

Текстовая расшифровка видео:

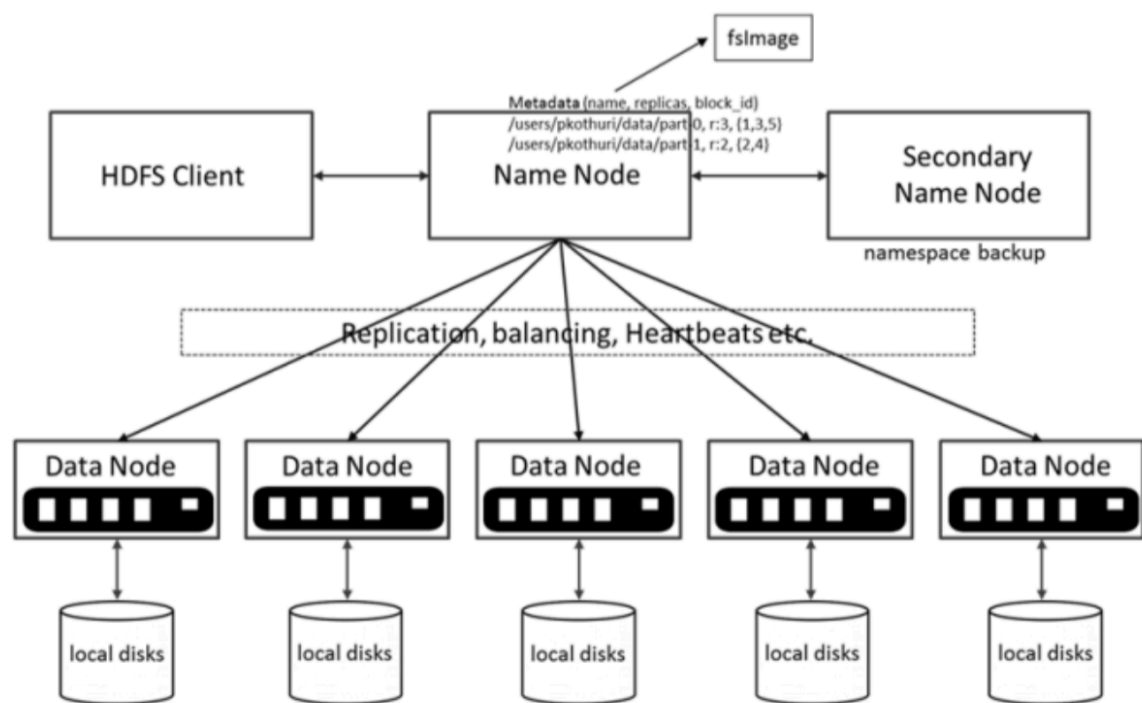
ПОДНИМАЕМ HDFS У СЕБЯ

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.

КупиСкачай 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 окружения 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:

```

1 Licensed to the Apache Software Foundation (ASF) under one
2 # or more contributor license agreements. See the NOTICE file
3 # distributed with this work for additional information
4 # regarding copyright ownership. The ASF licenses this file
5 # to you under the Apache License, Version 2.0 (the
6 # "License"); you may not use this file except in compliance
7 # with the License. You may obtain a copy of the License at
8 #
9 # http://www.apache.org/licenses/LICENSE-2.0
10 #
11 # Unless required by applicable law or agreed to in writing, software
12 # distributed under the license is distributed on an "AS IS" BASIS,
13 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14 # See the License for the specific language governing permissions and
15 # limitations under the License.
16
17 # Set Hadoop-specific environment variables here.
18
19 # The only required environment variable is JAVA_HOME. All others are
20 # optional. When running a distributed configuration it is best to
21 # set JAVA_HOME in this file, so that it is correctly defined on
22 # remote nodes.
23

```

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:

```
25 export JAVA_HOME=/usr/lib/jvm/default
26 #export JAVA_HOME=${JAVA_HOME}
```

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:

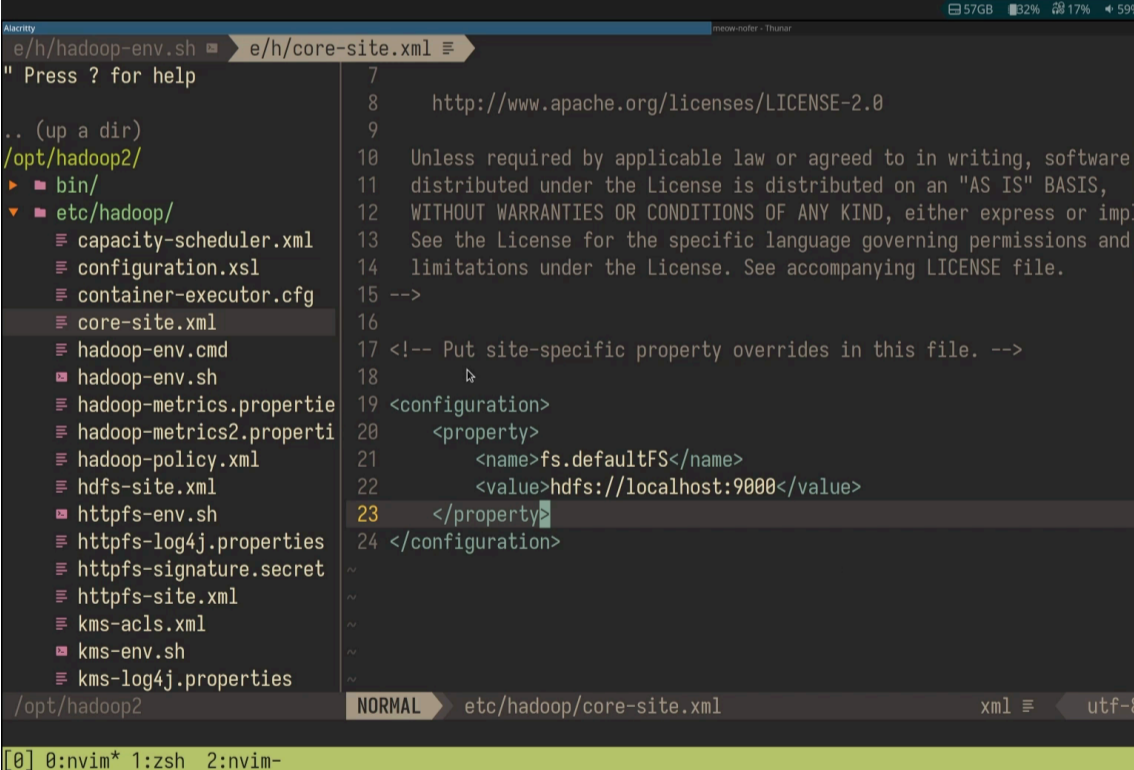
etc/hadoop/core-site.xml

```
<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»:



```
Activity
e/h/hadoop-env.sh e/h/core-site.xml
" Press ? for help
.. (up a dir)
/opt/hadoop2/
  bin/
  etc/hadoop/
    capacity-scheduler.xml
    configuration.xml
    container-executor.cfg
    core-site.xml
    hadoop-env.cmd
    hadoop-env.sh
    hadoop-metrics.properties
    hadoop-metrics2.properties
    hadoop-policy.xml
    hdfs-site.xml
    https-env.sh
    https-log4j.properties
    https-signature.secret
    https-site.xml
    kms-acls.xml
    kms-env.sh
    kms-log4j.properties
  /opt/hadoop2
  NORMAL ete/hadoop/core-site.xml xml utf-8
[0] 0:nvim* 1:zsh 2:nvim-
```

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:

```

Alacritty
e/h/hdfs-site.xml  e/h/hadoop-env.sh  e/h/core-site.xml
" Press ? for help
.. (up a dir)
/opt/hadoop2/
├─ bin/
└─ etc/hadoop/
    ├── capacity-scheduler.xml
    ├── configuration.xml
    ├── container-executor.cfg
    ├── core-site.xml
    ├── hadoop-env.cmd
    ├── hadoop-env.sh
    ├── hadoop-metrics.properties
    ├── hadoop-metrics2.properties
    ├── hadoop-policy.xml
    ├── hdfs-site.xml
    ├── httpfs-env.sh
    ├── httpfs-log4j.properties
    ├── httpfs-signature.secret
    ├── httpfs-site.xml
    ├── kms-acls.xml
    ├── kms-env.sh
    └── kms-log4j.properties
/opt/hadoop2
NORMAL  etc/hadoop/hdfs-site.xml  xml  utf-8
"etc/hadoop/hdfs-site.xml" 24L, 867B
[0] 0:nvim* 1:zsh 2:nvim-

```

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»):

```

Alacritty
meow-nofer@pikachu /opt/hadoop2 which hdfs
/opt/hadoop2/bin/hdfs
meow-nofer@pikachu /opt/hadoop2 hdfs namenode -format
23/06/10 15:07:03 INFO namenode.FSNamesystem: dfs.namenode.safemode.threshold-pct = 0.9990000128746033
23/06/10 15:07:03 INFO namenode.FSNamesystem: dfs.namenode.safemode.min.datanodes = 0
23/06/10 15:07:03 INFO namenode.FSNamesystem: dfs.namenode.safemode.extension = 30000
23/06/10 15:07:03 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.window.num.buckets = 10
23/06/10 15:07:03 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.num.users = 10
23/06/10 15:07:03 INFO metrics.TopMetrics: NNTop conf: dfs.namenode.top.windows.minutes = 1,5,25
23/06/10 15:07:03 INFO namenode.FSNamesystem: Retry cache on namenode is enabled
23/06/10 15:07:03 INFO namenode.FSNamesystem: Retry cache will use 0.03 of total heap and retry cache entries
23/06/10 15:07:03 INFO util.GSet: Computing capacity for map NameNodeRetryCache
23/06/10 15:07:03 INFO util.GSet: VM type = 64-bit
23/06/10 15:07:03 INFO util.GSet: 0.0299999999329447746% max memory 889 MB = 273.1 KB
23/06/10 15:07:03 INFO util.GSet: capacity = 2^15 = 32768 entries
23/06/10 15:07:03 INFO namenode.FSImage: Allocated new BlockPoolId: BP-1749854144-127.0.1.1-1686398823374
23/06/10 15:07:03 INFO common.Storage: storage directory /tmp/hadoop-meow-nofer/dfs/name has been successfully formatted.
23/06/10 15:07:03 INFO namenode.FSImageFormatProtobuf: Saving image file /tmp/hadoop-meow-nofer/dfs/name/current/fsimage.ckpt_000000000000000000 using no compression
23/06/10 15:07:03 INFO namenode.FSImageFormatProtobuf: Image file /tmp/hadoop-meow-nofer/dfs/name/current/fsimage.ckpt_000000000000000000 of size 357 bytes saved in 0 seconds.
23/06/10 15:07:03 INFO namenode.NNStorageRetentionManager: Going to retain 1 images with txid >= 0
23/06/10 15:07:03 INFO util.ExitUtil: Exiting with status 0
23/06/10 15:07:03 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****
SHUTDOWN_MSG: Shutting down NameNode at pikachu.localdomain/127.0.1.1
*****/
[0] 0:nvim- 1:java* 2:nvim "pikachu" 15:07 18-июн-23

```

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

```
meow-nofer@pikachu /opt/hadoop2$ ./sbin/start-dfs.sh
Starting namenodes on [localhost]
localhost: starting namenode, logging to /opt/hadoop2/logs/hadoop-meow-nofer-namenode-pikachu.out
localhost: starting datanode, logging to /opt/hadoop2/logs/hadoop-meow-nofer-datanode-pikachu.out
Starting secondary namenodes [0.0.0.0]
0.0.0.0: starting secondarynamenode, logging to /opt/hadoop2/logs/hadoop-meow-nofer-secondarynamenode-pikachu.out
meow-nofer@pikachu /opt/hadoop2$
```

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

- NameNode;
- Secondary NameNode;
- DataNode;

```
meow-nofer@pikachu /opt/hadoop2$ jps
1354705 NameNode
1355363 Jps
1355034 SecondaryNameNode
1354843 DataNode
meow-nofer@pikachu /opt/hadoop2$
```

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:

```
meow-nofer@pikachu /opt/hadoop2$ jps
1354705 NameNode
1355363 Jps
1355034 SecondaryNameNode
1354843 DataNode
meow-nofer@pikachu /opt/hadoop2$ ls
bin etc include lib libexec LICENSE.txt logs NOTICE.txt README.txt sbin share
meow-nofer@pikachu /opt/hadoop2$ cd logs
meow-nofer@pikachu /opt/hadoop2/logs$ ls
hadoop-meow-nofer-datanode-pikachu.log          hadoop-meow-nofer-secondarynamenode-pikachu.out
hadoop-meow-nofer-datanode-pikachu.out        hadoop-meow-nofer-secondarynamenode-pikachu.out.1
hadoop-meow-nofer-datanode-pikachu.out.1     hadoop-meow-nofer-secondarynamenode-pikachu.out.2
hadoop-meow-nofer-datanode-pikachu.out.2     hadoop-meow-nofer-secondarynamenode-pikachu.out.3
hadoop-meow-nofer-datanode-pikachu.out.3     hadoop-meow-nofer-secondarynamenode-pikachu.out.4
hadoop-meow-nofer-datanode-pikachu.out.4     hadoop-meow-nofer-secondarynamenode-pikachu.out.5
hadoop-meow-nofer-datanode-pikachu.out.5     SecurityAuth-meow-nofer.audit
hadoop-meow-nofer-namenode-pikachu.log       userlogs
hadoop-meow-nofer-namenode-pikachu.out      yarn-meow-nofer-nodemanager-pikachu.log
hadoop-meow-nofer-namenode-pikachu.out.1    yarn-meow-nofer-nodemanager-pikachu.out
hadoop-meow-nofer-namenode-pikachu.out.2    yarn-meow-nofer-nodemanager-pikachu.out.1
hadoop-meow-nofer-namenode-pikachu.out.3    yarn-meow-nofer-resourcemanager-pikachu.log
hadoop-meow-nofer-namenode-pikachu.out.4    yarn-meow-nofer-resourcemanager-pikachu.out
hadoop-meow-nofer-namenode-pikachu.out.5    yarn-meow-nofer-resourcemanager-pikachu.out.1
hadoop-meow-nofer-secondarynamenode-pikachu.log
```

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:

```
meow-nofer@pikachu /opt/hadoop2$ hdfs dfs -ls /
meow-nofer@pikachu /opt/hadoop2$
```

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:

```
meow-nofer@pikachu /opt/hadoop2/logs cd ..
meow-nofer@pikachu /opt/hadoop2 hdfs dfs -ls /
meow-nofer@pikachu /opt/hadoop2 hdfs dfs -mkdir -p /user/meow-nofer/input
meow-nofer@pikachu /opt/hadoop2 hdfs dfs -ls /
Found 1 items
drwxr-xr-x  - meow-nofer supergroup          0 2023-06-10 15:12 /user
meow-nofer@pikachu /opt/hadoop2
```

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

```
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop ls
01-wordcount 02-students 76-0.txt cleanup.sh notebooks
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop hdfs dfs -put ./76-0.txt /user/meow-nofer/i
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop
```

Проверим залилось ли:

```
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop ls
01-wordcount 02-students 76-0.txt cleanup.sh notebooks
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop hdfs dfs -put ./76-0.txt /user/meow-nofer/i
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop hdfs dfs -ls /user/meow-nofer/input
Found 1 items
-rw-r--r--  1 meow-nofer supergroup    627169 2023-06-10 15:13 /user/meow-nofer/input/76-0.txt
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop
```

Мы убедились, что файл лежит.

Посмотрим напрямую с HDFS:

```
(jup) meow-nofer@pikachu ~/Experiments/slurm/hadoop hdfs dfs -cat /user/meow-nofer/input/76-0.t
The Project Gutenberg EBook of Adventures of Huckleberry Finn, Complete
by Mark Twain (Samuel Clemens)

This eBook is for the use of anyone anywhere at no cost and with almost
no restrictions whatsoever. You may copy it, give it away or re-use
it under the terms of the Project Gutenberg License included with this
eBook or online at www.gutenberg.net

Title: Adventures of Huckleberry Finn, Complete
Author: Mark Twain (Samuel Clemens)
Release Date: August 20, 2006 [EBook #76]
Last Updated: August 17, 2016]
Language: English
Character set encoding: UTF-8

*** START OF THIS PROJECT GUTENBERG EBOOK HUCKLEBERRY FINN ***

Produced by David Widger
```

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

Как вам урок?



Изучил, далее >

