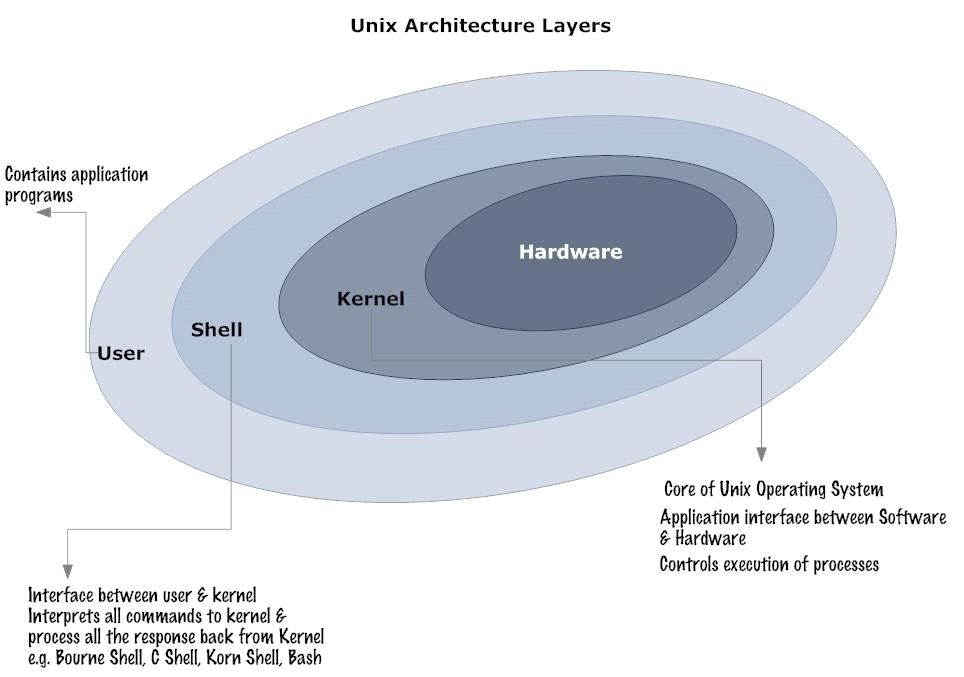
**LINUX – Hands on**

UNIX is a computer Operating System which is capable of handling activities from multiple users at the same time performing multitasking of programs. UNIX was originated around in 1969 at AT&T Bell Labs by Ken Thompson and Dennis Ritchie.

**Linux Architecture**



**Architecture Components**

* **Kernel:** The kernel is the heart of the operating system. It interacts with hardware and most of the tasks like memory management, task scheduling and file management.
* **Shell:** The shell is the utility that processes your requests. When you type in a command at your terminal, the shell interprets the command and calls the program that you want. C Shell, Bourne Shell and Korn Shell are most famous shells which are available with most of the Unix variants.
* **Commands and Utilities:** There are various command and utilities which you would use in your day to day activities. **cp, mv, cat** and **grep** etc. are few examples of commands and utilities. There are over 250 standard commands plus numerous others provided through 3rd party software. All the commands come along with various optional options.
* **Files and Directories:** All data in UNIX is organized into files. All files are organized into directories. These directories are organized into a tree-like structure called the filesystem.

**All about HADOOP**

OOP Model

Data warehouse

DBMS

Traditional files

RDBMS

BIG DATA

Distributed system

3

2

1

**BIG DATA** is a phrase or methodology defines the way of acquire, cure, store, process, analyze huge volume of heterogeneous data flow in different frequency managed in a stipulated period.

**Characteristics of Big Data:**

1. Velocity (The processing speed is high as the nature is distributed)
2. Variety (supports various kind of data like structured/semi /un structured.
3. Volume (supports huge data processing)
4. Veracity (trustable/high availability)
5. Visibility (the process is visible)
6. Value

**Tools in Big data:**

1. HADOOP
2. SPARK
3. AMAZON EMR
4. Teradata aster
5. Oracle BD appliance
6. MS HD insight
7. IBM Biginsight
8. Nokia - Disco/Misc

**Hadoop Intro:**

Hadoop is an open source apache framework provides distributed storage and processing on huge data sets that run on clusters of commodity hardware.

**Development Commands**

**ls** [**-AabCcdFfghikLlmnopqRrstux1**] [**-timeout** *seconds*] [**-X** *attr*] [*pathname*...]

**DESCRIPTION**

**ls** lists files and directories. If the *pathname* is a file, **ls** displays information on the file according to the requested options. If the *pathname* is a directory, **ls** displays information on the files and subdirectories therein. You may obtain information on a directory itself using the **-d** option.

If you do not specify any options, **ls** displays only the file name(s). When **ls** sends output to a pipe or a file, it writes one name per line; when it sends output to the terminal, it uses the **-C** (multi-column) format.

**Options**

**-A**

lists all entries including those starting with periods (.), but excluding any . or .. entries.

**-a**

**lists all entries including those starting with a period (.). 🡪 hidden files**

**-b**

displays non-printable characters as octal bytes with the form *\ooo*.

**-C**

puts output into columns, sorted vertically; this is the default output format to the terminal.

**-c**

uses the creation time of the file for sorting (**-t**) or displaying (**-l**).

**-d**

does not display the contents of named directories, but show information on the directories themselves.

**-F**

puts a / after each directory name, a \* after every executable file, a | after every FIFO file, a @ after every symbolic link and a = after every socket.

**Note:**

Windows systems do not support FIFO files or sockets (as files) and consider all files that are not directories or symbolic links to be executable files. This option of the **ls** utility labels the files and directories display accordingly.

**-f**

enables the **-a** option and disables the **-C**, **-g**, **-l**, **-n**, **-o**, **-r**, **-s**, and **-t** options. For each argument that is a directory, all directory entries are listed in the same order they are retrieved from the system (POSIX-compliant and UNIX systems only).

**-g**

displays only the group ID numbers (on 7/2008R2/8/2012/10/2016, POSIX-compliant and UNIX systems).

**-h**

displays file sizes using more human-friendly units. Units used are:

B Bytes

KB Kilobytes

MB Megabytes

GB Gigabytes

TB Terabytes

PB Petabytes

EB Exabytes

When this option is specified, each file size is displayed using the most appropriate unit and is rounded to two decimal places.

**-i**

displays inode numbers along with file names (only on systems that support inode numbers, such as POSIX-compliant and UNIX systems).

**-k**

displays size in kilobytes instead of blocks when specified with the **-s** option. If the **-s** option is not specified, this option has no effect.

**-L**  follows symbolic links.

**Note:**

Symbolic links are only available on 7/2008R2/8/2012/2016 systems with the NTFS file system.

**-l**

displays permissions, links, owner, group, size, time, name; see [*Long Output Format*](https://www.mkssoftware.com/docs/man1/ls.1.asp#Long_Output_Format).

**-m**

displays names in single line, with commas separating names.

**-n**

displays user ID and group IDs as numbers instead of names.

**-o**

displays only the user ID of owner (POSIX-compliant and UNIX systems only).

**-p**

puts / after directory names.

**-q**

displays non-printable characters as ?.

**-R**

lists subdirectories recursively.

**-r**

sorts in reverse of usual order; you can combine this with other options that sort the list.

**-s**

displays size in blocks (after the inode number, but before other information). If the **-k** option is also specified, the size is displayed in kilobytes instead.

**-t**

sorts entries by time. By default, this option sorts the output by the modification times of files. You can change this with the **-c** and **-u** options.

**-timeout** *seconds*

specifies the number of seconds that **ls** has to complete the operation before timing out and issuing an error. *seconds* is given in seconds.

**-u**

uses the last access time for sorting (**-t**) or displaying (**-l**).

**-X** *attr*

displays extended security attributes under Windows. This option turns on the **-l** (long listing) option. The *attr* argument is a string containing some of these characters:

|  |  |
| --- | --- |
| **Character** | **Meaning** |
| **A** | file attributes in verbose format |
| **a** | file attributes in terse format |
| **D** | verbose file Discretionary Access Control Entries (7/2008R2/8/2012/10/2016 only) |
| **d** | terse file Discretionary Access Control Entries (7/2008R2/8/2012/10/2016 only) |

If both **A** and **a** or both **D** and **d** are in *attr*, **ls** uses the last one given on the command line. See [*Extended Attribute Descriptions*](https://www.mkssoftware.com/docs/man1/ls.1.asp#Extended_Attribute_Descriptions).

**-x**

puts output into columns sorted across the rows.

**-1**

forces single column output.

**Note:**

When you specify options that are mutually exclusive (for example, **-c** and **-u**), the option that appears last on the command line is used.

**Long Output Format**

The output from **ls** **-l** summarizes all the most important information about the file on one line. If the specified *pathname* is a directory, **ls** displays information on every file in that directory (one file per line). It precedes this list with a status line that indicates the total number of file system blocks (512 byte units) occupied by the files in that directory. Here is a sample of the output along with an explanation.

-rw-rw-rw- 1 root dir 104 Dec 25 19:32 file

Eg:

Ls –lart

-rw– r-— r—- 1 hduser hduser 176 Jul 18 2013 .bash\_profile

-rw– r-— r—- 1 hduser hduser 176 Jul 18 2013 .bash\_logout

drwx rwx rwx 1 hduser hduser 176 Jul 18 2013 .themes

The first character identifies the file type:

- Regular file

b Block special file

c Character special file

d Directory

l Symbolic link

n Network file

p FIFO

s Socket

The next nine characters are in three groups of three; they describe the permissions on the file. The first group of three describes owner permissions; the second describes group permissions; the third describes other (or *world*) permissions. Because Windows systems do not support group and other permissions, these are copies of the owner's permissions. Characters that may appear are:

r Permission to read file

w Permission to write to file

x Permission to execute file

a Archive bit is on (file has not been backed up)

c Compressed file

s System file

h Hidden file

t Temporary file

On Windows systems, most of the permissions shown are artificial, with no real meaning. The w bit is set according to the ReadOnly attribute, and the rx bits are always set on.

You can change some permissions with the **[chmod](https://www.mkssoftware.com/docs/man1/chmod.1.asp)** command.

After the permissions comes the number of links to the file.

Next comes the name of the owner of the file or directory. On file systems that don't support 7/2008R2/8/2012/10/2016 security, the owner name cannot be determined and the owner ID number is displayed instead. Under 7/2008R2/8/2012/10/2016 the name of the owner of a file is displayed if the file's SIDs can be obtained and if these SIDs have an associated name in the SAM database. If the file has a SID associated with it, but the name of the SID cannot be determined, then the value of the SID is displayed. (This can happen when the current user is not in the domain that was used when the file was created.) If the file does not have a SID (for example, if it is on a non-NTFS file system), or if the file security information cannot be accessed because the file is locked by another process, then the user name appears as <unavail>.

**Note:**

When a listed file is owned by the local computer, the owner is displayed as *computer\_name*\ where *computer\_name* is the name of the local computer.

Then comes the name of the group that owns the file or directory. On Windows systems, the same rules are followed for the group name as for the owner name.

Following this is the size of the file, expressed in bytes.

After this comes a date and time. For a file, this is the time that the file was last changed; for a directory, it is the time that the directory was created. The **-c** and **-u** options can change which time value is used. If the date is more than six months old or if the date is in the future, the year is shown instead of the time.

The last item on the line is the name of the file or directory.

**Extended Attribute Descriptions**

On 7/2008R2/8/2012/10/2016 systems, **ls** supports the **-X** option, which displays the extended attributes of the file(s). This description is shown beneath the long output format already described.

For **ls -X A** or **ls -X a**, the display format is:

Attributes: *attributelist*

where *attributelist* is a comma-separated list of one or more of:

|  |  |
| --- | --- |
| **Verbose** | **Terse** |
| Archive | A |
| Compressed | C |
| Directory | D |
| Encrypted | E |
| Hidden | H |
| Normal | N |
| Offline | O |
| ReparsePoint | R |
| ReadOnly | RO |
| Sparse | SP |
| System | S |
| Temporary | T |

For **ls -X D** or **ls -X d**, the display format is:

ACE: *perm usergrpname ace\_flags accessmode*

where:

*perm*

is one of Allow or Deny.

*usergrpname*

is the user or group name as taken from the 7/2008R2/8/2012/10/2016 Security Access Manager registry. Any spaces embedded in the name are replaced by + characters.

*ace\_flags*

is a comma-separated list of the flags for Discretionary Access Control Entities:

|  |  |
| --- | --- |
| **Verbose** | **Terse** |
| ContainerInheritAce | CIA |
| FailedAccessAceFlag | FAAF |
| InheritOnlyAce | IOA |
| NoPropagateInheritAce | NPIA |
| ObjectInheritAce | OIA |
| SuccessAccessAceFlag | SAAF |

If no flags are set, **ls** displays a single -. ACE stands for Access Control Entry; for more information, see your 7/2008R2/8/2012/10/2016 system documentation.

*accessmode*

is the set of access permissions (if Allow is displayed) or restrictions (if Deny is displayed) for this ACE. These can be any or none of the following:

|  |  |  |
| --- | --- | --- |
| **Verbose** | **Terse** | **Note** |
| Read | R | Read file data or list directory entries |
| Write | W | Write file data or create new file in directory |
| eXecute | X | Execute a file or traverse a directory |
| Append | A | Append data to a file or add subdirectory |
| DeleteChild | DC | For a directory, delete entries in directory |
| ReadEa | RE | Read extended attributes |
| WriteEa | WE | Write extended attributes |
| ReadAttr | RA | Read attributes |
| WriteAttr | WA | Write attributes |
| Delete | D | Delete access |
| ReadControl | RC | Read access to the owner, group, and discretionary access control list (ACL) of the security descriptor |
| WriteDac | WD | Write access to the owner, group, and discretionary access control list (ACL) of the security descriptor |
| takeOwnership | O | Write access to the owner |
| Synchronize | S | Synchronize access |

**File handling**

**Create/Edit file**

**vi file1**

Add some content in the above file after typing '**i**'. Delete a line : esc+dd

Delete a word : esc+dw Delete a character : esc+x

Search for a word: esc /wordtofind Goto a line number: esc :linenumber Undo: esc+u

Redo: ctrl+R

To save type **Shift+: wq**

To quit without saving **Shift+: q!**

**Create/Replace a file with few content**

**echo 'welcome to unix' > filename1**

🡪 this will replace the data if filename1 exists already

**Append content to a file**

**echo "this is the second line" >> filename1**

* This will append the line to the file filename1 if its existing already.

**Create Empty file**

**touch filename2**

* this file will be a zero byte file. it cant contain any data. If we append any data into it, then it will no longer be a zero byte file.
* This is usually used as a flag. It will be created once a successful execution is completed to indicate that its completed.

**Display content of a file**

**Display whole content**

**cat filename**

**Display incremental content**

**more filename**

**Display first 10 lines**

**head filename**

eg : cat sample1 | head -4

**Display last 10 lines**

**tail filename**

eg : cat sample1 | tail -4

cat sample1 | tail -5 | head -2

**Counting number of lines in a file**

**wc -l filename**

eg: cat sample1 | wc –l

wc –l sample1

**File operation (copy, move, rename, delete)**

**cp filename file2 mv file2 file3**

**rm file3**

**Directory Commands**

**You can go in your home directory anytime using the following command −**

**cd ~**

Here ~ indicates home directory. If you want to go in any other user's home directory then use the following command −

**cd ~hduser**

**To go in your last directory you can use following command −**

**cd -**

**To go to the parent directory**

**cd .. cd ../..**

**Create Dir Relative path**

Access the rest of the child path from the parent path.

**cd ~**

**mkdir dirname**

**Absolute path**

Fully qualified path start with ‘/’ provided from root till the child.

**mkdir /home/hduser/dirname2**

**Create directory structure from parent directory, here all three dir1,2,3 will be created using option ‘p’.**

**mkdir -p /home/hduser/dir1/dir2/dir3**

**Change Dir**

**cd dirname cd ..**

**Move dir**

**mv dirname dirname1**

**Remove Dir**

**rmdir ~/dirname1 🡪remove dir**

**rm -r ~/dirname1**

**Admin commands:**

**Create Users, set password, switch user and exit out**

**sudo useradd inceptez sudo passwd inceptez su inceptez**

**exit**

**Permissions**

Types - Owner, group and others

**ls -l /home/hduser**

Here first column represents different access mode ie. permission associated with a file or directory.

The permissions are broken into groups of threes, and each position in the group denotes a specific permission, in this order: read (r), write (w), execute (x) −

* The first three characters (2-4) represent the permissions for the file's owner. For example - rwxr-x**r--** represents that onwer has read (r), write (w) and execute (x) permission.
* The second group of three characters (5-7) consists of the permissions for the group to which the file belongs. For example -rwxr-x**r--** represents that group has read (r) and execute (x) permission but no write permission.
* The last group of three characters (8-10) represents the permissions for everyone else. For example -rwxr-x**r--** represents that other world has read (r) only permission.

**Change mode:**

|  |  |  |
| --- | --- | --- |
| **Number** | **Octal Permission Representation** | **Ref** |
| **0** | No permission | --- |

|  |  |  |
| --- | --- | --- |
| **1** | Execute permission | --x |
| **2** | Write permission | -w- |
| **4** | Read permission | r-- |

Change permission of the owner and provide read and write access.

**chmod 000 filename chmod u+rw filename**

Change permission of the group and provide read and write access.

**chmod g+rw filename**

Change permission of the others and provide read and write access.

**chmod o+rw filename**

**Changing owners and Groups**

Change owner from hduser to inceptez

**sudo chown inceptez filename1**

Change group from hduser to inceptez

**sudo chgrp inceptez filename1 Misc commands**

**Identify all running process**

**ps -ef**

**Identify specific running process (bash)**

**ps -ef | grep bash**

**Kill a process running**

**kill -9 processid Disk size commands**

**Disk free**

**df -k /**

**Disk usage**

**du -k /tmp**

**History of the commands used**

**history**

**Grep**

**grep line22 filename**

**Compression**

**gzip filename gunzip filename.gz**

**HDFS Dev & Admin Commands**

**Open a terminal window to the current working directory.**

cd /home/hduser

1. **Print the Hadoop version**

hadoop version

1. **Report the amount of space used and # available on currently mounted filesystem**

hadoop fs -df hdfs:/

1. **Count the number of directories, files and bytes under # the paths that match the specified file pattern #**

hadoop fs -count hdfs:/

1. **Count the number of directories, files and bytes under # the paths that match the specified file pattern #**

hadoop fs -mkdir /user/hduser/hadoop

hadoop fs -mkdir -p /user/hduser/hadoop/dir1/dir2

1. **Create a sample file in linux and place it into hadoop directory**

echo "sampledata" > sample.txt

hadoop fs -put ~/sample.txt /user/hduser/hadoop

hadoop fs -copyFromLocal -f ~/sample.txt /user/hduser/hadoop

1. **List the contents of this new directory in HDFS.**

hadoop fs -ls /user/hduser/hadoop

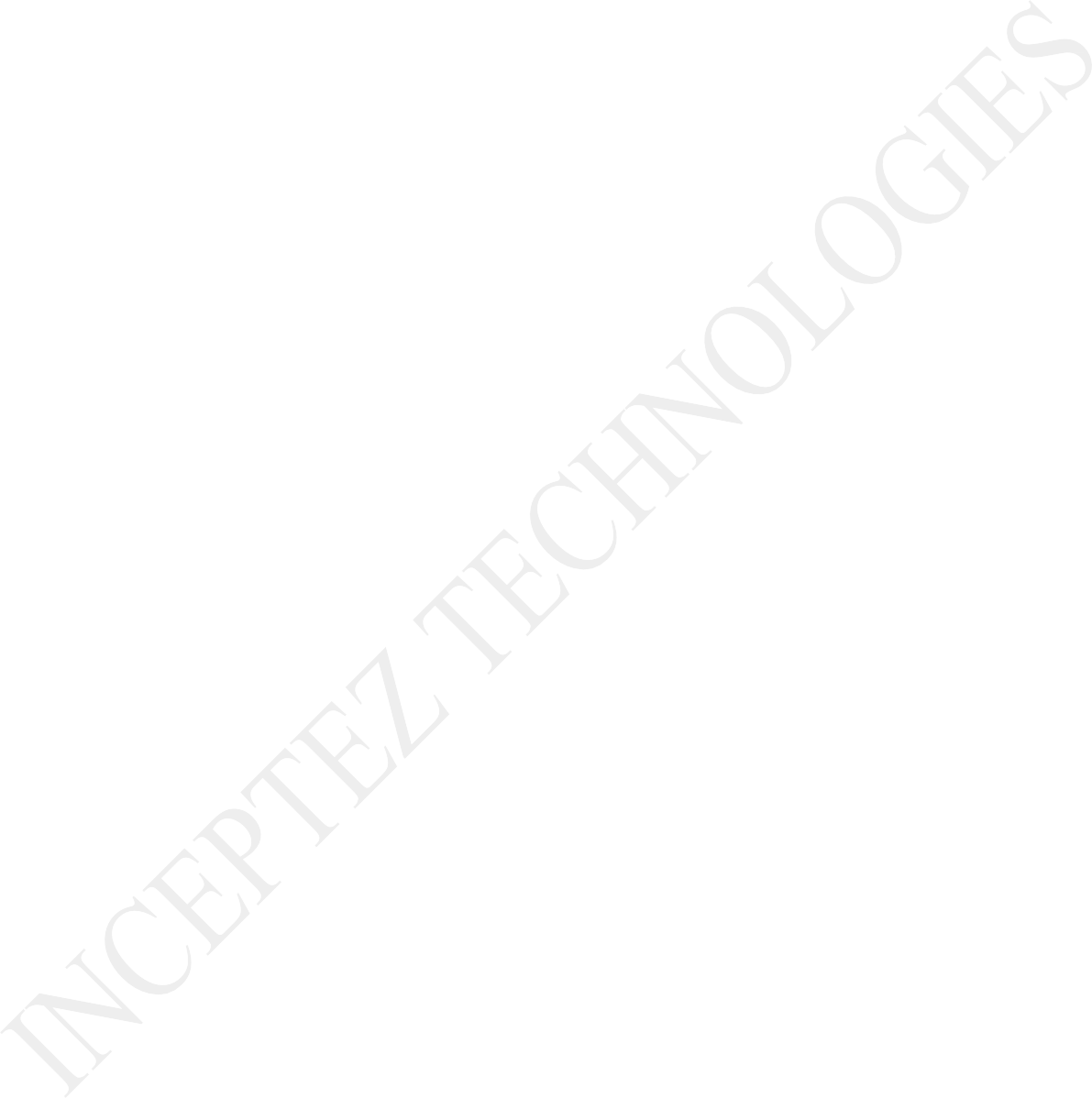
1. **Copy a directory from local to hadoop.**

hadoop fs -put /home/hduser/mrdata /user/hduser/hadoop

1. **Remove a HDFS file**

hadoop fs -copyFromLocal test1.txt hadoop/ hadoop fs -rm hadoop/test1.txt

1. **HDFS file permission setup, default is 666**



**10. Remove the entire directory and all of its contents in hadoop.**

**hadoop fs -mkdir hadoop/test**

**hadoop fs -put ~/sample.txt hadoop/test**

**hadoop fs -rm -r hadoop/test**

1. **Copy the file from hadoop to local.**

**hadoop fs -copyToLocal /user/hduser/test.txt /tmp**

1. **Remove all files from hadoop directory ending with .txt**

**hadoop fs -rm hadoop/\*.txt**

1. **cp is used to copy files between directories present in HDFS**

**hadoop fs -cp /user/hduser/test.txt /user/hduser/test2.txt**

1. **Get command to copy the file from hadoop to local.**

**hadoop fs -get test2.txt /home/hduser/test3.txt**

1. **Display last few lines in hadoop**

**hadoop fs -put filename**

**hadoop fs -tail filename**

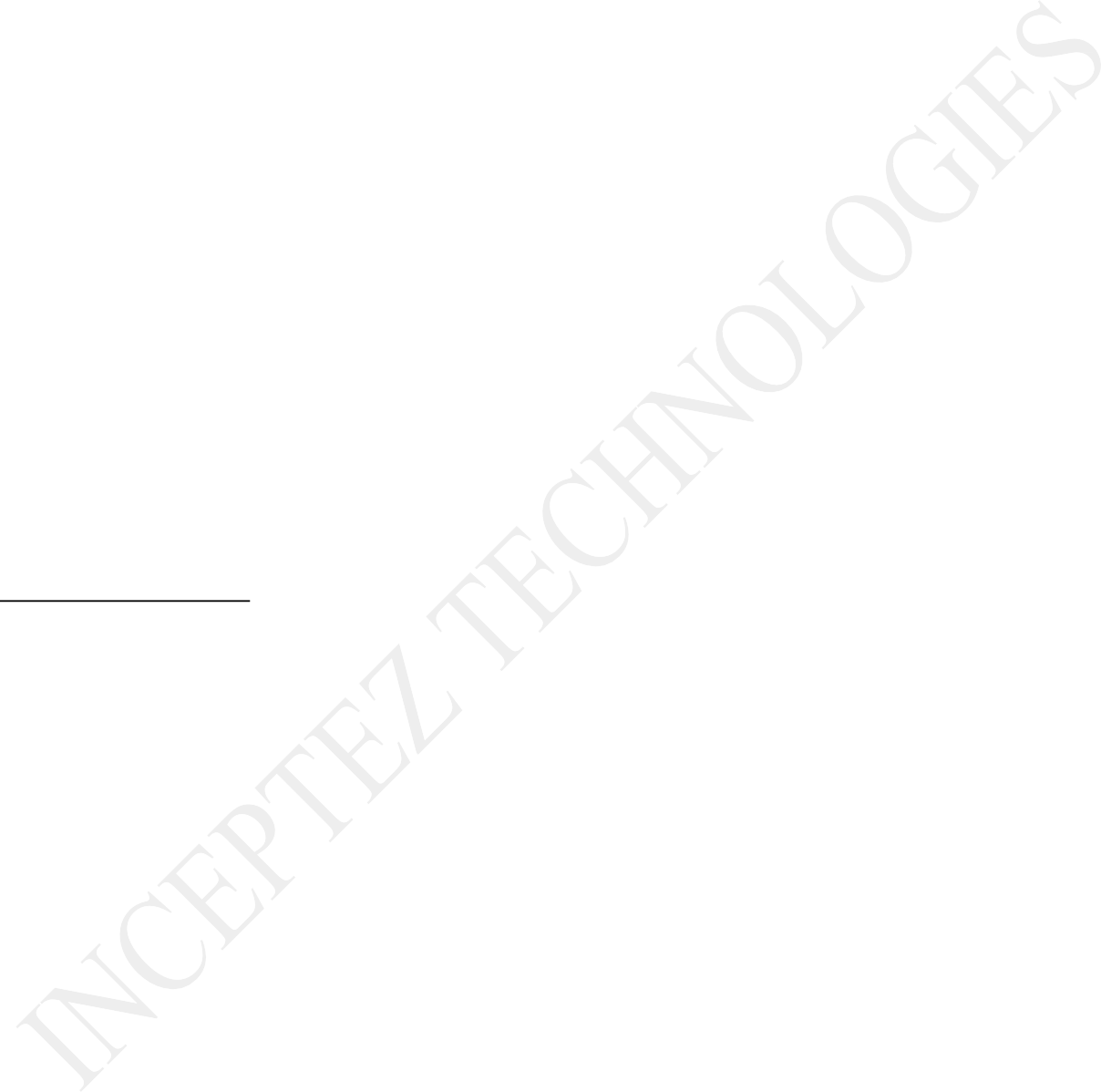
hadoop fs -touchz hadoop/test4.txt hadoop fs -ls hadoop/test4.txt

hadoop fs -chmod 600 hadoop/test4.txt

1. **View the content of copied file.**

hadoop fs -cat /user/hduser/testing/test.txt

1. **Move file from local to hdfs**



**hadoop fs -moveFromLocal ~/test.txt /user/hduser/test.txt**

**19. Append file from local to hdfs.**

**cd ~**

**echo somedata > test1.txt**

**hadoop fs -appendToFile test1.txt /user/hduser/testing/test.txt**

**20. Create new file with zero content.**

**hadoop fs -touchz text.txt**

**ADMIN COMMANDS**

1. **See how much space this directory occupies in HDFS.**

**hadoop fs -du -s -h hadoop**

1. **Run a DFS filesystem checking utility**

**hadoop fsck - /**

1. **Run a cluster balancing utility**

**hadoop balancer**

**20. Default names of owner and group are hduser,**

hadoop fs -ls test2.txt

hadoop fs -chown inceptez:inceptez test2.txt

**21. Change the group of a file in hadoop**

hadoop fs -ls test2.txt

hadoop fs -chgrp hadoop test2.txt

**HDFS commands Hands on**

1.create a file in linux copying the first 50 lines from and existing file

head -50 got.txt>got1

2. Create a dir in Hdfs

hadoop fs -mkdir /user/hduser/kanna1

3. Copy the file got.txt from linux to hdfs in the name gotstory

hadoop fs -copyFromLocal ~/got.txt /user/hduser/kanna1/gotstory

4. Display only records 10 to 20 from hdfs

hadoop fs -cat /user/hduser/kanna1/gotstory | head -20 | tail -10

5.edit the files in hdfs to remove first 30 lines

hadoop fs -get /user/hduser/kanna1/gotstory ~/kanna1/gotstoryedit

ls

got1 gotstoryedit got.txt

vi gotstoryedit (edit the content) save by :wq

wc -l gotstoryedit

20 gotstoryedit

hadoop fs -put ~/gotstoryedit.txt /user/hduser/kanna1/gotstoryedited

6. set the block size 128MB while writing the file in HDFS

hadoop fs -D dfs.block.size=134217728 -put ~/gotstoryedit.txt /user/hduser/kanna1/gotstoryedited

now check the block size in the UI

7. set the block size 128MB while writing the file in HDFS

hadoop fs -D dfs.block.size=134217728 -put ~/gotstoryedit.txt /user/hduser/kanna1/gotstoryedited

now check the block size in the UI

8. set the replication factor to 3 while writing file into hdfs

hadoop fs -D dfs.replication=3 -put ~/gotstoryedit.txt /user/hduser/kanna1/gotstoryedited

9. how to check the block information (in which data node block is present, no of blocks,size,replicaiton etc)

hdfs fsck /user/hduser/kanna1/hadoop-2.7.1.tar.gz -files -locations -blocks

Connecting to namenode via http://localhost:50070/fsck?ugi=hduser&files=1&locations=1&blocks=1&path=%2Fuser%2Fhduser%2Fkanna1%2Fhadoop-2.7.1.tar.gz

FSCK started by hduser (auth:SIMPLE) from /127.0.0.1 for path /user/hduser/kanna1/hadoop-2.7.1.tar.gz at Mon Mar 25 14:40:59 IST 2019

/user/hduser/kanna1/hadoop-2.7.1.tar.gz 210606807 bytes, 2 block(s): OK

0. BP-1284683981-127.0.0.1-1553442766038:blk\_1073741869\_1045 len=134217728 repl=1 [DatanodeInfoWithStorage[127.0.0.1:50010,DS-e593dba0-6cf4-46b9-9864-532fa99eeeac,DISK]]

1. BP-1284683981-127.0.0.1-1553442766038:blk\_1073741870\_1046 len=76389079 repl=1 [DatanodeInfoWithStorage[127.0.0.1:50010,DS-e593dba0-6cf4-46b9-9864-532fa99eeeac,DISK]]

Status: HEALTHY

Total size: 210606807 B

Total dirs: 0

Total files: 1

Total symlinks: 0

Total blocks (validated): 2 (avg. block size 105303403 B)

Minimally replicated blocks: 2 (100.0 %)

Over-replicated blocks: 0 (0.0 %)

Under-replicated blocks: 0 (0.0 %)

Mis-replicated blocks: 0 (0.0 %)

Default replication factor: 1

Average block replication: 1.0

Corrupt blocks: 0

Missing replicas: 0 (0.0 %)

Number of data-nodes: 1

Number of racks: 1

FSCK ended at Mon Mar 25 14:40:59 IST 2019 in 2 milliseconds

The filesystem under path '/user/hduser/kanna1/hadoop-2.7.1.tar.gz' is HEALTHY

10. Distributed copy (distcp) is a tool used for large inter/inter-cluster copying

hadoop distcp hdfs://localhost:54310/user/hduser/kanna1/gotstoryedited hdfs://localhost:54310/user/hduser/kannan1/edited

11. choose to overwrite the target file unconditionally even if it exits

hadoop distcp -overwrite hdfs://localhost:54310/user/hduser/kanna1/gotstoryedited hdfs://localhost:54310/user/hduser/kannan1/edited

12.to view the content of fsimage file, need to convert into text or xml fies using fsimage image viewer

--convert to xml

cd /usr/local/hadoop\_store/hdfs/namenode/current

hdfs oiv i fsimage\_0000000000000093142 -o fsimage.txt -p XML