



La computación como herramienta de trabajo del profesional de Ingeniería

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M. I. Marco Antonio Martínez Quintana

Asignatura: Fundamentos de Programación

Grupo: 3

No. de Práctica(s): 2

Integrante(s): Sánchez García Rocío

*No. de Equipo de
cómputo empleado:* No aplica

No. de Lista: 47

Semestre: 2021-1

Fecha de entrega: Viernes 16 de octubre de 2020

Observaciones:

CALIFICACIÓN: _____

Practica 2

GNU / Linux

Objetivo

Conocer la importancia del sistema operativo de una computadora, así como sus funciones. Explorar un sistema operativo GNU/Linux con el fin de conocer y utilizar los comandos básicos en GNU/Linux.

Actividades

- ❖ Iniciar sesión en un sistema operativo GNU/Linux y abrir una “terminal”.
- ❖ Utilizar los comandos básicos para navegar por el sistema de archivos.
- ❖ Emplear comandos para manejo de archivos.

Introducción

El sistema operativo es el conjunto de programas y datos que administra los recursos tanto de hardware (dispositivos) como de software (programas y datos) de un sistema de cómputo y/o comunicación. Además, funciona como interfaz entre la computadora y el usuario o aplicaciones.

En la actualidad existen diversos sistemas operativos; por ejemplo, para equipos de cómputo están Windows, Linux, Mac OS entre otros. Para el caso de dispositivos móviles se encuentran Android, IOS, Windows Phone entre otros. Cada uno de ellos tiene diferentes versiones y distribuciones que se ajustan a los diversos equipos de cómputo y comunicación en los que trabajan.

Los componentes de un sistema operativo, de forma general, son:

- Gestor de memoria.
- Administrador y planificador de procesos.
- Sistema de archivos.
- Administración de E/S.

Comúnmente, estos componentes se encuentran en el kernel o núcleo del sistema operativo.

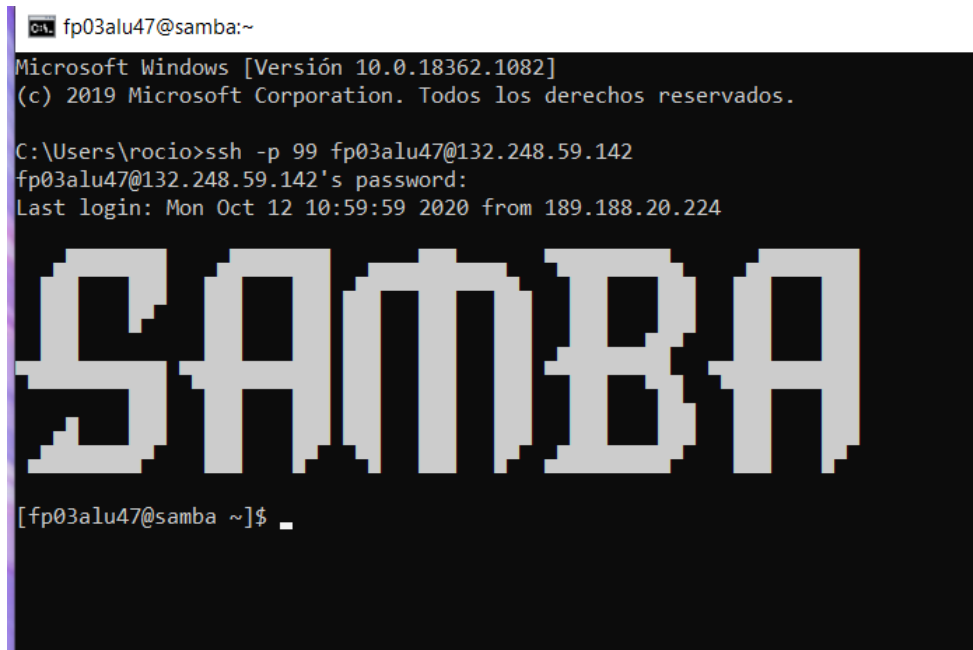
En cuanto a la Interfaz con el usuario, las hay de tipo texto y de tipo gráfico. En la actualidad, es común trabajar con la interfaz gráfica ya que facilita mucho seleccionar la aplicación a utilizar; inclusive esta selección se hace “tocando la pantalla” (técnica touch).

Sin embargo, cuando se desarrollan proyectos donde se elaborarán documentos y programas es necesario el uso de dispositivos de entrada y salida (hardware) y aplicaciones en modo texto (software).

Desarrollo

Comandos básicos

Abrir una “terminal” o “consola”, ventana donde aparece la “línea de comandos” en la cual se escribirá la orden o el comando.



```
C:\> fp03alu47@samba:~  
Microsoft Windows [Versión 10.0.18362.1082]  
(c) 2019 Microsoft Corporation. Todos los derechos reservados.  
  
C:\Users\rocio>ssh -p 99 fp03alu47@132.248.59.142  
fp03alu47@132.248.59.142's password:  
Last login: Mon Oct 12 10:59:59 2020 from 189.188.20.224  
  
Samba  
  
[fp03alu47@samba ~]$
```

Comando **ls**

- ☐ Permite listar los elementos que existen en alguna ubicación del sistema de archivos de Linux.
- ☐ Linux nombra la ubicación actual con un punto (.) por lo que **ls** y **ls .** realizan lo mismo.

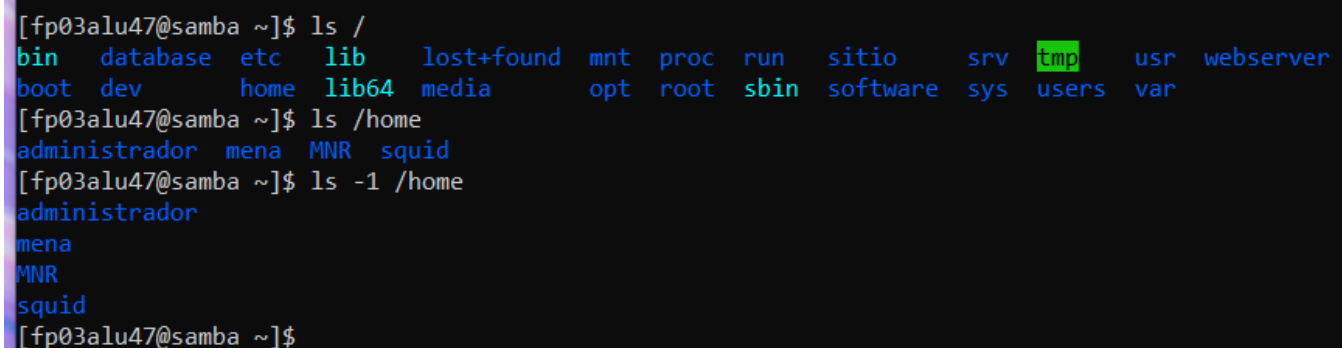
El comando **ls** realiza acciones distintas:

→ Si se utiliza la opción **l** se genera un listado largo de la ubicación actual.



```
[fp03alu47@samba ~]$ ls -l
Escritorio
[fp03alu47@samba ~]$ ls -l .
Escritorio
[fp03alu47@samba ~]$ ls -l -l
Escritorio
[fp03alu47@samba ~]$
```

Para listar los elementos que existen en cualquier ubicación del sistema de archivos hay que ejecutar el comando especificando como argumento la ubicación donde se desea listar los elementos.



```
[fp03alu47@samba ~]$ ls /
bin  database  etc  lib  lost+found  mnt  proc  run  sitio  srv  tmp  usr  webserver
boot  dev      home  lib64  media  opt  root  sbin  software  sys  users  var
[fp03alu47@samba ~]$ ls /home
administrador  mena  MNR  squid
[fp03alu47@samba ~]$ ls -l /home
administrador
mena
MNR
squid
[fp03alu47@samba ~]$
```

- Para ver los archivos que se encuentran en la raíz se usa: **ls /**
- Para ver los usuarios del equipo local se usa: **ls /home**
- Para una ejecución más específica, se pueden combinar tanto las opciones como los argumentos usando: **ls -l /home**
- GNU/Linux proporciona el comando **man**, el cual permite visualizar la descripción de cualquier comando así como la manera en la que se puede utilizar. **man ls**

```

fp03alu47@samba:~
LS(1)                                User Commands                                LS(1)

NAME
    ls - list directory contents

SYNOPSIS
    ls [OPTION]... [FILE]...

DESCRIPTION
    List information about the FILES (the current directory by default). Sort entries alphabetically if none of
    -cftuvSUX nor --sort is specified.

    Mandatory arguments to long options are mandatory for short options too.

    -a, --all
        do not ignore entries starting with .

    -A, --almost-all
        do not list implied . and ..

    --author
        with -l, print the author of each file

    -b, --escape
        print C-style escapes for nongraphic characters

    --block-size=SIZE
        scale sizes by SIZE before printing them. E.g., '--block-size=M' prints sizes in units of 1,048,576
        bytes. See SIZE format below.

    -B, --ignore-backups
        do not list implied entries ending with ~

    -c      with -lt: sort by, and show, ctime (time of last modification of file status information) with -l: show
            ctime and sort by name otherwise: sort by ctime, newest first

    -C      list entries by columns

    --color[=WHEN]
        colorize the output. WHEN defaults to 'always' or can be 'never' or 'auto'. More info below

    -d, --directory
        list directory entries instead of contents, and do not dereference symbolic links

    -D, --dired
        generate output designed for Emacs' dired mode

    -f      do not sort, enable -aU, disable -ls --color

```

Aprender a navegar por el sistema de archivos Linux en modo texto

→ Para ver la lista de archivos del directorio **usr**, usar el comando: **ls /usr**

Cuando se especifica la ubicación de un archivo que parte de la raíz, se dice que se está indicando la “ruta absoluta” del archivo.

```

[fp03alu47@samba ~]$ ls /usr
bin  etc  games  include  lib  lib64  libexec  local  sbin  share  src  tmp
[fp03alu47@samba ~]$

```

Otra forma para especificar la ubicación de un archivo, es emplear la “ruta relativa”.

→ Para listar los archivos que dependen del directorio padre se escribe el siguiente comando: **ls ..** o **ls ../**

→ Si la cuenta depende de **home**, la ruta relativa para listar los archivos del directorio **usr** es: **ls ../../usr**

```
[fp03alu47@samba ~]$ ls ..
fp03alu01 fp03alu07 fp03alu13 fp03alu19 fp03alu25 fp03alu31 fp03alu37 fp03alu43 fp03alu49 fp03alu55
fp03alu02 fp03alu08 fp03alu14 fp03alu20 fp03alu26 fp03alu32 fp03alu38 fp03alu44 fp03alu50 fp03alu56
fp03alu03 fp03alu09 fp03alu15 fp03alu21 fp03alu27 fp03alu33 fp03alu39 fp03alu45 fp03alu51 fp03alu57
fp03alu04 fp03alu10 fp03alu16 fp03alu22 fp03alu28 fp03alu34 fp03alu40 fp03alu46 fp03alu52
fp03alu05 fp03alu11 fp03alu17 fp03alu23 fp03alu29 fp03alu35 fp03alu41 fp03alu47 fp03alu53
fp03alu06 fp03alu12 fp03alu18 fp03alu24 fp03alu30 fp03alu36 fp03alu42 fp03alu48 fp03alu54
[fp03alu47@samba ~]$ ls ../
fp03alu01 fp03alu07 fp03alu13 fp03alu19 fp03alu25 fp03alu31 fp03alu37 fp03alu43 fp03alu49 fp03alu55
fp03alu02 fp03alu08 fp03alu14 fp03alu20 fp03alu26 fp03alu32 fp03alu38 fp03alu44 fp03alu50 fp03alu56
fp03alu03 fp03alu09 fp03alu15 fp03alu21 fp03alu27 fp03alu33 fp03alu39 fp03alu45 fp03alu51 fp03alu57
fp03alu04 fp03alu10 fp03alu16 fp03alu22 fp03alu28 fp03alu34 fp03alu40 fp03alu46 fp03alu52
fp03alu05 fp03alu11 fp03alu17 fp03alu23 fp03alu29 fp03alu35 fp03alu41 fp03alu47 fp03alu53
fp03alu06 fp03alu12 fp03alu18 fp03alu24 fp03alu30 fp03alu36 fp03alu42 fp03alu48 fp03alu54
[fp03alu47@samba ~]$ ls ../../usr
ls: no se puede acceder a ../../usr: No existe el fichero o el directorio
[fp03alu47@samba ~]$
```

Comando *touch*

Permite crear un archivo de texto, su sintaxis es la siguiente: *touch nombre_archivo[.ext]*

Comando *mkdir*

- Permite crear una carpeta, su sintaxis es la siguiente: *mkdir nombre_carpeta*
- Para crear una carpeta en nuestra cuenta que tenga como nombre “tarefas” se escribe el siguiente comando: *mkdir tareas*



```
[fp03alu47@samba ~]$ mkdir nombre_carpeta
[fp03alu47@samba ~]$ mkdir tareas
[fp03alu47@samba ~]$
```

Comando *cd*

- Permite ubicarse en una carpeta, su sintaxis es la siguiente: *cd nombre_carpeta*

```
[fp03alu47@samba ~]$ cd nombre_carpeta  
[fp03alu47@samba nombre_carpeta]$
```

- Por lo que si queremos situarnos en la carpeta “**tareas**” creada anteriormente, se escribe el comando: **cd tareas**

```
[fp03alu47@samba ~]$ cd tareas  
[fp03alu47@samba tareas]$ _
```

- Si se desea situar en la carpeta de inicio de la cuenta, que es la carpeta padre, se escribe el comando: **cd ..**

```
[fp03alu47@samba ~]$ cd ..  
[fp03alu47@samba fp03]$ _
```

Comando ***pwd***

- Permite conocer la ubicación actual, su sintaxis es la siguiente: **pwd**

```
[fp03alu47@samba ~]$ pwd  
/users/fp03/fp03alu47  
[fp03alu47@samba ~]$
```

comando ***find***

- Permite buscar un elemento dentro del sistema de archivos, su sintaxis es la siguiente: **find . -name cadena_buscar**

Para este comando hay que indicar en qué parte del sistema de archivos va a iniciar la búsqueda. En el ejemplo anterior la búsqueda se inicia en la posición

actual (uso de `.`). Además utilizando la bandera **-name** permite determinar la cadena a buscar.

- Si queremos encontrar la ubicación del archivo **tareas** se escribe el siguiente comando: **`find . -name tareas`**

```
[fp03alu47@samba ~]$ find . -name cadena_buscar
[fp03alu47@samba ~]$ find . -name tareas
./tareas
[fp03alu47@samba ~]$
```

Comando **clear**

- Permite limpiar la consola, su sintaxis es la siguiente: **`clear`**

```
C:\> fp03alu47@samba:~
[fp03alu47@samba ~]$
```

Comando **cp**

- Permite copiar un archivo, su sintaxis es la siguiente:
`cp archivo_origen archivo_destino`

Si queremos una copia del archivo **datos.txt** con nombre **datosViejos.txt** en el mismo directorio, entonces se escribe el comando: **`cp datos.txt datosViejos.txt`**

```
[fp03alu47@samba ~]$ cp archivo_origen archivo_destino
cp: se omite el directorio «archivo_origen»
[fp03alu47@samba ~]$ cp datos.txt datosViejos.txt
cp: se omite el directorio «datos.txt»
[fp03alu47@samba ~]$
```


Si se requiere la copia de un archivo que está en la carpeta padre en la ubicación actual y con el mismo nombre, entonces es posible emplear las rutas de la siguiente forma: *cp ../archivo_a_copiar*

Comando *mv*

- Mueve un archivo de un lugar a otro en el sistema de archivos, su sintaxis es la siguiente:
mv ubicación origen/archivo ubicación_destino
- Si se requiere reubicar, un archivo que se encuentra en la carpeta padre, en el directorio actual y con el mismo nombre se debe utilizar las rutas relativas de la siguiente forma: *mv ../archivo_a_reubicar*
- El comando puede ser usado para cambiar el nombre del archivo: *mv nombre_actual_archivo nombre_nuevo_archivo*

```
[fp03alu47@samba ~]$ mv tareas tarea  
[fp03alu47@samba ~]$
```

Comando *rm*

- Permite eliminar un archivo o un directorio, con la siguiente sintaxis:
rm nombre_archivo
rm nombre_carpeta
- Cuando la carpeta a borrar contiene información se utiliza la bandera *-f* para forzar la eliminación.
- Si la carpeta contiene otras carpetas se utiliza la función *-r*, para realizar la eliminación recursiva

```
[fp03alu47@samba ~]$ rm tarea -r  
[fp03alu47@samba ~]$ rm tarea -f  
[fp03alu47@samba ~]$
```

Bibliografía

Manual de prácticas del Laboratorio de Fundamentos de Programación, Facultad de ingeniería UNAM, recuperada el 14 de octubre, en <http://lcp02.fib.unam.mx/>