Fundamentals of Linux

DAY 1

Linux Ideas and History

Open Source

- The term open source software or OSS, refers to software in which the source code is freely available to all.

- Linux is distributed under the GNU license [Gnu Not Unix],

- it is an open source movement started by Richard Stallman in the year 1984,

- The main aim of GNU is to provide the operating system. and it's source code freely.

- Linus Torvalds released the Linux under GNU movement.

Linux Origins

1984 : The GNU project and the free software foundation

Create open source version of Unix utilities

Create the General Public License [GPL]

Software license enforcing open source principles

1991 : Linus Torvalds

Create open source, unix-like kernel, released under the GPL

Today :

Linux kernel + GNU utilities=complete open source, unix-like operating system

Linux Principles

- Everything is a file (including hardware)
- Small, single-purpose programs
- Configuration data stored in text
- Ability to chain programs together to perform complex tasks

Linux Usage Basics

Logging in to a Linux System

Two types of logging screens

1. Virtual Console [text-based],[CUI]
2. Graphical Console [X console],[GUI]

CUI- Character User Interface

GUI- Graphical User Interface

Switching between virtual consoles and the graphical environment

- Switch among Virtual consoles by typing:

  Ctrl + Alt + F (2-6)

- Access the graphical console by typing:

  Ctrl + Alt + F1

- Login using login name and password
- Server systems often have only virtual consoles
- Desktops and workstations typically have both

Users

Two type of users

1.Admin User

eg – root

2.Non admin user(system user)

eg – apache, squid etc

- The root user: a special administrative account
- Also called the superuser
- root has near complete control over the system

Bash Syntax
(Running Commands)

- Commands have the following syntax:

  command *options arguments*

- Each item is separated by a space
- Options modify a command's behavior
- Single-letter options usually preceded by **-**
- Multiple commands can be separated by **;**
- Eg: # reboot -f

  # mkdir red hat; cp -v /etc/passwd

Get Help in a Textual Environment

Getting Help

- Don't try to memorize everything!
- Many levels of help
  - whatis
  - *command –help*
  - man and info

The whatis Command

- Displays short descriptions of commands
- Often not available immediately after install
- Eg :

```
# whatis ls
```

The --help Option

- Displays usage summary and argument list
- Used by most, but not all, commands
- Eg :

```
# ls --help
```

The man Command

- Manual page for any specified command
- Provides documentation for commands
- Almost every command has a man "page"
- Eg :

```
# man ls
```

Manage Files from the Command Line

Linux File Hierarchy Concepts

- Files and directories are organized into a single-rooted tree structure
- Filesystem begins at the root directory, represented by a lone / (forward slash) character
- Paths are delimited by /

Some Important Directories

- Home Directories: /root, /home/*username*
- Configuration: /etc
- User Executables: /bin, /usr/bin
- System Executables: /sbin, /usr/sbin
- Temporary Files: /tmp
- Kernels and Bootloader: /boot
- Server Data: /srv

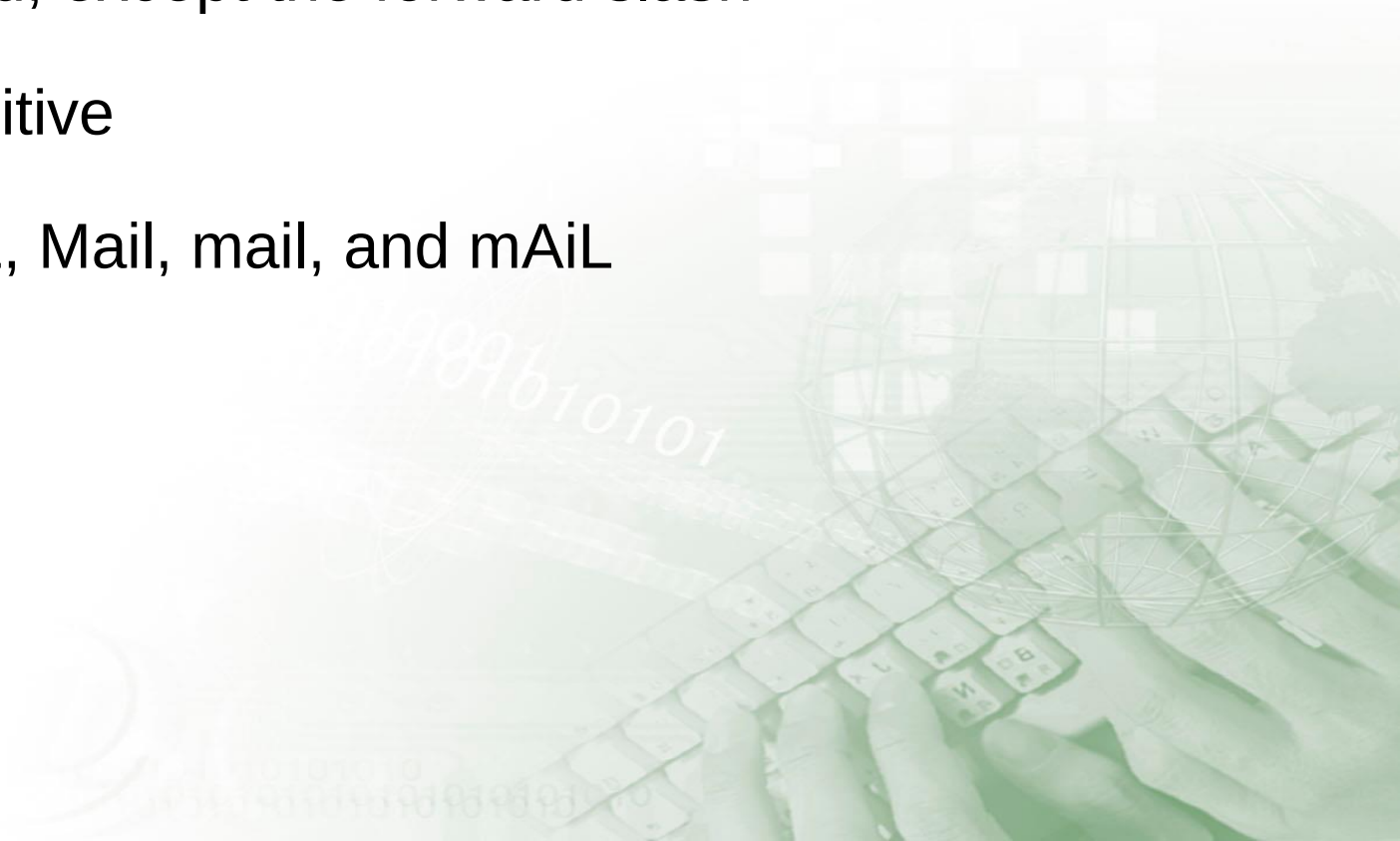
Some Important Directories

- Variable data: /var (such as logs and web site content)
- System Information: /proc, /sys
- Device information:/dev
- Other Mountpoints: /media, /mnt

File and Directory Names

- Names may be up to 255 characters
- All characters are valid, except the forward slash
- Names are case-sensitive

  Example: MAIL, Mail, mail, and mAiL



Current Working Directory

•Each shell and system process has a current working directory (cwd)

•**pwd**

Displays the absolute path to the shell's cwd

•Eg :

```
# pwd
```

Absolute and Relative Pathnames

- **Absolute pathnames**
  - Begin with a forward slash
  - Complete "road map" to file location
  - Can be used anytime you wish to specify a file name
- **Relative pathnames**
  - Do not begin with a slash
  - Specify location relative to your current working directory
  - Can be used as a shorter way to specify a file name

Changing Directories

- **cd** changes directories
  - To an absolute or relative path:
    - **cd** /home/joshua/work
    - **cd** project/docs
  - To a directory one level up:
    - **cd ..**
  - To your home directory:
    - **cd**
  - To your previous working directory:
    - **cd -**

Listing Directory Contents

•Lists the contents of the current directory or a specified directory

•Usage:

ls [options] [*files_or_dirs*]

•Example:

ls

ls -a (include hidden files)

ls -l (display extra information)

ls -i (display index number)

ls -R (recurse through directories)

Creating and Removing Files

- **touch -** create empty files
- **rm -** remove files
- Usage

  rm [options] <*file*>...
- Example

  rm -i *file (interactive)*

  rm -f *file (force)*
- *Eg:*

  # touch redhat

  # touch rhce{1..10}

  # rm redhat

  # rm -f redhat

Other Methods

- For creating file
- cat redhat: read file
- gedit filename
- Vim filename
- Vi filename
- Nano filename

Creating and Removing Directories

- **mkdir** creates directories
- **rmdir** removes empty directories
- **rm -r** recursively removes directory trees
- Eg:

```
# mkdir redhat
# rmdir redhat
# mkdir linux
# cd linux
# touch date
# rm -r linux
# rm -rf linux
```

Copying Files and Directories

- **cp -** copy files and directories
- Usage:

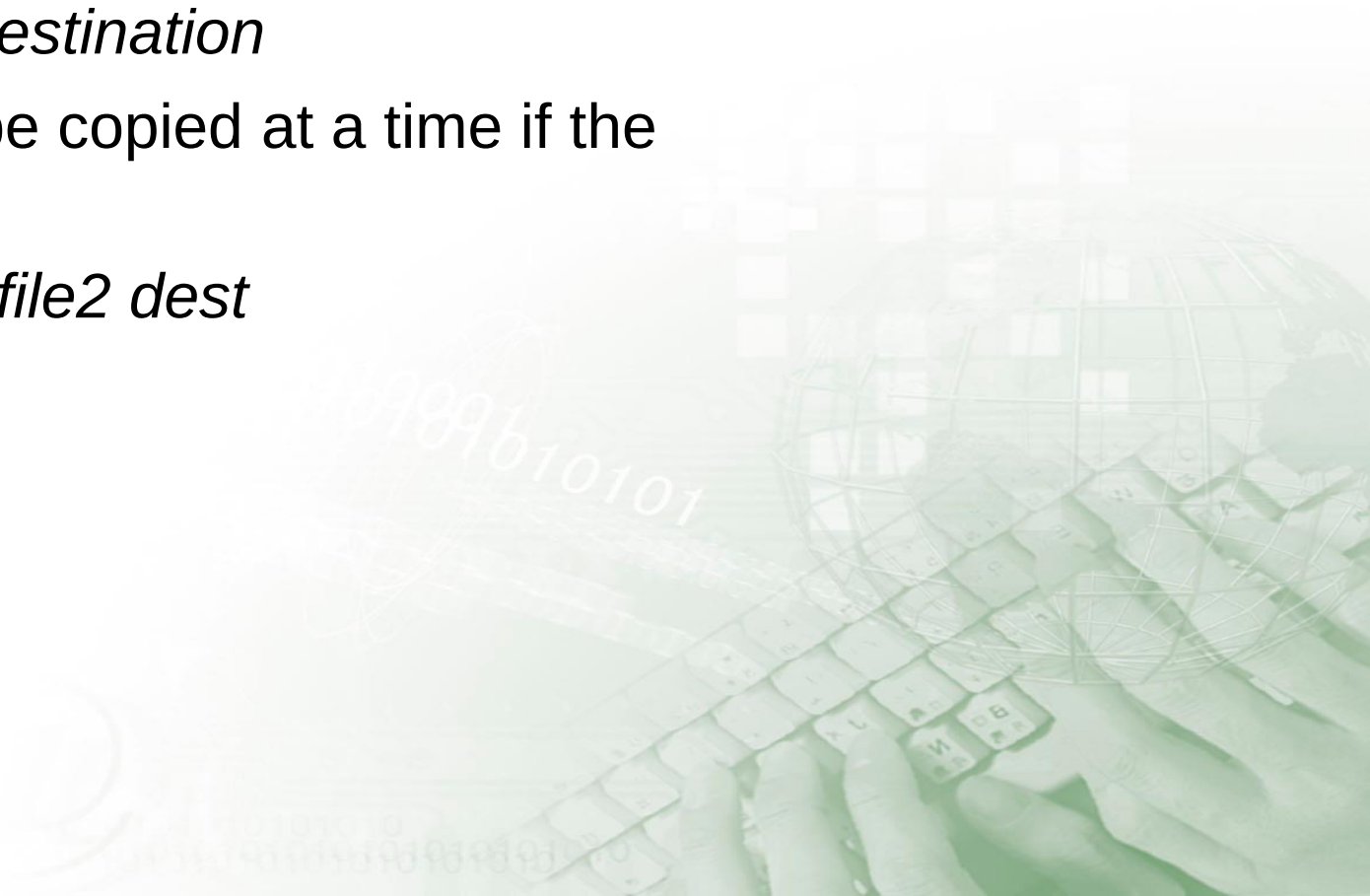
  cp [options] *file destination*

- More than one file may be copied at a time if the destination is a directory:

  cp [options] *file1 file2 dest*

- *Command options:-*

  -r (récursive)



Copying Files and Directories: The Destination

- If the destination is a directory, the copy is placed there
- If the destination is a file, the copy overwrites the destination
- If the destination does not exist, the copy is renamed
- Eg:

```
# mkdir /redhat

# cp /etc/passwd   /redhat
```

Moving and Renaming Files and Directories

- **mv -** move and/or rename files and directories
- Usage:

  mv [options] *file destination*

- More than one file may be moved at a time if the destination is a directory:

  mv [options] *file1 file2 destination*

- Destination works like **cp**
- Eg:-

```
# touch rhce
# mv rhce   /home
```