoop
    The authenticity of host 'node2 (192.168.0.15)' can't be established.
    ECDSA key fingerprint is
SHA256:ipeXngFsI78NussSM1FtMEK0gI5Z32MZptr7XJ79pW8.
    Are you sure you want to continue connecting (yes/no/[fingerprint])?
yes
    Warning: Permanently added 'node2,192.168.0.15' (ECDSA) to the list of
known hosts.
    hadoop-3.3.5.tar.gz
100%  674MB  45.6MB/s   00:14
```

Descomprimir y Mover los Binarios en los Nodos Trabajadores

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.15 -p 22
root@nodo2:~# cd /home/hadoop
root@nodo2:/home/hadoop# ls
    hadoop-3.3.5.tar.gz
root@nodo2:/home/hadoop# tar -xzf hadoop-3.3.5.tar.gz
root@nodo2:/home/hadoop# mv hadoop-3.3.5 hadoop
```

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 22
root@nodo1:/home/hadoop# tar -xzf hadoop-3.3.5.tar.gz
root@nodo1:/home/hadoop# mv hadoop-3.3.5 hadoop
```

Copiar la Configuración de Hadoop desde el Nodo Maestro a los Trabajadores

```
root@master:~# for node in nodo1 nodo2;
> do scp -r /home/hadoop/hadoop/etc/hadoop/*
$node:/home/hadoop/hadoop/etc/hadoop/
> done
    capacity-scheduler.xml                        100% 9213   427.6KB/s
00:00
    configuration.xsl                             100% 1335   679.1KB/s
00:00
    container-executor.cfg                        100% 2567     1.0MB/s
00:00
    ...
    ...
    yarnservice-log4j.properties                  100% 2567     1.0MB/s
00:00
```

p) Format HDFS: ejecutar sobre master, hdfs namenode -format

> **p) Format HDFS**: ejecutar sobre master, hdfs namenode -format

`hdfs namenode -format`: Este comando formatea el sistema de archivos del NameNode. Es importante notar que este comando debe ejecutarse una sola vez durante la configuración inicial del clúster Hadoop. Si se ejecuta nuevamente, se perderán todos los datos existentes en el HDFS.

```
root@master:~# su - hadoop

hadoop@master:~$ hdfs namenode -format
    WARNING: /home/hadoop/hadoop/logs does not exist. Creating.
    2024-05-15 05:21:45,699 INFO namenode.NameNode: STARTUP_MSG:
    /************************************************************
    STARTUP_MSG: Starting NameNode
```

```
    STARTUP_MSG:   host = master.hadoop.local/192.168.0.11
    STARTUP_MSG:   args = [-format]
    STARTUP_MSG:   version = 3.3.5
    STARTUP_MSG:   classpath = /home/hado
    ...
    2024-05-15 05:21:50,074 INFO namenode.NameNode: SHUTDOWN_MSG:
    /************************************************************
    SHUTDOWN_MSG: Shutting down NameNode at
master.hadoop.local/192.168.0.11
    ************************************************************/
```

> **q) Start and Stop HDFS**: sobre master ejecutar start-dfs.sh (stop-dfs.sh para pararlo). Esto inicia NameNode & SecondaryNameNode sobre master, y DataNode sobre node1 & node2, de acuerdo a la configuración realizada.. Se puede ver con el comando jps y sobre master se verá algo como:

Iniciar HDFS:

Ejecuto el siguiente comando para iniciar los demonios de HDFS.

```
hadoop@master:~$ $HADOOP_HOME/sbin/start-dfs.sh
    Starting namenodes on [master]
    master: Warning: Permanently added 'master,192.168.0.11' (ECDSA) to
the list of known hosts.
    master: hadoop@master: Permission denied (publickey).
    Starting datanodes
    localhost: Warning: Permanently added 'localhost' (ECDSA) to the list
of known hosts.
    localhost: hadoop@localhost: Permission denied (publickey).
    nodo1: WARNING: /home/hadoop/hadoop/logs does not exist. Creating.
    nodo1: mkdir: cannot create directory ‘/home/hadoop/hadoop/logs’:
Permission denied
    nodo1: ERROR: Unable to create /home/hadoop/hadoop/logs. Aborting.
    nodo2: ERROR: JAVA_HOME /usr/lib/jvm/java-11-openjdk-amd64 does not
exist.
    Starting secondary namenodes [master.hadoop.local]
    master.hadoop.local: Warning: Permanently added 'master.hadoop.local'
(ECDSA) to the list of known hosts.
    master.hadoop.local: hadoop@master.hadoop.local: Permission denied
(publickey).
```

Problema

```
hadoop@master:~$ ls -l ~/.ssh/id_rsa
ls: cannot access '/home/hadoop/.ssh/id_rsa': No such file or directory
hadoop@master:~$ ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/home/hadoop/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
```

```
Enter same passphrase again:
Your identification has been saved in /home/hadoop/.ssh/id_rsa
Your public key has been saved in /home/hadoop/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:mqxvpm+6DxuWEzlDgOduPD9VHLHaXA35Qg+MpLM0144
hadoop@master.hadoop.local
The key's randomart image is:
+---[RSA 2048]----+
| ..    .o+..     |
|. ..   .oo=o     |
| o  . =.ooo+.    |
|  .. o Oo+. o    |
| o  = o.E ..     |
|  =  *.o         |
| . o*.+          |
|   .o*+          |
|    *%+          |
+----[SHA256]-----+
```

```
hadoop@master:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
hadoop@master:~$ chmod 700 ~/.ssh
hadoop@master:~$ chmod 600 ~/.ssh/authorized_keys
hadoop@master:~$ chmod 600 ~/.ssh/id_rsa
hadoop@master:~$ ssh hadoop@master.hadoop.local
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-181-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Thu May 16 17:36:42 UTC 2024

  System load:  0.0               Processes:             113
  Usage of /:   87.0% of 4.67GB   Users logged in:       1
  Memory usage: 6%                IPv4 address for eth1: 84.88.58.69
  Swap usage:   0%

  => / is using 87.0% of 4.67GB

 * Strictly confined Kubernetes makes edge and IoT secure. Learn how
MicroK8s
   just raised the bar for easy, resilient and secure K8s cluster
deployment.

   https://ubuntu.com/engage/secure-kubernetes-at-the-edge

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
```

```
See https://ubuntu.com/esm or run: sudo pro status

New release '22.04.3 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


*** System restart required ***

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 22
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-181-generic x86_64)

root@nodo1:~# su - hadoop
hadoop@nodo1:~$ mkdir -p ~/.ssh
hadoop@nodo1:~$ nano ~/.ssh/authorized_keys
hadoop@nodo1:~$ nano ~/.ssh/authorized_keys
hadoop@nodo1:~$ chmod 700 ~/.ssh
hadoop@nodo1:~$ chmod 600 ~/.ssh/authorized_keys
hadoop@nodo1:~$ exit
logout
root@nodo1:~# exit
logout
Connection to 192.168.0.14 closed.
```

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.26 -p 22
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-182-generic x86_64)

root@nodo2:~# su - hadoop
hadoop@nodo2:~$ mkdir -p ~/.ssh
hadoop@nodo2:~$ nano ~/.ssh/authorized_keys
hadoop@nodo2:~$ nano ~/.ssh/authorized_keys
hadoop@nodo2:~$ chmod 700 ~/.ssh
hadoop@nodo2:~$ chmod 600 ~/.ssh/authorized_keys
hadoop@nodo2:~$ exit
logout
root@nodo2:~# exit
logout
Connecion to 192.168.0.26 closed.
```

```
root@master:~# su - hadoop
```

```
hadoop@master:~$ ssh hadoop@nodo2
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-182-generic x86_64)
hadoop@nodo2:~$ exit
logout

hadoop@master:~$ ssh hadoop@nodo1
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-181-generic x86_64)
hadoop@nodo1:~$ exit
logout
Connection to nodo1 closed.
```

Problemas solucionados...

Arrancamos

```
hadoop@master:~$ start-dfs.sh
    Starting namenodes on [master]
    Starting datanodes
    nodo2: datanode is running as process 3399.  Stop it first and ensure
/tmp/hadoop-hadoop-datanode.pid file is empty before retry.
    nodo1: datanode is running as process 10977.  Stop it first and ensure
/tmp/hadoop-hadoop-datanode.pid file is empty before retry.
    Starting secondary namenodes [master.hadoop.local]

hadoop@master:~$ jps
    29872 Jps
    29698 SecondaryNameNode
    29364 NameNode
    29500 DataNode

hadoop@master:~$ ssh hadoop@nodo1
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-181-generic x86_64)

hadoop@nodo1:~$ jps
    10977 DataNode
    12364 Jps

hadoop@master:~$ ssh hadoop@nodo2
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-182-generic x86_64)

hadoop@nodo2:~$ jps
    3399 DataNode
    4730 Jps
```

> **r) Monitor del HDFS Cluster**: se puede obtener información con hdfs dfsadmin -report. Y también desde otra MV que ya tenga disponible con un navegador en la URL y que esté conectada a la red privada http://192.168.0.1:9870, donde 192.168.0.1 es la IP del master.

Obtener información del clúster HDFS

```
# reporte
hadoop@master:~$ hdfs dfsadmin -report
Configured Capacity: 15033864192 (14.00 GB)
Present Capacity: 2016595968 (1.88 GB)
DFS Remaining: 2016522240 (1.88 GB)
DFS Used: 73728 (72 KB)
DFS Used%: 0.00%
Replicated Blocks:
    Under replicated blocks: 0
    Blocks with corrupt replicas: 0
    Missing blocks: 0
    Missing blocks (with replication factor 1): 0
    Low redundancy blocks with highest priority to recover: 0
    Pending deletion blocks: 0
Erasure Coded Block Groups:
    Low redundancy block groups: 0
    Block groups with corrupt internal blocks: 0
    Missing block groups: 0
    Low redundancy blocks with highest priority to recover: 0
    Pending deletion blocks: 0

-------------------------------------------------
Live datanodes (3):

Name: 192.168.0.11:9866 (master.hadoop.local)
Hostname: master.hadoop.local
Decommission Status : Normal
Configured Capacity: 5011288064 (4.67 GB)
DFS Used: 24576 (24 KB)
Non DFS Used: 4361019392 (4.06 GB)
DFS Remaining: 633466880 (604.12 MB)
DFS Used%: 0.00%
DFS Remaining%: 12.64%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 0
Last contact: Thu May 16 17:57:17 UTC 2024
Last Block Report: Thu May 16 17:50:20 UTC 2024
Num of Blocks: 0


Name: 192.168.0.14:9866 (nodo1.hadoop.local.)
Hostname: nodo1.hadoop.local
Decommission Status : Normal
Configured Capacity: 5011288064 (4.67 GB)
DFS Used: 24576 (24 KB)
Non DFS Used: 4320751616 (4.02 GB)
DFS Remaining: 673734656 (642.52 MB)
```

```
DFS Used%: 0.00%
DFS Remaining%: 13.44%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 0
Last contact: Thu May 16 17:57:18 UTC 2024
Last Block Report: Thu May 16 17:50:18 UTC 2024
Num of Blocks: 0


Name: 192.168.0.26:9866 (nodo2.hadoop.local)
Hostname: nodo2.hadoop.local
Decommission Status : Normal
Configured Capacity: 5011288064 (4.67 GB)
DFS Used: 24576 (24 KB)
Non DFS Used: 4285165568 (3.99 GB)
DFS Remaining: 709320704 (676.46 MB)
DFS Used%: 0.00%
DFS Remaining%: 14.15%
Configured Cache Capacity: 0 (0 B)
Cache Used: 0 (0 B)
Cache Remaining: 0 (0 B)
Cache Used%: 100.00%
Cache Remaining%: 0.00%
Xceivers: 0
Last contact: Thu May 16 17:57:18 UTC 2024
Last Block Report: Thu May 16 17:50:16 UTC 2024
Num of Blocks: 0
```

Monitorear HDFS desde una URL Para monitorear el HDFS desde una URL en otra máquina que esté en la red privada, abre un navegador web y accede a la siguiente URL (reemplaza 192.168.0.1 con la IP del maestro):

```
http://192.168.0.1:9870
```

> **s) Subir y obtener datos del HDFS:** para leer y escribir en el HDFS se hace con hdfs dfs -comando. Primero se crea un directorio y el resto de los comandos serán en relación a este.

Para leer y escribir en el HDFS, sigue los siguientes pasos:

```
# 1. Crear un Directorio en HDFS
hadoop@master:~$ hdfs dfs -mkdir -p /user/hadoop

# 2. Crear un Subdirectorio books
hadoop@master:~$ hdfs dfs -mkdir /user/hadoop/books
```

```
# 3. Descargar los Libros del Proyecto Gutenberg
hadoop@master:~$ cd /home/hadoop
hadoop@master:~$ wget -O alice.txt https://www.gutenberg.org/files/11/11-
0.txt
--2024-05-16 18:03:47--  https://www.gutenberg.org/files/11/11-0.txt
Resolving www.gutenberg.org (www.gutenberg.org)... 152.19.134.47,
2610:28:3090:3000:0:bad:cafe:47
Connecting to www.gutenberg.org (www.gutenberg.org)|152.19.134.47|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 154638 (151K) [text/plain]
Saving to: ‘alice.txt’

alice.txt                           100%
[=================================================================>]
151.01K   396KB/s    in 0.4s

2024-05-16 18:03:48 (396 KB/s) - ‘alice.txt’ saved [154638/154638]

hadoop@master:~$ wget -O holmes.txt
https://www.gutenberg.org/files/1661/1661-0.txt
--2024-05-16 18:03:55--  https://www.gutenberg.org/files/1661/1661-0.txt
Resolving www.gutenberg.org (www.gutenberg.org)... 152.19.134.47,
2610:28:3090:3000:0:bad:cafe:47
Connecting to www.gutenberg.org (www.gutenberg.org)|152.19.134.47|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 607504 (593K) [text/plain]
Saving to: ‘holmes.txt’

holmes.txt                          100%
[=================================================================>]
593.27K   736KB/s    in 0.8s

2024-05-16 18:03:57 (736 KB/s) - ‘holmes.txt’ saved [607504/607504]

hadoop@master:~$ wget -O frankenstein.txt
https://www.gutenberg.org/files/84/84-0.txt
--2024-05-16 18:04:03--  https://www.gutenberg.org/files/84/84-0.txt
Resolving www.gutenberg.org (www.gutenberg.org)... 152.19.134.47,
2610:28:3090:3000:0:bad:cafe:47
Connecting to www.gutenberg.org (www.gutenberg.org)|152.19.134.47|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 448642 (438K) [text/plain]
Saving to: ‘frankenstein.txt’

frankenstein.txt                    100%
[=================================================================>]
438.13K   691KB/s    in 0.6s

2024-05-16 18:04:04 (691 KB/s) - ‘frankenstein.txt’ saved [448642/448642]
```

```
# 4. Subir los Libros al Directorio books en HDFS
hadoop@master:~$ hdfs dfs -put alice.txt holmes.txt frankenstein.txt
/user/hadoop/books

# 5. Listar el Contenido del Directorio books en HDFS
hadoop@master:~$ hdfs dfs -ls /user/hadoop/books
Found 3 items
-rw-r--r--   2 hadoop supergroup     154638 2024-05-16 18:06
/user/hadoop/books/alice.txt
-rw-r--r--   2 hadoop supergroup     448642 2024-05-16 18:06
/user/hadoop/books/frankenstein.txt
-rw-r--r--   2 hadoop supergroup     607504 2024-05-16 18:06
/user/hadoop/books/holmes.txt

# 6. Mover el Contenido del HDFS al Sistema de Archivos Local
hadoop@master:~$ hdfs dfs -get /user/hadoop/books/alice.txt
get: alice.txt: File exists

# 7. Visualizar el Contenido desde el HDFS
hadoop@master:~$ hdfs dfs -cat /user/hadoop/books/alice.txt

    *** START OF THE PROJECT GUTENBERG EBOOK ALICE'S ADVENTURES IN
    WONDERLAND ***
    [Illustration]

    Alice’s Adventures in Wonderland
    by Lewis Carroll

    THE MILLENNIUM FULCRUM EDITION 3.0

    Contents

    CHAPTER I.     Down the Rabbit-Hole
    CHAPTER II.    The Pool of Tears
    CHAPTER III.   A Caucus-Race and a Long Tale
    CHAPTER IV.    The Rabbit Sends in a Little Bill
    CHAPTER V.     Advice from a Caterpillar
    CHAPTER VI.    Pig and Pepper
    CHAPTER VII.   A Mad Tea-Party
    CHAPTER VIII.  The Queen’s Croquet-Ground
    CHAPTER IX.    The Mock Turtle’s Story
    CHAPTER X.     The Lobster Quadrille
    CHAPTER XI.    Who Stole the Tarts?
    CHAPTER XII.   Alice’s Evidence

# 8. Obtener Ayuda de los Comandos de HDFS
hadoop@master:~$ hdfs dfs -help
```

> **t) Run Yarn:** HDFS es un distributed storage system, pero no ejecuta ni planifica las tareas, esto lo realiza el YARN y para iniciarlo: start-yarn.sh (y para pararlo stop-yarn.sh). Se puede verificar con el comando jps y se verá que ahora hay un ResourceManager sobre master, y NodeManager sobre node1 & node2.

Esto iniciará el ResourceManager en el nodo maestro y el NodeManager en los nodos trabajadores. Puedes verificar que los servicios se hayan iniciado correctamente utilizando el comando jps.

```
hadoop@master:~$ start-yarn.sh
    Starting resourcemanager
    Starting nodemanagers

hadoop@master:~$ jps
    30498 ResourceManager
    29698 SecondaryNameNode
    30660 NodeManager
    29364 NameNode
    31020 Jps
    29500 DataNode

hadoop@nodo2:~$ jps
    5072 Jps
    4883 NodeManager
    3399 DataNode

hadoop@nodo1:~$ jps
    10977 DataNode
    12712 Jps
    12521 NodeManager
```

u) Para ver los nodos yarn node -list y las aplicaciones con yarn application -list (para más opciones/detalles yarn -help) También desde un navegador http://193.168.0.1:8088, donde 192.168.0.1 es la IP del nodo master.

Para ver los nodos y aplicaciones en YARN

```
hadoop@master:~$ yarn node -list
2024-05-16 18:16:06,819 INFO client.DefaultNoHARMFailoverProxyProvider:
Connecting to ResourceManager at /192.168.0.11:8032
Total Nodes:3
         Node-Id         Node-State Node-Http-Address   Number-of-Running-
Containers
nodo1.hadoop.local:42871            RUNNING nodo1.hadoop.local:8042
0
nodo2.hadoop.local:40215            RUNNING nodo2.hadoop.local:8042
0
master.hadoop.local:43885           RUNNING master.hadoop.local:8042
0


# Listar las aplicaciones en YARN:
hadoop@master:~$ yarn application -list
2024-05-16 18:16:45,852 INFO client.DefaultNoHARMFailoverProxyProvider:
```

```
Connecting to ResourceManager at /192.168.0.11:8032
Total number of applications (application-types: [], states: [SUBMITTED,
ACCEPTED, RUNNING] and tags: []):0
                Application-Id      Application-Name        Application-
Type          User       Queue               State         Final-State
Progress                        Tracking-URL

# También puedes acceder a la interfaz web de YARN ResourceManager en tu
navegador web:
http://192.168.0.11:8088
```

v) Enviar MapReduce Jobs a YARN

```
root@master:~# sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
root@master:~# sudo iptables-save

# regla conexiones puerto 22
root@master:~# sudo iptables -L -v -n
Chain INPUT (policy ACCEPT 142 packets, 22619 bytes)
 pkts bytes target     prot opt in     out     source
destination
   12   804 ACCEPT     tcp  --  *      *       0.0.0.0/0
0.0.0.0/0           tcp dpt:8088
17092 1154K ACCEPT     icmp --  *      *       0.0.0.0/0
0.0.0.0/0           icmptype 8
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0
0.0.0.0/0           tcp dpt:8088
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0
0.0.0.0/0           tcp dpt:22

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source
destination
 196K  383M ACCEPT     all  --  eth1   eth0    0.0.0.0/0
0.0.0.0/0           state RELATED,ESTABLISHED
 118K 6988K ACCEPT     all  --  eth0   eth1    0.0.0.0/0
0.0.0.0/0

Chain OUTPUT (policy ACCEPT 109 packets, 15881 bytes)
 pkts bytes target     prot opt in     out     source
destination
   12   804 ACCEPT     tcp  --  *      *       0.0.0.0/0
0.0.0.0/0           tcp dpt:8088
17086 1154K ACCEPT     icmp --  *      *       0.0.0.0/0
0.0.0.0/0           icmptype 0


root@master:~# systemctl restart ssh
```

```
# agrego
hadoop@master:~$ nano ~/.bashrc
    export HADOOP_HOME=/home/hadoop/hadoop
    export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin

# cargo variables
root@master:~# source ~/.bashrc

# verifico haddop
root@master:~# hadoop version
    Hadoop 3.3.5

# start
root@master:~# start-dfs.sh
    Starting namenodes on [master]
    master: namenode is running as process 29364.  Stop it first and
ensure /tmp/hadoop-hadoop-namenode.pid file is empty before retry.
    Starting datanodes
    localhost: datanode is running as process 29500.  Stop it first and
ensure /tmp/hadoop-hadoop-datanode.pid file is empty before retry.
    nodo1: datanode is running as process 10977.  Stop it first and ensure
/tmp/hadoop-hadoop-datanode.pid file is empty before retry.
    nodo2: datanode is running as process 3399.  Stop it first and ensure
/tmp/hadoop-hadoop-datanode.pid file is empty before retry.
    Starting secondary namenodes [master.hadoop.local]
    master.hadoop.local: secondarynamenode is running as process 29698.
Stop it first and ensure /tmp/hadoop-hadoop-secondarynamenode.pid file is
empty before retry.

# Start
root@master:~# start-yarn.sh
    Starting resourcemanager
    ERROR: Attempting to operate on yarn resourcemanager as root
    ERROR: but there is no YARN_RESOURCEMANAGER_USER defined. Aborting
operation.
    Starting nodemanagers
    ERROR: Attempting to operate on yarn nodemanager as root
    ERROR: but there is no YARN_NODEMANAGER_USER defined. Aborting
operation.
```

errores

```
root@master:~# rm -f /tmp/hadoop-hadoop-namenode.pid
root@master:~# rm -f /tmp/hadoop-hadoop-datanode.pid
root@master:~# rm -f /tmp/hadoop-hadoop-secondarynamenode.pid

root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 22
root@nodo1:~# rm -f /tmp/hadoop-hadoop-datanode.pid
root@nodo1:~# exit
    logout
    Connection to 192.168.0.14 closed.
```

```
root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.26 -p 22
root@nodo2:~# rm -f /tmp/hadoop-hadoop-datanode.pid
root@nodo2:~# exit
    logout
    Connection to 192.168.0.26 closed.
root@master:~# nano ~/.bashrc
root@master:~# source ~/.bashrc
root@master:~# start-dfs.sh
    Starting namenodes on [master]
    Starting datanodes
    Starting secondary namenodes [master.hadoop.local]
root@master:~# start-yarn.sh
    Starting resourcemanager
    resourcemanager is running as process 32209.  Stop it first and ensure
/tmp/hadoop-hadoop-resourcemanager.pid file is empty before retry.
    Starting nodemanagers
    localhost: nodemanager is running as process 32361.  Stop it first and
ensure /tmp/hadoop-hadoop-nodemanager.pid file is empty before retry.
    nodo1: nodemanager is running as process 12966.  Stop it first and
ensure /tmp/hadoop-hadoop-nodemanager.pid file is empty before retry.
    nodo2: nodemanager is running as process 5338.  Stop it first and
ensure /tmp/hadoop-hadoop-nodemanager.pid file is empty before retry.

root@master:~# jps
    32209 ResourceManager
    44065 DataNode
    44705 Jps
    32361 NodeManager
    44302 SecondaryNameNode
    43919 NameNode

root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.14 -p 22
root@nodo1:~# jps
    12966 NodeManager
    15015 Jps
    14745 DataNode
root@nodo1:~# exit
    logout
    Connection to 192.168.0.14 closed.

root@master:~# ssh -i /root/.ssh/id_rsa root@192.168.0.26 -p 22
root@nodo2:~# jps
    7652 Jps
    7366 DataNode
    5338 NodeManager
root@nodo2:~# exit
    logout
    Connection to 192.168.0.26 closed.

root@master:~# yarn node -list
    2024-05-17 10:46:47,186 INFO
client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
at /192.168.0.11:8032
```

```
    Total Nodes:3
            Node-Id      Node-State Node-Http-Address   Number-of-Running-
Containers
    nodo2.hadoop.local:34205            RUNNING nodo2.hadoop.local:8042
0
    nodo1.hadoop.local:33669            RUNNING nodo1.hadoop.local:8042
0
    master.hadoop.local:43897           RUNNING master.hadoop.local:8042
0

root@master:~# yarn application -list
    2024-05-17 10:46:53,374 INFO
client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
at /192.168.0.11:8032
    Total number of applications (application-types: [], states:
[SUBMITTED, ACCEPTED, RUNNING] and tags: []):0
                    Application-Id      Application-Name
Application-Type          User       Queue               State
Final-State        Progress                        Tracking-URL
```

```
root@master:~# sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT
root@master:~# sudo iptables -A INPUT -p tcp --dport 55000 -j ACCEPT
root@master:~# sudo iptables-save
```

```
hadoop@master:~$ nano $HADOOP_HOME/etc/hadoop/yarn-site.xml
    <configuration>
        <!-- Site specific YARN configuration properties -->
        <property>
            <name>yarn.acl.enable</name>
            <value>0</value>
        </property>
        <property>
            <name>yarn.resourcemanager.hostname</name>
            <value>192.168.0.11</value>
        </property>
        <property>
            <name>yarn.resourcemanager.address</name>
            <value>0.0.0.0:8032</value>
        </property>
        <property>
            <name>yarn.resourcemanager.scheduler.address</name>
            <value>0.0.0.0:8030</value>
        </property>
        <property>
            <name>yarn.resourcemanager.resource-tracker.address</name>
            <value>0.0.0.0:8031</value>
        </property>
        <property>
```

```
            <name>yarn.resourcemanager.admin.address</name>
            <value>0.0.0.0:8033</value>
        </property>
        <property>
            <name>yarn.resourcemanager.webapp.address</name>
            <value>0.0.0.0:8088</value>
        </property>
        <property>
            <name>yarn.nodemanager.aux-services</name>
            <value>mapreduce_shuffle</value>
        </property>
    </configuration>
```

```
hadoop@master:~$ yarn jar
/home/hadoop/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-
3.3.5.jar wordcount "books/*" output

2024-05-17 13:25:14,949 INFO mapreduce.Job:  map 100% reduce 100%
2024-05-17 13:25:14,968 INFO mapreduce.Job: Job job_1715950294901_0001
completed successfully
2024-05-17 13:25:15,440 INFO mapreduce.Job: Counters: 56
    File System Counters
        FILE: Number of bytes read=475214
        FILE: Number of bytes written=2054985
        FILE: Number of read operations=0
        FILE: Number of large read operations=0
        FILE: Number of write operations=0
        HDFS: Number of bytes read=1211125
        HDFS: Number of bytes written=271943
        HDFS: Number of read operations=14
        HDFS: Number of large read operations=0
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Killed map tasks=1
        Launched map tasks=3
        Launched reduce tasks=1
        Data-local map tasks=2
        Rack-local map tasks=1
        Total time spent by all maps in occupied slots (ms)=51830
        Total time spent by all reduces in occupied slots (ms)=5239
        Total time spent by all map tasks (ms)=51830
        Total time spent by all reduce tasks (ms)=5239
        Total vcore-milliseconds taken by all map tasks=51830
        Total vcore-milliseconds taken by all reduce tasks=5239
        Total megabyte-milliseconds taken by all map tasks=53073920
        Total megabyte-milliseconds taken by all reduce tasks=5364736
    Map-Reduce Framework
        Map input records=23429
        Map output records=212209
        Map output bytes=2030419
```

```
        Map output materialized bytes=475226
        Input split bytes=341
        Combine input records=212209
        Combine output records=32641
        Reduce input groups=24742
        Reduce shuffle bytes=475226
        Reduce input records=32641
        Reduce output records=24742
        Spilled Records=65282
        Shuffled Maps =3
        Failed Shuffles=0
        Merged Map outputs=3
        GC time elapsed (ms)=684
        CPU time spent (ms)=4100
        Physical memory (bytes) snapshot=931336192
        Virtual memory (bytes) snapshot=10590457856
        Total committed heap usage (bytes)=631255040
        Peak Map Physical memory (bytes)=273195008
        Peak Map Virtual memory (bytes)=2646167552
        Peak Reduce Physical memory (bytes)=148111360
        Peak Reduce Virtual memory (bytes)=2652889088
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=1210784
    File Output Format Counters
        Bytes Written=271943

hadoop@master:~$ hdfs dfs -ls output
    Found 2 items
    -rw-r--r--   2 hadoop supergroup          0 2024-05-17 13:25
output/_SUCCESS
    -rw-r--r--   2 hadoop supergroup     271943 2024-05-17 13:25
output/part-r-00000
```

Creamos un puente local ~ Nebula

```
➜  ~ ssh -L 8088:localhost:8088 -i ~/.ssh/id_rsa -p 55000
hadoop@84.88.58.69
```

```
http://localhost:8088/cluster
```

> **2.** Sobre el Cluster Hadoop y teniendo como referencia el programa de contar palabras, desarrollar el Map y el Reduce en Python como se indican en los apuntes Arquitecturas de software para el Big Data punto 2.3.1. Bajar el archivo de recetas en formato CSV desde

https://analisi.transparenciacatalunya.cat/Salut/Receptes-facturades-al-Servei-Catal-de-la-Salut/thrd-jj3r

desde la pestaña Exportar -> CSV. En base al material mencionado desarrollar el map y reduce (mapper-rec.py y reducerrec.py) y probarlos que funcionan externamente (con Python y sin Hadoop) y luego ejecutarlos en Hadoop. Analizar el rendimiento tanto con la ejecución con y sin Hadoop y extraer conclusiones.

```
root@master:~# wget -O rece.csv
https://analisi.transparenciacatalunya.cat/Salut/Receptes-facturades-al-
Servei-Catal-de-la-Salut/thrd-jj3r/export?format=csv
```

mapper-rec.py

```
#!/usr/bin/env python3

import sys

for line in sys.stdin:
    line = line.strip()  # quitar espacios en blanco al inicio y final de
la línea

    if line[0] == "a":  # quitar la cabecera que comienza por 'a'
        continue

    words = line.split(",")  # separar los campos por ','

    wordFinal = []  # array de campos finales (incluidos los separados por
',')
    union = ""  # string temporal para ir juntando las partes de los
campos
    for word in words:  # para todos los campos
        if word[0] == "\"":  # si el campo comienza por '"'
            union = word.replace(word[0], "")  # se agrega a unión y se
quita la '"' y continua
            continue
        if union != "" and word[len(word) - 1] != "\"":  # si ya se tienen
las partes y no se ha llegado al final
            union = union + word
        elif union != "" and word[len(word) - 1] == "\"":  # si ya se
tienen partes y se ha llegado al final
            word = word.replace('\"', "")
            wordFinal.append(union + word)
            union = ""  # se borra unión para el próximo
        else:
            wordFinal.append(word)  # en caso contrario se agrega
```

```
    wordsNoWhite = wordFinal[13].replace(" ", "_")  # cambia los espacios
en blanco por "_"
    print('%s\t%s' % (wordsNoWhite, 1))  # se generan las tuplas con el
campo 13.
```

reducer-rec.py

```
#!/usr/bin/env python3

from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

for line in sys.stdin:  # input comes from stdin
    line = line.strip()  # remove leading and trailing whitespace
    word, count = line.split('\t', 1)  # parse the input we got from
mapper.py
    try:
        count = int(count)
    except ValueError:
        continue

    if current_word == word:
        current_count += count
    else:
        if current_word:
            print('%s\t%s' % (current_word, current_count))
        current_count = count
        current_word = word

if current_word == word:
    print('%s\t%s' % (current_word, current_count))
```

```
# permisos de los archivos para que sean ejecutables
➜  git git:(main) ✗ chmod +x mapper-rec.py
➜  git git:(main) ✗ chmod +x reducer-rec.py

# pruebo localmente
➜  git git:(main) ✗ cat receptes.csv | ./mapper-rec.py | sort | ./reducer-
rec.py
    ACTH    198
    AGONISTES_OPIACIS   8720
    Acido_aminosalicilico_y_agentes_similares   23647
    Acido_ascorbico_(vitamina_C)_monofarmaco    1134
    Acido_folico_y_derivados    28610
```

```
    Acido_salicilico_y_derivados    10300
    Acidos_biliares_y_derivados 22180

Adrenergicos_en_combinacion_con_anticolinergicos_combinaciones_con_cortico
steroides_incl. 18289

Adrenergicos_en_combinacion_con_corticosteroides_u_otras_agentes_excluyend
o_los_anticolinergicos  33709
    Agentes_adrenergicos_y_dopaminergicos   25414
    Agentes_antialergicos_excluyendo_corticosteroides   19471
    Agentes_antiinflamatorios_no_esteroideos    21848

Agentes_antiinflamatorios_no_esteroideos_y_antiinfecciosos_en_combinacion
925
    ...
    ...
    ...
```

Ejecutar los Scripts en Hadoop

```
➜  git git:(main) ✗ scp -P 55000 receptes.csv
hadoop@84.88.58.69:/home/hadoop/
    receptes.csv                              100% 1197MB   3.0MB/s
06:35

➜  git git:(main) ✗ scp -P 55000 mapper-rec.py
hadoop@84.88.58.69:/home/hadoop/
    mapper-rec.py                             100% 1324    45.8KB/s
00:00

➜  git git:(main) ✗ scp -P 55000 reducer-rec.py
hadoop@84.88.58.69:/home/hadoop/
    reducer-rec.py                            100%  681    22.0KB/s
00:00
```

```
hadoop@master:~$ ls -l /home/hadoop/
    total 1917372
    -rw-rw-r--  1 hadoop hadoop     154638 Feb  4 09:09 alice.txt
    drwxrwxr-x  4 hadoop hadoop       4096 May 16 17:50 data
    -rw-rw-r--  1 hadoop hadoop     448642 Dec  2  2022 frankenstein.txt
    drwxr-xr-x 11 hadoop hadoop       4096 May 15 05:21 hadoop
    -rw-rw-r--  1 hadoop hadoop  706533213 Mar 15  2023 hadoop-
3.3.5.tar.gz
    -rw-rw-r--  1 hadoop hadoop     607504 Oct 10  2023 holmes.txt
    -rwxr-xr-x  1 hadoop hadoop       1324 May 17 18:16 mapper-rec.py
    -rw-r--r--  1 hadoop hadoop 1255600000 May 17 18:04 receptes.csv
    -rwxr-xr-x  1 hadoop hadoop        681 May 17 18:16 reducer-rec.py
```

```
hadoop@master:~$ hadoop fs -mkdir -p /user/hadoop/recetas
hadoop@master:~$ hadoop fs -put /home/hadoop/receptes.csv
/user/hadoop/recetas
hadoop@master:~$ hadoop fs -put /home/hadoop/mapper-rec.py
/user/hadoop/recetas
hadoop@master:~$ hadoop fs -put /home/hadoop/reducer-rec.py
/user/hadoop/recetas
hadoop@master:~$ hadoop fs -ls /user/hadoop/recetas
    Found 3 items
    -rw-r--r--   2 hadoop supergroup       1324 2024-05-17 18:27
/user/hadoop/recetas/mapper-rec.py
    -rw-r--r--   2 hadoop supergroup 1255600000 2024-05-17 18:27
/user/hadoop/recetas/receptes.csv
    -rw-r--r--   2 hadoop supergroup        681 2024-05-17 18:28
/user/hadoop/recetas/reducer-rec.py
```

Continuar con la Ejecución del Trabajo en Hadoop

```
hadoop jar /home/hadoop/hadoop/share/hadoop/tools/lib/hadoop-streaming-
3.3.5.jar \
-mapper /user/hadoop/recetas/mapper-rec.py \
-reducer /user/hadoop/recetas/reducer-rec.py \
-input /user/hadoop/recetas/receptes.csv \
-output /user/hadoop/recetas/output
```

PROBLEMAS

He tenido que incrementar la memoria de cada máquina virtual a 10Gb

```
hadoop@master:~$ start-dfs.sh
    Starting namenodes on [master]
    Starting datanodes
    Starting secondary namenodes [master.hadoop.local]

hadoop@master:~$ start-yarn.sh
    Starting resourcemanager
    Starting nodemanagers

hadoop@master:~$ yarn node -list
    2024-05-18 08:01:45,729 INFO
client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
at /0.0.0.0:8032
    Total Nodes:1
            Node-Id      Node-State Node-Http-Address   Number-of-Running-
Containers
    master.hadoop.local:34691           RUNNING master.hadoop.local:8042
0
```

```
hadoop@master:~$ yarn application -list
    2024-05-18 08:01:50,820 INFO
client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
at /0.0.0.0:8032
    Total number of applications (application-types: [], states:
[SUBMITTED, ACCEPTED, RUNNING] and tags: []):0
                  Application-Id      Application-Name
Application-Type          User       Queue               State
Final-State        Progress                        Tracking-URL
```

Tubería local ~ master

```
➜  ~ ssh -L 8088:localhost:8088 -i ~/.ssh/id_rsa -p 55000
hadoop@84.88.58.69
    Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-182-generic x86_64)
hadoop@master:~$
```

```
http://localhost:8088/cluster/
```

```
hadoop@master:~$ hadoop fs -ls /user/hadoop/recetas
    Found 4 items
    -rwxr-xr-x   2 hadoop supergroup       1324 2024-05-17 18:27
/user/hadoop/recetas/mapper-rec.py
    drwxr-xr-x   - hadoop supergroup          0 2024-05-18 09:17
/user/hadoop/recetas/output
    -rw-r--r--   2 hadoop supergroup 1255600000 2024-05-17 18:27
/user/hadoop/recetas/receptes.csv
    -rwxr-xr-x   2 hadoop supergroup        681 2024-05-17 18:28
/user/hadoop/recetas/reducer-rec.py
```

```
hadoop@master:~$ hadoop fs -chmod +x /user/hadoop/recetas/mapper-rec.py
hadoop@master:~$ hadoop fs -chmod +x /user/hadoop/recetas/reducer-rec.py

hadoop@master:~$ hadoop fs -copyToLocal /user/hadoop/recetas/mapper-rec.py
.
    copyToLocal: mapper-rec.py: File exists

hadoop@master:~$ hadoop fs -copyToLocal /user/hadoop/recetas/reducer-
rec.py .
    copyToLocal: reducer-rec.py: File exists
```

Instalando python master, nodo1, nodo2

```
root@master:~# sudo apt update
root@master:~# sudo apt install python3
root@master:~# python3 --version
    Python 3.8.10
```

```
root@nodo1:~# sudo apt update
root@nodo1:~# sudo apt install python3
root@nodo1:~# python3 --version
    Python 3.8.10
```

```
root@nodo2:~# sudo apt update
root@nodo2:~# sudo apt install python3
root@nodo2:~# python3 --version
    Python 3.8.10
```

```
hadoop@master:~$ hadoop fs -rm -r /user/hadoop/recetas/output
    Deleted /user/hadoop/recetas/output

hadoop@master:~$ yarn jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-
streaming-3.3.5.jar \
> -input /user/hadoop/recetas/receptes.csv \
> -output /user/hadoop/recetas/output \
> -mapper /home/hadoop/mapper-rec.py \
> -reducer /home/hadoop/reducer-rec.py

    packageJobJar: [/tmp/hadoop-unjar16061167299127096850/] []
/tmp/streamjob5434767623560579355.jar tmpDir=null
    2024-05-18 10:23:59,908 INFO
client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
at /0.0.0.0:8032
    2024-05-18 10:24:00,180 INFO
client.DefaultNoHARMFailoverProxyProvider: Connecting to ResourceManager
at /0.0.0.0:8032
    2024-05-18 10:24:00,518 INFO mapreduce.JobResourceUploader: Disabling
Erasure Coding for path: /tmp/hadoop-
yarn/staging/hadoop/.staging/job_1716019302363_0008
    2024-05-18 10:24:01,935 INFO impl.YarnClientImpl: Submitted
application application_1716019302363_0008
    2024-05-18 10:24:01,968 INFO mapreduce.Job: The url to track the job:
http://master.hadoop.local:8088/proxy/application_1716019302363_0008/
    2024-05-18 10:24:01,970 INFO mapreduce.Job: Running job:
job_1716019302363_0008
    2024-05-18 10:24:12,281 INFO mapreduce.Job: Job job_1716019302363_0008
running in uber mode : false
    2024-05-18 10:24:12,285 INFO mapreduce.Job:  map 0% reduce 0%
    ...
```

```
    ...
    2024-05-18 10:26:51,946 INFO mapreduce.Job:  map 100% reduce 47%
    2024-05-18 10:26:57,992 INFO mapreduce.Job:  map 100% reduce 78%
    2024-05-18 10:27:04,023 INFO mapreduce.Job:  map 100% reduce 97%
    2024-05-18 10:27:05,035 INFO mapreduce.Job:  map 100% reduce 100%
    2024-05-18 10:27:06,056 INFO mapreduce.Job: Job job_1716019302363_0008
completed successfully
    2024-05-18 10:27:06,248 INFO mapreduce.Job: Counters: 55
    2024-05-18 10:27:06,257 INFO streaming.StreamJob: Output directory:
/user/hadoop/recetas/output

hadoop@master:~$ hadoop fs -ls /user/hadoop/recetas/output
    Found 2 items
    -rw-r--r--   2 hadoop supergroup          0 2024-05-18 10:27
/user/hadoop/recetas/output/_SUCCESS
    -rw-r--r--   2 hadoop supergroup      17639 2024-05-18 10:27
/user/hadoop/recetas/output/part-00000
```

Comparativa de Rendimiento: Ejecución Local vs Hadoop

```
→  git git:(main) ✗ time (cat receptes.csv | python3 mapper-rec.py | sort
| python3 reducer-rec.py > reducer_output.txt)

( cat receptes.csv | python3 mapper-rec.py | sort | python3 reducer-rec.py
> )  69.69s user 4.18s system 111% cpu 1:06.01 total
```

```
hadoop@master:~$ time (yarn jar
$HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-3.3.5.jar \
> -input /user/hadoop/recetas/receptes.csv \
> -output /user/hadoop/recetas/output \
> -mapper /home/hadoop/mapper-rec.py \
> -reducer /home/hadoop/reducer-rec.py)

packageJobJar: [/tmp/hadoop-unjar1803497871655201916 4/] []
/tmp/streamjob14178942302182188566.jar tmpDir=null
2024-05-18 10:38:37,756 INFO mapreduce.Job: Running job:
job_1716019302363_0009
2024-05-18 10:38:48,109 INFO mapreduce.Job: Job job_1716019302363_0009
running in uber mode : false
2024-05-18 10:38:48,113 INFO mapreduce.Job:  map 0% reduce 0%
...
...
2024-05-18 10:41:44,744 INFO mapreduce.Job:  map 100% reduce 98%
2024-05-18 10:41:45,754 INFO mapreduce.Job:  map 100% reduce 100%
2024-05-18 10:41:45,767 INFO mapreduce.Job: Job job_1716019302363_0009
completed successfully
2024-05-18 10:41:45,994 INFO mapreduce.Job: Counters: 55
    File System Counters
        FILE: Number of bytes read=238201145
```

```
        FILE: Number of bytes written=479465569
        HDFS: Number of write operations=2
        HDFS: Number of bytes read erasure-coded=0
    Job Counters
        Killed map tasks=3
        Launched map tasks=13
        Total megabyte-milliseconds taken by all map tasks=870288384
        Total megabyte-milliseconds taken by all reduce tasks=89778176
    Map-Reduce Framework
        Map input records=6160558
        Map output records=6160557
        Map output bytes=225862484
        Map output materialized bytes=238201199
        Input split bytes=1030
        Combine input records=0
        Combine output records=0
        Reduce input groups=443
        Reduce shuffle bytes=238201199
        Reduce input records=6160557
        Reduce output records=443
        Spilled Records=12321114
        Shuffled Maps =10
        Failed Shuffles=0
        Merged Map outputs=10
        GC time elapsed (ms)=7108
        CPU time spent (ms)=93140
        Physical memory (bytes) snapshot=3103965184
        Virtual memory (bytes) snapshot=29151420416
        Total committed heap usage (bytes)=2274152448
        Peak Map Physical memory (bytes)=284626944
        Peak Map Virtual memory (bytes)=2670575616
        Peak Reduce Physical memory (bytes)=478236672
        Peak Reduce Virtual memory (bytes)=2669363200
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=1255636864
    File Output Format Counters
        Bytes Written=17639
2024-05-18 10:41:46,003 INFO streaming.StreamJob: Output directory:
/user/hadoop/recetas/output

real    3m13.954s
user    0m5.767s
sys 0m0.440s
```

Análisis de Rendimiento

Ejecución Local: Real: 1 minuto 6 segundos **Ejecución en Hadoop:** Real: 3 minutos 13 segundos

Razones del Mayor Tiempo en Hadoop

1. **Overhead de Configuración y Comunicación:** Hadoop tiene un overhead significativo relacionado con la configuración del trabajo, la comunicación entre nodos y la gestión de recursos.
2. **Distribución de Datos:** El tiempo que tarda Hadoop en dividir, distribuir y leer los datos desde HDFS puede ser considerable, especialmente para conjuntos de datos más pequeños.
3. **Latencia de Red:** La comunicación entre los nodos puede añadir latencia adicional, lo cual no ocurre en una ejecución local donde todo sucede en la misma máquina.
4. **Tiempo de Arranque de Tareas:** Iniciar y finalizar tareas en un entorno distribuido tiene un costo en términos de tiempo. En un entorno local, el tiempo de arranque de los procesos es mucho menor.
5. **Manejo de Fases Intermedias:** Hadoop maneja fases intermedias de Shuffle y Sort, lo cual es crucial para grandes volúmenes de datos, pero puede ser innecesariamente costoso para conjuntos de datos más pequeños.

Conclusiones

En **Conjuntos de Datos Pequeños** la ejecución local puede ser más rápida y eficiente. Sin embargo en **Grandes Volúmenes de Datos** Hadoop es más adecuado debido a su capacidad de escalar horizontalmente. El tiempo adicional en Hadoop se debe principalmente a la sobrecarga inicial y a la gestión de recursos en un entorno distribuido.

Optimización en Hadoop

- **Configuración del Número de Splits:** Ajustar el número de splits para que los datos se procesen de manera más eficiente.
- **Tamaño del Bloque de HDFS:** Ajustar el tamaño del bloque de HDFS para optimizar el rendimiento.
- **Uso de Combiner** : Implementar un combiner para reducir la cantidad de datos que necesitan ser transferidos entre el mapper y el reducer.

En definitiva creo que el uso de Hadoop está justificado principalmente cuando se manejan grandes volúmenes de datos distribuidos. Para tareas más pequeñas o pruebas iniciales, la ejecución local es más rápida y sencilla. A medida que los datos y la complejidad crecen, Hadoop ofrece las herramientas necesarias para manejar y procesar datos a gran escala de manera eficiente.