Tech Mahindra

Unix Fundamentals & Commands

Day 1

TLS

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award

Objectives

- At the end of the session, you will be able to:
  - Use general purpose Unix commands

Agenda

- Introduction to Unix
- Navigating the File System
- Use Unix commands

UNIX

Day 1

TLS

Rewards and Recognition

Objectives

- At the end of the session, you will be able to:
  - Use general purpose Unix commands

Agenda: Day 1

- Unix Overview
- Unix File System
- Unix Basic Commands
- Unix File Management Commands

Unix Overview

TLS

Rewards and Recognition

Winner - 2013

History of Unix

- Developed in AT&T Bell Labs by Ken Thomson as a single user OS in 1969
- Initially written in assembly language
- Developed as multi-user OS later
- Rewritten in C in 1973
- Licensed to university for educational purposes in 1974
- POSIX (Portable Operating System for Unix) was developed

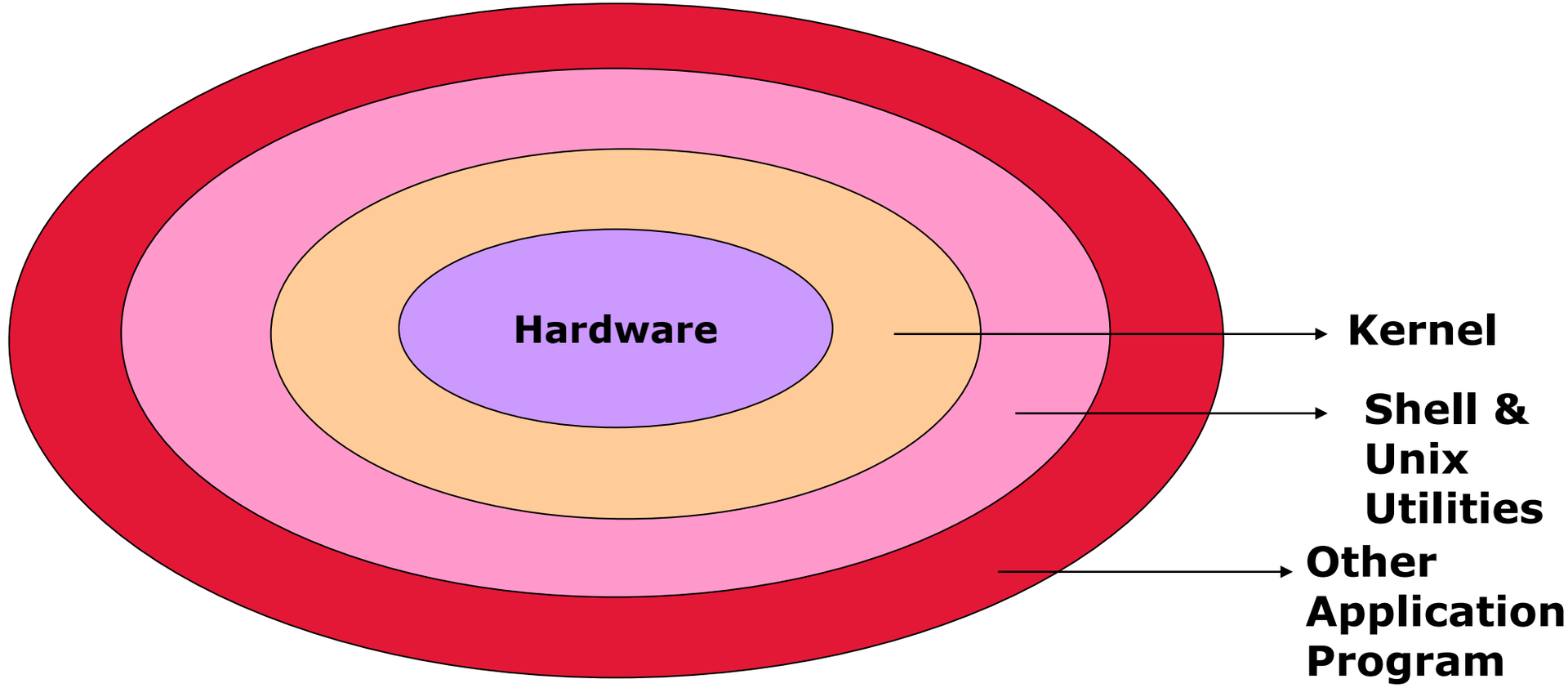
Unix Flavors

- AIX by IBM
- Solaris by Sun Microsystems
- HP-UX by Hewlett-Packard Company
- IRIX by Silicon Graphics, Inc.
- FreeBSD by FreeBSD Group
- GNU/Linux by Open Source Movement
- SCO Unix by The Santa Cruz Operation Inc.

Features of Unix

- Multi-user
- Multi-tasking
- Portable
- Interactive
- Shell
- Security
- Hierarchical File System

Unix Architecture



Unix File System

TLS

Rewards and Recognition

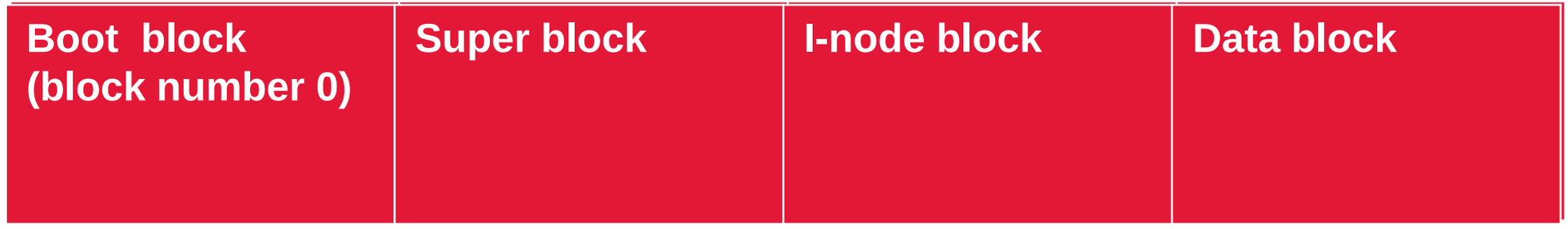
Winner - 2013

Types of file system in Unix

- UFS (Unix File system)
- HSFS(High Sierra File system) and ISO 9660 file system used on CD-ROM and is read only file system
- PCFS(PC file system) - allows read/write access to data & programs on DOS formatted disks
- UDF (Universal Disk Format) used for storing information on optical media technology called DVD
- NFS (Network file system) used to share file system in a network
- SWAPFS (Swap File system) used for virtual memory by the kernel
- PROCFS(Process File system) - resides in memory, contains the list of active processes, by process number, in the /proc directory

Components of the UFS

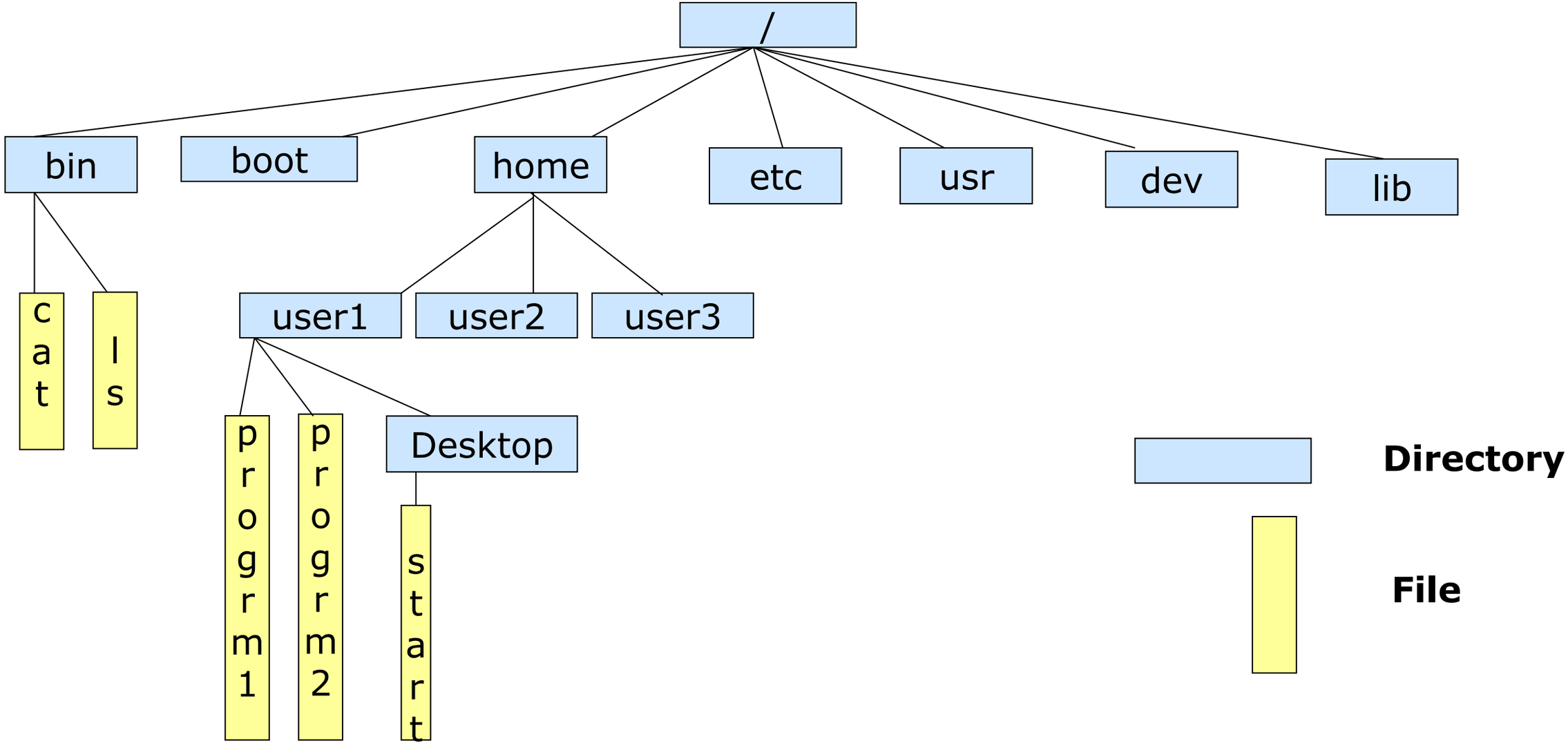
- The UFS has the following four types of blocks
  - Boot Block – Stores information used when booting the system
  - Super block – Stores much of the information about the file system
  - I-node – Stores all information about a file except its name.
  - Storage or data block – stores data for each file.



Disk Partition

- Partitions are logical containers used to house file systems
- A disk may have single or multiple partitions
- OS may span over multiple partitions on a disk or vice versa
- A partition on a hard drive consists of
  - a range of cylinders of HDD
  - each partition is defined by both a start and end cylinder (the size of cylinders varying from disk to disk)

Root Directory Structure



File Basics

- Everything on Unix is a file. File structure is hierarchical like an upside down tree.
- File is just a sequence of bytes.
- The meaning of the bytes depends solely on the programs that interpret the file.
- The format of a file is determined by the programs that use it.

File Attributes

- Name (such as “test.log”)
- An Owner (such as “John”)
- Access Rights (such as read, write, execute)
- Other attributes (such as date of creation)

File Types

- Every item in a UNIX file system belongs to one of the three possible types:

1. Ordinary/Regular files
2. Directory files
3. Device/Special files

Ordinary File

- Contains text, data, or program information
- Cannot contain another file or directory
- Can be thought of as one-dimensional array of bytes

Directory File

- Contains directory(s) and/or file(s) within it.
- Has one line for each item contained within the directory.
- Each line in a directory file contains only the name of the item, and a numerical reference to the location of the item, called inode number.
- Inode number is an index to a table known as the inode table. Inode stores all information about the file except its name.

Device File

- Physical devices (printers, terminals etc) are represented as “files”.

- Two types of device files:
  1. Character Special
  2. Block Special

Links

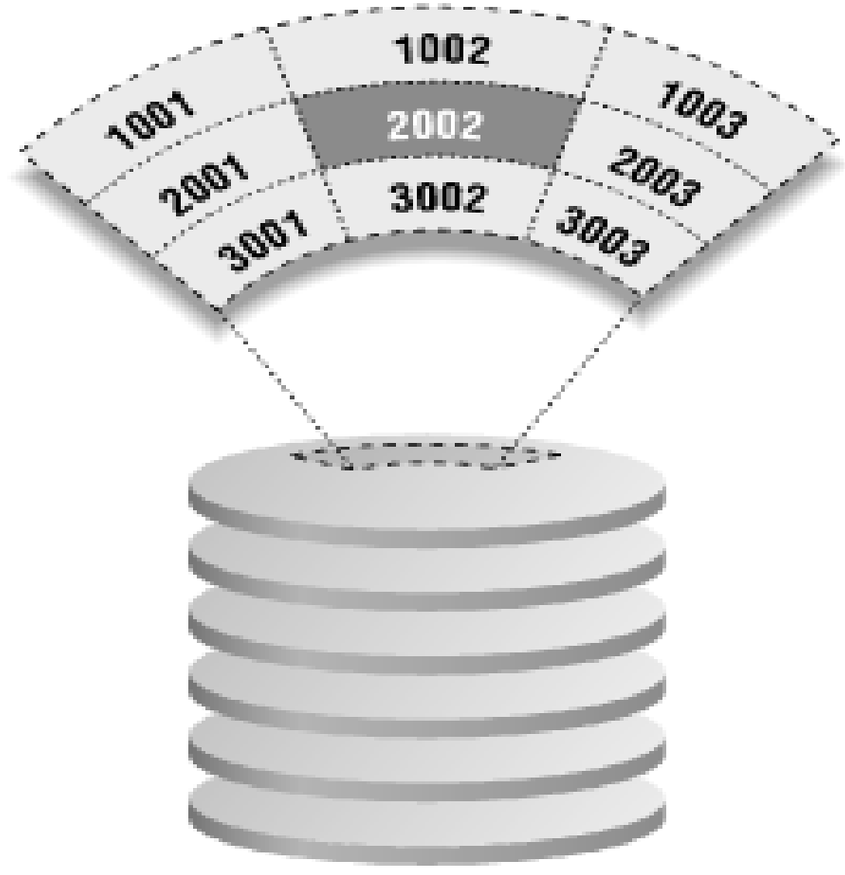
- Links make the same file available in multiple directories at the same time
  - Two types of Links:
    1. Hard Link
       - A hard link is another name given to the existing file
       - These names share the same inode
    2. Soft Link
       - A soft link to a file has a separate inode than the file
       - It stores the target file's path in its inode

Inode

- Inode is a data structure containing useful information about an item in the Unix File System.

- Inodes reside on disk and do not have names. Instead, they have indices (numbers) indicating their positions in the array of inodes as shown in the next slide.

Inode



inode 2002

Item Location	Item Type	Item Size *(bytes)*
Time Inode Modified *(ctime)*	Time Contents Modified *(mtime)*	Time File Accessed *(atime)*
File's Owner *(UID)*	File's Group *(GID)*	Per-missions *(mode bits)*
Reference Count	Location of Data on Disk	

Pathname

- Every item in the file system with a name can be specified with a pathname.
- Pathname represents the path to the entry from the root of the file system. By following this path, the system can find the inode of the referenced entry.
- Pathnames can be absolute or relative.

Unix Users

- Super User
- Owner
- Group
- Others

Unix Users

- Superuser
  - Can also be referred to as a System Administrator
  - Has an overall authority on Unix OS
  - Responsible for OS maintenance, backup and recovery, user management etc.
  - Superuser login is root and prompt is #

- Owner
  - Is a user who creates a file
  - For every Unix file there can be only one owner
  - File owner can assign the file permissions to group and other users

Unix Users

- Group
  - In Unix, groups can be formed based on area of work
  - Superuser can create a group and assign members to it
  - Owner of a file can decide what permissions to be given to group members

- Others
  - User who is not a owner and does not belong to any specific group is referred to as other user
  - Owner of a file can decide what permissions to be given to other users

Unix Commands

TLS

Rewards and Recognition

Winner - 2013

Basic Commands

- $ pwd

  Shows working i.e current directory

- $ who

  Shows who is logged on

- $ who am i

  Shows the login name of the current user

- $ tty

  Print the file name of the terminal connected to standard input

Basic Commands

- $ man command formats and displays online manual pages

- The manual pages are divided into logical grouping of the commands called the sections. Sections are numbered from 1 through 9.

  For example :-

  Commands are 1, System Calls are 2, Library Function are 3, File Formats are 5, & Management Commands are 8.

- If section is specified, man only looks in that section of the manual.

- The command man 2 open displays the manual for the system call open.

Basic Commands

- $ date: command prints or sets the system date and time

- For example:-

  $ date

  $ date –r TestFile

  Displays the last modification time of the file “TestFile”

- This command can be used with suitable format specifiers as argument. Each format is preceded by a + symbol, followed by the % operator, and a single character describing the format.

Basic Commands

Sequence	Interpretation
%a	abbreviated weekday name (Mon .. Sun)
%b or %h	abbreviated month name (Jan .. Dec)
%d	day of month (01 .. 31)
%r	time (12- hour)
%T	time (24- hour)
%y	last two digits of year (00 .. 99)
%D	date (mm/dd/yy)

e. g. $ date “+Today is %a %m %Y”

Basic Commands

```
echo –e “Hi $USER \n Welcome to Unix”

echo –e ‘Hi $USER \n Welcome to Unix’

echo –e Hi $USER \n Welcome to Unix

clear
tput clear

tput cup <row_coordinate> <column_coordinate>
tput cup 15 20
```

Basic Commands

$ which <command_name>

- Shows the path of the specified command

$ type <command_name>

- Shows the type of the specified command – shell built-in or file path if it is external

$ file list1

- Shows the type of the file whether regular file or directory file or some other file

Basic Commands

$ cal

- Shows the calendar of the current month/year

$ bc

2.3+5.4

- Shows the mathematical calculations involving integers as well as floats
- bc is a filter command

$ write *user1*

- Lets to write a message to the other user; user1 in this example

File Management Commands

$ ls

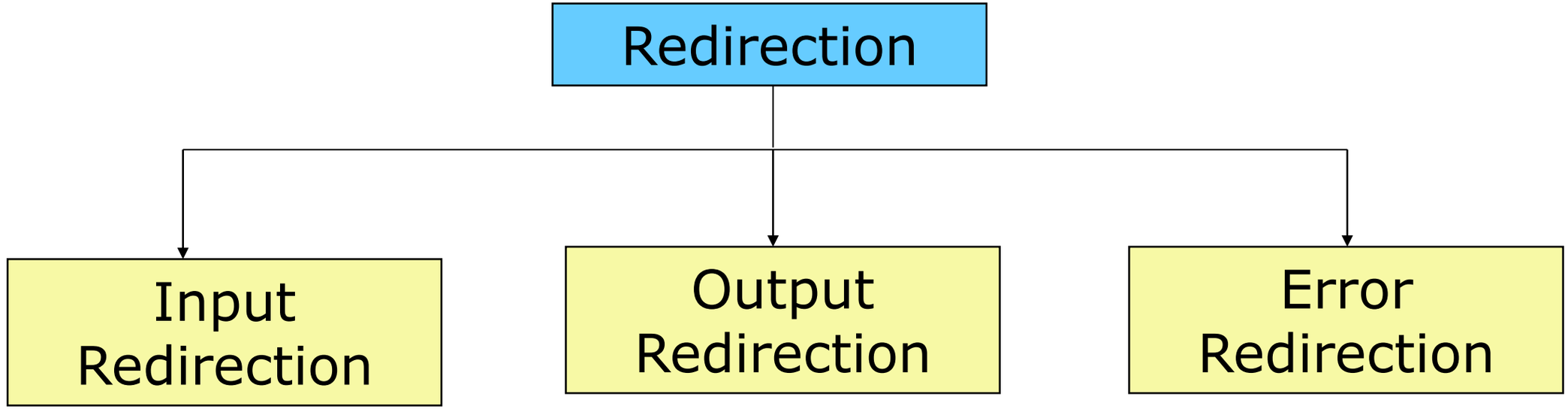
- Command to list files and directories

Option	Description
-l	**list in long format**
-C	**multicolumn output**
-F	**indicates type of file by /, ***
-R	**recursive listing of all subdirectories encountered**
-a	**list all files including hidden files**
-i	**List all files along with i-node number**

Wild Card Characters

- * - zero or more characters – *e.g.* **ls m*** lists all the files starting with m
- ? – single Character – *e.g.* **ls m?** lists all the files starting with m and having one character after that
- []- any character from all the characters within [] *e.g.*
  ls [aeiou]* lists all files starting with a or e or i or o or u
- - Specifies range
- ! – works as not operator *e.g.*
  ls [!x-z]* will list all the files not starting with any character from the range x-z

Redirection



Input Redirection

- Input Redirection specifies that the standard input, instead of coming from the standard input file, comes from a disk file

- $cat < filename
  - e.g. cat < data1
  - e.g. cat 0< data1

Output Redirection

- The output redirection specifies that the standard output of a given command is written to the disk file, instead of monitor

- $ cat filename1 > filename2
  - e.g. :-

cat abc 1 > xyz

Error Redirection

- The error redirection specifies that an error is written on the standard error-msg file, instead of the monitor

- $ cat filename 2>error-msg file
  - e.g. :-

    cat pqr 2> error-msg

File Management Commands

- cat
  - To concatenate files
  - To display contents of one or more ordinary files
  - To create an ordinary file

$ cat *file1 file2 …*

- It displays contents of all files specified on the command line one below the other

$ cat > *file*

- It creates a new file by accepting text from the standard input
- Press CTRL-d to save and exit the file

File Management Commands

- touch
  - Update the access and modification times of each FILE to the current time.
  - If the file is NOT present creates a ZERO byte file
  - Eg:-
    touch sample.txt

File Management Commands

- mkdir [-p] *dirname*
  - Makes a directory of a given *dirname*
  - The *dirname* can contain the full path prefix of the directory to be created
  - More than one directories can be created at the same time
  - When executed with –p option, it does not give any error if the directory *dirname* already exists
  - When executed with option –p, it makes parent directories in the path, if needed (if any parent directory in the path is not available)

$ mkdir dir1 dir2 dir3

- Makes directories dir1 dir2 and dir3 in the current directory

File Management Commands

- rmdir *dirname*
  - Removes an empty directory
  - Avoids the chances of accidental removal of a directory with some useful data files.

$rmdir dir1 dir2 dir3

- removes directories dir1 dir2 and dir3 from the current directory, given that they are empty

File Management Commands

- cd [directory]
  - changes working directory to the directory, if specified; otherwise to the home directory
  - cd .. moves to the parent directory
  - cd or cd ~ changes to the home directory

File Management Commands

- cp command copies files and directories

- $ cp –i file1 file2
  - Copies file1 to file2
  - -i - informs user before overwriting, if file2 exists

- $ cp file1 file2 … dest_directory
  - Copies multiple files in the specified existing directory

- $ cp -r directory1 directory2 … dest_directory
  - Recursively copies files from directory1, directory2 etc. to the dest_directory

Note: Shell Meta-characters can also be used with “cp”

File Management Commands

- mv command changes name of the file or moves the file to the specified destination path

- $ mv file1 new-file
  - Renames file1 as new-file

- $ mv file1 file2 … dest_directory
  - Moves multiple files to the specified existing directory

- $ mv directory1 directory2 … dest_directory
  - Moves one or more directory subtrees to an existing or new dest_directory

Note: Shell Meta-characters can also be used with “mv”

File Management Commands

- rm command is used for deleting unwanted files/directories

- $ rm [-i] file …
  - It is interactive removal (option –i) of specified files

- $ rm -r directory …
  - It is recursive deletion of all the files within the specified directories and also the directories themselves

Note: Shell Meta-characters can also be used with “rm”

Summary

- In this session, we have covered:
  - Unix Overview
  - Unix File System
  - Unix Basic Commands
  - Unix File Management Commands

Unix Fundamentals and Commands

Thank you

Rewards and Recognition

2012 ASTD BEST AWARD WINNER

Training TOP 125 Winner - 2013

Tata Institute of Social Sciences CLO Awards India Best Virtual Learning Award


Tech
Mahindra