

Linux Overview

CSC 310 - Programming Languages

Linux

- Linux is an operating system, like Windows or OSX.
- Linux and Kutztown University
 - The Computer Science department has a Linux server named acad that CS and IT students can access
 - This is where you will do your assignments for this class
 - The interface to the Linux server is a command line interface, not a graphical user interface

Connecting to acad

- Windows: Open the command prompt program
- OSX: Open the terminal program
- In the command prompt or terminal program

- Run the command:

```
ssh username@host
```

- Your username is the first part of your KU email address
 - The host name is `acad.kutztown.edu`
 - Example:

```
ssh abcde123@acad.kutztown.edu
```

Connecting to acad (Continued)

- The first time you connect to acad you may see something similar to:

```
The authenticity of host 'csit.kutztown.edu (156.12.127.127)'
ECDSA key fingerprint is SHA256:2oJ7zjD4/XbLbyWwWbv15+0
Are you sure you want to continue connecting (yes/no/[fingerprint])
```

- If you see this, type in “yes”, then press enter.
- Then you will be prompted for password. Your password is the same password that you use to access other Kutztown resources. Type in your password, then press enter.
- On a successful login, you should see something similar to:

```
Last login: Sun Sep 26 12:16:10 2021 from kuvapcsitr01
[user@kuvapcsitr01 ~]$
```

Command Line

- A command line is a text based interface to the operating system
- Commands are entered by typing the command on the keyboard then pressing enter
- The command line presents you with a prompt; as a command is typed it is displayed after the prompt.
- Example: in the previous slide, the prompt is:

```
[user@kuvapcsitrd01 ~]$
```

Directories and Files

- Linux uses directories (similar to Windows folders) to organize files
- Directories can contain files and other directories (called subdirectories)
- A directory is also a file; it is a special file that can contain other files
- A directory or file that has a name that starts with a . (dot) is called a hidden directory or file
- Directories and files should be named using the letters, digits, underscores, and dots; other names are possible, but are not as nice to deal with using the command line interface

Directory System Structure

- The Linux file system is hierarchical; all directories and files are organized in a tree-like structure with the directories corresponding to branches and the files corresponding to leaves
- The topmost directory is call the root directory and is denoted with / (slash)
- Subdirectories are located “under” the root directory
- The working directory is the current location in the file system
- When you first log in, your working directory is your home directory

Absolute Path Names

- Every directory or file has a full name called an absolute pathname
- An absolute pathname uniquely identifies a specific directory or file
- Absolute pathnames always start with a slash (/) followed by any subdirectories in its path separated by slashes and ending with the name of the directory or file

Absolute Path Name Example

- The following absolute path is similar to the absolute path of your home directory

`/home/students.kutztown.edu/username/f.txt`

- This can be read as:
 - There is a directory under the root directory named `home`
 - that contains a subdirectory named `students.kutztown.edu`
 - that contains a subdirectory named `username`
 - that contains a file named `f.txt`

Relative Path Names

- Directories and files can also be referred to using relative pathnames
- A relative pathname is a pathname relative to the working directory
- Example: If the working directory is `username` and contains a file named `f.txt`, then the relative path for `f.txt` is simply `f.txt`

Linux Commands

- A Linux command consists of three parts:
 - Command: name of the command; this always comes first
 - Options (or flags): an flag has a dash (-) in it; flags are usually optional and alter the way the command executes relative to its default behavior
 - Arguments: some commands need specific information to run, for example a file name, which are specified as command line arguments

Conventions for Command Descriptions

- The following conventions will be used in the command descriptions that follow:
 - [] (square brackets) are used to indicate that something is optional
 - <> (angle brackets) are used to indicate a path (relative or absolute)
 - \$ (dollar sign) indicates the prompt in examples

pwd

- Print the absolute pathname of the working directory
- Example:

```
$ pwd
```

```
/home/students.kutztown.edu/username
```

ls

- List the contents of a directory
 - `ls`: list the contents of the working directory
 - `ls <directory>`: list the contents of the specified directory
 - `ls -l [<directory>]`: list the contents in long format
 - `ls -a [<directory>]`: list the contents including hidden directories and files

cd

- Change directory
 - `cd`: change to home directory
 - `cd <directory>`: change to the directory indicated by the path `<directory>`
- Special directory names
 - `.` (dot): the working directory
 - `..` (dot dot): the parent of the working directory
 - `~` (tilde): your home directory

file

- Determine the file type

- `file <path>`: determine the file type of the file at `<path>`

- Example:

```
$ file f.txt
```

```
f.txt: ASCII text
```


mkdir

- Make directory

- `mkdir <dirname>`: create the directory at the path `<dirname>` if it does not already exist in the parent directory
- `mkdir -p <dirname>`: create the directory at the path `<dirname>` and also create all non-existing parent directories in the path

rmmdir

- Remove directory
 - `rmmdir <path>`: remove the directory at `<path>` as long as it is empty
- Note: the `rmmdir` command will permanently remove the directory; you cannot undo this action

touch

- Create an empty file

- `touch <filename>`: create an empty file named `<filename>`

- Example:

```
$ touch a.txt
```

```
$ ls
```

```
a.txt
```

cp

- Copy files and directories

- `cp <source> <destination>`: copy the file at <source> to <destination>
- `cp -r <source> <destination>`: copy the directory at <source> to <destination>

mv

- Move (rename) a file or directory
 - `mv <source> <destination>`: move (rename) the file or directory at `<source>` to `<destination>`
- Note: unlike the `cp` command, the `mv` command does not need the `-r` flag when moving (renaming) a directory

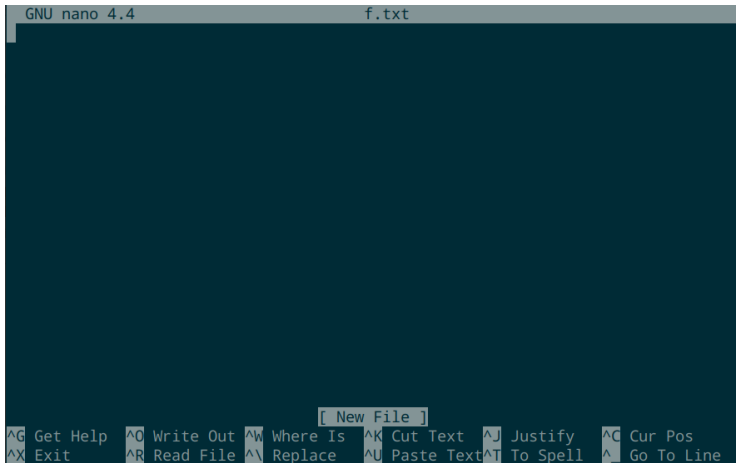
rm

- remove files or directories
 - `rm <file>`: remove the file at the path `<file>`
 - `rm -r <directory>`: remove the directory at path `<directory>` and its contents
 - `rm -i <file>`: (RECOMMENDED) remove the file at the path `<file>` in interactive mode; this will prompt the user before removing the file
 - `rm -rI <directory>`: (RECOMMENDED) remove the directory at the path `<directory>` in interactive mode; this will prompt the user before removing the directory and all its contents
- Note: the `rm` command will permanently delete the file or directory; you cannot undo this action

nano

- nano is a command line text editor
 - `nano <filename>`: open the file `<filename>` in a nano interface; if the file `<filename>` does not exist, then it is created
- A text editor is a program that allows a user to edit the contents of a text file
- There are several text editor programs installed on the Linux server; nano is the simplest one for beginners

The nano interface



The image shows the GNU nano 4.4 text editor interface. The top status bar displays "GNU nano 4.4" on the left and "f.txt" on the right. The main editing area is a large, empty dark blue rectangle. At the bottom, a status bar contains a list of keyboard shortcuts and their corresponding actions. A small menu box with the text "[New File]" is currently open, highlighting the "New File" option. The status bar shortcuts are arranged in two rows: the first row includes ^G Get Help, ^O Write Out, ^W Where Is, ^K Cut Text, ^J Justify, and ^C Cur Pos; the second row includes ^X Exit, ^R Read File, ^\ Replace, ^U Paste Text, ^T To Spell, and ^_ Go To Line.

```
GNU nano 4.4                                f.txt
```

[New File]

^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^\ Replace	^U Paste Text	^T To Spell	^_ Go To Line

The nano interface (Continued)

- The top line of the interface has:
 - The name of the program and version number
 - The name of the file
 - An indicator that the file has been modified since it was last saved
- The third line from the bottom is a “system message”
- The last two lines are the shortcut lines; this is what makes nano more user-friendly compared to other Linux text editors

nano Shortcuts

- All nano shortcuts are prefixed with either ^ (caret) or M
- ^ refers to the control key; when typing a control sequence, you must hold down the control key and press the accompanying key at the same time
- M refers to the alt key; when typing a control sequence, you must hold down the alt key and press the accompanying key at the same time
- Example: ^G means hold down the control key and press g (not shift+g)

Common nano Shortcuts

- ^S: save
- ^O: save as
- ^X: exit (if the file is modified a prompt will appear)
- ^G: display the help text