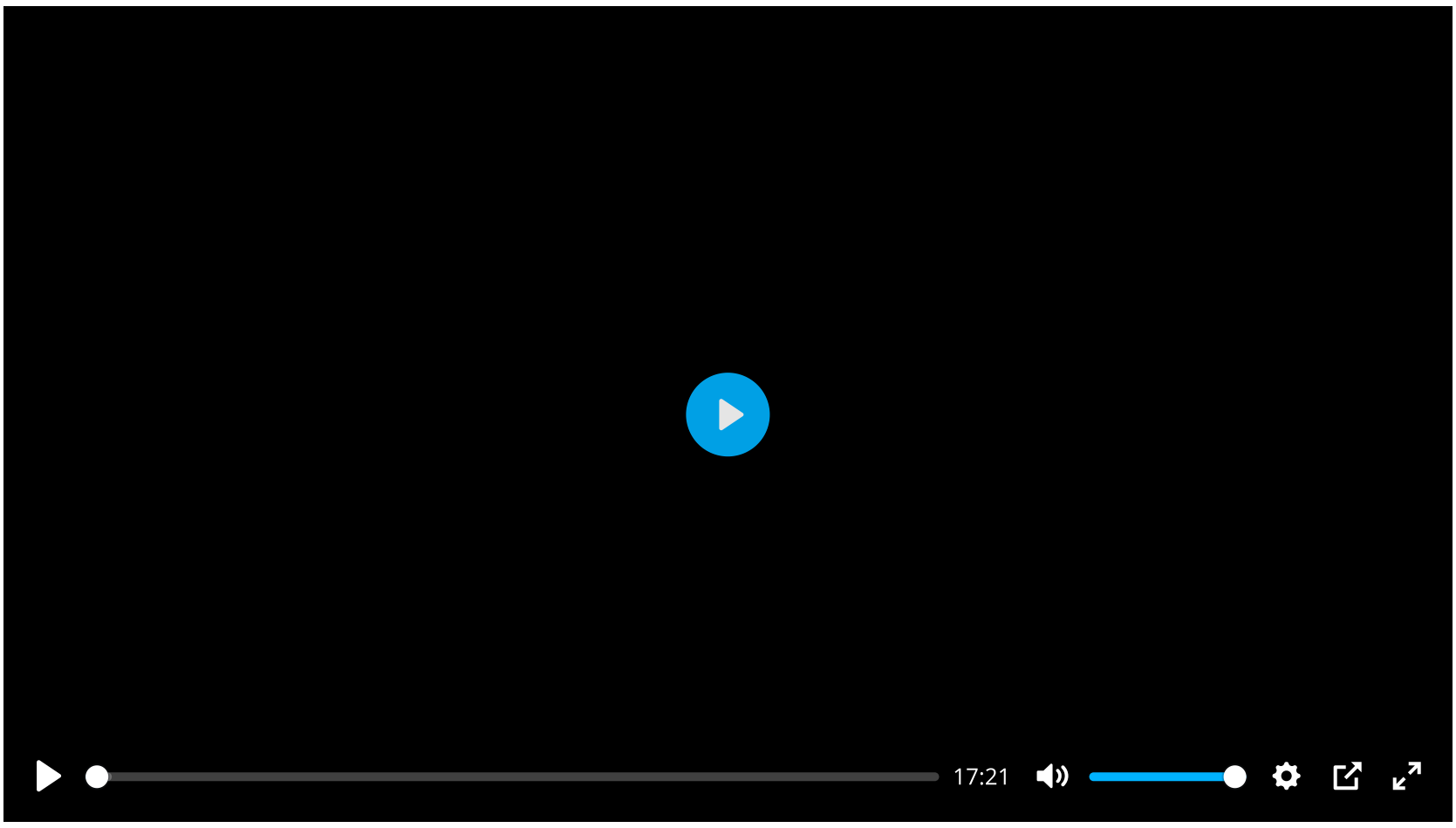
3.3 Технология Map-Reduce



Текстовая расшифровка видео:

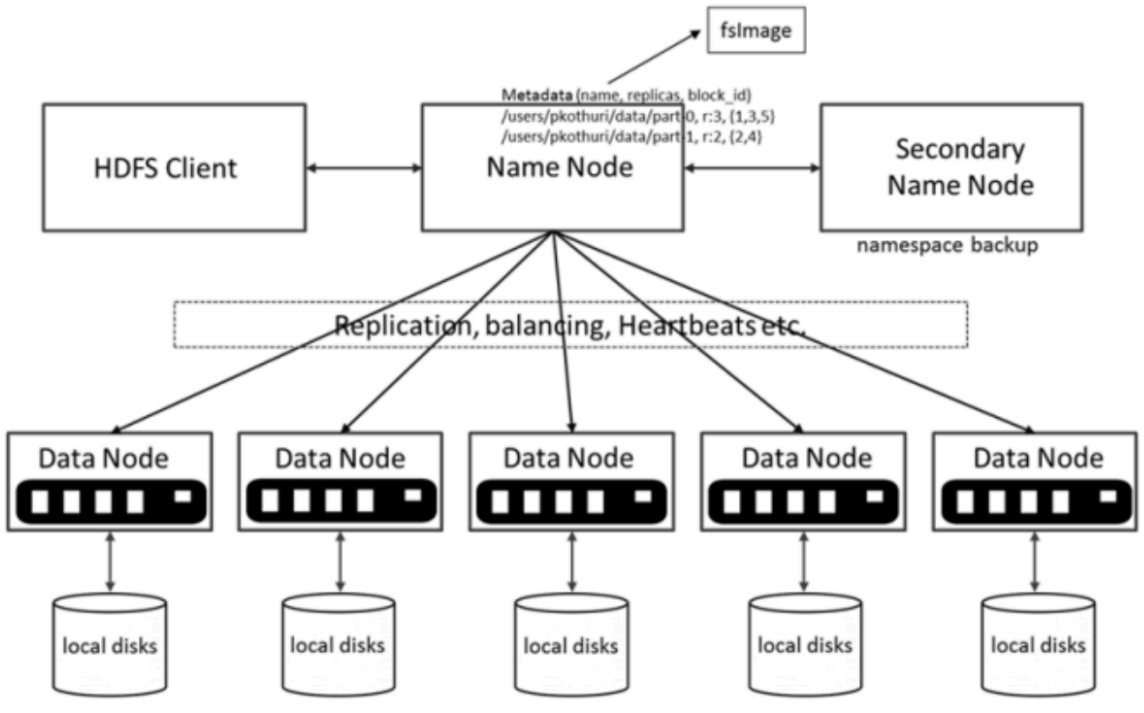
ПОДНИМАЕМ HDFS У СЕБЯ

План:

- NameNode, SNameNode, DataNode;
- КупиСкачай слона;
- Переменные и конфиги;
- Запускаем HDFS;

NameNode, SNameNode, DataNode

Возможно, вы помните эту схему из предыдущих уроков по HDFS:



Вспомним:

- У нас есть несколько компонентов;

- У нас есть **NameNode**, который хранит метаданные и **Secondary NameNode**, который дополнительно их кэширует и хранит копию;
- У нас есть **DataNode**, где данные нарезаются на кусочки (они копируются несколько раз, отправляются на DataNode);
- У нас есть Data Locality – главная фишка. Зная по метаданным, где и какие кусочки лежат, мы можем запускать код там же, где лежат физические данные.

~~Купи~~Скачай слона

Для того, чтобы скачать Apache Hadoop нужно перейти на официальный сайт и скачать дистрибутив, который уже будет включать в себя HDFS, фреймворк MapReduce, шедулер YARN и т.д.

Его можно развернуть на одной машине, однако нужна **Java** версий **8** и **11**.

В операционных системах есть дополнительные инструменты для того, чтобы переключаться между актуальными версиями Java.

Также можно указать путь в переменных окружениях до необходимой вам версии Java:

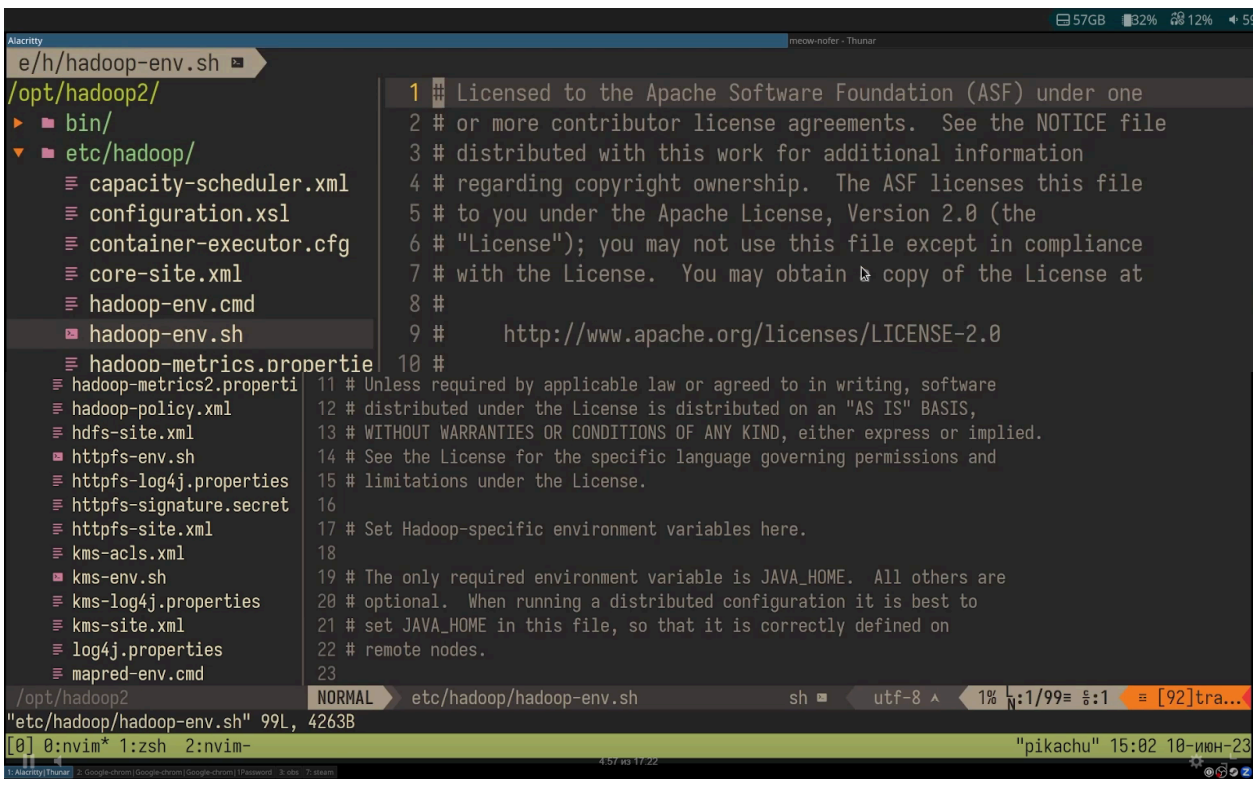
```
# добавить в etc/hadoop/hadoop-env.sh
export JAVA_HOME=/usr/lib/jvm/default
```

У вас путь может быть другой

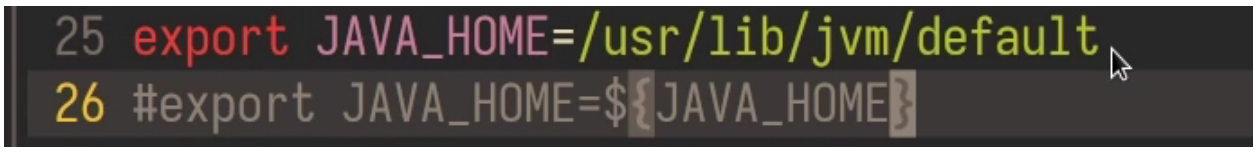
Нюансы:

- должен работать SSH на localhost без пароля;
- есть два варианта переменных окружений – переменное окружение на вашей машине и переменное окружение, задающееся глобально для кластера в целом.

Рассмотрим пример:



Мы разместили его в /opt/hadoop2. Пойдем в /etc/hadoop и пропишем версию Java. По умолчанию она берется из переменного окружения, мы заменяем на собственный путь к развернутой jvm. Это базовый конфигурационный файл для всего Hadoop:



Важно: на вашей машине jvm может находиться в другом месте! Если будете разворачивать на своей машине, то для начала найдите, где стоит Java и потом уже используйте данный путь.

Переменные и конфиги

Чтобы запустить HDFS нужно настроить следующее:

Endpoint для API (подключение к HDFS). По умолчанию это Port 9000, но мы укажем, что Hadoop будет запускаться на той же машине:

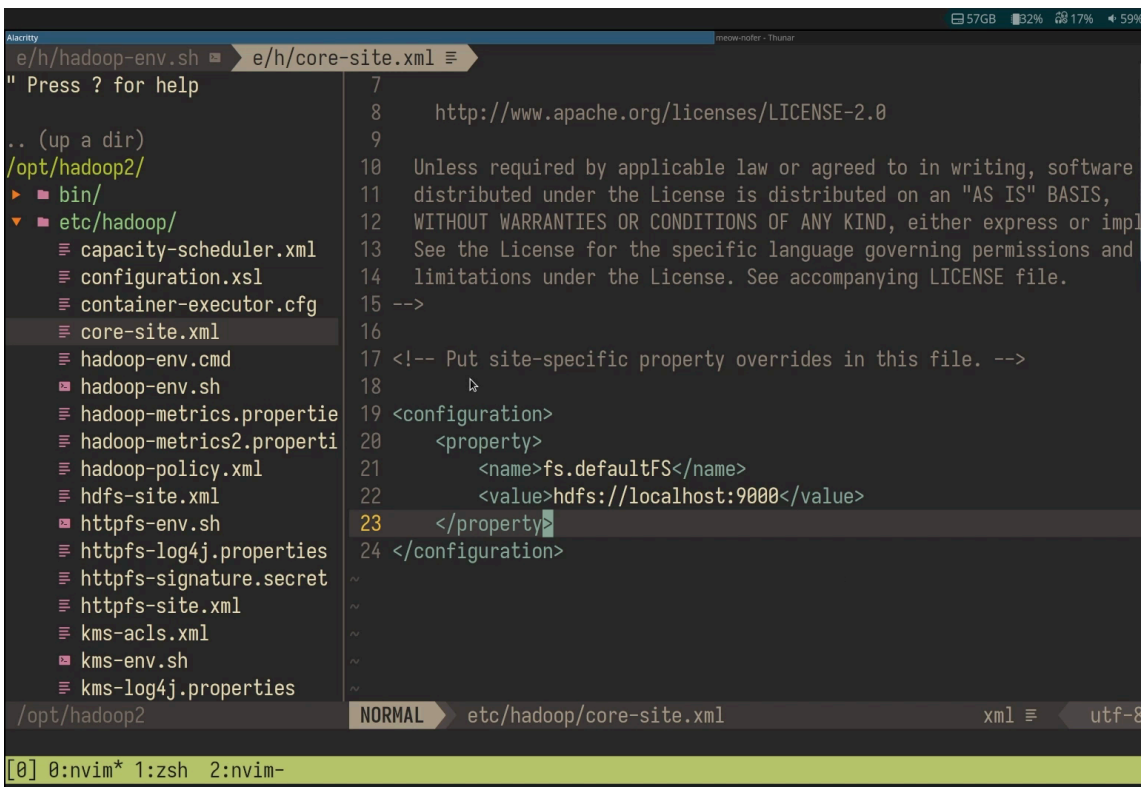
etc/hadoop/core-site.xml

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

etc/hadoop/hdfs-site.xml

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
</configuration>
```

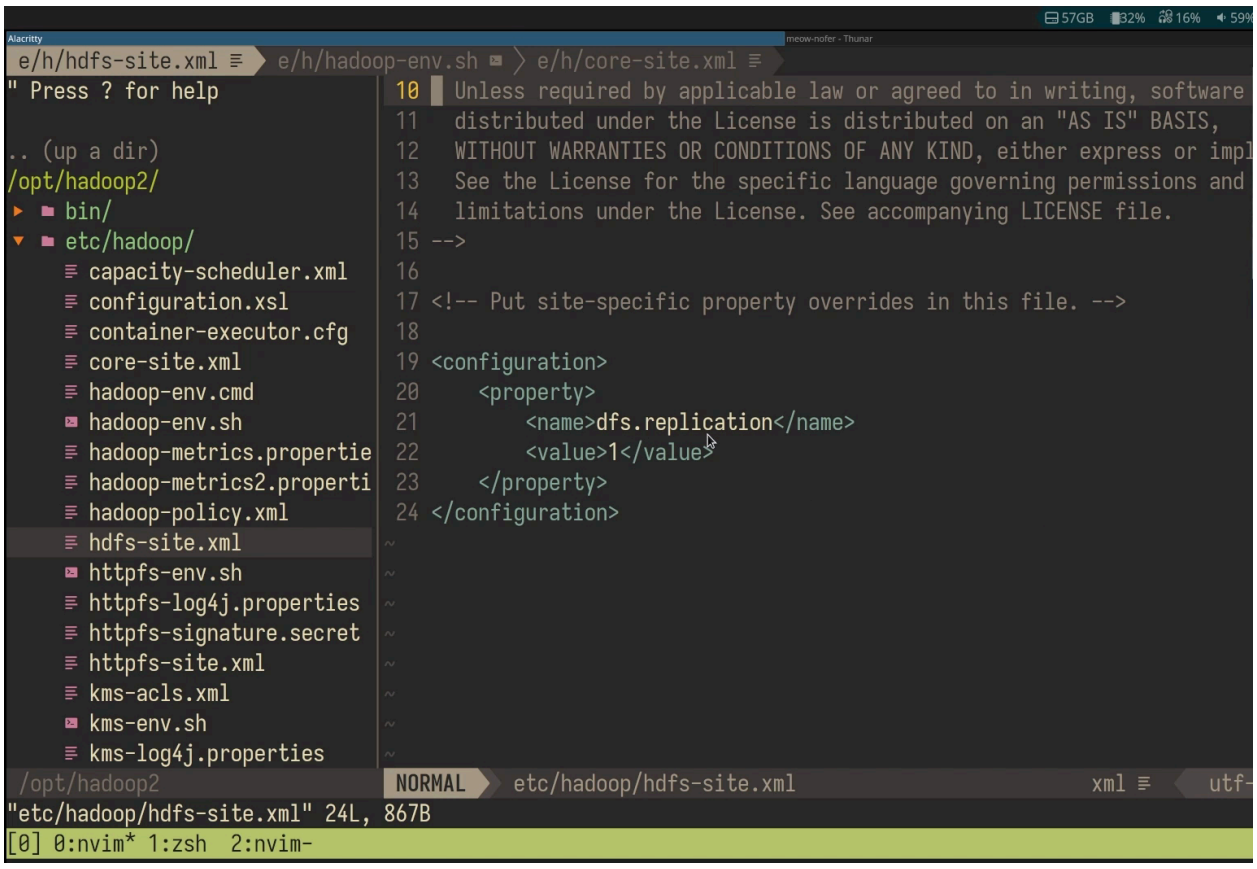
Пропишем это в файле «core-site.xml». По умолчанию в скаченном дистрибутиве configuration пустой. Мы добавляем «property»:



Таким образом мы укажем, что HDFS – локальный.

Нюанс: в HDFS работает репликация и кусочки файлов в кластере раскидываются на разные машины. В данном примере мы разворачиваем все на одной машине, поэтому репликацию необходимо будет выключить.

Пропишем, что replication factor – 1, а копий не будет:



Далее, задаем пачку переменных окружений (возможно, у вас все сработает без этого):

```
export JAVA_HOME=/usr/lib/jvm/default
export HADOOP_HOME=/opt/hadoop2
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export HADOOP_OPTS="$HADOOP_OPTS -Djava.library.path=$HADOOP_HOME/lib/native"
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```

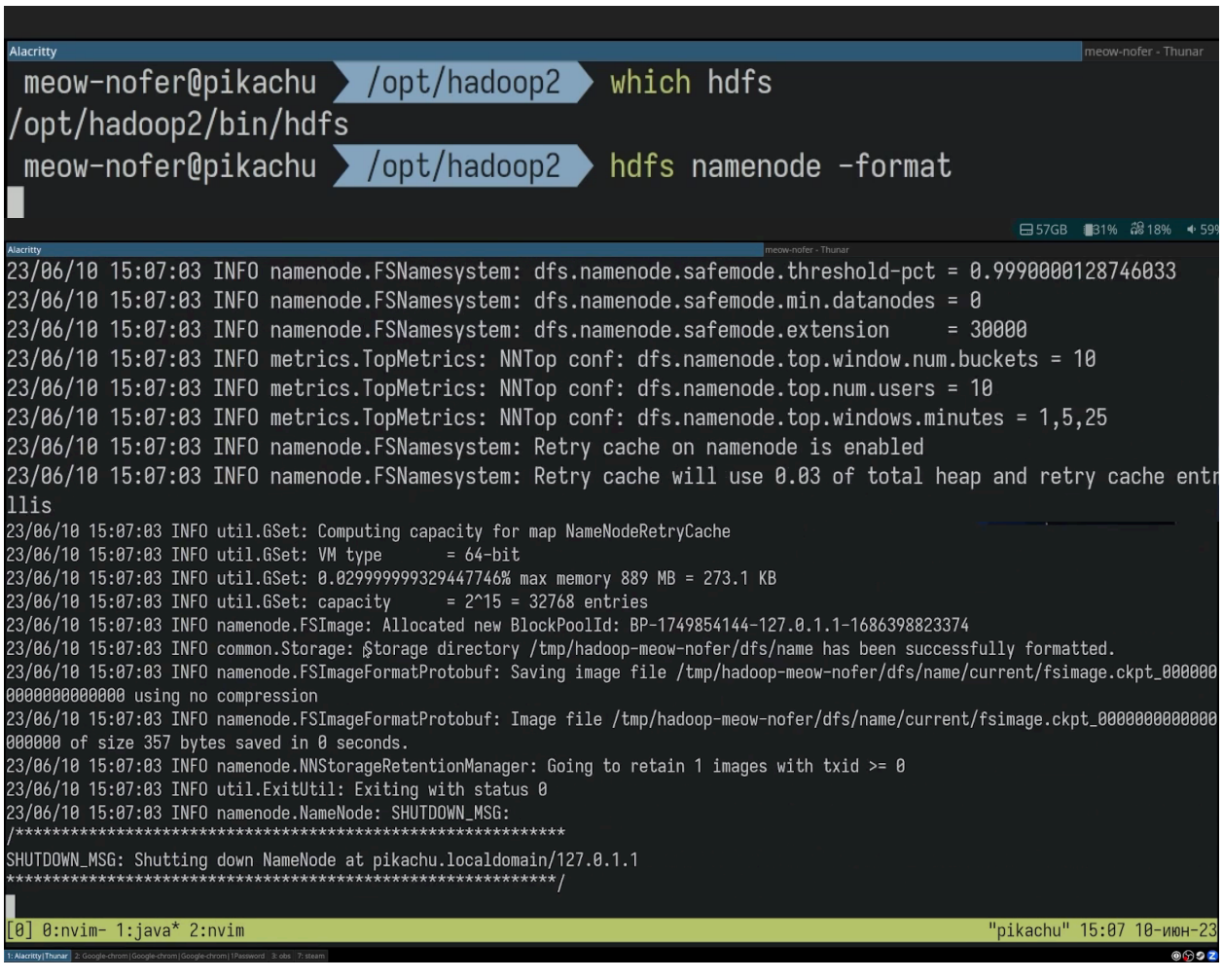
Это разные настройки для MapReduce, дополнительных фреймворков и т.д.

Это один из исчерпывающих наборов.

Запускаем HDFS

Для запуска HDFS необходимо запустить бинарник и провести форматирование. Просим HDFS создать все необходимые каталоги и метаданные на локальной машине.

HDFS-бинарник автоматически указывает на HDFS в Hadoop2. У нас появились необходимые элементы в локальном каталоге (в данном примере – временный каталог «tmp»):

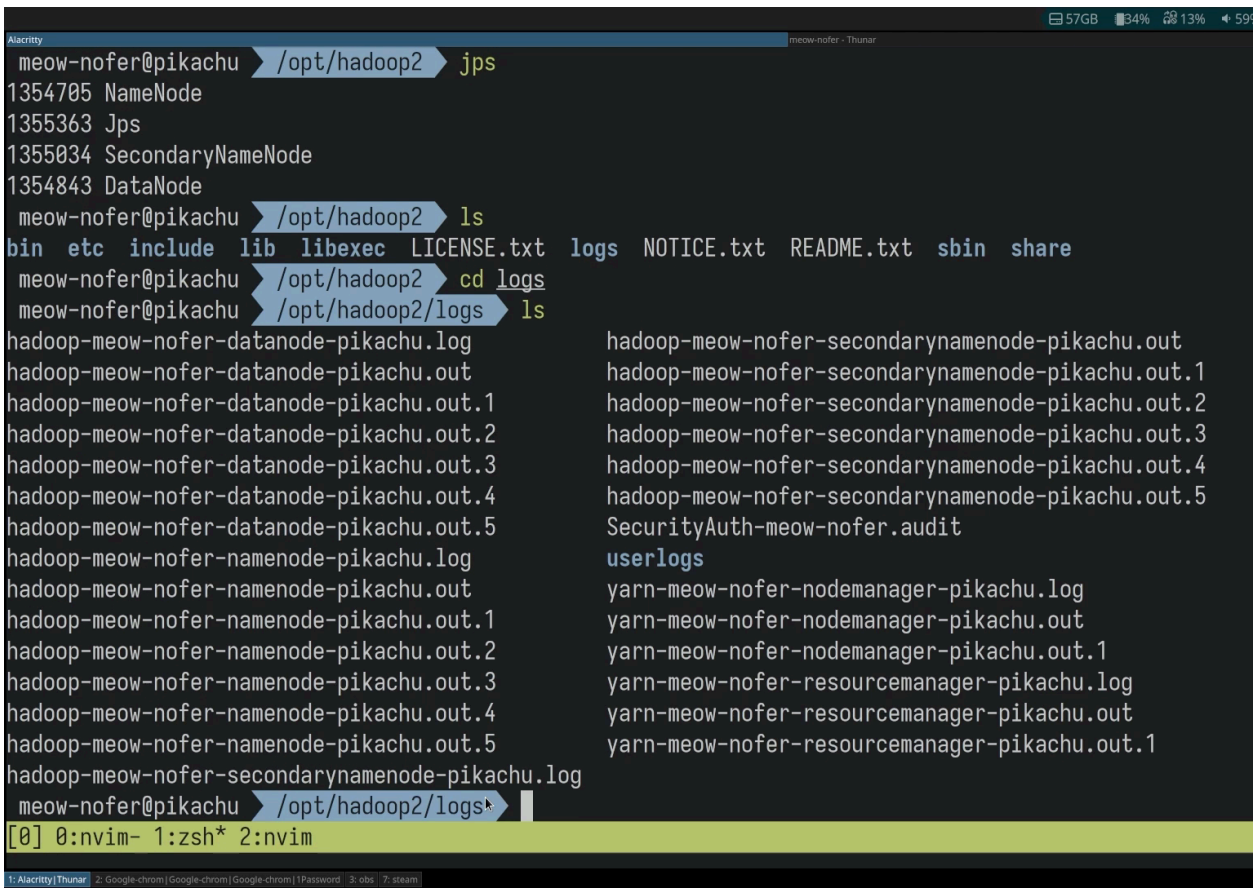


Далее запускаем и пишем: **/sbin/start-dfs.sh**

Чтобы проверить был запуск или нет, мы можем использовать утилиту jps. Так, мы видим, что запустилось следующее:

- NameNode;
- Secondary NameNode;
- DataNode:

Если у вас что-то не запускается, можно зайти в локальный каталог в папку «logs». В ней есть логи по каждому отдельному компоненту и тому, как каждый запускался:



Также мы можем открыть web-интерфейс и просмотреть всю нужную нам информацию: http://localhost:50070/

Вопрос: как перекидывать файлы?

Ответ: для этого используются те же команды, что и с файловой системой Linux. Однако вспомним, что HDFS работает только через API, поэтому мы используем следующий синтаксис:

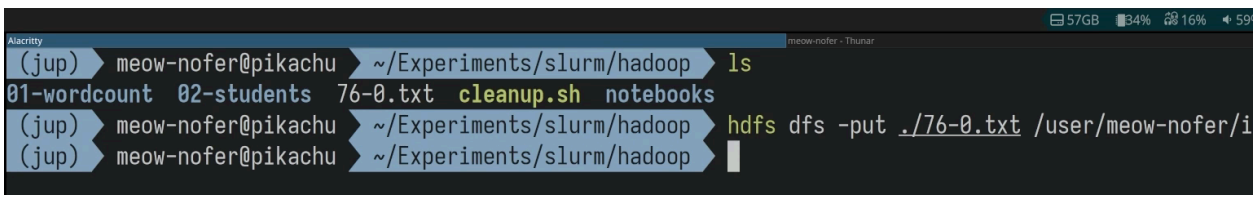
```
~$ bin/hdfs dfs -mkdir /user
~$ bin/hdfs dfs -mkdir /user/meow-nofer
~$ bin/hdfs dfs -mkdir /user/meow-nofer/input
~$ bin/hdfs dfs -put ./76-0.txt /user/meow-nofer/input
```

Посмотрим, как это работает:

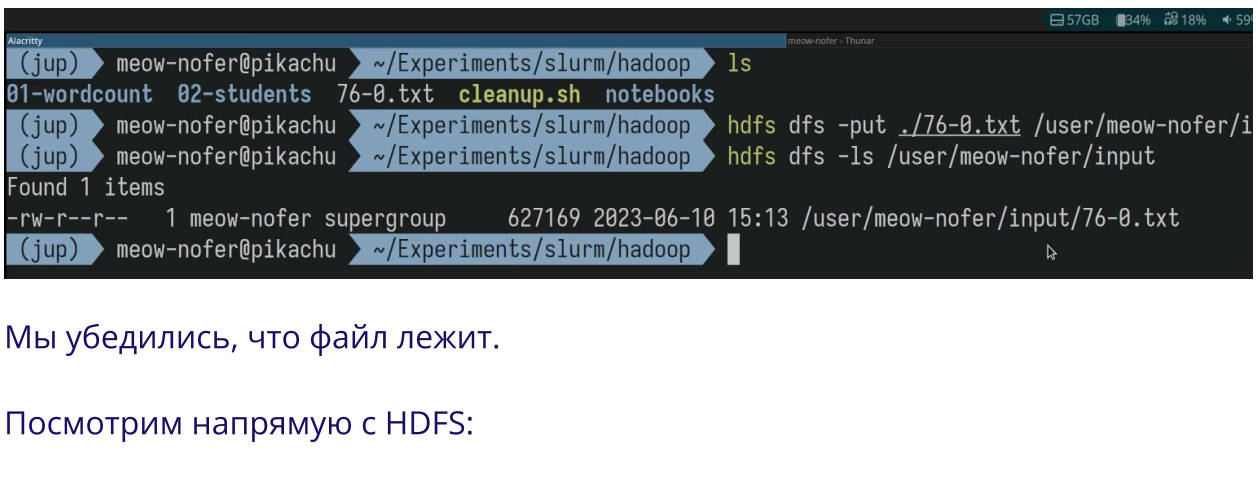
В Hadoop по умолчанию считается, что у каждого пользователя есть свой домашний каталог, который находится в /user/user-name. Если вы запускаете job, например, MapReduce, то очень часто подразумевается, что входные данные лежат именно в этом каталоге.

Мы знаем, что есть опция «-mkdir -p». Можно сделать следующее:

Для того, чтобы копировать файлы туда-обратно не используется команда «cp» (она служит для копирования файлов внутри системы), используются команды «put» и «get»:

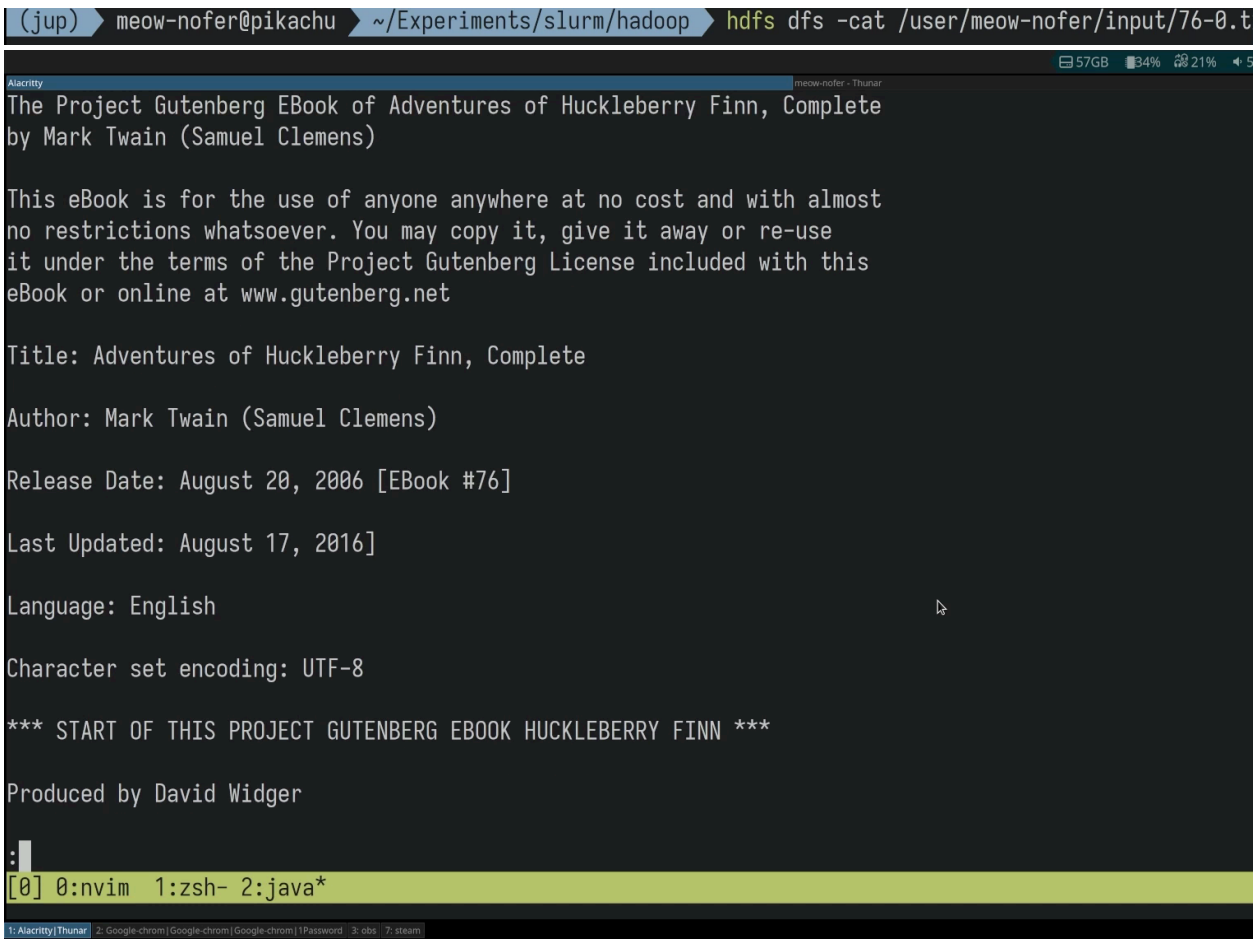


Проверим залилось ли:



Мы убедились, что файл лежит.

Посмотрим напрямую с HDFS:



Нюанс: важно докрутить до конца, иначе «cut» будет падать.

Как вам урок?

Изучил, далее