Vagrant (https://www.vagrantup.com/)

Vagrant è uno strumento per creare e gestire macchine virtuali.

Per funzionare, Vagrant ha bisogno che sia installato uno dei software di virtualizzazione supportati. Nel seguito assumeremo che sia usato Virtual Box (https://www.virtualbox.org/)

Sul sito di Vagrant c'è una documentazione per iniziare ad usare questo strumento: https://www.vagrantup.com/intro/getting-started/index.html

Un progetto Vagrant consiste di una cartella con all'interno un Vagrantfile – un file di testo per la configurazione del progetto.

Per iniziare un progetto, occorre prima di tutto creare la cartella (es. "hadoop-master"):

```
mkdir hadoop-master
```

Quindi si entra nella cartella

```
cd hadoop-master
```

e si inizializza il progetto (nell'esempio sottostante si è scelto di creare una macchina Ubuntu Xenial a 64 bit)

```
vagrant init ubuntu/xenial64
```

Questo comando crea nella cartella corrente (del progetto) un Vagrantfile. Qui sotto c'è un esempio di Vagrantfile che è stato modificato per specificare l'hostname (hadoop-master), per assegnare un indirizzo IP privato alla macchina virtuale (192.168.33.10) e per configurare la RAM (4GB):

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# All Vagrant configuration is done below. The "2" in Vagrant.configure
# configures the configuration version (we support older styles for
# backwards compatibility). Please don't change it unless you know what
# you're doing.
Vagrant.configure("2") do |config|
  # The most common configuration options are documented and commented below.
  # For a complete reference, please see the online documentation at
  # https://docs.vagrantup.com.

  # Every Vagrant development environment requires a box. You can search for
  # boxes at https://vagrantcloud.com/search.
  config.vm.box = "ubuntu/xenial64"
  config.vm.hostname = "hadoop-master"
```

```
  # Create a private network, which allows host-only access to the machine
  # using a specific IP.
  config.vm.network "private_network", ip: "192.168.33.10"

  # Provider-specific configuration so you can fine-tune various
  # backing providers for Vagrant. These expose provider-specific options.
  # Example for VirtualBox:
  #
  config.vm.provider "virtualbox" do |vb|
  #   # Display the VirtualBox GUI when booting the machine
  #   vb.gui = true
  #
  #   # Customize the amount of memory on the VM:
    vb.memory = "4096"
  end
end
```

A questo punto è possibile avviare la macchina virtuale:

```
vagrant up
```

Dopo un po' di tempo (es. qualche minuto), la macchina virtuale sarà pronta e sarà possibile connettersi ad essa tramite ssh

```
vagrant ssh
```

Per chiudere la sessione ssh, è sufficiente eseguire il comando `exit` durante la sessione remota

```
exit
```

Per spegnere la macchina virtuale

```
vagrant halt
```

Per distruggere una macchina virtuale (es. cancellare i dischi virtuali)

```
vagrant destroy
```

Apache Hadoop (http://hadoop.apache.org/)

Occorre scaricare la versione 2.9.2 (https://www.apache.org/dyn/closer.cgi/hadoop/common/hadoop-2.9.2/hadoop-2.9.2.tar.gz)

Per precauzione, verificare il checksum (https://dist.apache.org/repos/dist/release/hadoop/common/hadoop-2.9.2/hadoop-2.9.2.tar.gz.mds) o la firma digitale

(https://dist.apache.org/repos/dist/release/hadoop/common/hadoop-2.9.2/hadoop-2.9.2.tar.gz.asc), seguendo le istruzioni sulla pagina di download (https://hadoop.apache.org/releases.html)

Nel seguito supporremo di usare una macchina virtuale Vagrant e che la distribuzione Hadoop sia stata copiata all'interno della cartella del progetto.

Ci si connetta alla macchina virtuale:

```
vagrant ssh
```

Installare Java 8:

```
sudo apt-get install openjdk-8-jdk-headless
```

spostarsi nella cartella `/vagrant` (corrispondente alla cartella del progetto al di fuori della macchina virtuale):

```
cd /vagrant
```

Scompattare la distribuzione di Hadoop

```
tar xzf hadoop-2.9.2.tar.gz
```

Potrebbero comparire i seguenti messaggi di errore:

> tar: hadoop-2.9.2/lib/native/libhdfs.so: Cannot create symlink to 'libhdfs.so.0.0.0': Protocol error
>
> tar: hadoop-2.9.2/lib/native/libhadoop.so: Cannot create symlink to 'libhadoop.so.1.0.0': Protocol error

Definire le variabili di ambiente `JAVA_HOME` e `HADOOP_HOME` all'interno del file `.profile`:

```
nano /home/vagrant/.profile
```

All'interno del file aggiungere le seguenti righe:

```
export HADOOP_HOME=/vagrant/hadoop-2.9.2
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/
```

Un'altra cosa da modificare è il file `/etc/hosts` (mi raccomando, solo nel caso delle machine virtuali, non fatelo sul vostro computer!), in modo che l'hostname della macchina e quello di altre macchine nel cluster sia associato all'indirizzo IP statico (le colonne sono separate da TAB)

```
192.168.33.10   hadoop-master   hadoop-master
```

Per rendere effettive queste modifiche, occorre uscire dalla macchina virtuale e rientrarvi.

Durante la nuova sessione, eseguire questo comando per verificare che la variabile di ambiente `HADOOP_HOME` sia definita:

```
echo $HADOOP_HOME
```

Ricordare che tutte le modifiche ai file di configurazione vanno riportate su tutte le macchine appartenenti al cluster.

Come prima cosa vogliamo configurare e far partire HDFS. Dobbiamo modificare il file `$HADOOP_HOME/etc/hadoop/core-site.xml` come segue:

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://192.168.33.10:9000</value>
    </property>
</configuration>
```

Nel caso si voglia lavorare con una sola macchina, è possibile usare `localhost` al posto dell'indirizzo IP (o dell'hostname della macchina).

Nel file `$HADOOP_HOME/etc/hadoop/core-site.xml` si può indicare, per esempio, il numero di repliche predefinito. Qui ho indicato soltanto 2 (invece delle classiche 3):

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>2</value>
    </property>
</configuration>
```

Sul nodo dove sarà in esecuzione il NameNode, è necessario formattare il filesystem:

```
$HADOOP_HOME/bin/hdfs namenode -format MyCluster
```

A questo punto possiamo avviare il NameNode:

```
$HADOOP_HOME/bin/hadoop namenode
```

E su ciascuna macchine del cluster il DataNode:

```
$HADOOP_HOME/bin/hadoop namenode
```

I due script di cui sopra sono bloccanti: sarà quindi necessario aprire altre sessioni SSH.

Per verificare il funzionamento del filesystem, collegarsi all'interfaccia web:

http://localhost:50070

Il comando `hadoop` possiede una shell dedicata al filesystem (http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-common/FileSystemShell.html):

- Creare la cartella `input`:
  `$HADOOP_HOME/bin/hadoop fs -mkdir -p input`
  Da notare che il path di destinazione è relativo, perciò verrà creata la cartalla `/user/<nome utente>/input`
- Caricare il file `input.nq` dentro la cartella `input`
  `$HADOOP_HOME/bin/hadoop fs -copyFromLocal input.nq input`

- In caso di problemi con i permessi, è possibile disabilitarli (nel file `hdfs-site.xml`)
  `dfs.permissions.enabled = true`

Come seconda cosa vogliamo configurare e far partire YARN.

Attenzione: si sconsiglia di configurare YARN al di fuori della macchina virtuale, perché è configurato per accettare richieste di job provenienti dall'esterno. Dovrebbe essere possibile restringerlo alle richieste locali agendo sulle proprietà di configurazione che determinano gli hostname dei diversi processi

Link interessante: https://wiki.apache.org/hadoop/ConnectionRefused (da cui si è preso spunto per modificare il file `/etc/hosts`)

Dobbiamo modificare o creare il file `$HADOOP_HOME/etc/hadoop/yarn-site.xml`

```
<configuration>
      <property>
            <name>yarn.resourcemanager.hostname</name>
            <value>192.168.33.10</value>
      </property>
     <property>
         <name>yarn.nodemanager.aux-services</name>
         <value>mapreduce_shuffle</value>
     </property>
      <property>
            <name>yarn.nodemanager.resource.cpu-vcores</name>
            <value>2</value>
      </property>
      <property>
            <name>yarn.nodemanager.resource.memory-mb</name>
            <value>4096</value>
      </property>
      <property>
            <name>yarn.scheduler.minimum-allocation-mb</name>
            <value>1024</value>
      </property>
      <property>
             <name>yarn.nodemanager.vmem-pmem-ratio</name>
             <value>2.1</value>
      </property>

</configuration>
```

Dobbiamo modificare o creare il file `$HADOOP_HOME/etc/hadoop/mapred-site.xml`

```
<configuration>

    <property>

        <name>mapreduce.framework.name</name>

        <value>yarn</value>

    </property>
```

```
        <property>
              <name>mapreduce.map.memory.mb</name>
              <value>2048</value>
        </property>
        <property>
              <name>mapreduce.reduce.memory.mb</name>
              <value>2048</value>
        </property>
        <property>
              <name>mapreduce.map.java.opts</name>
              <value>-Xmx1900m</value>
        </property>
        <property>
              <name>mapreduce.reduce.java.opts</name>
              <value>-Xmx1900m</value>
        </property>
    </configuration>
```

I due file di configurazione di sopra sono basati sulle caratteristiche della macchina virtuale appena creata. In caso la macchina abbia caratteristiche diverse, occorre aggiustare i parametri di configurazione. Questo post spiega come configurare YARN: https://it.hortonworks.com/blog/how-to-plan-and-configure-yarn-in-hdp-2-0/

A questo punto possiamo avviare i due processi principali:

- ResourceManager
  `$HADOOP_HOME/sbin/yarn-daemon.sh start resourcemanager`
- NodeManager (su ciascuna macchina esecutrice)
  `$HADOOP_HOME/sbin/yarn-daemon.sh start nodemanager`

Accedere alla user interface web del resource manager:

http://192.168.33.10:8088/

Per lanciare un job mapreduce su YARN ed HDFS, si può usare un comando tipo questo:

```
%HADOOP_HOME%/bin/hadoop jar ia2-2018_2019-mapreduce-examples-0.0.1-
SNAPSHOT.jar
it.uniroma2.art.teaching.ia2_2018_2019.mapreduce.examples.ResourceCounter
hdfs:// 192.168.33.10:9000/user/vagrant/input hdfs://
192.168.33.10:9000/user/vagrant/output
```

Dove bisogna prestare attenzione a mettere i pathname e gli indirizzi IP (o hostname) corretti.