zulip

Aquellos que mandaron el email con los datos, deberían haber recibido la invitación a participar de zulip.

`https://fis-unne.zulipchat.com/`

- Hay información de como usar el zulip. Las guías etc.
- Pueden mandar consultas de los ejercicios desde allí.
- Pueden entablar conversaciones entre uds.

En la página web de la materia:

`https://pulidom.github.io/programacion`

Tienen la filminas nuevas en la solapa clases y la nueva guía.

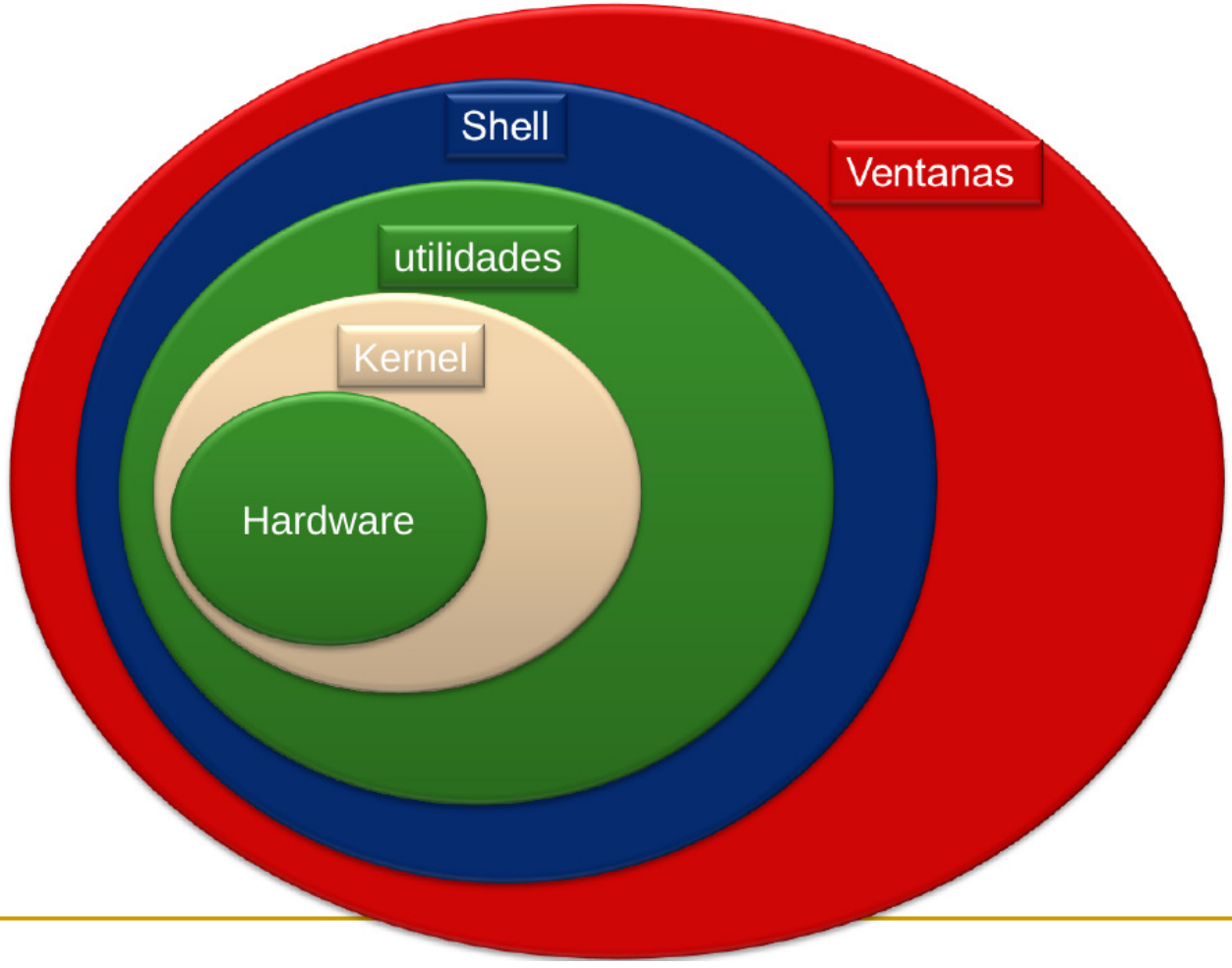
Si no mandan esta semana el email quienes faltan se quedaran sin las herramientas de trabajo.

Temario de la clase

Temario

- Estructura de directorios y atributos de archivos en linux
- Comandos básicos de shell para la administración de archivos y directorios (ls, mkdir, mv, cp, etc.)
- Búsqueda de archivos y cadenas en archivos (find, grep)
- Scripts en bash
- Trabajo remoto: ssh, scp, screen, nano.

Sistema Operativo



El sistema operativo propiamente es el kernel (nucleo), el resto incluyendo el administrador de archivos, bash, administrador de pantalla son aplicaciones.

Distribuciones del linux

- Debian
- Slackware
- Opensuse (HPC/Servidores).
- Ubuntu
- Lubuntu (Light-ubuntu).
- Centos, Fedora, Arch Linux, Mandriva, etc

Organización de los directorios y usuarios del linux

- user: Puede configurar sus propias aplicaciones y utilizar el sistema sin realizar cambios.
- root (superuser): Es el único que tiene permiso de cambiar variables del sistema operativo.

Estructura de directorios principales

- /root
- /home
- /usr
- /media
- /etc
- /proc

/home/[nombre-usuario] Es nuestro directorio de cabecera y alli tendremos nuestra estructura interna.

Permisos de los archivos

Permisos de los directorios o archivos: Lectura(r), Escritura(w), Ejecución(x)

El único lugar que tendremos permiso de escritura es el /home/[nombre-usuario] ceconea: /home1/prog/[nombre-estudiante]
En el resto de los directorios generalmente se tiene permiso de lectura pero no de escritura.

Hay archivos que pueden ejecutarse (con instrucciones que entienda la máquina). Todos los directorios son ejecutables.
Permisos independientes: Para el propio usuario (u/user). Para el grupo (g/group). Para el resto (o/others).

Terminal: bash/c-shell

Es un lenguaje de programación que sirve para manejar a la computadora a través de cómandos básicos. Generalmente, se utiliza para configurar los sistemas, para manejar los dispositivos, manejo de usuarios, etc.

Lenguajes de programación: Interpretés y Compiladores.

Existen dos tipos de forma de dar las instrucciones en una terminal bash:

1. Interactiva: Ejecutan cada instrucción.
2. Script: Es un programa (conjunto de instrucciones) que se guarda en un archivo. El archivo/file debe tener permiso de “ejecución”.

ls: list

El comando ’ls’ produce una lista de los archivos del directorio donde nos encontramos.

```
$ ls [directorio]
```

nos lista los archivos dentro del directorio especificado.

```
$ ls [file]
```

nos lista el file especificado (si es que se encuentra alli o da un error si el file no existe).

Modificatorios/Opciones del comando:

```
$ ls -t
```

Lista los archivos ordenados por el último modificado primero.

Generalmente las opciones o modificatorios van con un ’-’

Atributos de un archivo

```
$ ls -l
```

Realiza la lista de archivos con los atributos de cada archivo.

Atributos de un archivo:

- Permisos (lectura, escritura, ejecución).
- Usuario y grupo (de usuarios) del archivo
- Tamaño de los archivos en bytes.
- Última fecha de modificación del file

El asterisco (comodín del poker)

```
$ ls a*
```

Lista todos los archivos que comienzan con 'a'

```
$ ls /usr/*.txt
```

Lista todos los archivos con extensión txt que se encuentran en el directorio usr

```
$ ls /usr/a*.txt
```

Igual que el anterior pero solo los que empiezan con a

```
$ ls /usr/a?.txt
```

El signo de interrogación es un comodín de un solo caracter.

cd: Change Directory

cd cambia el directorio.

Si queremos ir a un directorio específico:

```
$ cd /usr/local
```

Si queremos volver en un directorio de /usr/local a /usr:

```
$ cd ../
```

(los dos puntos vuelven para atrás en la estructura de directorio).

Si queremos ir a nuestro home directory /home/[usuario] usar la tilde:

```
$ cd
```

o

```
$ cd ~
```

Si queremos ir a algun directorio de nuestro home,

```
$ cd ~/programacion/guia1
```

(la tilde es una abreviatura de /home/usuario)

Mover, copiar, borrar archivos

Para mover un archivo o directorio: mv (move)

```
$ mv [file] [directory]
```

```
$ mv [directory] [directory]
```

Para copiar un archivo o directorio: cp (copy)

```
$ cp [file] [directory]
```

Para borrar archivos o directorio: rm (remove) NO se recomienda

```
$ rm -i [file]
```

La opción -i hace interactivo (pregunta si realmente lo quieren borrar).

Elimina totalmente los archivos o directorios (no se pueden recuperar).

Alternativa es: mover al trash [Directorio con archivos que se borran]

```
$ mkdir ~/trash
```

```
$ mv file ~/trash/
```

Varios

- man man=manual, nos da información sobre el comando, la instrucción e.g. man ls.
- mkdir Crea un directorio. MaKe DIRectory
- rmdir Borra un directorio. ReMove DIRectory
- pwd: muestra el nombre del directorio de trabajo (Print Working Directory)
- touch: Si el archivo no existe, lo crea. Si ya existía le cambia el tiempo al actual.
- echo: Imprime/Muestra en pantalla. echo 'Hola' echo Hola echo $HOLA
  Cualquier palabra con un $ se esta refiriendo a una variable del sistema $var (variable var). Entonces imprime el contenido de la variable 'HOLA' (si no esta definida no imprime nada).

Comando chmod

chmod: Change file mode. Cambia los permisos de un archivo.
Obviamente solo nos permite cambiar los permisos de los archivos que somos propietarios.
e.g. chmod u+w file, Primer carácter: a,u,g,o Segundo carácter: + o -
Tercero: r,w,x,X

chown: Change owner nos permite cambiar al propietario de un archivo
`$ chown usuario:users programacion` [Hace que programación pertenezca al 'usuario' y al grupo 'users'] (solo para archivos dentro de nuestro directorio)

Comando find

find Encontrar, es decir se utiliza para buscar archivos o directorios.

```
$ find /home/[usuario] -name "*.txt"
```

En este caso busca todos los archivos que tengan como extensión txt en nuestro home directory.

Comando grep

grep Busca en un archivo o grupo de archivos, una palabra o patrón.
Sintaxis: `$ grep -[opciones] [patron] [archivo]`

Supongamos queremos encontrar en un conjunto de archivos donde tenemos la información del experimento “Maxwell”.
Entonces lo que hacemos es buscar el patron: ’Maxwell’ en los directorios exp*.txt.

```
$ grep ’Maxwell’ exp*.txt
```

Este comando imprimirá como salida todas las líneas en los archivos exp*.txt que tengan la palabra Maxwell (El linux distingue entre mayúsculas y minúsculas)
Si se quiere buscar palabras con mayúsculas o minúsculas (ambas), maxwell, MAXWELL o Maxwel, agregar la opción -i.

Redireccionamiento de la salida de un comando

Existen comandos para redireccionar la salida de un comando.

'>' Manda la salida de un comando a un archivo:

Ejemplo:

```
$ ls > listado.txt
```

Guarda la salida del comando en un archivo llamado listado.txt. Si no esta lo genera al archivo. De lo contrario lo sobreescribe.

'|' manda la salida a otro comando Ejemplo:

```
$ ls *.txt | grep 'Ejercicio'
```

Busca en todos los archivos .txt la palabra Ejercicio. Es decir lo que hace el comando | es pasarle la lista de archivos encontrados al 'grep'.

Script

Para generar un script abrimos/creamos un archivo, prueba.sh (la extensión .sh nos indica por convención que esta escrito en bash/shell, pero se puede usar cualquier cosa)
Luego conviene poner en la primera línea del archivo:

```
#!/bin/bash
```

Esta instrucción dice que lo que viene esta en lenguaje de programación 'bash'.
Script para que la computadora salude y se presente: saludo.sh

```
#!/bin/bash
echo 'Hola. Soy Alexa'
```

Ejecución del script

Por otro lado se deben asegurar de que el archivo tenga permiso de ejecución.

```
$ chmod a+x saluda.sh
```

Luego para ejecutarlo solo tienen que hacer:

`$ ./saluda.sh` (si estan en el mismo directorio)

Si estan en cualquier lugar:

```
$ ~/programacion/saluda.sh
```

(Se pone todo el path o camino)

Scripts con variables de entrada

Modificamos el programa anterior para que interactue con el usuario.

```
#!/bin/bash
echo 'Hola '$1'. Yo soy Alexa'
```

Ejecutando como: ./saluda.sh [tuNombre]

Es decir, que $1 es una variable de entrada en el script.

Cálculos con el bash

Los cálculos en bash NO son recomendados. Vamos a utilizar python que es mucho mas potente. Bash tiene una forma bastante primitiva de cálculos:

```
$ myvar = 2
$ expr $myvar + 1
$ expr $myvar / 3
$ expr $myvar \* 3
```

Acceso remoto a un servidor

- Las máquinas que estan en linux pueden accederse en forma remota.
- Se puede trabajar y ejecutar programas en forma remota.
- En general se accede a una terminal y solo se utiliza esta.
- Se puede exportar aplicaciones con ventanas. Pero hay que evitarlas! (por la velocidad de la red).

Es decir que todo lo que podemos hacer lo tenemos que hacer en una terminal!

secure shell (ssh) desde windows

- El protocolo que se utiliza actualmente para acceder a una computadora o servidor remoto es ssh (secure shell).
- Es una shell como la que conocemos excepto que transmite la información encriptada por seguridad.
- Desde windows se puede acceder a maquinas linux instalando aplicaciones en general libres.
- Una muy recomendable para esto es PUTTY.
- Desde el linux se puede usar desde una terminal y hacer ssh (Instalar la aplicación si no se encuentra)

Domicilio de los servidores remotos (o nombre identificatorio)

- Existen dos tipos de nombres para los servidores en la red. Uno basado en caracteres (nombre de dominio) y otro basado en números.
- Numero de Internet Protocol (IP). e.g. 171.25.10.155 (4 números cada uno de 8 bits es decir desde 0 hasta 255 aunque hay numeros reservados por protocolo).
- Por nombres: (estos estan definidos en un domain name server: servidor de nombres de dominios). e.g maquina.unne.edu.ar
- Los nombres siguen una estructura de dominios (gran dominio ar, luego edu etc).

Cambio de clave

La primera actividad que deben hacer en el servidor es poner una clave personal y privada.
Debe contener 8 caracteres como minimo, mayusculas, minusculas y numeros (1 o 2 a lo sumo).

Transferencia de archivos

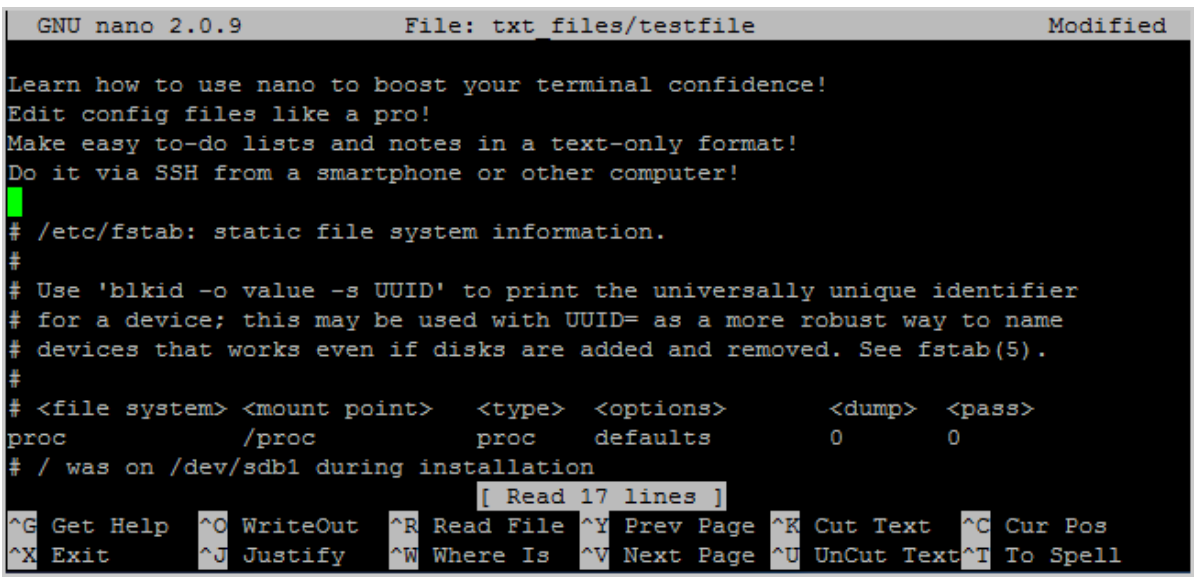
Si uno quiere pasar archivos entre la computadora que esta trabajando y el servidor se puede utilizar scp / sftp. Son protocolos de transferencia de archivos encriptados.
En general como el servidor es el que tiene una direccion publica siempre usan el comando desde su computadora.

- Para transferir desde la computadora al servidor: `scp archivo.txt 200.45.54.94:directorioservidor/`
- Para transferir desde el servidor a la computadora: `scp 200.45.54.94:directorioservidor/archivo.txt ./`

En el caso de windows pueden instalar PuTTY scp que es para transferir archivos (pscp).

Editores de textos en una terminal

- Para programar se requiere de un editor de textos.
- Los editores para programar son: nano rpvi o vim, e emacs.
- El mas sencillo es nano (recomendado)
- Para programadores mas experimentados: vi o vim, emacs.
- Puede ser para uso local. Linux incluido. Windows: https://files.lhmouse.com/nano-win/
- Se puede editar un programa en forma remota en el servidor.
- Para editar un archivo (existente o nuevo) hacen en la terminal: nano /directorio/archivo



Nano

Abajo tienen los shorcuts en general Ctrl + algo (o con las Funciones).

- Ctrl + g es la Documentación.
- Ctrl + o escribir (grabar)
- Ctrl + x Salir de nano (pueden grabar los cambios).
- Ctrl + w Buscar palabra o texto.

Copia y pegada de textos:

- Ctrl + 6: Para marcar un pedazo de texto
- Alt + 6: Para copiar el texto marcado
- Ctrl + k cortar texto.
- Ctrl + u pegar texto copiado/cortado
- Alt + m habilita el copiado y pegado con el mouse (al menos en linux)

Trabajando en forma remota

Editando multiples archivos en Nano

Alt + f Para trabajar con varios archivos a la vez.

Alt + . Alt + , se usa para navegar entre los archivos.

Ctrl + r Se usa para abrir nuevos archivos (en nuevos buffers).

Se puede copiar y pegar entre distintos archivos abiertos

Múltiples terminales: screen

Desde línea de comando:

```
$ screen
```

- Si quieren crear una nueva terminal: Ctrl + a + c
- Si se quiere volver a la terminal anterior: Ctrl + a + p
- Si se quiere ir a la terminal siguiente: Ctrl + a + n