

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«МИРЭА – Российский технологический университет»**

**РТУ МИРЭА**

Институт: КБСП

Направление: 09.03.02 «Информационные системы и технологии»

Кафедра: КБ-14 «Средства визуализации и обработки данных»

Дисциплина: «Методы Big Data»

**ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №1**

**«Разработка клиента HDFS»**

Выполнил студент БСБО-08-18 Шайко Т.И.

Руководитель работы: Лебедев А.С.

Москва 2021 г.

Оглавление

[Цель работы 3](#_Toc86869209)

[Задача работы 3](#_Toc86869210)

[Теоретическое введение 4](#_Toc86869211)

[HDFS 4](#_Toc86869212)

[bash 5](#_Toc86869213)

[Ход выполнения 6](#_Toc86869214)

[Список использованной литературы 10](#_Toc86869215)

[Приложение 11](#_Toc86869216)

# Цель работы

Разработать клиент для управления распределённой файловой системой Hadoop, сокращенно — HDFS. Он должен поддерживать выполнение следующих операций:

* *mkdir* <имя каталога в HDFS> (создание каталога в HDFS);
* *put* <имя локального файла> (загрузка файла в HDFS);
* *get* <имя файла в HDFS> (скачивание файла из HDFS);
* *append* <имя локального файла> <имя файла в HDFS> (конкатенация файла в HDFS с локальным файлом);
* *delete* <имя файла в HDFS> (удаление файла в HDFS);
* *ls* (отображение содержимого текущего каталога в HDFS с разделением файлов и каталогов);
* *cd* <имя каталога в HDFS> (переход в другой каталог в HDFS, ".." — на уровень выше);
* *lls* (отображение содержимого текущего локального каталога с разделением файлов и каталогов);
* *lcd* <имя локального каталога> (переход в другой локальный каталог, ".." — на уровень выше).

Имена файлов и каталогов не содержат путей и символа "/". Параметры командной строки клиента: сервер, порт, имя пользователя. Пример запуска клиента:

./myhdfscli.rb localhost 50070 aslebedev

Использовать WebHDFS REST API и любой язык программирования.

# Задача работы

Используя выбранный язык программирования реализовать клиент HDFS. Для простоты реализации выберем язык командной оболочки Линукса — bash. Остаётся лишь создать скрипт .sh и определить его поведение — взаимодействие с файловой системой Хадупа.

# Теоретическое введение

## HDFS

Для хранения информации большого объёма и проведения основных операций с такими данными: чтения и записи, — ориентированных на быстродействие всей системы, была придумана файловая система распределённого типа — DFS. Она относится к важному проекту по хранению и переработке данных большого количества — Hadoop. Поэтому полное название файловой системы, о которой будет вестись речь в этой работе, выглядит так — **HDFS**.

HDFS — это файловая система, которая хранит в себе и управляет файлами, физически расположенными на разных машинах, то есть серверах. Причём эти серверы в основном дешёвого качества — и масштабирование в рамках HDFS описывают как **горизонтальное**, иными словами улучшение системы производится не улучшением общего качества оборудования (его производительности, объёмом памяти и другими характеристиками), а увеличением количества этих относительно недорогих машин.

Помимо этого, ещё одной спецификой HDFS является **потоковая обработка данных**. Это значит, что данные идут одним потоком при передаче по сети между машинами, включенными в эту систему: их нельзя прочесть, допустим, с середины или начать записывать в файл информацию не с конца, а с начала или с той же середины файла — записывать можно только в конец существующего файла или в новый файл.

И стоит ещё раз упомянуть про объём данных, с которым работает HDFS. Он большой и исчисляется **терабайтами и петабайтами**. Это привычный объём данных для этой файловой системы.

Дополню, что HDFS обладает также свойством хорошей **отказоустойчивости и** **надёжности** при неожиданно случающихся сбоях и неисправностях оборудования серверного кластера. Она достигается путём репликации (копирования) всех данных на разные машины, и при сбое одной из них — файлы с нужной информацией берутся из их копий, расположенных на других машинах.

В пределах одного сервера различают понятия **datanode** и **namenode**, или узел данных и узел имён. Первый узел — datanode — хранит данные, в него можно записать информацию и из него можно прочитать её. Namenode, или узел имён, хранит метаданные всех файлов, причём файлов не только той машины, на которой он расположен, но и файлов, находящихся на других машинах кластера. Метаданные — это атрибуты файлов, к примеру, их размер, дата создания, владелец и другая информация.

Для предотвращения сбоев придумали резервный узел имён — **secondary namenode**. Он время от времени копирует информацию с основного узла имён, который работает сейчас — вторичный узел имён считывает метаданные, включая расположение отдельных блоков файлов на разных серверах кластера. Но стоит учесть тот факт, что при сбое основного, первичного узла имён, всю информацию с вторичного узла имён восстановить не получится, потому что обновляется она не настолько часто. Но всё равно бОльшую часть этих данных при какой-нибудь технической неполадке сервера вернуть себе не составит труда.

К узлу имён в первую очередь при записи данных обращается клиент. А он ищет доступные datanode’ы в кластере и предлагает их для записи клиенту. (Файлы в файловой системе Hadoop’а располагаются каждый сразу на разных серверах в виде их частей, или **блоков**). При чтении файла происходит то же обращение сначала к namenode’у — ведь он хранит метаданные этого файла. [3] [4]

## bash

Ещё немного расскажу про язык сценариев, который понимает и читает оболочка командной строки Linux — bash. Он так и называется. С его помощью можно писать свои последовательности привычных нам команд Линукса и тем самым автоматизировать свою работу за компьютером.

В понятия и термины этого языка входят знакомые программисту слова «переменные», «структуры ветвления программы» (if then else; case), «циклы» (for и while), «потоки ввода-вывода и их перенаправление» (STDIN, STDOUT, STDERR), «функции и модули» и множество других понятий и определений.

Подробнее читай в многочисленных пособиях по bash в Интернете. [2] [1]

# Ход выполнения

Работать с HDFS в рамках этой лабораторной работы мы будем с помощью сценария bash.

Итак, создаём новую директорию в каталоге пользователя «*root*» — «*/root/mysources/hdfsClient*». В нём создаём, используя утилиту «*touch*», новый файл с расширением «*.sh*». Потом задаём права на выполнение этого нового файла-скрипта bash:

#chmod +x client.sh

Всё, теперь можно приступать к написанию кода для реализации клиента HDFS.

Сначала предусматриваем случаи возникновения ошибок, таких как ввод не всех параметров командной строки (их должно быть у нас по заданию три штуки: хост, порт и имя пользователя Хадупа) или ввод неверного имени пользователя (то есть того, который не указан в Хадупе — который не устанавливал его и не пользуется им).

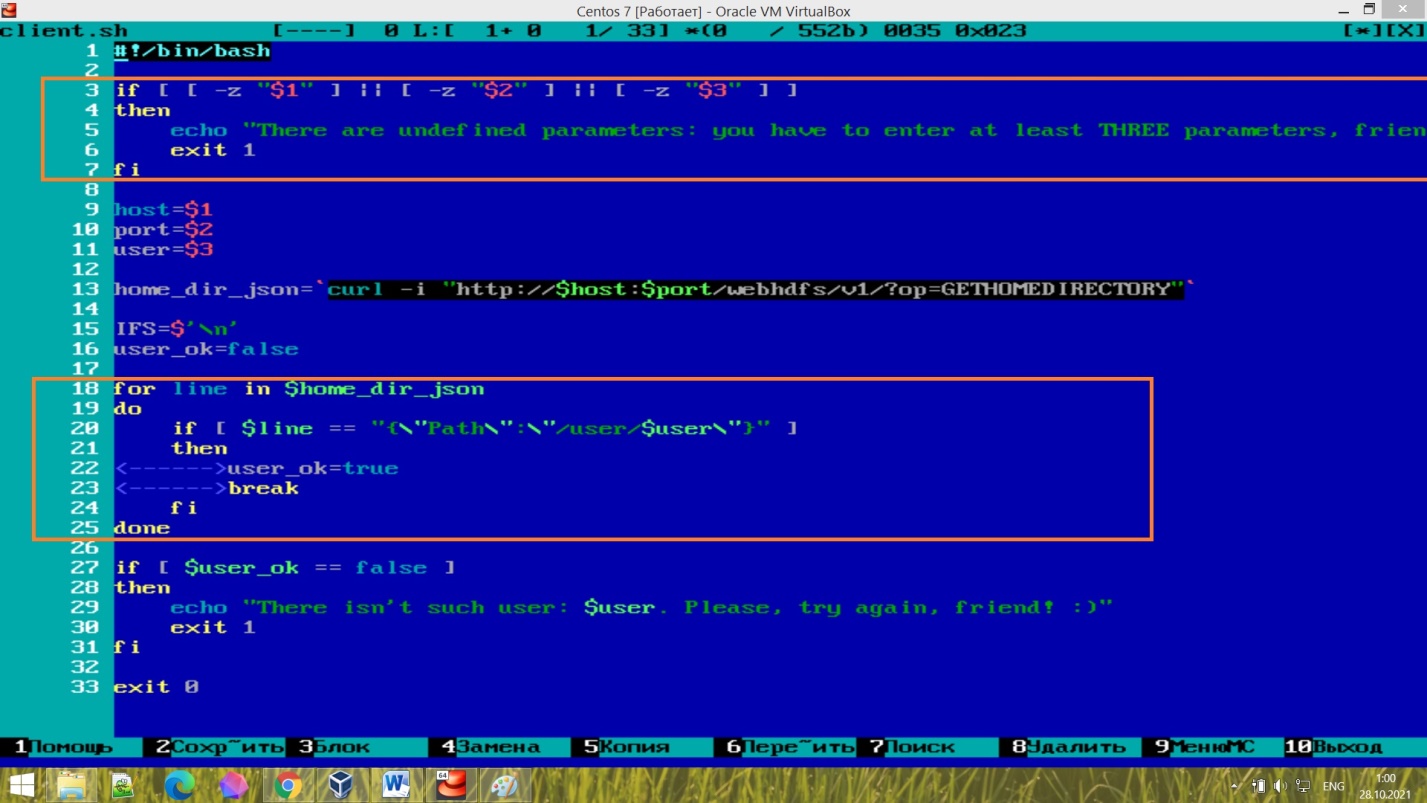


Рисунок 1 — Начальный проверочный код нашего скрипта — клиента HDFS

На рисунке 1 изображён начальный кусок bash-сценария, который проверяет случаи возникновения подобных ошибок.

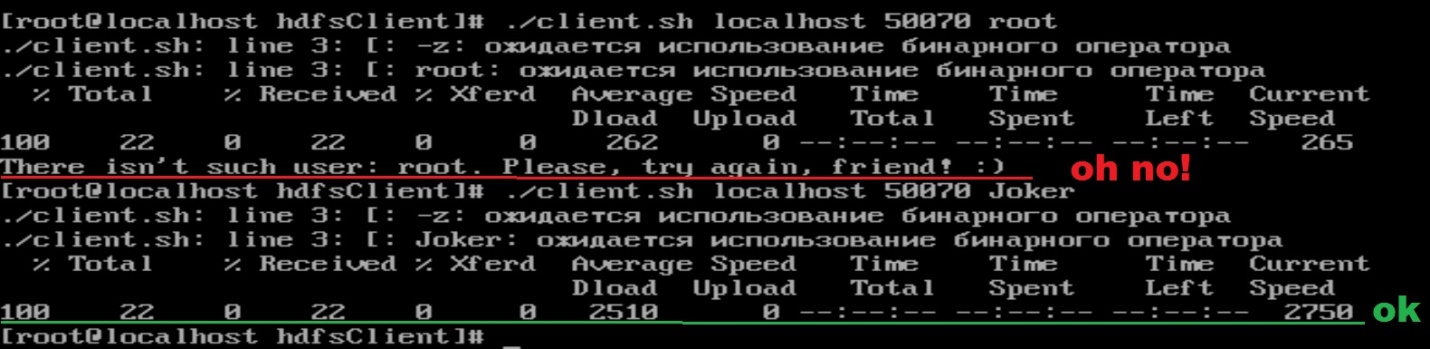
В блоке, начинающемся с 3-ьей строки кода и заканчивающемся 7-ой строкой, проверяется указание пользователем всех трёх параметров командной строки при запуске этого скрипта.

Потом на 13-ой строчке обращаемся через http-запрос к HDFS и просим вернуть нам домашнюю директорию пользователя Хадупа. Система вернёт нам json-объект, в котором будет прописана эта домашняя директория, и который мы сохраним в переменную «*home\_dir\_json*» для последующей обработки в цикле.

Этот обрабатывающий цикл мы поместим следом за выполнением этого запроса — в блоке с 18-ой строки кода по 25-ую строку. Он считает каждую строку сохранённого json-объекта и, когда встретит строку с названием домашней директории HDFS (строка 20 — условие, где как раз указана эта строка), вернёт нам флаг («*user\_ok*»), который будет истинным («true») при совпадении введённого имени пользователя с именем пользователя в json-объекте и ложным («false») при вводе не того пользователя.

Если этот флаг равен «false», то дальше — в строках с 27-ую по 31-ую — мы выйдем из нашего клиента из-за ошибки ввода имени пользователя и по этой причине невозможности продолжить надлежащую и корректную работу с HDFS.

Проверяем работу этого куска кода и убеждаемся, что всё работает, как мы и задумали:



Дальше мы пишем набор требуемых bash-функций для выполнения требуемых задач нашим клиентом. В каждой из них мы вызываем встроенную в Линукс команду «curl» и указываем в качестве её параметров, помимо ключей *–I*,*–X* и *–T*, http-запрос к узлу имён нашей распределённой системы (в рамках этих лабораторных работ по BigData он будет всегда находиться у нас на нашем же компьютере) с просьбой выполнить определённую операцию, которая в запросе указывается в http-параметре *«op»*.

И каждая bash-функция будет запрашивать «NameNode» с помощью конкретного http-запроса выполнить ту или иную задачу, выданную преподавателем. К примеру, реализация команды нашего клиента *«ls»* требует bash-функции с названием *«LS()»* и http-запроса с параметром *«op»*, равным *«LISTSTATUS»* (см. рисунок 2).

Некоторые функции HDFS-клиента реализованы с помощью обыкновенной команды *«hdfs dfs <…>»* для разнообразия и простоты.

Весь код скрипта-клиента указан в приложении.

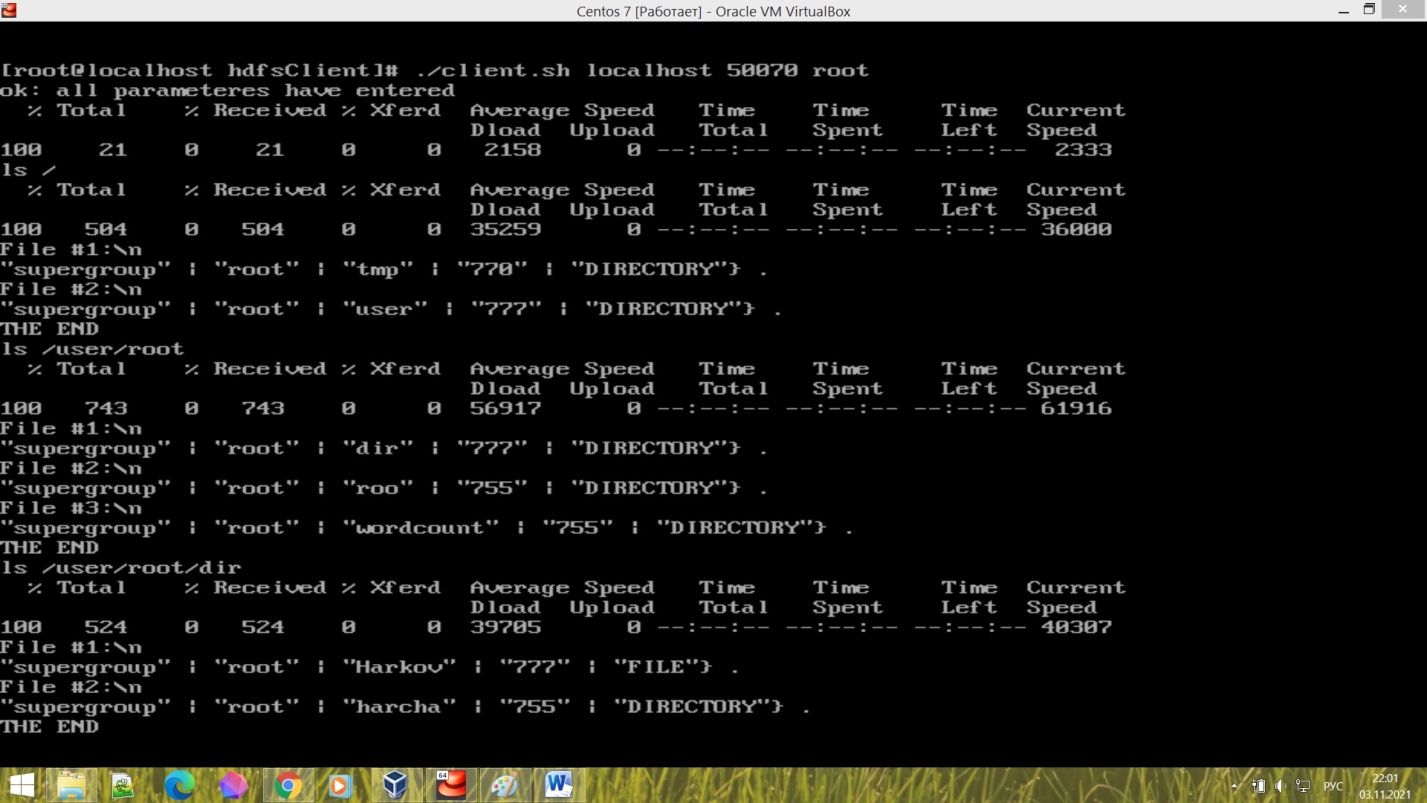


Рисунок 2 — Тестирование написанной функции клиента *«LS()»*

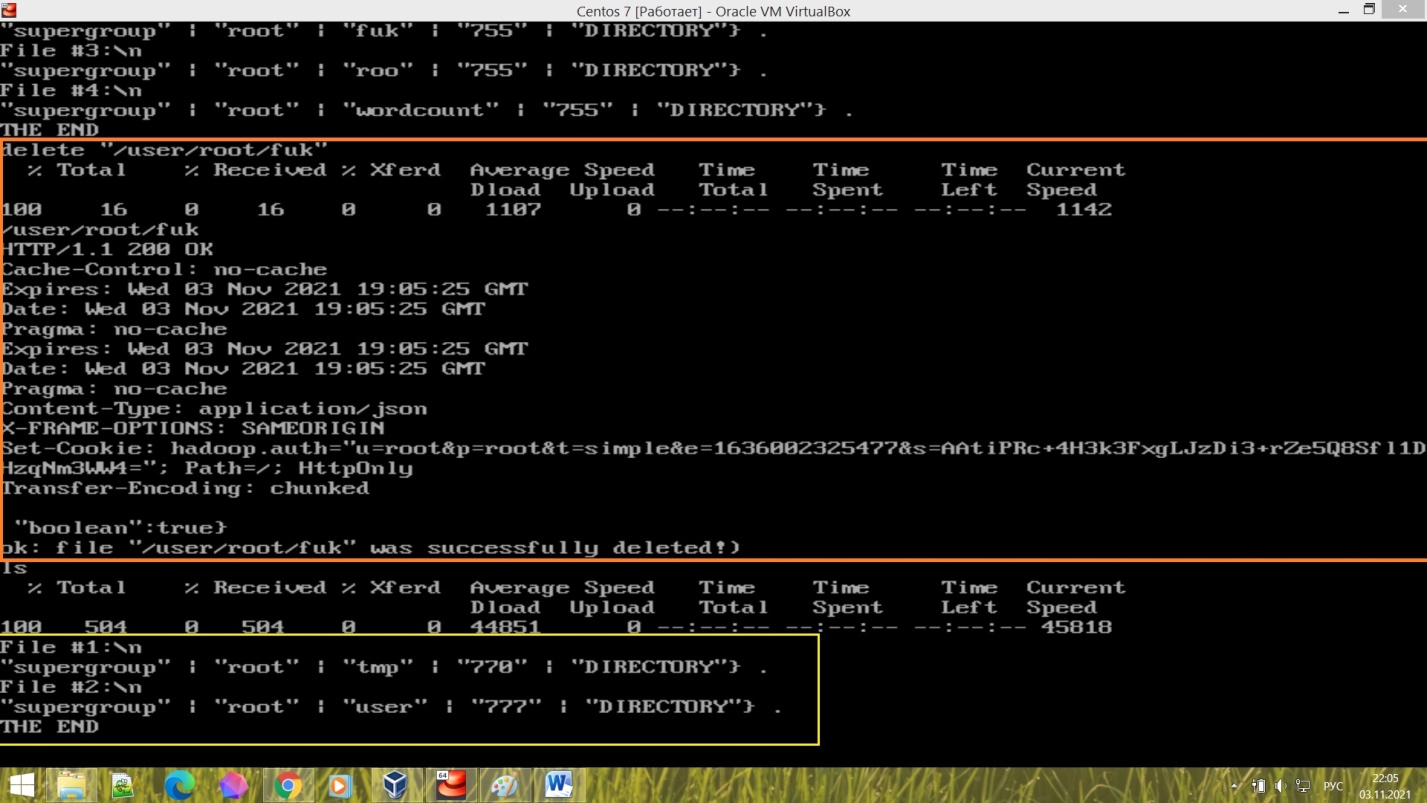


Рисунок 3 — Команда *«DELETE»* клиента HDFS

Также успешно работают функции *«CD()»*, *«GET()»* (см. рисунок 4), *«PUT()»*, *«LCD()»*, *«CREATE()»*, *«DELETE()»* (см. рисунок выше) и другие.

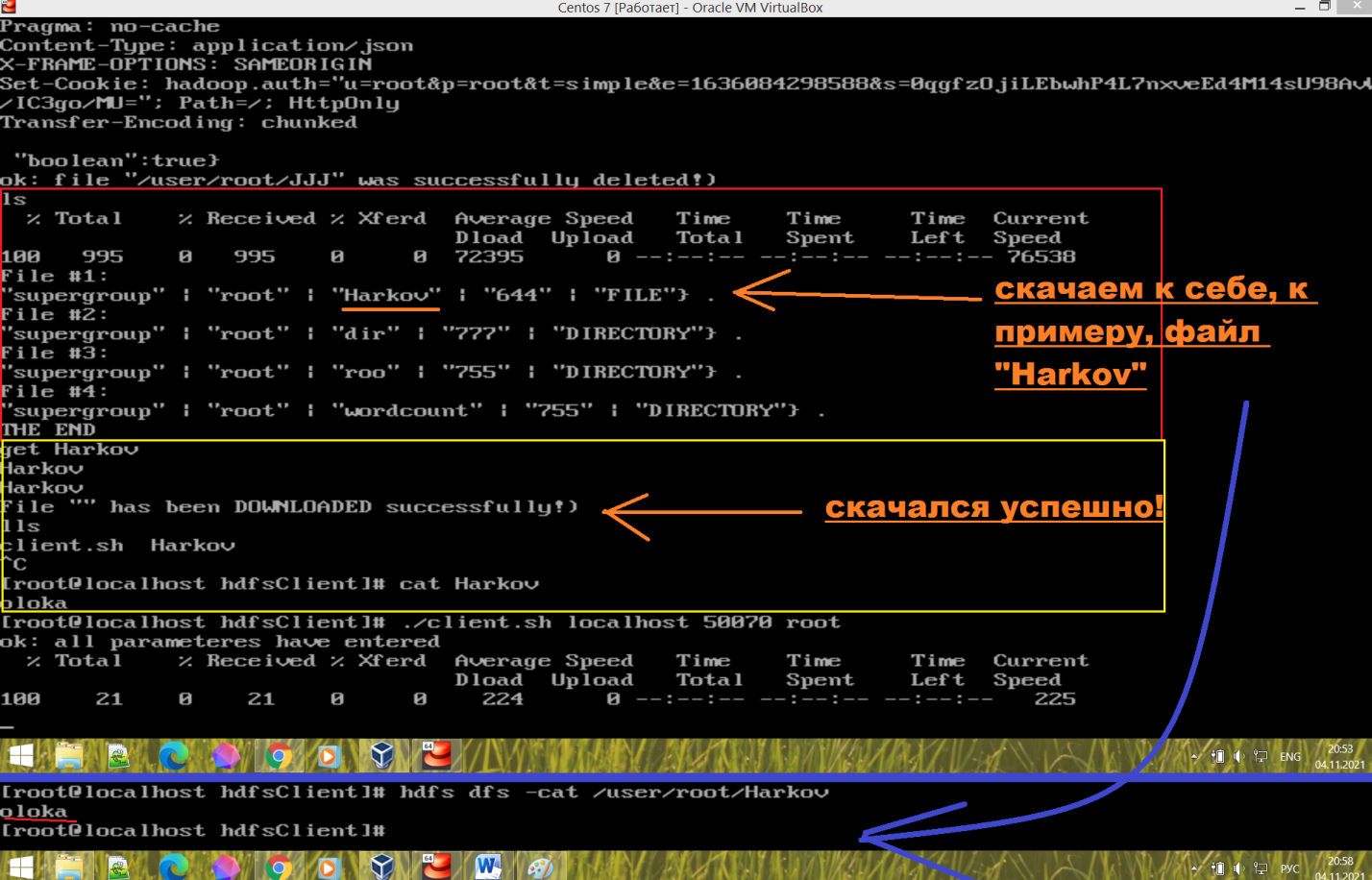


Рисунок 4 — bash-функция *«GET()»* в работе

# Список использованной литературы

1. likegeeks.com // LINUX Shell scripting step by step tutorial URL: https://likegeeks.com/bash-script-easy-guide/ (дата обращения: 27.10.2021).
2. Блог компании RUVDS.com — Bash-скрипты: начало // ХАБР URL: https://habr.com/ru/company/ruvds/blog/325522/ (дата обращения: 27.10.2021).
3. Лебедев А.С. Методы Big Data [Электронный ресурс]: Учебно-методическое пособие / Лебедев А.С., Магомедов Ш.Г. – М.: МИРЭА – Российский технологический университет, 2021. – 1 электрон. опт. диск (CD-ROM).
4. Лекция 2 — HDFS // Лебедев Артём Сергеевич — Big Data (РТУ МИРЭА) URL: https://online-edu.mirea.ru/pluginfile.php?file=%2F795313%2Fmod\_resource%2Fcontent%2F1%2F%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D1%8F%202.%20%D0%A0%D0%B0%D1%81%D0%BF%D1%80%D0%B5%D0%B4%D0%B5%D0%BB%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F%20%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2%D0%B0%D1%8F%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20HDFS.pdf (дата обращения: 26.10.2021).
5. WebHDFS REST API // Hadoop URL: https://hadoop.apache.org/docs/r1.0.4/webhdfs.html (дата обращения: 03.11.2021).

# Приложение

