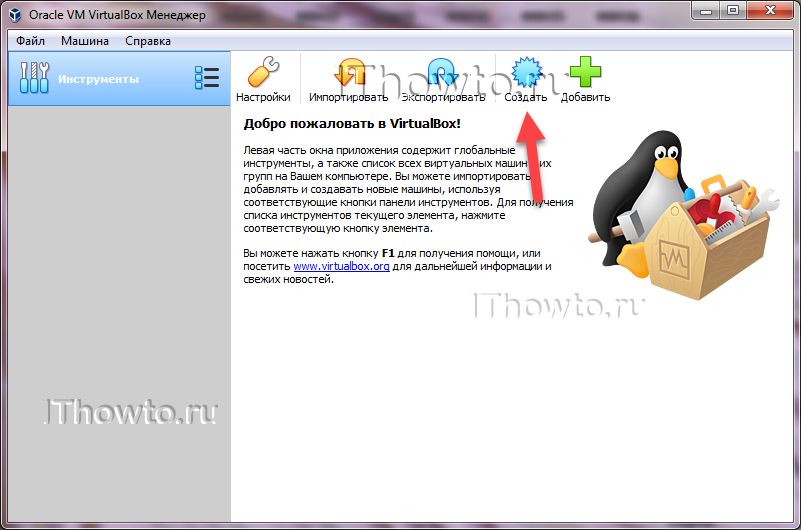
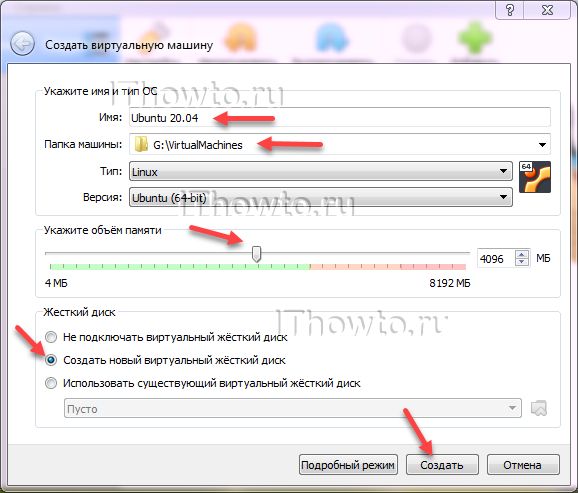
**Установка Ubuntu 20.04 в VirtualBox**

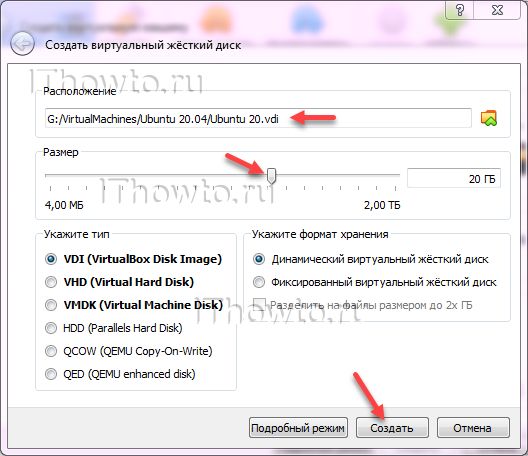
1. Скачайте образ диска Ubuntu 20.04 с официальной страницы дистрибутива и сохраните его в папке на жестком диске.
2. Сначала необходимо создать виртуальную машину для установки. Для этого запустите VirtualBox и нажмите на пункт "Создать":



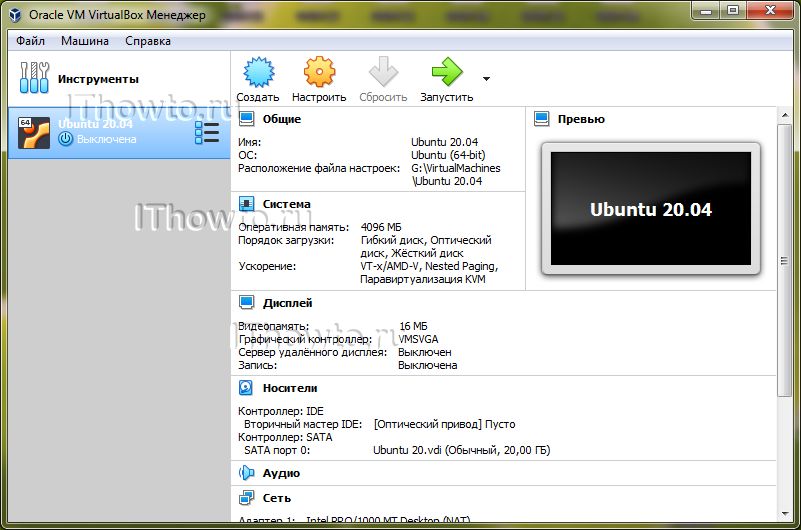
1. Называем нашу виртуальную машину согласно версии - "Ubuntu 20.04". Указываем где будет располагаться файл с содержимым виртуальной машины. Для того, чтобы система, при работе, не испытывала нужды в памяти, устанавливаем размер памяти в 4 Гб. Отмечаем пункт "Создать новый виртуальный диск" и нажимаем на кнопку "Создать":



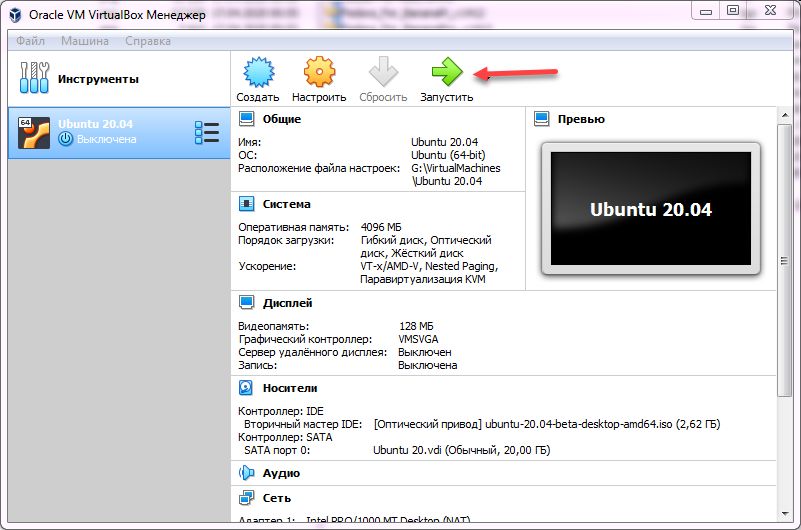
1. VirtualBox автоматически создаст подпапку в выбранной ранее папке, и даст название файлу с виртуальной машиной. Нажимаем на кнопку "Создать":



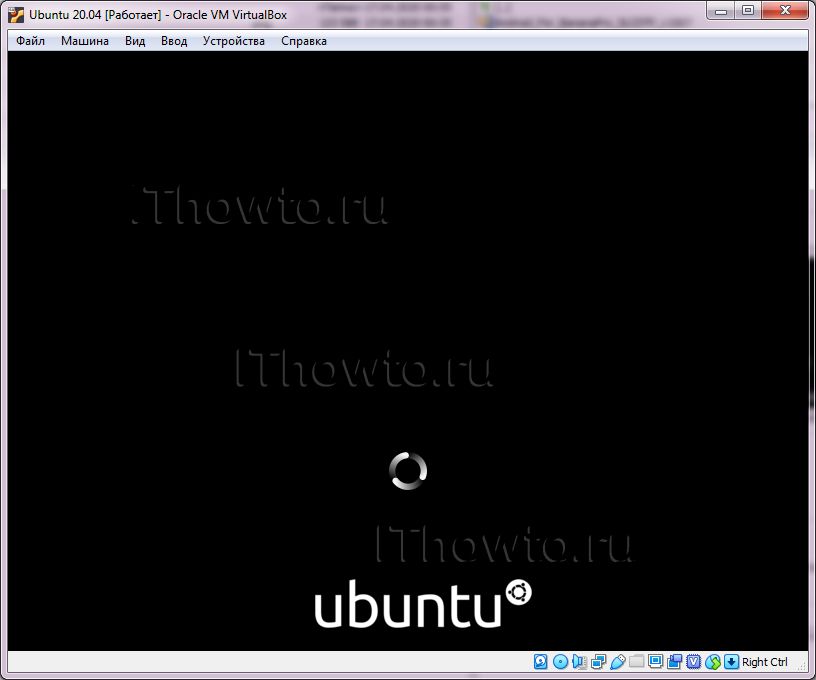
1. Видим, что виртуальная машина для Ubuntu 20.04 создана:



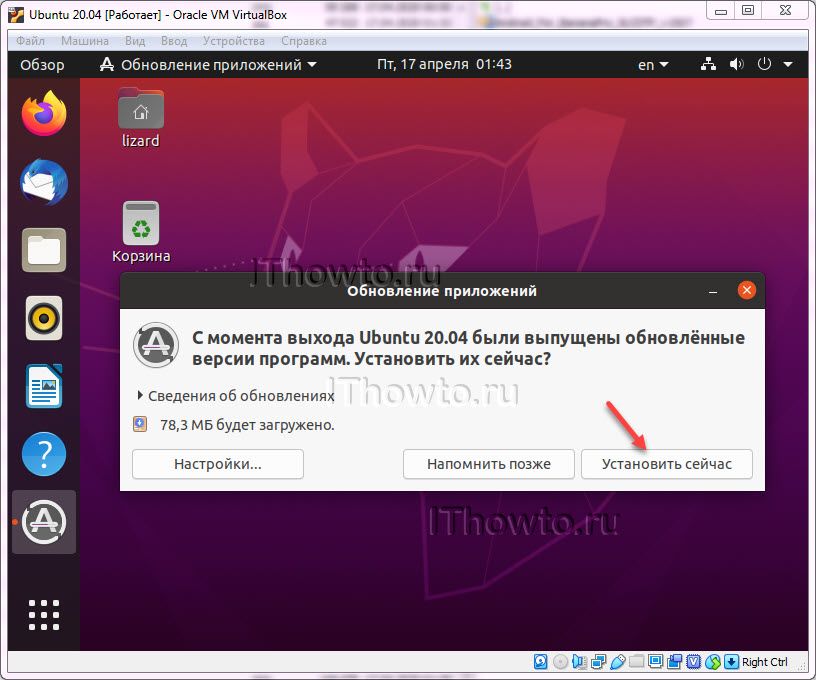
1. Настала пора запускать созданную нами виртуальную машину с Ubuntu 20.04 и приступать к ее установке:



1. При старте установщика Ubuntu 20.04 будет произведена проверка диска. Далее, некоторое время мы будем наблюдать вращающийся "спиннер" на экране:



1. Возможно, что после установки Ubuntu 20.04, на экране появится уведомление о доступных обновлениях. Просто подтвердите их установку:



**Установка Hadoop в Ubuntu**

1. Установить Java в Ubuntu

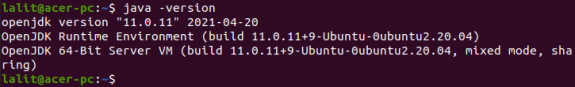
Фреймворк Hadoop написан на Java, поэтому для его установки в вашей системе требуется Java.

Вы можете использовать следующую команду, чтобы установить его в своей системе:

*sudo apt install default-jdk -y*

Вы можете проверить установку Java, используя следующую команду:

*java -version*



1. Установите OpenSSH на Ubuntu

Если SSH не установлен в вашей системе, вы можете установить его с помощью следующей команды:

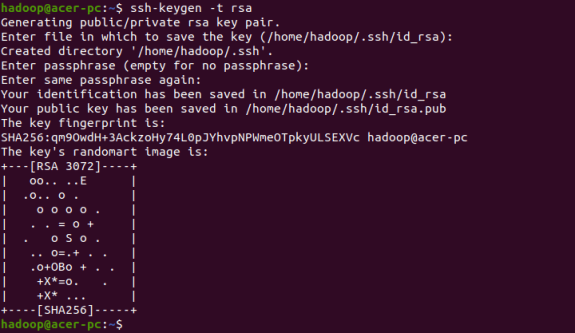
*sudo apt install openssh-server openssh-client -y*

1. Включить SSH без пароля для пользователя Hadoop (или тот, который вы задали при установке).

Вам необходимо настроить SSH без пароля для пользователя Hadoop для управления узлами в кластере или локальной системе. Сгенерируйте пары ключей SSH —

*ssh-keygen -t rsa*

Он попросит вас ввести имя файла и парольную фразу, просто нажмите Enter, чтобы завершить процесс.



Теперь добавьте сгенерированные открытые ключи из id\_rsa.pub в author\_keys —

*cat ~/.ssh/id\_rsa.pub >> ~/.ssh/authorized\_keys*

Теперь установите правильные разрешения для author\_keys —

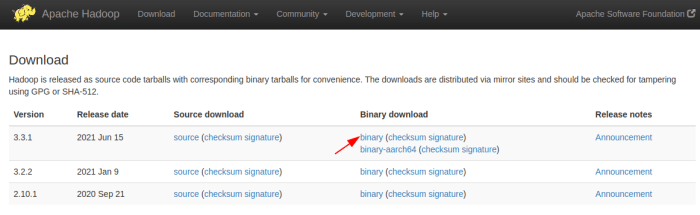
*chmod 640 ~/.ssh/authorized\_keys*

Проверьте аутентификацию SSH, используя следующую команду:

*ssh localhost*

1. Загрузите и установите Хадуп

Перейдите на официальную страницу загрузки Hadoop и выберите загрузку последнего бинарного файла, нажав на указанную ссылку, как вы можете видеть на данном изображении —



В качестве альтернативы используйте команду wget, чтобы загрузить его с вашего терминала —

*wget https://dlcdn.apache.org/hadoop/common/hadoop-3.3.1/hadoop-3.3.1.tar.gz*

После загрузки извлеките его с помощью данной команды –

*sudo tar -xvzf hadoop-3.3.1.tar.gz*

Переименуйте извлеченный каталог в hadoop —

*sudo mv hadoop-3.3.1 hadoop*

1. Настройка переменных среды Hadoop

Нам нужно отредактировать данные файлы, чтобы настроить среду Hadoop.

bashrc

hadoop-env.sh

core-site.xml

hdfs-site.xml

mapred-сайт-xml

yarn-site.xml

Итак, давайте начнем настройку по одному —

1. Отредактируйте файл bashrc

Сначала откройте файл bashrc с помощью текстового редактора.

*sudo nano .bashrc*

Добавьте указанные строки в конец этого файла –

*export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64*

*export HADOOP\_HOME=/home/hadoop/hadoop*

*export HADOOP\_INSTALL=$HADOOP\_HOME*

*export HADOOP\_MAPRED\_HOME=$HADOOP\_HOME*

*export HADOOP\_COMMON\_HOME=$HADOOP\_HOME*

*export HADOOP\_HDFS\_HOME=$HADOOP\_HOME*

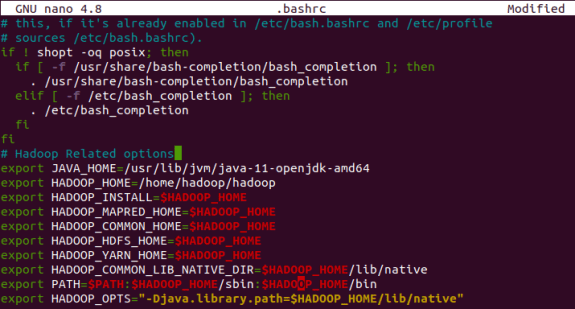
*export HADOOP\_YARN\_HOME=$HADOOP\_HOME*

*export HADOOP\_COMMON\_LIB\_NATIVE\_DIR=$HADOOP\_HOME/lib/native*

*export PATH=$PATH:$HADOOP\_HOME/sbin:$HADOOP\_HOME/bin*

*export HADOOP\_OPTS="-Djava.library.path=$HADOOP\_HOME/lib/native"*

Сохраните этот файл и выйдите из редактора.



Активируйте переменную среды, выполнив следующую команду:

*source ~/.bashrc*

1. Изменить файл переменной среды Hadoop

Затем откройте файл переменной среды Hadoop, т.е. hadoop-env.sh

*nano $HADOOP\_HOME/etc/hadoop/hadoop-env.sh*

и установите переменную JAVA\_HOME, как указано ниже –

*export JAVA\_HOME=/usr/lib/jvm/java-11-openjdk-amd64*

Сохраните и закройте файл.

1. Отредактируйте файл core-site.xml

Сначала создайте namenode и datanode каталоги внутри домашнего каталога Hadoop с помощью данной команды –

*mkdir -p ~/hadoopdata/hdfs/namenode*

*mkdir -p ~/hadoopdata/hdfs/datanode*

1. Откройте и отредактируйте файл core-site.xml —

*nano $HADOOP\_HOME/etc/hadoop/core-site.xml*

Здесь измените значение в соответствии с вашим именем хоста –

*<configuration>*

*<property>*

*<name>fs.defaultFS</name>*

*<value>hdfs://127.0.0.1:9000</value>*

*</property>*

*</configuration>*

Сохраните и закройте этот файл.

1. Отредактируйте файл hdfs-site.xml

Откройте файл hdfs-site.xml —

*nano $HADOOP\_HOME/etc/hadoop/hdfs-site.xml*

И измените namenode и datanode пути к каталогам –

*<configuration>*

*<property>*

*<name>dfs.replication</name>*

*<value>1</value>*

*</property>*

*<property>*

*<name>dfs.name.dir</name>*

*<value>file:///home/hadoop/hadoopdata/hdfs/namenode</value>*

*</property>*

*<property>*

*<name>dfs.data.dir</name>*

*<value>file:///home/hadoop/hadoopdata/hdfs/datanode</value>*

*</property>*

*</configuration>*

Сохраните этот файл и выйдите из редактора.

1. Изменить mapred-site.xml

Затем откройте и отредактируйте файл mapred-site.xml —

*nano $HADOOP\_HOME/etc/hadoop/mapred-site.xml*

Внесите изменения, как указано ниже –

*<configuration>*

*<property>*

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

*<value>yarn</value>*

*</property>*

*</configuration>*

Сохраните и закройте также этот файл.

1. Отредактируйте yarn-site.xml

*Теперь отредактируйте файл yarn-site.xml –*

*nano $HADOOP\_HOME/etc/hadoop/yarn-site.xml*

И внесите данные изменения –

*<configuration>*

*<property>*

*<name>yarn.nodemanager.aux-services</name>*

*<value>mapreduce\_shuffle</value>*

*</property>*

*</configuration>*

Сохраните файл и закройте редактор.

1. Запустите кластер Hadoop

Перед запуском кластера Hadoop важно отформатировать namenode.

Выполните следующую команду, чтобы отформатировать namenode –

*hdfs namenode -format*

После успешного форматирования используйте следующую команду для запуска кластера Hadoop.

*start-dfs.sh*

Затем запустите службу YARN с помощью данной команды –

*start-yarn.sh*

После запуска вышеуказанных служб вы можете проверить, работают ли они или нет, используя —

*jps*



1. Доступ к Hadoop из вашего браузера

Откройте браузер в своей системе и введите указанный URL-адрес, чтобы получить доступ к веб-интерфейсу Hadoop в браузере.

*http://localhost:9870*

Это обеспечит полный обзор всего кластера.

Порт по умолчанию для datanode это 9864, поэтому для доступа к нему используйте -

*http://localhost:9864*

Менеджер ресурсов пряжи доступен на порту с номером 8088, поэтому для доступа к нему используйте:

*http://localhost:8088*

Здесь вы можете отслеживать все процессы, запущенные в вашем кластере Hadoop.

**Выполнить пример WordCount v1.0**

1. Создать каталог WordCountTutorial с файлом WordCount.java со следующим кодом:

*import java.io.IOException;*

*import java.util.StringTokenizer;*

*import org.apache.hadoop.conf.Configuration;*

*import org.apache.hadoop.fs.Path;*

*import org.apache.hadoop.io.IntWritable;*

*import org.apache.hadoop.io.Text;*

*import org.apache.hadoop.mapreduce.Job;*

*import org.apache.hadoop.mapreduce.Mapper;*

*import org.apache.hadoop.mapreduce.Reducer;*

*import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;*

*import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;*

*public class WordCount {*

*public static class TokenizerMapper*

*extends Mapper<Object, Text, Text, IntWritable>{*

*private final static IntWritable one = new IntWritable(1);*

*private Text word = new Text();*

*public void map(Object key, Text value, Context context*

*) throws IOException, InterruptedException {*

*StringTokenizer itr = new StringTokenizer(value.toString());*

*while (itr.hasMoreTokens()) {*

*word.set(itr.nextToken());*

*context.write(word, one);*

*}*

*}*

*}*

*public static class IntSumReducer*

*extends Reducer<Text,IntWritable,Text,IntWritable> {*

*private IntWritable result = new IntWritable();*

*public void reduce(Text key, Iterable<IntWritable> values,*

*Context context*

*) throws IOException, InterruptedException {*

*int sum = 0;*

*for (IntWritable val : values) {*

*sum += val.get();*

*}*

*result.set(sum);*

*context.write(key, result);*

*}*

*}*

*public static void main(String[] args) throws Exception {*

*Configuration conf = new Configuration();*

*Job job = Job.getInstance(conf, "word count");*

*job.setJarByClass(WordCount.class);*

*job.setMapperClass(TokenizerMapper.class);*

*job.setCombinerClass(IntSumReducer.class);*

*job.setReducerClass(IntSumReducer.class);*

*job.setOutputKeyClass(Text.class);*

*job.setOutputValueClass(IntWritable.class);*

*FileInputFormat.addInputPath(job, new Path(args[0]));*

*FileOutputFormat.setOutputPath(job, new Path(args[1]));*

*System.exit(job.waitForCompletion(true) ? 0 : 1);*

*}*

*}*

1. Создаем файл input.txt в каталоге WordCountTutorial/input\_data/input.txt с произвольным текстом.
2. Создание каталога в файловой системе HDFS

*hadoop fs -mkdir /WordCountTutorial*

1. Cоздание каталога внутри для ввода

*hadoop fs -mkdir /WordCountTutorial/Input*

1. Открываем браузер и проверяем что создались каталоги

*localhost:50070*

Переходим в Utilities > Browser the file system

1. Загружаем файлы в каталоги

*hadoop fs -put '/home/user/Desktop/WordCountTutorial/input\_data/input.txt' /WordCountTutorial/Input*

1. Переходим в каталог учебника

*cd /home/home/user/Desktop/WordCountTutorial*

1. Выполняем java код

*javac -classpath ${HADOOP\_CLASSPATH} -d '/home/user/Desktop/WordCountTutorial/tutorial\_classes' '/home/user/Desktop/WordCountTutorial/WordCount.java'*

1. Созданные файлы в tutorial\_classes помещаем в один файл jar

*jar -cvf firstTutorial.jar -C tutorial\_classes/ .*

1. Запускаем файл .jar

*hadoop jar '/home/user/Desktop/WordCountTutorial/firstTutorial.jar' WordCount /WordCountTutorial/Input /WordCountTutorial/Output*

1. Вывод результата

*hadoop dfs -cat /WordCountTutorial/Output/\**

Дополнительные источники:

<https://www.youtube.com/watch?v=Slbi-uzPtnw>

<https://www.youtube.com/watch?v=6sK3LDY7Pp4>

Создание .jar через IDE <https://www.projectpro.io/hadoop-tutorial/hadoop-mapreduce-wordcount-tutorial>

<https://phoenixnap.com/kb/install-hadoop-ubuntu>

<https://medium.com/@festusmorumbasi/installing-hadoop-on-ubuntu-20-04-4610b6e0391e>