



BTL HDH nhóm 6 - bài tập lớn đề tài hadoop

hệ điều hành (Học viện Công nghệ Bưu chính Viễn thông)



Scan to open on Studocu

Học Viện Công Nghệ Bưu Chính Viễn Thông
Khoa An Toàn Thông Tin



Báo cáo An toàn Hệ Điều Hành

Đề tài 6: Tìm hiểu nền tảng tính toán phân tán Hadoop: Giới thiệu, kiến trúc, các thành phần, cơ chế hoạt động, cài đặt, ưu và nhược điểm.

Nhóm thực hiện: Nhóm 6

Các Thành viên:

Đoàn Thị Hà My	msv: B21DCAT134
Vũ Thành Long	msv: B21DCAT012
Phạm Lê Nam	msv: B21DCAT142
Đỗ Thị Phương	msv: B21DCAT150
Mai Xuân Nhật	msv: B21DCAT147

Giảng viên hướng dẫn: Thầy Hoàng Xuân Diệu

Chương 1: Giới thiệu đề tài

1.1 Giới thiệu:

Tên đề tài: “ Tìm hiểu nền tảng tính toán phân tán Hadoop: Giới thiệu, kiến trúc, các thành phần, cơ chế hoạt động, cài đặt, ưu và nhược điểm.”

Sơ lược: Trong đề tài này, nhóm sẽ thực hiện tìm hiểu các kiến trúc cơ bản, các thành phần, cơ chế hoạt động, ưu nhược điểm khi sử dụng nền tảng Hadoop. Sau đó nhóm tiến hành cài đặt nền tảng.

1.2 Lý do thực hiện đề tài:

Trong thời đại chúng ta đang sống, ngành công nghệ máy tính phát triển một cách nhanh chóng . Việc sử dụng máy tính và các tài nguyên trực tuyến để xử lý công việc, giải trí... hiện nay đã dần trở nên phổ biến. Theo ước tính đến nay, đã có 79% dân số Việt Nam sử dụng Internet. Lượng dữ liệu mỗi ngày cần xử lý và quản lý là rất lớn.

Với việc khối lượng dữ liệu của một hệ thống gia tăng tới một mức độ nhất định, thì việc hệ thống sẽ phải đối mặt với thách thức phải làm sao để lưu trữ và phân tích dữ liệu. Để đối mặt với những thách thức này, các hệ thống tính toán phân tán trở thành một phần quan trọng của cơ sở hạ tầng công nghệ thông tin. Hadoop là một trong số những nền tảng này.

Đề tài sẽ khám phá sâu rộng về Hadoop, một hệ thống mã nguồn mở được thiết kế để xử lý và lưu trữ lượng lớn dữ liệu trên nền tảng phân tán. Nhóm sẽ tìm hiểu về kiến trúc của Hadoop, cách nó quản lý dữ liệu, cơ chế hoạt động của nó và đưa ra những đánh giá về ưu nhược điểm của nền tảng tính toán Hadoop.

Chương 2: Khái niệm

2.1 Khái niệm

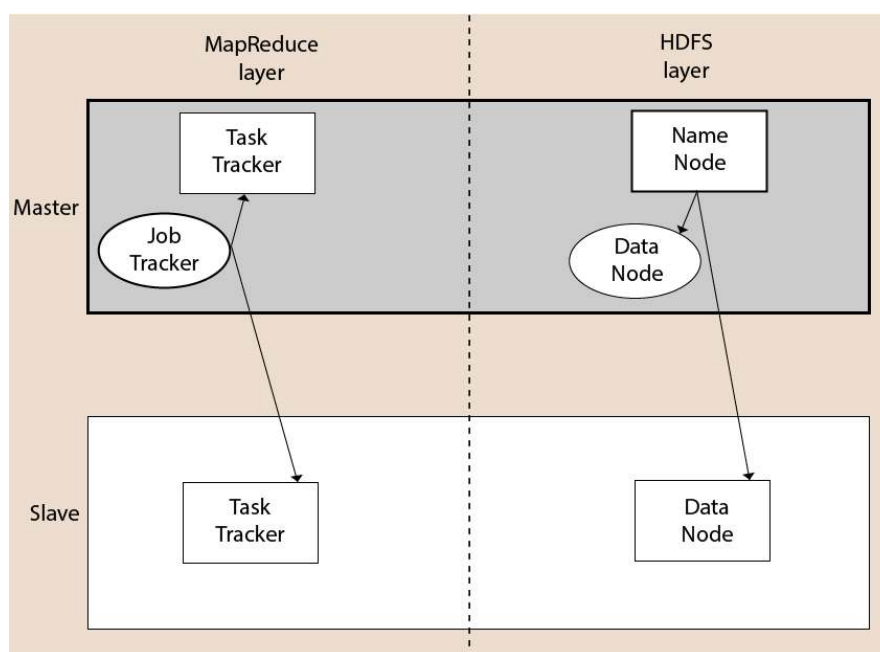
- **Hadoop** là một Apache framework mã nguồn mở cho phép phát triển các ứng dụng phân tán (distributed processing) để lưu trữ và quản lý các tập dữ liệu lớn.
- Hadoop hoạt động trên Thuật toán lập trình **MapReduce**, mô hình mà ứng dụng sẽ được chia nhỏ ra thành nhiều phân đoạn khác nhau được chạy song song trên nhiều node khác nhau.

2.2 Kiến trúc và các thành phần của hadoop

2.2.1 Kiến trúc

Một cụm Hadoop nhỏ gồm 1 master node và nhiều slave node. Toàn bộ cụm chứa 2 lớp, một lớp MapReduce Layer và lớp kia là HDFS Layer

- Master node: JobTracker, TaskTracker, NameNode, DataNode.
- Slave node: DataNode, và TaskTracker.



Hình 1: Một cụm Hadoop

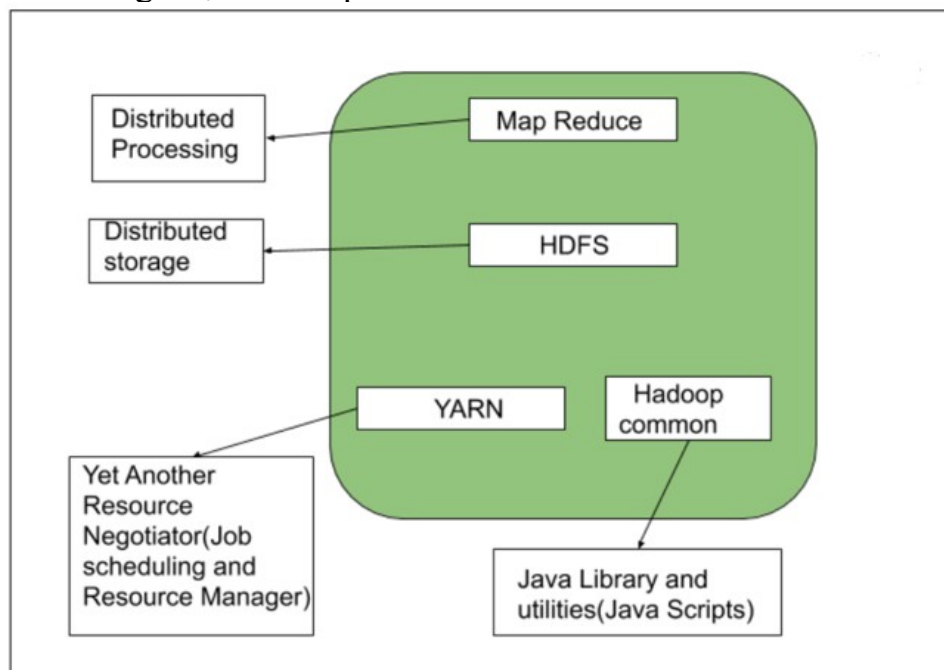
2.2.2 Các thành phần

Hadoop được chia làm 4 lớp đặc biệt:

1. MapReduce
2. HDFS

3. YARN

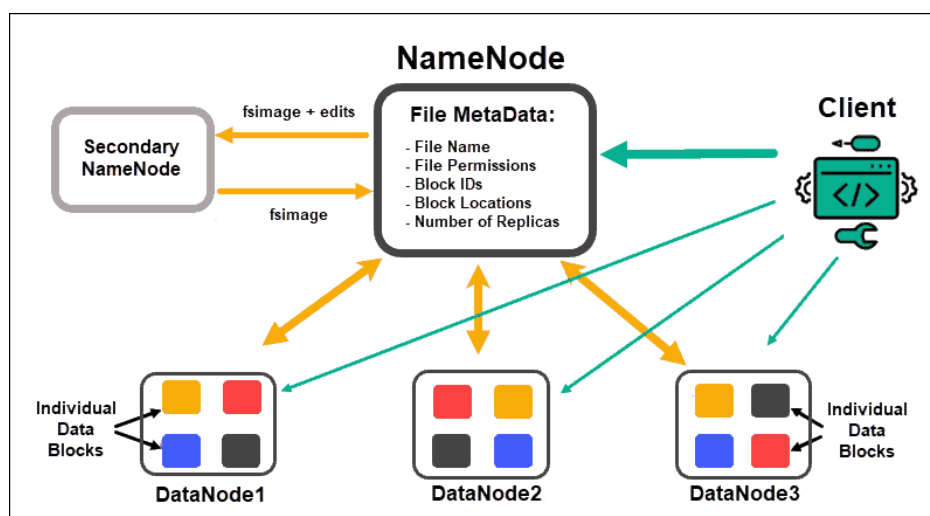
4. Tiện ích chung hoặc Hadoop Common



Hình 2: Các lớp kiến trúc Hadoop

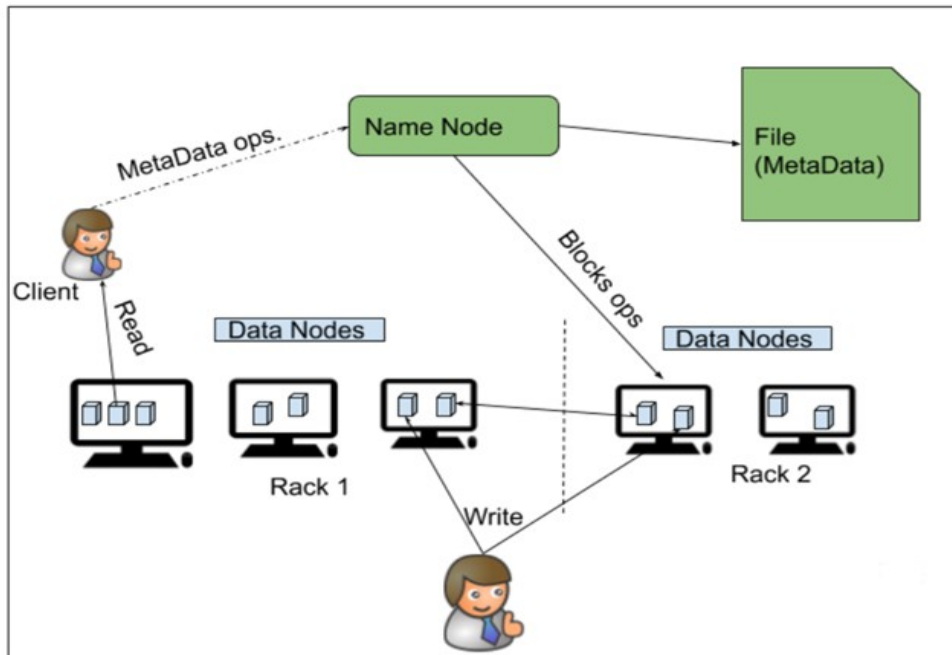
a. Hadoop Distributed File System (HDFS)

- Đây là hệ thống file phân tán cung cấp truy cập thông lượng cao cho ứng dụng khai thác dữ liệu.
- HDFS trong Hadoop cung cấp khả năng chịu lỗi và tính sẵn sàng cao cho lớp lưu trữ và các thiết bị khác có trong cụm Hadoop đó.
- Các nút lưu trữ dữ liệu trong HDFS:
 - NameNode (Master)
 - DataNode (Slave)



Hình 3:

DataNodes xử lý và lưu trữ các khối dữ liệu, trong khi **NameNodes** quản lý nhiều DataNode, duy trì siêu dữ liệu khối dữ liệu và kiểm soát quyền truy cập của máy khách.



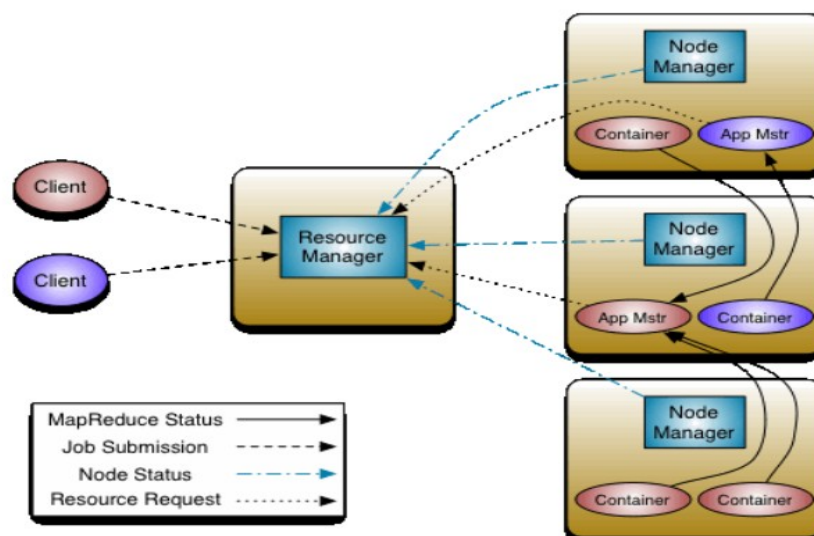
Hình 4: Kiến trúc HDFS

Một tập tin với định dạng HDFS được chia thành nhiều khối và những khối này được lưu trữ trong một tập các DataNodes.

→ NameNode định nghĩa ánh xạ từ các khối đến các DataNode → Các DataNode điều hành các tác vụ đọc và ghi dữ liệu lên hệ thống file. Chúng cũng quản lý việc tạo, huỷ, và nhân rộng các khối thông qua các chỉ thị từ NameNode.

b. Hadoop YARN

- Ý tưởng cơ bản của YARN là chia nhỏ các chức năng quản lý tài nguyên và lập kế hoạch / giám sát công việc thành các daemon riêng biệt.
- Gồm: + ResourceManager (toàn cầu)
+ ApplicationMaster (mỗi ứng dụng)



- ResourceManager và NodeManager tạo thành khung tính toán dữ liệu.

Trong đó:

- ResourceManager là cơ quan cuối cùng phân xử tài nguyên giữa tất cả các ứng dụng trong hệ thống.
- NodeManager là tác nhân khung trên mỗi máy chịu trách nhiệm về các container, giám sát việc sử dụng tài nguyên của chúng (cpu, bộ nhớ, đĩa, mạng) và báo cáo tương tự cho ResourceManager / Scheduler.
- ResourceManager có hai thành phần chính:
 - + Scheduler: chịu trách nhiệm phân bổ tài nguyên cho các ứng dụng đang chạy khác nhau tùy thuộc vào các ràng buộc quen thuộc về dung lượng, hàng đợi,...
 - + ApplicationsManager: chịu trách nhiệm chấp nhận đệ trình công việc, thương lượng container đầu tiên để thực thi ứng dụng cụ thể ApplicationMaster và cung cấp dịch vụ khởi động lại vùng chứa ApplicationMaster khi thất bại.

c. Hadoop MapReduce

- MapReduce là một thành phần thiết yếu của kiến trúc Hadoop được thiết kế để xử lý và phân tích một lượng lớn dữ liệu theo kiểu phân tán và song song.
- MapReduce chia các nhiệm vụ thành hai giai đoạn chính: **Map** và **Reduce**.

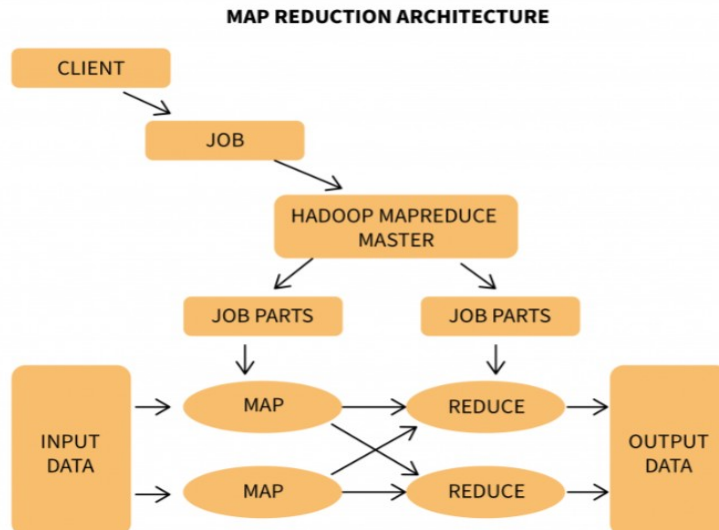
Trong đó:

Nhiệm vụ Map:

- RecordReader
- Map
- Combiner
- Partitioner

Nhiệm vụ Reduce:

- Shuffle and Sort
- Reduce
- OutputFormat



Hình 5: Quá trình MapReduce

MapReduce liên quan đến một máy khách gửi công việc đến trình quản lý Hadoop MapReduce → Công việc sau đó được chia thành các phần công việc nhỏ hơn bởi Hadoop MapReduce Master → Dữ liệu đầu vào được xử lý thông qua các hàm Map() và Reduce(), dẫn đến dữ liệu đầu ra → Hàm Map chia nhỏ dữ liệu thành các cặp khóa-giá trị, sau đó được xử lý thêm bởi hàm Reduce.

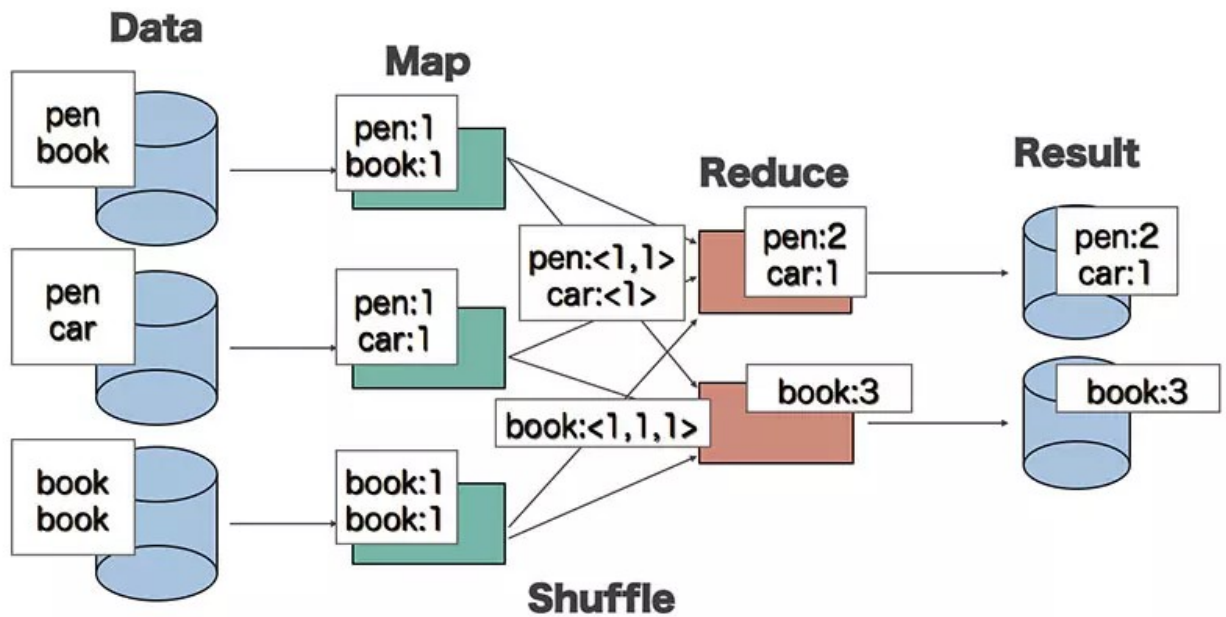
d. Hadoop Common

Đây là các thư viện và tiện ích cần thiết của Java để các module khác sử dụng. Những thư viện này cung cấp hệ thống file và lớp OS trừu tượng, đồng thời chứa các mã lệnh Java để khởi động Hadoop.

Chương 3: Cơ chế hoạt động

3.1 Quá trình hoạt động:

Quá trình hoạt động của Hadoop gồm 3 giai đoạn:



Quy trình hoạt động của Hadoop (Cho vào slide)

Giai đoạn 1:

- Một user hay một ứng dụng sẽ submit một job lên hệ thống Hadoop (hadoop job client) cùng với những yêu cầu xử lý các thông tin cơ bản gồm:

- Truyền dữ liệu lên máy chủ (input) để bắt đầu phân tán dữ liệu và xuất kết quả (output).
- Các dữ liệu sẽ được xử lý thông qua 2 hàm chính là map và reduce. Trong đó:
 - Map có chức năng quét qua toàn bộ dữ liệu và phân tán chúng thành các dữ liệu con.
 - Reduce có chức năng thu thập các dữ liệu còn lại và sắp xếp chúng.
- Các thiết lập cụ thể liên quan đến job thông qua các thông số được truyền vào.

Giai đoạn 2:

- Hệ thống Hadoop tiến hành submit job bao gồm file jar, file thực thi và bắt đầu thiết lập lịch làm việc (JobTracker) sau đó đưa job vào hàng đợi.
- Sau khi tiếp nhận yêu cầu từ JobTracker, Máy chủ “mẹ” (master hay chính là NameNode) sẽ phân chia công việc cho các máy chủ “con” (slave hay chính là DataNode). Các máy chủ con bắt đầu thực hiện các job được giao và trả kết quả cho máy chủ “mẹ”.

Giai đoạn 3:

- TaskTrackers được nằm trên các node sẽ tiến hành thực thi cho các tác vụ MapReduce để có thể trả về được những kết quả thuộc dạng output đã được lưu trữ ngay trong hệ thống file. Khi thực hiện chạy Hadoop thì cũng đồng nghĩa là đang chạy các tập trình nền hoặc những chương trình thuộc dạng thường trú khác ngay trên các máy chủ khác nhau nằm trên mạng của mình. Các trình nền này thường sẽ có những vai trò cụ thể nên những chỉ số tồn tại trên máy chủ có thể tồn tại ở nhiều máy chủ khác nhau. Một số daemon (Daemon là gì? Daemon hoặc Disk And Execution MONitor là một chương trình chạy như một tiến trình nền chứ không phải là một tiến trình tương tác) khác bao gồm:
 - NameNode
 - DataNode
 - SecondaryNameNode
 - JobTracker
 - TaskTracker

Chương 4: Cách cài đặt Hadoop

4.1 Cài đặt hadoop:

- Bước 1: cài đặt java:

- `sudo apt-get install default-jdk`

khi cài đặt hoàn tất ta kiểm tra version của java

- `java -version`

```
(VietnamKali) [ ~ ]
$ java -version
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
openjdk version "17.0.6" 2023-01-17
OpenJDK Runtime Environment (build 17.0.6+10-Debian-1)
OpenJDK 64-Bit Server VM (build 17.0.6+10-Debian-1, mixed mode, sharing)
```

- Bước 2: cài đặt hadoop

Tải hadoop:

wget <https://dlcdn.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz>

```
(victim@kali)-[~]
$ wget https://dlcdn.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz
--2024-02-08 22:15:17-- https://dlcdn.apache.org/hadoop/common/hadoop-3.3.6/hadoop-3.3.6.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org) ... 151.101.2.132, 2a04:4e42::64
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 730107476 (696M) [application/x-gzip]
Saving to: 'hadoop-3.3.6.tar.gz'

hadoop-3.3.6.tar.gz 100%[====>] 696.28M 4.06MB/s in 2m 51s

2024-02-08 22:18:08 (4.07 MB/s) - 'hadoop-3.3.6.tar.gz' saved [730107476/730107476]
```

Giải nén file:

tar -xzvf hadoop-3.3.6.tar.gz:

```
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.3.3.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Null.java
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_2.8.3.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.3.5.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_2.8.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.0.3.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/hadoop-hdfs_0.22.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_2.9.1.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.1.1.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/hadoop-hdfs_0.20.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.0.0-alpha4.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.2.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_2.9.2.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.0.0-alpha2.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.0.2.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_2.10.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.1.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.0.1.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.2.1.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.2.4.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/hadoop-hdfs_0.21.0.xml
hadoop-3.3.6/share/hadoop/hdfs/jdiff/Apache_Hadoop_HDFS_3.1.3.xml
hadoop-3.3.6/share/hadoop/hdfs/hadoop-hdfs-client-3.3.6-tests.jar
hadoop-3.3.6/share/hadoop/hdfs/hadoop-hdfs-httpfs-3.3.6.jar
```

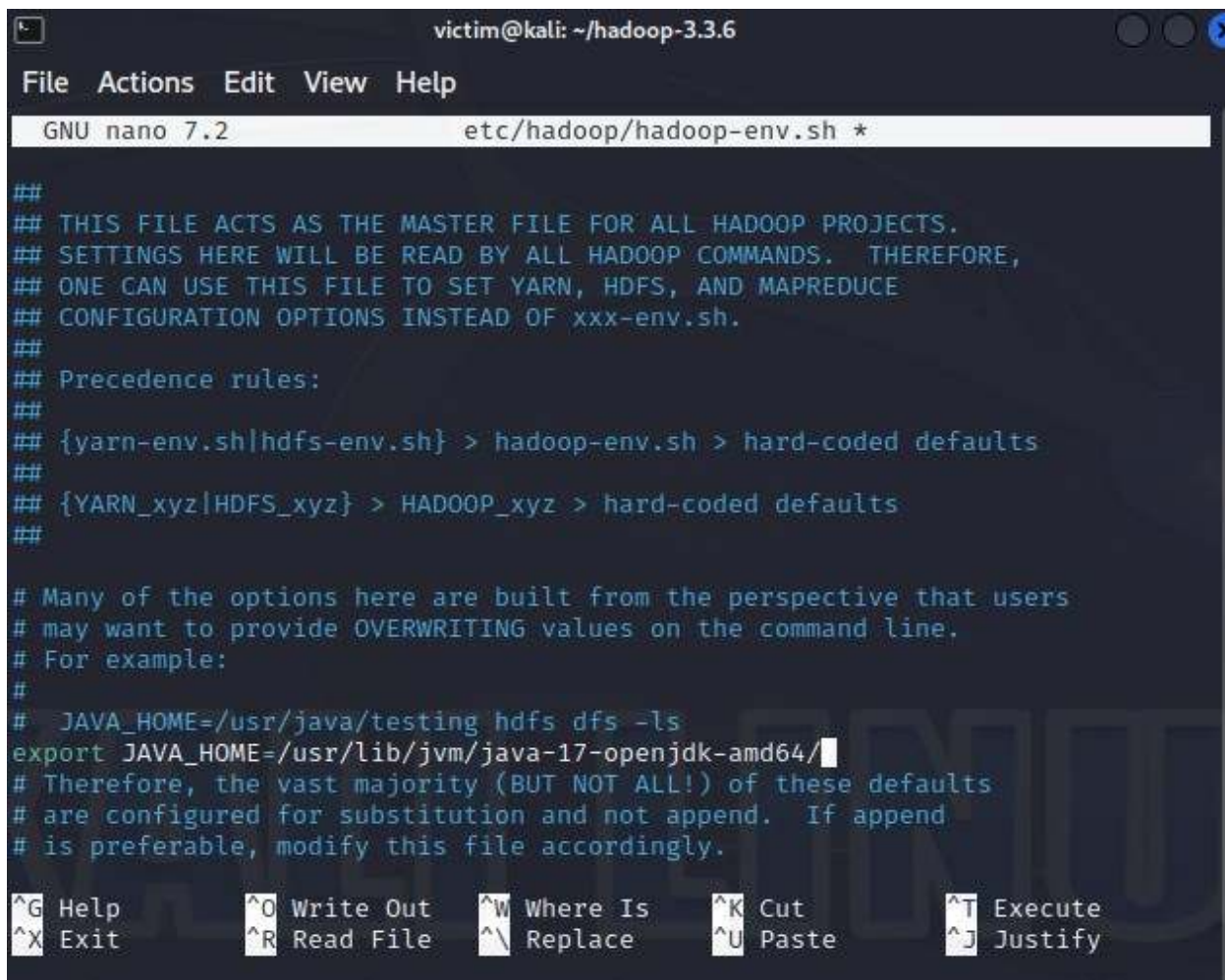
- **Bước 3:** cấu hình biến môi trường JAVA_HOME

Tìm đường dẫn mặc định của Java path:

readlink -f /usr/bin/java | sed "s:bin/java::"


```
(victim@kali)-[~]  
$ readlink -f /usr/bin/java | sed "s:bin/java::"  
/usr/lib/jvm/java-17-openjdk-amd64/
```

Đặt giá trị JAVA_HOME trong file hadoop-env.sh trở vào đường dẫn trên



```
victim@kali: ~/hadoop-3.3.6  
File Actions Edit View Help  
GNU nano 7.2 etc/hadoop/hadoop-env.sh *  
##  
## THIS FILE ACTS AS THE MASTER FILE FOR ALL HADOOP PROJECTS.  
## SETTINGS HERE WILL BE READ BY ALL HADOOP COMMANDS. THEREFORE,  
## ONE CAN USE THIS FILE TO SET YARN, HDFS, AND MAPREDUCE  
## CONFIGURATION OPTIONS INSTEAD OF xxx-env.sh.  
##  
## Precedence rules:  
##  
## {yarn-env.sh|hdfs-env.sh} > hadoop-env.sh > hard-coded defaults  
##  
## {YARN_xyz|HDFS_xyz} > HADOOP_xyz > hard-coded defaults  
##  
# Many of the options here are built from the perspective that users  
# may want to provide OVERWRITING values on the command line.  
# For example:  
#  
# JAVA_HOME=/usr/java/testing hdfs dfs -ls  
export JAVA_HOME=/usr/lib/jvm/java-17-openjdk-amd64/  
# Therefore, the vast majority (BUT NOT ALL!) of these defaults  
# are configured for substitution and not append. If append  
# is preferable, modify this file accordingly.  
^G Help      ^O Write Out  ^W Where Is   ^K Cut        ^T Execute  
^X Exit      ^R Read File  ^\ Replace    ^U Paste      ^J Justify
```

- **Bước 4:** chạy thử hadoop:

Vào thư mục bin của hadoop và gõ dòng lệnh

./hadoop

```
(victim@kali)-[~/hadoop-3.3.6/bin]
$ ./hadoop
Usage: hadoop [OPTIONS] SUBCOMMAND [SUBCOMMAND OPTIONS]
or      hadoop [OPTIONS] CLASSNAME [CLASSNAME OPTIONS]
where CLASSNAME is a user-provided Java class

OPTIONS is none or any of:

buildpaths          attempt to add class files from build tree
--config dir        Hadoop config directory
--debug            turn on shell script debug mode
--help             usage information
hostnames list[,of,host,names] hosts to use in worker mode
hosts filename      list of hosts to use in worker mode
loglevel level      set the log4j level for this command
workers            turn on worker mode

SUBCOMMAND is one of:

Admin Commands:

daemonlog           get/set the log level for each daemon
```

Bảng hướng dẫn hiện ra tức là ta đã cài đặt thành công hadoop trên máy

Chương 5: Phân tích ưu nhược điểm của nền tảng

5.1 Các ưu điểm của Hadoop:

- Khả năng mở rộng cao: Hadoop là một nền tảng lưu trữ có khả năng mở rộng cao bởi vì nó có thể lưu trữ và phân phối các bộ dữ liệu rất lớn trên hàng trăm máy chủ hoạt động song song. Không giống như các hệ thống cơ sở dữ liệu quan hệ truyền thống (RDBMS) không thể mở rộng xử lý lượng lớn dữ liệu, Hadoop cho phép các doanh nghiệp chạy các ứng dụng trên hàng ngàn node liên quan tới hàng ngàn Terabyte dữ liệu.
- Chi phí hiệu quả: Hadoop cũng cung cấp một giải pháp lưu trữ hiệu quả về chi phí cho các dữ liệu của doanh nghiệp. Vấn đề với các hệ thống quản lý cơ sở dữ liệu quan hệ truyền thống là nó được đầu tư tối đa chi phí để mở rộng tới một mức độ để xử lý khối lượng lớn dữ liệu. Để giảm chi phí, nhiều công ty sẽ phải phân loại dữ liệu thành các dạng mẫu dựa trên một số gải định khi dữ liệu đó có

giá trị nhất. Các dữ liệu thô sẽ được xóa bỏ vì nó tạo nên gánh nặng chi phí. Thêm vào đó, phương pháp này chỉ có thể đáp ứng trong ngắn hạn. Điều đó có nghĩa khi các ưu tiên kinh doanh thay đổi, các dữ liệu thô hoàn chỉnh cần thiết không có sẵn vì nó quá đắt để lưu trữ. Ngược lại, Hadoop được thiết kế như một kiến trúc quy mô có chi phí thấp nhưng có thể lưu trữ tất cả các dữ liệu của công ty để sử dụng sau. Hadoop giúp tiết kiệm chi phí đáng kinh ngạc: thay vì chi phí hàng ngàn đến hàng chục ngàn \$ mỗi Terabyte, Hadoop cung cấp khả năng tính toán và lưu trữ mỗi Terabyte chỉ với vài trăm \$.

- **Linh hoạt:** Hadoop cho phép các doanh nghiệp dễ dàng tiếp cận các nguồn dữ liệu mới và khai thác các loại dữ liệu khác nhau(cả hai đều có cấu trúc và phi cấu trúc) để tạo ra giá trị từ dữ liệu đó. Điều này có nghĩa các doanh nghiệp có thể sử dụng Hadoop để thu được những hiểu biết kinh doanh có giá trị từ dữ liệu đó. Điều đó có nghĩa các doanh nghiệp có thể sử dụng Hadoop để thu được những hiểu biết kinh doanh có giá trị từ các nguồn dữ liệu như phương tiện truyền thông xã hội, các cuộc hội thoại email hoặc dữ liệu kích chuột, thống kê sở thích, hoạt động của người dùng. Ngoài ra Hadoop có thể được sử dụng cho nhiều mục đích, chẳng hạn như xử lý đăng nhập, các hệ thống được đề nghị, kho dữ liệu phân tích chiến dịch thị trường và phát hiện gian lận.

- **Nhanh:** Phương pháp lưu trữ độc đáo của Hadoop dựa trên một hệ thống tập tin phân phối theo các “bản đồ” dữ liệu. Các công cụ xử lý dữ liệu thường nằm trên cùng một máy chủ với dữ liệu nên kết quả xử lý dữ liệu nhanh hơn nhiều. Nếu chúng ta đang làm việc với khối dữ liệu lớn phi cấu trúc, Hadoop có thể xử lý hiệu quả mỗi terabyte dữ liệu chỉ trong vài phút và mỗi petabyte trong vài giờ.

- **Khả năng chịu lỗi:** Khi dữ liệu được gửi đến một nút nào đó thì dữ liệu cũng được nhân rộng đến các node khác trong cluster. Vì vậy, trong trường hợp thất bại thì luôn có một bản sao có sẵn để sử dụng. Kiến trúc của Hadoop cung cấp cơ chế bảo vệ dữ liệu ở cả duy nhất một node và nhiều node. Khi nói đến xử lý dữ liệu một cách an toàn và chi phí hiệu quả. Hadoop có lợi thế hơn các hệ thống quản lý cơ sở dữ liệu quan hệ và giá trị của nó cho một doanh nghiệp sẽ tiếp tục tăng khi dữ liệu phi cấu trúc ngày càng nhiều thêm.

5.2 Nhược điểm và giới hạn của Hadoop:

- Hadoop không có mô hình bảo mật và tính phức tạp cao
- Hadoop không cung cấp lưu trữ hoặc mã hóa dữ liệu mức mạng, trong khi đây là mối quan tâm rất lớn đối với dữ liệu ứng dụng đối tượng khách hàng cần độ bảo mật thông tin như chính phủ, ngân hàng...
- HDFS không hiệu quả để xử lý các tập tin nhỏ
- MapReduce không thích hợp để sử dụng các trường hợp có nhu cầu truy cập dữ liệu thời gian thực
- MapReduce khó biểu diễn dữ liệu đầu ra theo nhu cầu cần sử dụng.

Chú thích tài liệu tham khảo:

1. <https://hadoop.apache.org/>
2. International Journal of Innovative Research in Computer Science & Technology (IJIRCST):
<https://www.ijircst.org/DOC/20-a-review-paper-on-hadoop-architecture.pdf>
3. [Hadoop là gì? Tìm hiểu chi tiết về công cụ phân tích Big Data tốt nhất thế giới \(tino.org\)](#)
4. [Cấu trúc và thành phần của công nghệ HADOOP ! \(itnavi.com.vn\)](#)
5. <https://lanit.com.vn/hadoop-la-gi.html>
6. <https://viblo.asia/p/tim-hieu-ve-hadoop-bWrZn1XwKxw>
7. [How Hadoop Works - Understand the Working of Hadoop - TechVidvan](#)