Hadoop File System

HDFS Basics

Hadoop File System or HDFS is a distributed file system that resides on top of the filesystems in of the compute nodes forming a Hadoop cluster.

HDFS has the following properties:

1. Distributed file storage. All data stored on HDFS is accessible from all Hadoop nodes.
2. Optimized for very large file storage. HDFS stores files in blocks of 64 MB. This means that a single disk read operation can bring 64 Mb of data directly to a compute node.
3. Support for write-once, read-many access parttern. HDFS is largely designed for the following pattern of use:
   - A data file is uploaded to HDFS once.
   - A large number of analytical tasks (individual MapReduce jobs) is performed using this file as input data.
4. Support for commodity hardware. HDFS assumes that it runs on commodity hardware, which has a high probability of failure. To compensate, HDFS supports a variety of data replication and recovery protocols that prevent data loss in case of hard disk failures.
5. Standard POSIX interface. HDFS supports the standard POSIX file system interface. This essentially means that with minor exceptions (HDFS needs to have commands for transfer of files to/from that are not present in regular file systems), standard UNIX/Linux file system commands are supported on HDFS.

HDFS is not very good for dealing with

1. Large numbers of small files. Each file will wind up being stored in a 64MB block.
2. Multiple active writes to HDFS files. Data files on HDFS are assumed to be static. HDFS is not very good at supporting active modification of data files.

HDFS organization. By default, HDFS is organized in a way similar to how Linux file system is organized. Each Hadoop user receives their own directory:

```
hdfs:///user/<loginId>
```

or, simply

```
/user/<loginId>
```

This is the default location for all file transfers/file operations for HDFS for user `<loginId>`. For example

```
$ hadoop fs -ls
```

command that I run (as user `dekhtyar`), is equivalent to running

```
$ hadoop fs -ls /user/dekhtyar
```

or

```
$ hadoop fs -ls hdfs:///user/dekhtyar
```

Permissions. HDFS supports the standard user-group-others POSIX file access model. By default, the group is set to `supergroup` and all Hadoop users usually are its members.

Working with HDFS

Hadoop provides three command-line methods for accessing HDFS:

- `hadoop fs` command
- `hadoop dfs` command
- `hdfs dfs` command

hadoop dfs and hdfs dfs commands. The `hadoop dfs` and `hdfs dfs` commands provide command-line access to HDFS and the files stored on it. These two commands provide the same set of functionality.

hadoop fs command. The `hadoop fs` command provides interface to any file system reachable from the node on which the command is run. Specifically, in addition to HDFS, `hadoop fs` can access files from the local file system.

Below, we use `hadoop fs` to represent the syntax of HDFS commands. The syntax of the other two commands is similar.

General file system access command format. The general format of an HDFS access command is:

```
$ hadoop fs -<command> [<arguments>]
```

Here, <command> is the file system access command, and <arguments> are the optional arguments to each command.

File system access commands.

HDFS supports the following file system access commands. (This is not a full list, but rather a list of most important commands.)

Command	Meaning
-help	help message, instructions on use of commands
-usage	display information about the usage of a specific command
-ls	display the lists of files/directories
-put, -copyFromLocal	copy file from local file system to HDFS
-get, -copyToLocal	copy file from HDFS to local file system
-moveFromLocal	move file from local file system to HDFS
-moveToLocal	move file from HDFS to local fils system
-mkdir	create a directory
-rmdir	remove a directory
-cp	copy files
-mv	move files
-rm	delete (remove) files
-touchz	create a zero length file
-chmod	change file access permissions
-chgroup	change file group
-chown	change file owner
-cat	display contents of file(s)
-text	output the contents of a file as text
-tail	display the last 1Kb of the file
-du	show file system usage statistics
-df	show free space on the file system

Viewing directory structure and files. To see what is in a specific HDFS directory, use the -ls command.

```
$ hadoop fs -ls <hdfsPath>
```

For example

```
$ hadoop fs -ls test/
```

shows the list of files and directories in the test directory located in the home directory of the current user.

A sample output may be:

```
dekhtyar@cslvm31:~/369/lab6$ hadoop fs -ls test/
Found 5 items
-rw-r--r--   2 dekhtyar supergroup         83 2016-02-04 14:59 test/data
drwxr-xr-x   - dekhtyar supergroup          0 2016-02-05 12:03 test/grep
drwxr-xr-x   - dekhtyar supergroup          0 2016-02-09 17:33 test/out01
drwxr-xr-x   - dekhtyar supergroup          0 2016-02-09 17:09 test/output
-rw-r--r--   2 dekhtyar supergroup       3302 2016-02-04 20:00 test/wc.jar
```

HDFS supports the `-ls -R` flag, which recursively lists all subdirectories.

```
dekhtyar@cslvm31:~/369/lab6$ hadoop fs -ls -R test/
-rw-r--r--   2 dekhtyar supergroup         83 2016-02-04 14:59 test/data
drwxr-xr-x   - dekhtyar supergroup          0 2016-02-05 12:03 test/grep
-rw-r--r--   2 dekhtyar supergroup          0 2016-02-05 12:03 test/grep/_SUCCESS
-rw-r--r--   2 dekhtyar supergroup          7 2016-02-05 12:03 test/grep/part-r-00000
drwxr-xr-x   - dekhtyar supergroup          0 2016-02-09 17:33 test/out01
-rw-r--r--   2 dekhtyar supergroup          0 2016-02-09 17:33 test/out01/_SUCCESS
-rw-r--r--   2 dekhtyar supergroup        114 2016-02-09 17:33 test/out01/part-r-00000
drwxr-xr-x   - dekhtyar supergroup          0 2016-02-09 17:09 test/output
-rw-r--r--   2 dekhtyar supergroup          0 2016-02-09 17:09 test/output/_SUCCESS
-rw-r--r--   2 dekhtyar supergroup         94 2016-02-09 17:09 test/output/part-r-00000
-rw-r--r--   2 dekhtyar supergroup       3302 2016-02-04 20:00 test/wc.jar
```

To view the contents of the file you can issue one of the following commands:

```
$ hadoop fs -cat <hdfsFile>
```

or

```
$ hadoop fs -text <hdfsFile>
```

To view only the end of a large file, use

```
$ hadoop fs -tail <hdfsFile>
```

Copying files. To put a file (or files) onto HDFS from a local system, use `-put`:

```
$ hadoop fs -put <localSource> <hdfsDestination>
```

Here, `<localSource>` is the file access path/pattern (can include wildcards) on the local system, and `<hdfsDestination>` is a destination (must be a directory if `<localSource>` matches multiple files) on HDFS, where the file(s) shall be uploaded.

For example,

```
$ hadoop fs -put data .
```

copies the file `data` from the current directory of the local filesystem to the home directory of the current user of HDFS.

To copy a file (or files) from HDFS to a local file system use `-get`:

```
$ hadoop fs -put <hdfsSource> <localDestination>
```

Here, `<hdfsSource>` is the file access path/pattern (can include wildcards) on the HDFS and `<localDestination>` is a destination (must be a directory if `<hdfsSource>` matches multiple files) on HDFS, where the file(s) shall be uploaded.

For example,

```
$ hadoop fs -get test/output/part-r-00000 .
```

copies the file part-r-00000 residing in /user/<loginId>/test/output directory into the current directory on the local file system.

Using -moveFromLocal instead of -put and -moveToLocal instead of -get erases the source file/files after they have been successfully transferred to the new destination.

hadoop fs -cp can be used to copy files within HDFS, as well as copy files between different file systems.

```
$ hadoop fs -cp foo bar
```

copies file foo on HDFS (/user/<loginId>/foo to a new file in the same directory named bar.

```
$ hadoop fs -cp file:///home/<loginId>/foo hdfs:///user/<loginId/
```

copies file foo from the home directory of the user <loginId> on local file system to HDFS. The inverse can be done using the following command:

```
$ hadoop fs -cp hdfs:///user/<loginId>/foo file:///home/<loginId/
```

hadoop fs -mv works the same way, only it removes the source file after the successful transfer.

Directory operations. Simple directory management is the same as in Linux.

To create a new HDFS directory:

```
$ hadoop fs -mkdir <hdfsDirectory>
```

To remove an empty HDFS directory:

```
$ hadoop fs -rmdir <hdfsDirectory>
```