:

“linux es sensible a las minusculas y minusculas, es diferente una mayuscula que minuscula”

el directorio /dev contiene dispositivos (enlaces a lispositivos del sistema (teclado. Discos etc)

“el directorio /etc guarda la configuracion del sistema”

“cat /etc/passwd :estan todos los usuarios del sistema”

“cat /etc/group : nos muestra los grupos del sistema”

“el directorio /bin son ejecutables de los comandos”

“cuando un programa falla deja un archivo en /var/crash”

para eliminarlos “sudo rm /var/crash/\*”

“ls -l /sbin/init : nos muestra el ejecutable que arranca la maquina”

“prioridades en linux van desde -20 al +19

**-20 tiene mayo prioridad**

**“**sudo fdisk -l : este comanod te muestra los discos de la computadora”

los comando sinternos de una shell son los que no tienen ejecutable, los externos son losque tienen ejecutable en el sistema (la terminal(cmd) es una shell bash)

type comando: nos dice si el comando es interno y externo si es externo nos dice deonde esta su ejecutable

los ejecutables de linux son .sh

en el shell scritp poner como header #!/bin/bash

#!/bin/sh es un enlase simbolico a ‘dash’

“sudo passwd” : para cambiar contrasena de root

uptime : nos da datos del funcionamiento

uptime -p: tiempo q paso desde que se arranco la maquina

uptime -s: momento esacto q paso desde que arranco la maquina

**PONER ñ en linux**

ctrl + shift + u + unicode code of character

00f1 (cero ceor efe uno) para ñ

00d1 (cero cero de uno) para Ñ

“para usar ssh es necesario instalar el cliente y el servidor ssh”

sudo apt-get install openssh-server openssh-client

“ssh [nombre\_usuario@numero\_ip](mailto:nombre_usuario@numero_ip)” : asi se hace acceso por ssh

COMANDOS RASPBERRRY

“sudo raspi-config” : entre a la configuracion de las raspbery

“sudo i2cdetect -y 1” : muestra las direcciones de los i2c de la raspberry

**HACER ZOOM EN TERMINAL**

**-**para hacer zoom en la terminal, posicionados en terminal usamos los siguientes comandos

ctrl + shift + +

-para agrandar

ctrl + -

-para alejar

ctrl + )

-para poner neutral (original)

**CREAR OTRA CUEN LINUX**

**1.** su: entrar como usuario root (si no hay root usar “sudo su”, contrasena es la de la compu)

2.useradd -m -d /nombre nombre: ponemos el nombre del nuevo usuario,

useradd -m nombre (usar solo este por lo general)

el -m crea su propio directorio

el -d le da un directorio diferente

3.passwd nombre: nos pedira la contrasena del usuario nombre dos veces

4.userdel nombre: borrar al usuario nombre

**DESACTIVAR FIRAWALL**

sudo ufw disable: desactiva el firewall

sudo ufw enable: activa el firewall

**COMANDOS DE LINUX**

pwd: avisa en que librería estamos.

Man: nos da una ayuda sobre el comando que va despues

ls -l: listado detallado

ls -r: ordena los ficheros de forma descendente según orden alfabetico

ls -d: nos muestra los nombres de los directorias sin su contenido

ls -a: nos muestra los ficheros ocultos tambien

los ficheros ocultos empiezan con un “.”

ls -al: es la combinacion de los dos anteriores

ls -S: ordena por tamano del ficheros

ls -t: ordena por tiempo

man -f \_\_\_: nos da una breve descripcion del comando siguiente

file \_\_: nos dice el tipo de fichero es el siguiente

WILDCARDS (metaracteres en ficheros)

\*: sustituye a un grupo de caracteres( ls b\*: nos muestra too los ficheros que empieza por b)

?:solo sustituye a **UN** carácter

[ ]: es “o” (ls [asd]\*:mostrara los archivos que empiecen con a, s o d)

(ls \*[0-2]: los ficheros que terminen en caracteres entre 0 y 2)

(ls \*[a-c]: los ficheros que terminen en caracteres entre a y c)

INFORMACION SOBRE EL FICHERO

-cuando hacemos el comando “ls -l” en la primera columna nos da las espicificaciones del archivos

el primero es el tipo

d: es un directorio

-: son ficheros normales(texto,ejecutable,binario)

l: son enlaces o acceso directo a otro archivo

estos solo estan en el directorio /dev

b: son tipo bloque

c: tipo carácter

los siguientes son los permisos que tienen los ficheros

habla de lo que hay dentro

si es 1 es un archivo (tipo -)

si es tipo – y el numero es mayor restamos 1 y el resultado es el numero de enlaces (fisico o simbolico) apuntandole

si es 2 es un directorio (tipo d)

si es tipo d y el numeor es mayor restamos 2 y el resultadoe s el numero de directorios dentro del mismo

el siguiente es el propietario

es el grupo

feche de modificacion o creacion

nombre

**CREAR FICHEROS**

mkdir: crea dirctorios

mkadir – p dir1/dir2: donde estemos, crea un directorio dir1 y dnetro de este crea dir2

rmdir: borra directorios

rm: borra ficheros

tambien funciona con meta caracteres

rm -r dir1:borra los direcctorios con sus contenidos

touch: crea ficheros si no existe, y si existe cambia la fecha de creacion

echo hola: manda la palabra hola a la salida estandar (pantalla)

echo -ne hola : imprime hola pero no hace enter (backslash \) de salto de lines y lo siguiente se imprime alado

echo -ne ‘hola \r’ : el \r hace que el cursor vuelva al principio

echo prueba > f1.txt: crea el fichero f1 si no existe y **REESCRIBE** en el la palabra “prueba”

echo prueba >> f1.txt: **AUMENTA**  al fichero la palabra prueba

cat f1.txt: muestra el contenido del fichero

cat -n f1.txt: nos muestra enumeando las filas

tail f1.txt: muestra las 10 ultimas lineas del ficheros

tail -n 20 f1.txt: muestra las 20 ultimas lineas

tail -f f1.txt: se mantiene funcionando y muestra todo lo que se vaya anhadiendo al ficheros

head f1.txt: muestra las primeras 10 lineas

head -n 20 f1.txt: muestra las primera 20 lineas

cp (fichero origen) /(directorio destino)/(fichero destino): copia un fichero si se le da ruta de destino, o si se le da otro nombre solo lo copia con el nuevo nombre

cp -i: pregunta si quiere sobreescribir en caso de ya existir un fichero con ese nombre

cp f\* (destino): copia todos los ficheros que enpiecen con “f” al destino

funciona con los metacaracteres

cp -r (directorio origen) (directorio destino) : con -r copia todo un directorio y lo que esta dentro

mv: mover o renombrar ficheros

\*: tambien funciona con metacaracteres

**EDITOR DE TEXTO**

less f1..txt: entra en un editor de texto

ES SOLO DE LECTRURA

q:sale de less

**INODOS**

son numeros que tienen los datos de un ficheros

tamanho

direccion de datos

usuario

grupo

ls -li f1.txt: nos dice el numero de inodo

enlace fisico (hard):es como una copia con otro nombre del mismo ficheros

el fichero sigue vivo si esq no se borran todos los enlaces

los enlaces se van actualizando al mismo tiempo

los enlaces creados son ficheros de tipo normal (-)

ln f1.txt f1\_enlace.txt: crea un enlace tipo fisico que se cambia uno de ellos

enlace sibolico (soft): es como un acceso director

si se borra el original, los enlaces ya no funcionan

los enlaces se van actualizando al mismo tiempo

los enlaces creados son de tipo link (l)

este enlace creado adquiere un nuevo inodo

ln -s f1.txt f1\_enlace.txt: crea un enlace tipo simbolico

**TAMANO DE DISCOS**

df : poniendo solo df nos muestra informacion de las particiones (discos) del sistema (tamano total,usado, libre)

-este comando list tanto los discos fisicos (pendrive o externos) , logicos(particiones) y unidades virtuales temporales.

df -h:nos lo pone en un formato entendible

df -h dir\_name : nos da la informacion solo del disco donde este el directorio pasado

df . -h : esto nos muestra los datos de la unidad montada al sistema (el disco duro prncipal)

-esto nos da las siguientes columnas

-filsystem : este es el nombre de cada unidad

-size : tamano del directorio

-used : tamano usado del directorio

-avail : tamano libre disponibles

-use% : porsentaje ocupado del directorios

-mounted on: este el path de donde esta ubicado el disco

**TAMANO DE FICHEROS**

du dir: te da el tamano de los subdirectorios dentro de dir

du -a dir:saca el tamano de cada uno de los ficheros dentro de dir

du -s dir: saca el tamano de los directorios directos sin entrar en cada uno de ellos

du -am -t 1m:pregunta por todos los ficheros que sean mayor a 1mega

du -sh . :no pasandole punto nos dice el tamano del directorio donde estoy

du -sh \* : pasando asterisco nos dice el tamano de los subficheros de donde este

**USAR MAS DE UN USUARIO (**terminales virtuales)

ctrl + alt + F3: entrar a otro usuario

ctrl + alt + F2: volver al usuario original

who:nos dice quien esta conectado a nuestro usuario en ese momento

**EDITOR DE TEXTO**

gedit nombre.txt: crea un archivo de texto con el nombre.txt

**BUSCAR FICHEROS EN EL PAD DEL SISTEMA**

which fichero: nos da la ruta a la ubicación de este fichero en el sistema

whereis fichero: nos avisa donde esta el ejecutable, manual y las librerias que usa

**CREAR COMANDOS**

alias nombre=’comando’: crea un comando con el nombre que hace lo que se ponga en comandos

alias nombre: nos dice que comandos hace el alias nombre

**TIEMPO**

time comando: nos dice cuanto tiempo le toma el proceso del comando

-este comando se lo puede utilzar con python u otros

time python3 programa.py

-esto nos devolvera cuanto tarda en ejecutar el programa

**calendario**

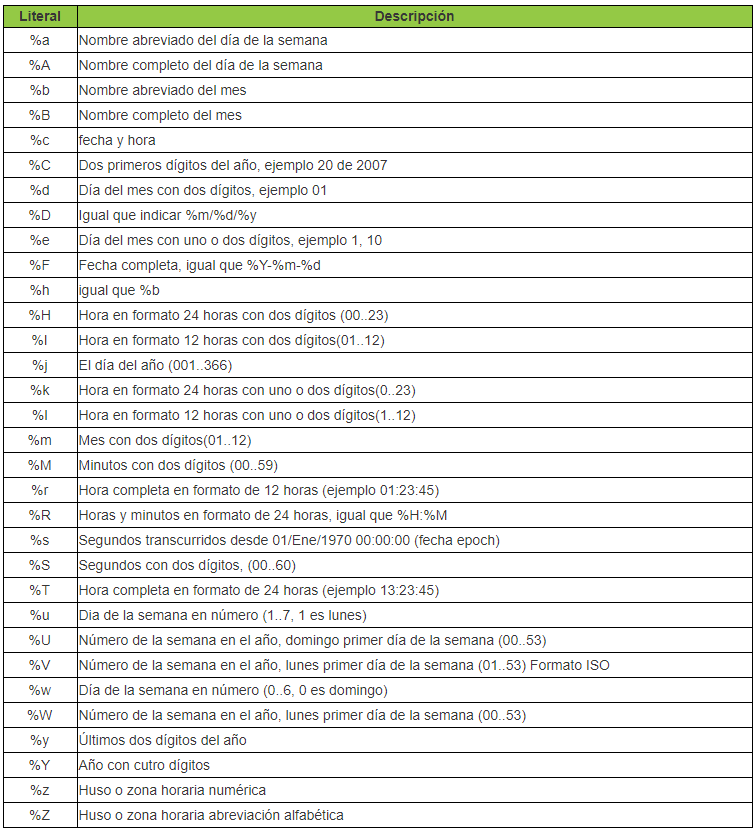
cal: nos da la fecha

cal -m 8 1999: nos da el mes 8 del ano 1999

ncal: formato anglosajon (desde lunes)

date: nos da dia, hora, fecha

date +'hoy es el dia %d del mes %B del ano %Y y la hora de %k:%M:%S'



**ENTRADAS Y SALIDAS**

STDIN(0)=entrada desde el teclado

STDOUT(1)=salida en la pantalla

STDERR(2)=salida de errores

tr a o:cambia la “a” por la “o”

tr aeiou AEIOU f1.txt : sustituya la ‘a’ po ‘A’, ‘b’ por ‘B’ y asi carácter por carácter

tr a-z A-Z f1.txt : convierte todo en mayuscula

tr -d a f1.txt : borra todas las a del fichero

tr -s a f1.txt : borra todas las a repeticas

tr -c a o f1.txt : todo lo que **no** es a sustituye por o

REDIRECCIONAR ENTRADA

tr a o < f1.txt: cambia las “a” por la “o” en el fichero f1.txt

tr a o << final : recibe palabras del teclado y sustituye las “a” por “o” cuando en lee “final”

en lugar que la entrada sea el teclado, la entrada es un ficheros

wc -l < f1.txt : cuenta cuantas lineas hay en el fichero f1.txt

wc -l << final : cuenta lineas hasta que aparesca la palabra “final”

REDIRECCIONAR SALIDA

echo hola > f1.txt: manda hola a la salida que es el fichero f1.txt

date > f1.txt:manda la fecha a fichero f1.txt

en lugar que la salida sea la pantalla, la salida es un fichero

ping 8.8.8.8 | tee ping.txt : todo lo que salga de ping lo envia a la pantalla **Y** al fichero ping.txt

salida de errores

error 2> error.txt : ponemos el 2 antes del > para q guarde el mensaje de error

error > salidanormal.txt 2> error.txt : la salida estandar lo manda a salinormal.txt y el erro a error.txt

error &> error.txt : manda tanto la salida estandar como el error a error.txt

error > /dev/null :lo borra el mensaje

para guardar un error pon

**TUBERIAS**

cat f1.txt | tr a o : dice que la salida del primer comando lo manda como entrada para el segundo comando

**FILTROS**

uniq f1.txt : filtra todas las lineas **consecutivas** iguales y las vuelve 1 linea

uniq -c f1.txt: nos avisa cuantas veces se repite cada linea

uniq -d f1.txt: nos avica que lineas estan repetidas y nos entrega esas lineas

wc f1.txt: nos cuenta

wc -l f1.txt: nos dice cuantas lineas hay

wc -w f1.txt: cuantas palabras hay

wc -m f1.txt: cuantos caracteres

wc -c f1.txt: cuantso bytes hay

grep palabra f1.txt :busca palabra en fichero f2.txt no puede entrar en directorios y tira warning

grep -s palabra f1.txt: no lansa los errores de los directorios y solo lanza resultados

grep -l palabra f1.txt: solo nos dice en que ficheros hay la palabra

grep -c palabra f1.txt: nos idce cuantas vecces aparece la palabra en cada ficheros

grep -i palabra f1.txt: para no separe mayusculas de minusculas

sort f1.txt:ordene resultados por orden alfabetico

sort -r f1.txt:ordena en forma alfabetica ascendente

sort -n f1.txt: ordena por numeros

sort -k2 : ordena según lo que haya en el campo 2

cut :busca espacios de forma de TAB

cut -f1 fichero.txt: estamos diceindo que nos devuelva el primer campo

cut -d ‘:’ -f1 fichero.txt: nos da el primer campo del fichero.txt

cur -d ‘:’ -f1,6 fichero.txt : nos saca el campo 1 y 6

cut -c 25-29 fichero.txt : le decimos que nos extraiga los datos que estan entre la posicion 25 y 29

diff

**PROCESOS**

jobs: nos muestra los procesos que se estan llevando acabo en el bash actual

jobs -l : nos muestra el identificador del proceso

ps: muestra los procesos que se estan llevando a cabo en el sistema

“espacio PID : es el identifacador del proceso”

“ espacio TTY: es la terminal de la cual se esta ejecutnado”

“espacio CMD : que comadno se esta corriendo”

ps -f : nos muestra los procesos ejecutandose en el bash actual(en esa ventana abierta

“espacio PPID: proceso padre el cual ejecuto este proceso hijo”

“espacio NI: nos dice la prioridad

ps -l : nos muestra mas detalles

“espacio F : nos dice los permisos que tiene el procesos”

“espacio S : nos dice el estado del proceso “

“espacio PRI: prioridad”

ps -ef: nos muestra too los procesos del sistema

ps -ef | grep 1234 : podemos buscar los proecesos por us identificador hasta llegar al padre original

ps -o pid, ni ,comm: nos mues los espacios pid ni y comm

ps -uxf: procesos que se estan ejecutando con el usuario curso

pstree: nos muestra en formato arbol todo los procesos de padre a hijo

pstree -p : nos muestra con los identificadores

pstree -s 123: muscamos toda la jerarquia del proceso con el identificador 123

PRIORIDADES DE PROCESOS

top : nos muestra los procesos en tiempo real

despues de dar top

^M : ordena por memoria cual proceso consume mayor memoria

^P : por consumo de CPU

c : muestra comando completo

INFORMACION DEL CPU

lscpu: este omando nos da la informacion del cpu

CAMBIAR DE PRIORIDADES A PROCESOS

(sleep 5: hace que la shell se quede esperando 5 segundod)

nice -10 sleep 10 & : asigna una prioridad al momento de correr

renice -n 10 1234 : el proceso con el ID 1234 cambia a prioridad 10

MATAR PROCESOS

para matar procesos mandamos senales a un procesos

kill -l: nos muestra las senales que hay en el sistema

man signal: para ver mas comandos de senales

kill -9 1234 : es para matar un procesos 1234 bruscamente

kill -15 1234: termina un proceso, no siempre termina, depende de prioridades

guarda lo necesario y termina el proceso mas limpio

PONER PROCESOS EN SEGUNDO PLANO

sleep 1000 &: pone el proceso sleep en segundo plano

nos entrega el numero de proceso entre corchetes [1] y el identificador del proceso

con procesos que muestran cosas en la pantalla por mas que lo pongamos en backgrpound sige lanzando cosas a la pantalla

du -am /: lanza los tamanos de todos los directorios de todo el sistema

du -am / > f1.txt 2> error.txt & :

du -am / &> f1.txt &

ponemos en segundo plano, y todos los mensajes y errores se guardan en ficheros y ya no nos lanza nada en la pantalla

fg 1 : nos pone en primero plano el proceso con el numero entre corchetes que puesimos en background

**USUARIOS**

id usuario: nos muestra las carateristicas del usuario

**PERMISOS**

acl: permisos de usuarios para usuariso

selinux: permisos para usuarios mas complejos



espacio d : nos dice el tipo de archivos

los 9 siguientes nos dice los permisos del fichero

los 3 primeros: son permisos de usuarios

los 3 siguientes son permisos de grupo

los 3 siguientes son permisos al mundo

son 3 tipos d permiso de **ficheros**

r (read): el fichero se puede leer copiar o vizualizar

w (write): se puede modificar, borrar o cambiar permisos

x (ejecucion): si el fichero es ejecutable

permisos para **directorios**

r : se puede ver el contenido (se le puede hacer ls)

w : se puede crear, borrar o modificar directorios o ficheros dentro de el

lo que este dentro de el puede que no se pueda modificar

x : si puedo entrar en el directorio

CAMBIAR PERMISOS

se peude cambiar de dos formas por carácter o octal

FORMA CARACTER

chmod u+w directorio: cambia los permisos + para para aumentar el premiso y – para quitar

chmod u+x directorio: da permiso de usuario de ejecucio

chmod g+x directorio: cambia permisos para el grupo

chmod o+x directorio: cambia permisos para otros

chmod a+x directorio : cambia permisos a todos

FORMA OCTAL

0=- - -

1=- - x

2=- w -

3=- w x

4=r - -

5=r - x

6=r w -

7=r w x

chmod 124 directorio: nos pones usuario ejecucion, grupo escritura y otros lectura

CAMBIAR PERMISOS POR DEFECTO

umask:nos muestra el umask de nuestra maquina

directorios

maximo valor octal:777

umask:022 (defecto)

777 – 022 = 755= r w x r – x r - x=el octal por defecto para directorios

ficheros

maximo valor octal: 666

umask:022 (defeccto)

666 – 022 = 644= r w – r - - r - -

sacamos umask nuevo para la nueva configuracion que querramos

umask 002: cambia el umask por defecto (022) a 002

CAMBIAR PROPIETARIO DE FICHEROS

sudo chown ususario fichero : cambia nuestro fichero al propiedad del usuario

sudo chgrp grupo fichero : cambia de nuestro fichero al grupo que pongamos

**VARIABLE DE ENTORNO**

printenv : nos muestra las variables de entorno

set : nos muestra las variables de entorno y las de shell

echo$PWD : con el signo de dólar decimos que nos muestra el valor de la variable PWD

$PWD : guarda la direccion donde estamos con la shell

$DISPLAY

$HOME : direccion de home

$SHELL : la tipo de shell que se esta guardando

$LANG : el lenguaje de la

$PATH : muestra los directorios donde encontrar comandos (ejecutables

solo en estas direcciones busca variables

cuando creamos ejecutables los tenemos que agregar la direccion de la carpeta donde esta los ejecutables q creamos

set PATH=$PATH :nueva direcciones

a la variable PATH le guardamos lo anterior juentp nuestra nueva direccion

$PS1 : muestra el pront del bash

$USER : muestra el usuario que somos

FICHEROS QUE CARGAN LOS EJECUTABLES

estas son variables para todo el sistema ( todos los usuarios pueden verlos)

/etc/profile : carga las variables de login al sistema

/etc/bash.bashrc : carga las variables de entorno de bash

/*etc*/bash\_profile : en otros sistemas se puede llamar asi

estas variables solo las puede ver un usuario en especifico,ya que tienen punto al inicio estan icultos

.profile : son las variables de entorno

se ejecuta solo cuando se inicia el sistema

.bashrc : son las variables de bash

se ejecuta cada vez que se abre un bash

**BUSCAR FICHEROS EN EL SISTEMA**

locate f1.txt : busca en una base de datos

/*var/*lib/mlocate/mlocate.db : es una base de datos que actualiza cada dia con los ficheros del sistemas

sudo updatedb : actualiza esta base de datos

locate -b f1.txt : solo busca en ficheros

locate -c f1.txt : nos dice cuantos ficheros o directorios hay con el nombre f1.txt

locate \f1 : busca los nombres solo con el texto f1

find : busca en todo los sistemas de directorios, entonces puede tardar mas que locate

find /directorio\_inicio -name f’1.txt’ : busca el fichero f1.txt a partir del directorio\_inicio

find /directorio\_inicio -not -name ‘f1.txt’ : busco todos los ficheros que no tengan la palabra f1.txt

find /directorio\_inicio -type -f: busca los que son ficheros

-d : directorio

-l : enlaces

c : tipo caracteres

monitor, teclado

b : tipo bloque

disco duro, memoria

find /directorio\_inicio -atime : fecha mas reciente acceso

-mtime : fecha mas reciente de creacion o modificacion

find /directorio\_inicio -atime +2 : hace mas de dos dias

1 : hace un dia esactamente

find /directorio\_inico -size +5k : busca archivos cn mas de 5 kilobytes

c : en bytes

k : en kilobytes

M : en megabytes

G : en gigabytes

b : trozos de 512 bytes

find /directorio\_inicio -user usuario : busca los ficheros pertenecioente al usuario

find /directorio\_inicio -group grupo : busca ficheros pertenecientes al grupo

find /directorio\_inicio -perm 642 : los ficheros que tengan esos permisos en octal

**EMPAQUETADOR**

tar cvf paquete.tar \*: sol empaqueta , pone varios ficheros dentro de uno, pero no comprime

tar cvfz paquete.tar.gz fichero : lo empaqueta todo y ademas comprime con el gzip

tar xvf paquete.tar.gz : descomprimimos y desempaquetamos un fichero que sea .tar.gz

tar tvf paquete.tar fichero : nos lista el contenido del .tar

tar xvf paquete.tar fichero : estraemos le contenido del .tar

con el \* significa que empaquete todo lo que este en nuestro directorio actual

c :crear

x : extraer

l : para listar

v : ver el avance

f : file

t: lista el contenido del .tar

**COMPRESOR**

gzip fichero : comprime **UN** fichero y borra el original y entrega un .gz

gzip -k fichero: comprime y **NO** borra el original

gzip -l fichero : muestra el porcentaje de comprecion

gunzip fichero : descomprime el fichero

gunzip -l fichero.gz : listo el contenido del .gz sin descomprimirlo

bzip2 fichero : comprime ficheros mas grandes, trabaja igual que gzip entrega un .bz2

bunzip2 fichero.bz2 : descomprime el fichero bzip2

zip compresion.zip fichero1 fichero2 fichero3 : comprime todo y no borra los originales

unzip compresion.zip : descomprime el .zip donde estemos

unzip -l compresion.zip : lista el contenido del .zip sin descomprimirlo

**ADMINISTRADOR DE TAREAS**

(free : permite ver la memoria libre)

watch -n 2 ‘comando’ : corre de nuevo el comando que pongamos cada 2 seg

CRONTAB

crontab: configurar trabajos o proyectos en un momento dado en el tiempo

cat /etc/crontab : muestra el archivo crontab donde esta la configuracion de que y cuando ejecutar

este dichero solo lo puede modificar root , nosotros nos tenemos que crear otro crontab

crontab -l : nos muestra los usuarios del crontab

crontab -e crea y modifica el crontab y nos lo guarda en el lugar ocrrecto

crontab fichero\_creado : este comando nos manda el fichero en formato crontab q tengamos

crontab -ri : borra el crontab y nos pregunta si queremos borrarlo

2 variables

SHELL: que shell utilizara para ejecutar

PATH: donde esta el ejecutable a usar

espacio a llenar

m : minuto en el que se ejecutara el script/comando

va del 0 a 59 si ponemos 1,2,3,4 lo hace al minuto 1 2 3 4 si ponemos 1-4 lo hace al minuto

h : la hora del dia que se ejecutable

va del 0 a 23 horas

dom : dia del mes a ejecutarse

mon : mes a ejecutarse

va del 1 al 12

dow : dia de la semana a jecutarse

va del 0 al 6 donde 0 es domingo

user : usuario que ejecutara

solo ponemos si somos root

command : comando o script que se ejecutara

FORMATO PARA EL CRONTAB

# m h dom mon dow user command

\* \* \* \* \* date >> ~/Escritorio/dir1/prueba\_crontab

**GESTION DE USUARIOS Y GRUPOS**

cat /etc/passwd : cada linea es por usuario

campo 1: nombre de usuarios

campo 2 : contrasena encriptada

campo 3 : identificador de usuario (UID)

campo 4 : identificador de grupo (GID)

campo 5 : comentario

campo 6 : directorio de carpeta de usuario personal

campo 7 : identifica la shell del usuario

sudo cat /etc/shadow :

campo 1 : nombre de usuario

campo 2 : contrasena encriptada

campo 3 : caundo se cambio la contrasena por ultima vez

el numero que sale son los dias pasados desde el 1 del 1 de 1970

campo 4 : cuantos dias tengo que esperar para cambiar la contrasena

campo 5 : dias que dura las contrasena

campo 6 : cuantos dias antes que caduque la contrasena nos avisa

campo 7 : dias pasados de la caducidad de la cuenta para que esta se de de baja

campo 8 : fecha de caducidad (poner dias desde el 1-1-1970 para poner fecha)

chage -l usuario : nos muestra los datos de ese usuario legible para el humano

cat /etc/group :

campo 1 : nombre de grupo

campo 2 : contrasena del grupo

campo 3 : identificador del grupo

campo 4 : usarios perteneciente al grupo separados por coma (,)

CREAR GRUPOS

sudo groupadd -g 3000 nombre\_grupo : creamos un grupo llamado nombre\_grupo y con identificador 3000

no menor de 1000

MODIFICAR GRUPO

groupmod -n nuevo\_nombre -g 123 nombre\_grupo

-p : poner contrasena

-g : modificar el identificador del grupo

-n : cambiar nombre del grupo

BORRAR GRUPO

sudo groupdel grupo : borrar grupo

-f: si un usuario tiene este grupo como principal le asigna otro y borra el que queremos

CREAR USUARIO

sudo useradd -u 3000 -d /home/usuario -g 2000 -s /bin/bash -c ‘usuario normal’ nombre\_usuario

-d : decimos cual es el directorio personal del usuarios

-u : identificador del usuario

-g : grupo principal

-G : grupos adicionales

-s : consola shell que suara

-c : comentario

-p : contrasena

passwd usuario: cambia la contrasena del usuario, te pedira la vieja y la nueva contrasena

-l : bloquea al usuarios

-u : desbloquea la contrasena

-s : estado de la contrasena

MODIFICAR USUARIO

sudo usermod : modificar usuario

se pone en lugar de useradd y ponemos lo espacios que queremos cambiar y al final el nombre del usuario del que queremos cambiar

recomendación ; no modificar el id del usuario

BORRAR USUARIOS

sudo userdel usuario: borra el usuario

-r : borrar los directorio y correo

**COMANDO SUDO**

el comando sudo nos permite ejecutar comandos como otro usuario

cat /etc/group | grep sudo : nos muestra los usuarios que pertenecen al grupo sudo y los que estan en el pueden ejecutarse como root

sudo -u usuario2 comando:ejecuta este comando com el usuario

su – usuario2 :crea un nuevo entorno y entras como usuario

su usuario2 : entra como el usuario2 peor arrastra

**EDITOR VI Y VIM**

distintos tipos de trabajo

modo comando:

ctrl + g : nos muestra el nombre del fichero que estamos y mas cosas

poner numero antes de un comando repite ese numero de veces el comando

moverse con flechas

h: izquierda

j: abajo

k: arriba

l:derechas

w: moverse de palabra en palabra hacia delante

b: palabra a palabra hacia atrás

$ : final de la linea

^ : inicio de linea

ctrl + f: pagina adelante

ctrl + b : pagina atrás

ctrl + d : media paguina adelante

ctrl + u : media pagina atrás

modo edicion:

edita texto

cuadno esty en una linea en blacon

i : entro en modo edicion a la izquierda del cursor

a : entra en modo edicion a la derecha del cursor

escape (esc) : sale del modo edicion

shift + i : entra edicion al principio de la linea

shift + a: entra en modo edicion al final de la linea

o : aumenta linea en blanco debajo donde estemos e inicia edicion ahi

shift + o : aumenta linea en blanco arriba donde estemos e inicia edicion ahi

x : borra hacia delante

shift + x : borra hacia atras

dw : borra desde donde estemos hasta el final de la palabra, para luego pegar

dd :suprime toda la linea, para luego pegar

u : deshace la ulitma operación que hicimos

r +letra: sustituye la letra donde estemos por la letre que ponemos en el comando

shift + r : sobre todo lo que este por lo que escribamos depues

salimos con escape (esc)

yy: copia toda la linea

yw: copia desde donde estamos hasta el final de la palabra

yb ; copia desde donde estemos hasta el inicio de la palabra

p : pega lo copiado justo debajo donde estemos

shift + p : pega encima de donde estemos

se abre en modo comando

/ palabra: ponemos lo que queremos buscar y damos a enter y el cursor se va a la primera coninsidencia que encuentre con el texto que pusimos

n : va a la siguiente coinsidencia

shift + n : busca la anterior coinsidencia

MODO XE : se entra con :

me situo en la liena que quiero sustituir y apreto : y salgo con u

:s/texto original/texto nuevo/

sustituye la primera coinsidencia en la linea

:s/texto original/texto nuevo/g

sustituye todas las coincidencias en toda la linea

:%s/texto original/texto nuevo/

sustituye la primera coincidencia de cada linea en todo el texto

:%s/texto original/texto nuevo/g

sustiruye toda las coincidencias en todo el texto

:r fichero: nos vamos a una linea en especifico del fichero acutal y hacemos este comando

inserta el contenido del fichero desde desde la linea

vi -O fichero fichero2 fichero3 : abre los tres ficheros en la misma ventana, separandolos por una linea bertical

vi -o fichero fichero2 fichero3 : abre los tres ficheros en la misma ventana, separandolos por una linea horizontales

ctrl + w + flecha : cambia de fichero a fichero

:w

graba todas las modificacione

:q

sale del VI si no tenemos modificaciones por guardar

:q!

Salir de VI sin guardar

:wq

guarda y sale de VI

:wq nuevo nombre

guarda con el nuevo nombre y sale de VI

:x

guarda y se sale de VI

RECUPERAR FICHEROS

cuando el vi se sale corruptamente, por fallos u otro error sin guardar

se genera un “.fichero.swp” con el mismo nombre del fichero que perdimos pero oculto

si intento volver a entrar al fichero nos lanza una advertencia y nos ofrece recuperar la ultima version **guardada** de ese fichero

:recover

recupera la ultima **modificacion** hecha en el fichero

despues es recomendable guardarlo una vez recuperado

despues tendemos que borar el .fichero.swp para que no haya problemas despues

IDENTAR VARIAS LINEAS AL MISMO TIEMPO

shft + v

-esto entra en visual line mode

flecha arriba o abajo

-para seleccionar todas las lineas que querramos identar

< o >

-en mi caso es (shift + ,) o (shift + .) esto para identar izquierda o derecha

.

-presionamos punto si queremos identar mas veces seguidas

OTROS COMANDOS EN VI

:set cursorcolumn

esto mustra una linea vertical a lo largo de la ventana justo donde esta el cursor

:set cursorline

esto muestra una linea horizontal a lo largo de la ventana justo donde esta el cursor

:set nu

activa la numeracion de lineas

:set nonu

desactiva la numeracion de lineas

:123

va a la linea 123

shift + g

va a la linea final del fichero

shift + j

la linea de abajo la pone despues de la que estamos

.

repite el ultimo comadno ejecutado

:set list

-esto muestra todos los caracteres especiales como espacio,salto de linea

:set listchars=eol:$,tab:>-,trail:~,extends:>,precedes:<,space=\_

con este comando configuramos que carácter mostrara donde haya:

eol = salto de linea

tab = tab

space = espacio

:set tabstop=4 shiftwidth=4 expandtab

:retab

con esto cambiamos los tabs por 4 espacios

**SHELL SCRIPT**

**. ./**script.sh : el punto espacio path de un script hace que recuerde las variables del script despues de ejecutar el mismo, y no las borre y olvide al terminar de ejecutarse dicho script

# : comentar una linea

#! : se pone esto en la primera linea para especificar el tipo de shell que correra el programa

#!/bin/bash : el programa se correra en la shell bash

#!/bin/ksh : con la korn shell

#!/bin/csh : con c shell

CAMBIAR LA VARIABLE $PATH

vamos al directorio principal (ponemos cd,nada mas)

ls -l .profile :

este es el fichero de ejecucion del usuario especifico de variables del sistema que se ejecuta 1 vez cuando se enciende la maquina

entramos a este fichero y **al final de todo el fichero** ponemos **:**

PATH = $PATH:(nueva doreccion)

hace que se aumente la nueva direccion a lo que ya hay

guardamos y salimos del fichero .profile y para que se ejecutenestos cambios tenenmos que reiniciar la maquina.

si solo actualizamos la variable $PATH como comando en la terminal, esta variable se guardara asi solo para esa ventana de terminal

es necesario anadir la ubicación (path) de donde estan los programas de mayor uso de lo contrario para correr un programa tendria que:

opsion 1:

poner la direccion completa del ejecutable para correrlo

/home/escritorio/programas/ejecutable.sh

opsion 2:

ir al directorio donde esta el ejecutable y poner

./ejecutable.sh

export variable= valor

para que el valor de la variable se reconosca en todo el sistema ponemos export

echo ${variable}hola

si quiero imprimir algo alado de la variable (sin espacio) ponemos la variable entre corchetes para que reconosca la variable

expr 1 + 2

hace que lo que esta a lado es numero pero tenenemos que separar cada parte por espacio

poner con espacios

let v3=10+20

guarda en v3 el resultado numerico de la operación

v1=10

v2=20

echo $v1 + $v2

no funciona porque linux lo reconoce como caracteres

imprime: 10 + 20

echo $(($v1+$v2))

poner asi para que reconosca a las variables como numeros

x=$(ls -l /var/lib)

guarda todo el resultado del comando en la variable x

x=`ls -l /var/lib`

guarda todo el resultado en el comando en la variable x

echo $? : nos muestra 0 si el **programa inmediato anterior ejecutado** es ejecuto bien

nos devuelve 2 si el comadno anterior se ejecuto mal

resultado=$?

Se lo puede guaradar en ua variable

CASE

case valor in

patron1)

comandos

;; : esto significa salte del case

patron2)

comandos

;;

\*) : esto significa default (q si no se cumple ninguna de las anteriores viene aquí

comandos

;;

esac

COSAS PARA PONER EN patron

[ABC]) : se refiere el valor es alguno de los caracteres A B o C

en esta caso puede entrar las entradas A B o C

[0-9]) : entra a este caso cuando la entrada esta entre 0-9

??) : en este caso estamos diciendo que es una **palabra de dos letras**

\*a) : entra en esta opsion cuando la entrada sea lo que sea que termine en ‘a’

x|X) : es cuando la entrada es ‘x’ o ‘X’

CONDICIONES

#

if [condicion] test

then

comando a ejcutar

elif [ codicion ]

then

comandos a ejecutar

else

comando alternativa

fi

#

test

-d fichero : si el fichero existe y es un directorio devuelve true

-e fichero : si el fichero existe, y es el tipo que sea devuelve true

-f fichero : si dichero existe y es un fichero normal devuelve true

-r fichero : si fichero existe y se puede leer devuelve true

-s fichero : si fichero existe y tiene tamano mayor a cero devuelve true

-w fichero : si fichero existe y se puede escribir en el devuelve true

-x fichero : si fichero existe y es ejecutable (en permisos) devuelve true

-L fichero : si fichero existe y es enlace simbolico devuelve true

COMPARACION DE CADENAS

test condicion : si queremos comprar

test string1 = string2

esto debuelve 0 si es verdadero y 1 si es falso

string : el string es nulo (\*\*)

-z string : la longitud del string es 0

-n string : la longitud del estring no es 0

string1 = string2 : los estring son iguales

string1 == string2

string1 != string2 : los string son distintos

ENTEROS

int1 -ep int2 :enteros son iguales

int1 -ne int2 : enteros son distintos

int1 -gt int2 : int1 es mayor que int2

int1 -ge int2 : int1 es mayor o igual que int2

int1 -lt int2 : int1 es menor que int2

int1 -le int2 : int1 es menor o igual que int2

OPERADORES LOGICOS

operation test []

and -a &&

or -o ||

not ! !

LEER DEL TECLADO

read variable : recibe del teclado y lo guarda en una variable

read -p ‘entroduce el valor de la variable’ variable

imprime un mensaje y recibe valores del teclado

read -d: var1 : este comando lee hasta encontrar el carácter ‘:’ o el que se ponga

read -s var1 : hace que lo que se escribe no se muestre en pantalla

**BUCLE FOR**

for variable in lista

do

comandos que utilizan variables

done

QUE PONER EN LISTA

\* : hace el comando ‘ls’ y usa esa lista

s\* : pone en la lista todos los ficheros con nombres que empiecen en ‘s’

{1..5} : la lista son numeros del 1 al 5

{a..z} : la lista son letras desde la a hasta la z

$(seq 1 4) : para listas numericas poner asi

$(seq 1 $(($variable))) : cuando un limite es una variable

**BUCLE WHILE**

while condicion

do

comandos

done

**BUCLE UNTIL**

until condicion

do

comandos

done

**PARAMETROS**

$1 : este el primer parametro que se pone al momento de correr el programas

$2 $3 : y asi en orden según se ponga los parametros

$0 : es un parametro reservado, en esta variable se guarda la direccion del programa (path completo)

$# : aquí se guarda la cantidad que le damos al programa

**CURSOS DE LINUX DE NETACAD**

aptitude : aptitude es un gestor de paquetes

aptitude moo : nos muestra easter eggs

EXPRESIONES REGULARES

. : (es un punto) cualquier individual

[ ] : cualquier carácter especificado

[^ ] : no el carácter especificado

\* : cero o mas d ellos caracteres previos

^ : si ^ es el primer carácter en el patron, entonces el patron debe estar al principio d el alinea que debe coincidir, de lo contrario solo un literal ^

grep

$ si $ es el ultimo carácter en el patron, entonces el patron debe estar al final de la linea que debe coincidir, de lo contrario solo un literal $

EXPRESIONES EXTENDIDAS

+ : uno o mas d ellos patrones previos

? : el patron anterior es opcional

{ } : especificar las coincidencias minimas, maximas o exactas del patron anteior

| : (es una barra vertical) alternancia

( ) : permite crear grupos

COMANDO :dd

dd : este comando es muy poderosa y delicada

**leer mas al respecto**

sudo shutdown : este comando apaga la computadora

shutdown [opciones] hora [mensaje]

**de manera segura**

avisara a todos los usuarios que la maquina se apagara en 5 min

si o si hay que poner un argumetno de hora

shotdown now “adios a todos”

ifconfig : da detalles de red

hostame : nos da el nombre de hostname

hostaname -I : nos da nuestra ip

iwconfig : se especializa en las conecciones inalambricas

ADMINISTRADORES DE PAQUETES

ej.-

ubuntu, debian, red hat

apt-cache search palabra\_clave

busca dentro de los paquetes con

apt-get install nombre\_paquete

este comando instala el paquete en caso de que no lo tengamos

en caso de que ya tengamos el paquete **lo actualiza**

apt-get update

actualiza los cache de los paquetes

es decir actualiza **la lista de los paquetes que tenemos disponibles para descargar**

esta lista se actualiza con servidores de internet

a la version mas reciente

tambien nos muestra donde conseguirlos

apt-get upgrade

actualiza todos los paquetes y sus dependencias a su version mas reciente disponibles

apt-get remove nombre\_paquete

**elimina un paquetes**

deja sus archivos de configuracion en el sistema

apt-get purge nombre\_paquete

**depura el paquete completo del sistema**

elimina sus archivos de configuracion del sistema

apt-get remove –purge nombre\_paquete

esto es util cuando arruinaste la configuracion de una aplicación y quieres iniciar desde cero con la misma aplicación

APRENDIDO POR EXPERIENCIA

“cat /etc/services” : este comando te muestra los puertos usados

**PREGUNTAR PUERTOS**

less /etc/services

-este comando nos muestra que puertos estan asignados a que servicio

sudo netstat -plut

-este comando nos permite supervisar la conexiones de red tanto entrates como salientes

-p :estamos diciendo que nos muestre los procesos que estan utlizando las conexiones

-l : muestra las conexiones en estado de escucha

-u : muestra las conexiones usando UDP

-t : muestra las conexiones usando TCP

sudo ss -plut

-este se usa ahora, **netstat a quedado en desuso**

-los argumentos funcionan igual en ss que en netstat

sudo ufw deny 8000/udp

-con este comando cerramos puertos

sudo ufw allow 800/tcp

-con este comando abrimos puertos al protocolo especificado

sudo ufw deny http

-niega todas las entradas con servicio http

-se puede usar esto tambien en allow

**LEER FICHEROS JSON**

jq ‘.’ fichero.json

-asi se lee el fichero y se lo imprime

**VER TODAS LAS INTERFACES DE RED**

ip link show

-esto mostrara todas las interfaces de red de la maquina y una descripcion de cada una

**VER TRAFICO DE RED DE LA COMPUTADORA**

sudo apt-get install nload

-nload es una herramienta de linux que sirve para ver el trafico de red en tiempo real.

nload

-este comando ejecuta la herramienta presionas flechas derecha o izquierda para cambiar entre los dispositivos de red que tenga la maquina.

-este comando muestra con graficas mas bonito

nload -m

-este comando muestra **solo** los datos de todos los dispositivos de red de la computadora

nload -a 400

-con el comando -a pones durante cuanto tiempo en segundos sacara el average (default es 300)

nload -t 600

-esto espara poner cada cuanto timepo en milisegundos se refrescara los datos(default es 500)

-no poner a menor de 100 milisegundos poq lo hara fallar

nload -u H

-esto significa poner las unidades legible para humanos

nload -u h

-esto significa poner las unidades legibles para maquina

sudo apt-get install vnstat

-esta es una herramiento que permite hacer historiales del trafico de red

sudo apt-get install vnstati

-esta es una herramienta que permite graficar los datos de vnstat

vnstati -5 -i wlp9s0 -o ~/Escritorio/summary.png

-este comando permite guardar el reporte de cada 5 segundos en una imagen

-se puede espsificar el tiempo de inicio y de fin con otros argumentos mas

**LIMITADOR DE TRAFICO EN PUERTO**

iptables: es un servicio de linux que nos brinda la posibilidad de administrar el trafico en los puertos

sudo iptables -L

-nos muestra los puertos que esta administrando iptables

sudo iptables -F

-borra todas las reglas que ayamos puesto con iptables

sudo iptables -A INPUT -p tcp –dport 25 -j ACCEPT

-con este comando ponemos una reglas de entrada para el puerto 25 y a los paquetes que quieran entrar

-con este comando estamos diciendo que todos los **paquetes tcp que ENTREN por el puerto 25 ceran aceptados (**ACCEPT)

-las reglas que se aplicaran al puerto especificado ya sea de entrada o de salida se espicifica en el ultimo comando:

-DROP : seran rechazados

-REJECT: rechaza el paso y le avisa al amisor que se le rechazo

-QUEUE: introduce el paquete a una cola dentro d ella biblioteca **libipq**

-RETURN el paquete vuelve a su origen y deja de cricula por la cadena

-LOG: crea un registro d ellos paquetes que circulan por la cadena

-DNAT: modifica la direccion de destino y puerto tambien

-SNAT: modifica la direccion de origen y su puerto tambien

-MASQUERADE: es una forma de SNAT y permite mas restricciones

sudo iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT

-este comando dice que todos los paquetes que entren por eth1 saldran por eth0

**VER IP PUBLICA**

sudo apt\_get update

sudo apt install curl

-necesitamos esta librería

curl ifconfig.me

-este comando nos devolvera en la ip publica

**VER TODAS LAS IP DE LA RED**

ip route

>>192.168.0.0/24

-con esto averiguaos que mascara tiene la red

NMAP

nmap -sP 192.168.0.0/24

-esto nos devolvera todos los dispositivos conectados a la red con esta ip

-este coamando utiliza paquetes **ISMP** para averiguar la presencia de un nodo en la red

-es decir que hace ping a todos los nodos de la red que se le especifique

sudo nmap -sT -p 80 192.168.0.0/24

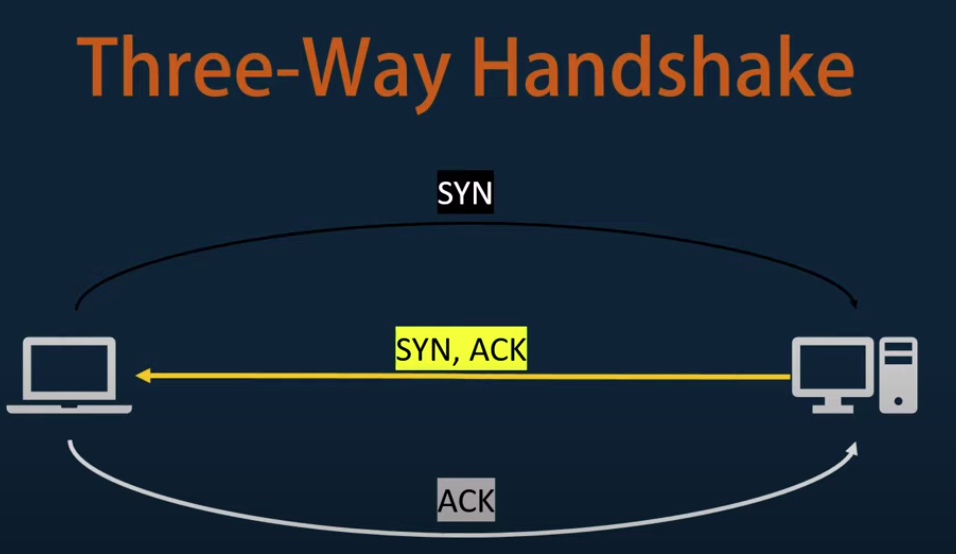
-la T es poq usa el protocolo TCP, revisa si el puerto esta completamente abierto

-esto utiliza el metodo de 3 way handshake completo ( la imagen de abajo)

-p 80,334 : para poner varias ips

-p 80-100 :para poner un rango de puertos

-con esto nos avisara el estado del perto especificado



sudo nmap -sS 192.168.0.24

-la S es por stealthy (cauteloso)

-si se queire pasar desapersivido en trakceo de la red mejor usar esto, poq el objetivo no sabra que se esta investigando

-esto **NO** esta basado en ismp

-esto se tiene que usar en super usuario

-esto utiliza metodo de **TREE WAY HANDSHAKE** **INCOMPLETO** para pregutnar por una ip, este metodo omite el ultimo ACK.

sudo nmap -PR 192,168,0,24

-esta es igual a la anterior pero en capa 2 usando **arp**

-si se tiene que revisar varias direcciones ip, se los puede pasar en un fichero

nmap -iL lista\_ips

-se puede poner que el resultado lo guarde en un fichero

nmap -o fichero\_salida.txt

-para especificar el nivel de detalle de los resultados podemos poner estos opsiones

-v : aumenta detalles

-vv :aun mas detalles

-d 0-9 : con esto ponemos el nivel de detalle 9 es el maximo

-mientras mas alto

sudo nmap -O 192,168,0,24

-esto nos devuelve el sistema operativo de la ip objetivo

sudo nmap -sS -D ip\_fake ip\_destino

sudo nmap -sS -D 192.168.0.30 192.168.0.35

-D :esta opsion hace el trabajo normal el -sS a la ip destino, pero hace otra ejecucion pero poniendo la ip\_fake como funte de la onsulta

-esto es importante usar para despistar al objetivo de quien es la fuente del trakeo

**SERVIDOR WEB EN LINUX (APACHE2)**

**-**todos los comandos con apache tiene que ser con permisos de superusuario

sudo apt-get update

sudo apt-get install apache2

-con esto instalamos apache en nuestra computadora

-por defecto apache ya se configura inicialmente

-incluso tiene una pagina por defecto para pruebas

sudo apt-get install apache2-doc

-con esto instalamos toda la documentacion de apache2

-ala cual podemos acceder por buscador

-para acceder a este manual ponemos en el buscador

la ip de la compu y despues ‘/manual/’

192.168.0.37/manual

localhost y despues ‘/manual’

127.0.0.1/manual

apache2 -v

-vemos la version de apache que se instalo

service apache2 status

-nos muestra el estatus de apache

ls -l etc/apache2/

-aqui estan los archivos de configuracion de apche2

-con este comando vemos esos archivos

-el fichero ports.conf es donde configuramos los servidores por puertos

-el fichero sites-available, aquí tenemos la configuracion de los servidores virtuales que tengamos

-el fichero sites-enable, aquí estan los servidores visibles

-solo tenemos que hacer un enlace simbolico a los sevidores dentro de ‘sites-available’ y poner este enlace dentro de ‘sites-enable’ para que este visibles

ls -l /var/www/

-aqui estan alojadas las paginas web

-aqui ponemos las paginas web que creemos

-aqui hay una carpeta llamada ‘html’

-aqui se aloja la la pagina web por defecto de apache

-para aceder a esta pagina solo se pone la ip o localhost

192.168.0.37

127.0.0.1

**SERVIDORES VIRTUALES POR NOMBRE**



-utilizar este metodo es hacer que nuestro servidor web aloje varios **DOMINIOS**

-todos los dominios apuntan al mismo IP

-pero el servidor sabe que dependiendo del dominio que le hagan solicitud va a mostrar una raspectiva pagina

PASOS(ejemplo de la foto)

1.crear la estructura de directorios donde estara las paginas web

mkdir /var/www/1redesplus

mkdir /var/www/2redesplus

2.poner las paginas en su respectivo directorios

touch /var/www/1redesplus/index.html

touch /var/www/2redesplus/index.html

3.crear los archivos de configuracion para que apunten a estas paginas

el fichero ‘/etc/apache2/sites-availabel/000-default.conf’ es un ejemplo de configuracion para aplicar

touch /etc/apache2/sites-available/1redesplus.conf

touch /etc/apache2/sites-available/2redesplus.conf

esto ponemos ahi:

<VirtualHost \*:80> //esto es una directiva que especifica que es un hostvirtual

// poner el \* dice que todas las ip accedan a este servidor, sin importar la **ip POR LA QUE VENGA** la solicitud

//pero solo por el puerto 80

ServerName 1redesplus.es //esto dice el nombre del dominio

ServerAdmin [root@1redesplus.es](mailto:root@1redesplus.es) //esto dice la direccion de correo ddel administrador

ServerAlias [www.1redesplus.es](http://www.1redesplus.es/) // esto dice otro dominio por el cual se puede acceder a este servidor

DocumentRoot /var/www/1redesplus //este es el path absoluto a la pagina

DirectoryIndex index.html //este dice cual es la pagina base, osea si hacemos un request al alias, sin pasarle otro directorio, entrara a este fichero

</VirtualHost>

4.hasta aquí la configuracion ya esta hecha, pero **no esta habilitada**

-activamos en el servidor

-para hacer esto solo tenemos que crear un enlace simbolico de este .conf y ponerlo en /etc/apache2/sites-enable/

-otra opcion es ejecutar este comandos

a2ensite 1redesplus.conf

a2ensite 2redesplus.conf

-este comando busca el fichero .conf especificado en ‘/etc/apache2/sites-available/’ y crea un enlace simbolico del mismo y lo pone en ‘/etc/apache2/sites-enable/’

-y esto nos devuelve un mensaje que se activo correctamente

-tenemos que deshabilitar la configuracion default en /etc/apache2/sites-enable con el siguiente comandos

a2dissite 000-default.conf

-este comando deshabilita una pagina del servidor, y nos avisa que deshabilito

-**en este mensaje nos da el comando que tenemos que poner para el siguiente paso, esto depende de la version de apache instalada**

-podemos hacer esto con las paginas que querramso desactivar

-este comando solo borra el enlace simbolico de ‘/etc/apache2/sites-enable/’

5.por ultimo despues de habilitar la pagina, tenemos que **activar la pagina dentro del servidor apache**

**-**ejecutamos el comando devuelto por el paso anterior

-tenemos que hacer la recarga apache2 con el siguiente comando:

systemctl reload apache2

6.por ultimo ya que estamos trabajando con servidores por dominio o nombre,

-tendriamos que habilitar una zona en el servidor dns

-o podemos habilitar los dominios en el archivo host

-modificamos el fichero ‘/etc/hosts’

sudo vi /etc/hosts

-si nunca se toco antes deberia salir esto:

127.0.0.1 localhost

127.0.1.1 rodrigo-VGN-NW270F

# The following lines are desirable for IPv6 capable hosts

::1 ip6-localhost ip6-loopback

fe00::0 ip6-localnet

ff00::0 ip6-mcastprefix

ff02::1 ip6-allnodes

ff02::2 ip6-allrouters

-deberiamos dejarlo asi:

127.0.0.1 localhost

127.0.1.1 rodrigo-VGN-NW270F

192.168.0.37 [www.1redesplus.es](http://www.1redesplus.es/)

192.168.0.37 [www.2redesplus.es](http://www.2redesplus.es/)

# The following lines are desirable for IPv6 capable hosts

::1 ip6-localhost ip6-loopback

fe00::0 ip6-localnet

ff00::0 ip6-mcastprefix

ff02::1 ip6-allnodes

ff02::2 ip6-allrouters

-esta configuracion solo funcionara a nivel de nuestra maquina o computadora

-significa que el servidor web de la computadora, cuando se le ahce una solicitud de una pagina, primero busca **el fichero ‘**/etc/hosts’, si encuentra la solicitud aquí la muestra

asi terminamos la configuracion de la pagina por nombre

**SERVIDORES VIRTUALES POR PUERTOS**



1. repetimos los paso 1 y 2 de servidores virtuales por nombre

mkdir /var/www/3redesplus

touch /var/www/3redesplus/index.html

-el index.html seria nuestra pagina

2.creamos su fichero de configuracion

touch /var/apache2/3redesplus.conf

-pero en la configuracion ponemos (dentro del 3redesplus.conf):

<VirtualHost \*:8081>//espesificamos el puerto para la pagina

ServerName 3redesplus.es

ServerAdmin root@3redesplus.es

ServerAlias www.3redesplus.es

DocumentRoot /var/www/3redesplus

DirectoryIndex index.html

</VirtualHost>

-todo es igual con sus respectivos datos, pero el la parte de puerto le asignamos su propio puerto

3.habilitamos el puerto en:

sudo vi /ate/apache2/ports.conf

-originalmete hay esto:

# If you just change the port or add more ports here, you will likely also

# have to change the VirtualHost statement in

# /etc/apache2/sites-enabled/000-default.conf

Listen 80

<IfModule ssl\_module>

Listen 443

</IfModule>

<IfModule mod\_gnutls.c>

Listen 443

</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

-lo dejamos asi:

# If you just change the port or add more ports here, you will likely also

# have to change the VirtualHost statement in

# /etc/apache2/sites-enabled/000-default.conf

Listen 80

Listen 8081

<IfModule ssl\_module>

Listen 443

</IfModule>

<IfModule mod\_gnutls.c>

Listen 443

</IfModule>

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet

-solo aumentamos que escuche en el puerto 8081

4.activamos el servidor

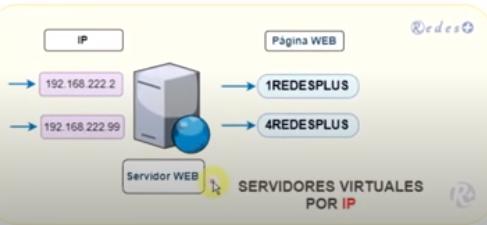
sudo a2ensite 3redesplus.conf

-sidamos:

sudo entstat -tuna

-veremos que ya estamos escuchando en el puerto 8081

**SERVIDORES VIRTUALES POR IP**

-para hacer este servidor es necesario que la maquina tenga dos interfases de red

-esto se puede hacer e una maquina virtual

-o en una computadora con con varias interfaces de red

COMPROBAR LOS HOST VIRTUALES ACTIVOS

apachectl -S

-este comando devuelve los **hostvirtuales activos actualmente**

-nos entrega lo siguietn:

VirtualHost configuration:

\*:80 is a NameVirtualHost

default server 1redesplus.es (/etc/apache2/sites-enabled/1redesplus.conf:1)

port 80 namevhost 1redesplus.es (/etc/apache2/sites-enabled/1redesplus.conf:1)

alias www.1redesplus.es

port 80 namevhost 2redesplus.es (/etc/apache2/sites-enabled/2redesplus.conf:1)

alias [www.2redesplus.es](http://www.2redesplus.es/)

-vemos que por defecto pone al servidor 1redesplus.es, esto solo porque es el primero en orden alfabetico

**SERVIDOR CON NETCAT**

**-**una conifguracion basica de linux es **netcat,** este nos permite lanzar un aservidor basico y simple

nc -lk port

nc -lk 9090

-esto abre un seridor escuchando en el puerto 9090

- una vez lanzado el seridor se puede mandar mensajes al cliente

-solo escribimos algo y damos enter para enviar mensajes

-esto tambien recive mensajes del cliente

nc localhost 9090

-esto crea un cliente que se conectara a localhost al puerto 9090 en netcat

-este tambien puede recibir y mandar mensajes

**detectar usb en linux**

lsusb -t

**-**con este comando se detecta los puertos usb fisicos utilizados en la maquina

usb-devices

-esto muestra datos mas detallados sobre los usb

xdg-open url

xdg-open imagen.png

-este comando abre el link, imagen o archivo en el programa predefinido

-si es link en el buscador

-si es imagen en el gestor de imagenes

**SCREEN**

-el objetivo de screen es la multiplexacion de la terminal

-divide la termianl fisica en multiples seciones virtuales

screen -ls

-muestra todas las sesiones que hay de screen

screen -r <numero de sesion>

-se vuele a meter a la secion especificada

ctrl + a + k

-esto mata el proceso de screen

**UTILIZAR LAS ESP8266 CON LINUX**

**-**descargmos de micropython.org el firmware para la tarjeta

micropython.org > download > ESP8266

-instalamos la herramienta para linux

pip3 install esptool

esptool.py --port /dev/ttyUSB0 erase\_flash

-asi borramos la memoria de la tarjeta

esptool.py --port /dev/ttyUSB0 --baud 460800 write\_flash --flash\_size=detect 0 /home/rodrigo/Descargas/esp8266-20210418-v1.15.bin

-asi escribimos el nuevo firmware

-el path es del fichero descargado de micropython.org

screen /dev/ttyUSB0 115200

-con esto nos conectamos a la tarjeta

-esto devuelve una shell de python

**-dar enter una vez que entra para que aparesca la teminal python**

**programar en micropython**

-podemos ver este video para iniciar https://www.youtube.com/watch?v=TsuT0QywNio

import machine

-esta librería nos permite utilizar los pines y demas

machine. TAB

-si escribimos ‘machine.’ y despues pulsamos TAB nos mostrara las funciones de machine

pin = machine.Pin(2 , machine.PWM)

-con este comando utilizamos los pines en pwm

import webrepl\_setup

-esto va a read-evaluate-print constantemente

-es un loop

-seguimos las instrucciones que salen

-esto hace que la esp8266 genere una red wifi y el codigo final del nombre de la red es la mac de la tarjeta

-nos pedira una contrasena (es con la n final mayuscula)

micropythoN

-para conectarnos por la computadora hacemos esto

git clone https://github.com/micropython/webrepl

-dentro de directorio descargado ponemos

xdg-open webrepl.html

xdg-open se explcia arriba es es

**subir un script de python a la esp8266**

pip3 install adafruit-ampy

-esta es la librería para cargar micropython a la esp866

ampy --port /dev/ttyUSB0 put ./main.py

-este comando sube el script a la esp8266

-el script se tiene que llamas main.py

-despues de subirlo presionamos reset en la tarjeta

ampy --port /dev/ttyUSB0 run main.py

-con este comando ejecutamos manualmente el script

-su hay errores este nos devuelve los errores

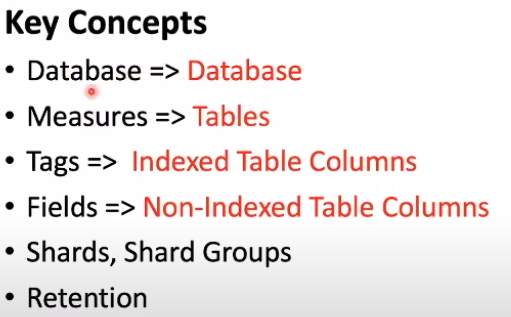
**INFLUXDB**

**-**es una base de datos de timeseries

-tiene una estructura similar que **SQLite**

**-**los timestamp se agrgan del momento que ingresa el dato, pero esto se lo puede conifgurar que se tengo el propio

**Como funciona influxdb**

****

-tags : solo pueden ser string

-fields : pueden ser de todo tipo

**-shards son la forma de alaasenamiento y de compresion**

**Conceptos de influxdb**

-tiene una version gratuita y una version comercial que soporta clustering y high availability

**-**es una base de datos de series de tiempo

-analogando a KIBANA influx tiene TICK stack

-telegraf : vendria siendo un pippeliner que meta datos a influxdb

-influxdb

-chronograf : provee una interfaz de usuario que permite vizualizar los datos de influxdb

-kapacitor : es una tecnologia que permite el procesamiento de los datos (alertas)

**Comandos en influxdb**

sudo service influxdb start

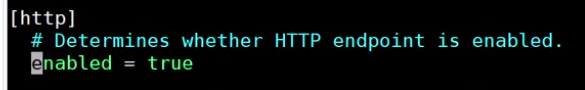
-esto es para inciar el servicio de influxdb

-de forma predeterminada influxdb escucha en el puerto **8086**

**-**el archivo de configuracion de influxdb esta por defecto en /etc/influxdb/influxdb.conf

-para habilitarlo como http endpoint y hacr los comandos por solicitudes post tenemos que modificar el influxdb,conf

-en el fichero debajo del **[http]** descomentar la linea **#enable = true,** y despues reiniciamos influxdb



-los comandos ejecutados con **curl** son los mismos que se usa en la terminal de influxdb

-solo pones lo que esta dentro de **q=** en la terminal de influxdb

curl -XPOST “<http://localhost:8086/query>” –data-urlencode “q=CREATE USER rodrigo WITH PASSWORD ‘rodrigo’ WITH AL PRIVILEGES”

-con este comando creamos el usuario administrador con todos los privilegios

influx -username ‘rodrigo’ -password ‘rodrigo’

-esto iniciara la terminal de influsdb con el usuario creado

-si no tenemos usuario creado podemos poner solo **influx**

curl -G <http://localhost:8086/query> -u rodrigo:rodrigo –data-urlencode “q=SHOW DATABASES”

-este comando mostrara todas las bases de datos en influxdb

**InfluxDB shell**

create database nombredb

-esto crea una base de datos

drop database nombredb

-esto borra la base de datos

use nombredb

-esto nos pone en uso la base de datos creada

show measurements

-eso muestra los me asurements de la base de datos

insert cpu,host=serverA,region=us\_west value=0.64 time=123456789

-el comando generico seria:

insert measurement\_name,tag1=tag1,tag2=tag2 value1=value2,1value2=value2 time=timestamp

-el orden tiene que ser primero los tags,luego los fields, y luego el timestamp

-este comando sirve para insertar datos en la base de datos

-cpu : es el measrement (nombre de la tabla)

-host , region : son los tags

-value : es el field

-time : podemos poner nuestro propio timestamp poniendolo al final del comandos

-tiene que ser en unix

-**SI NO PONEMOS TIEMSTAMP** influxdb pondra el tiempo en el ingreso el dato a la base de datos en **nanosegundos**

select \* from “cpu”

-esto nos muestra todo el contenido de las medidas (tabla) cpu

-es recomendable poner entre comillas

show tag keys

-esto nos muestra los tags de las base de datos de que tengamos guardados

show tag keys from “measurements\_name”

-esto nos muestra los tags de un measurement (tabla) en especifico

show field keys

-nos muestra los fields de todos los measurement de la base de datos y sus tipos

show field keys “measurement\_name”

-nos muestra los nombre y tipos de los fields del measurement especificado

show tag values from “measure\_name” with key in (“region”)

-aqui decimos que nos muestre lo valores de tag los cuales tengan valor en el tag region

-nos mostrara sin repetidos

show retention policies

-esto nos mostrara las politicas de retencion

-si en nombre sale autogen significa que se genero automaticamente

-las politicas de retencion se crean autmaticamente cuadno creamos una base de datos

-las politicas de retencion dicen las duracion de vida de los datos

-si en duration dice 0s significa que nunca se borrara automaticamente

show shards

-muestra los shard de todas las base de datos

-los shards son un contenedor donde se introducen los datos

show shard groups

-esto muestra lso shard groups

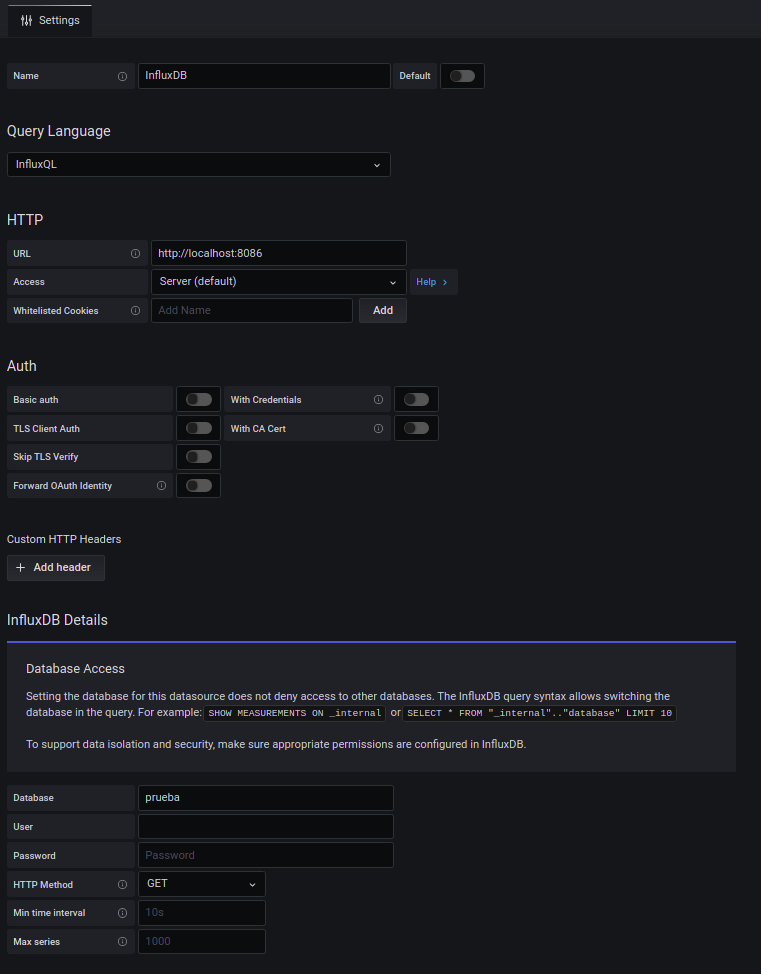
-el group shard el un grupo de contenedores de datos

**GRAFANA**

**-**para actualizar grafana o si no te sale las ultimas versiones de los plugins lo mejor es resintalar grafana de nuevo

-solo es descargar el ulitmo paquete de grafana y instalarlo

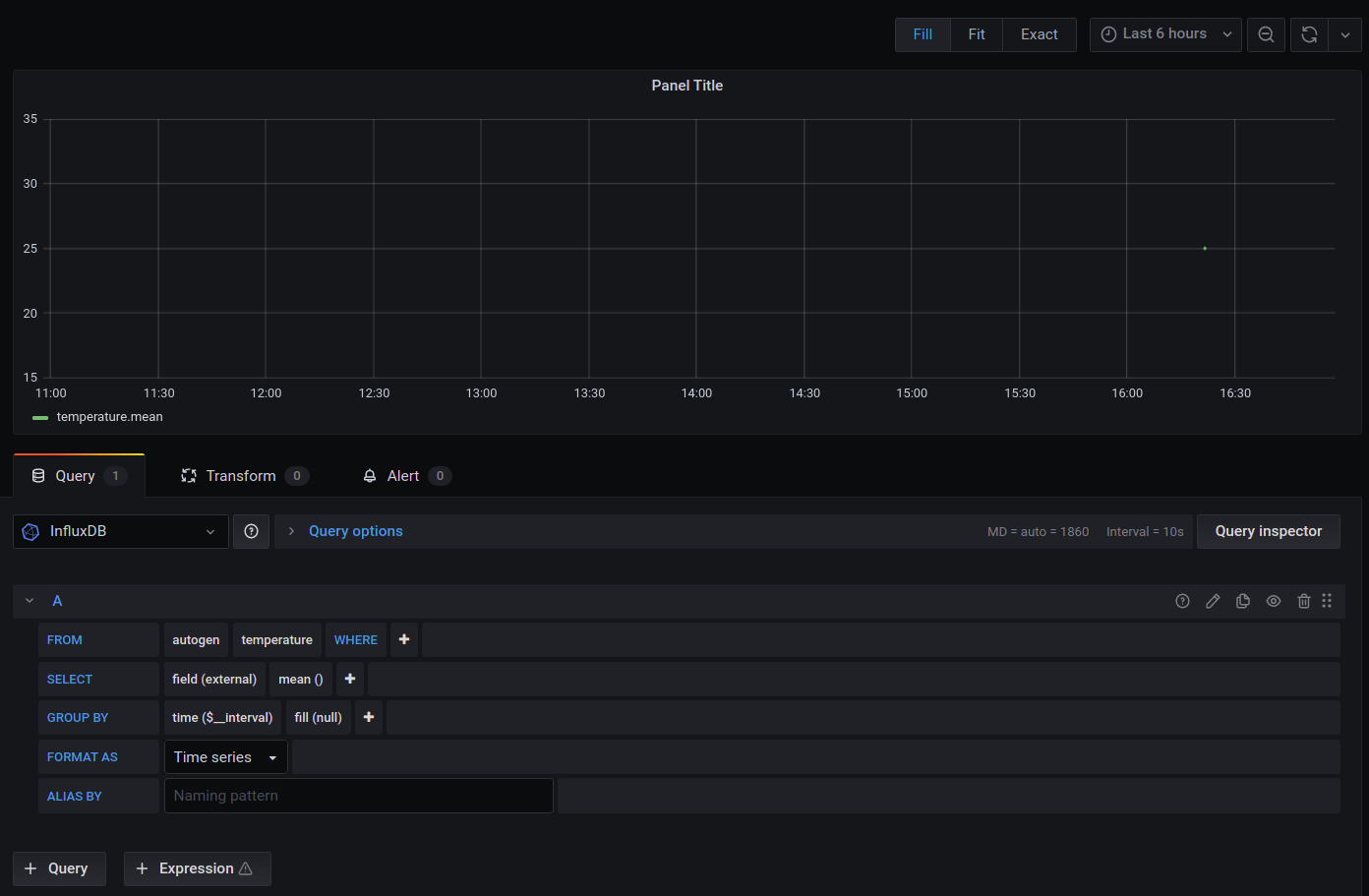
-para leer datos de influxdb en grafana entramos a la configuracion del plugin en new datasource y ponemos:



-ya que no pusimos usuario ni contrasena dejamos en blanco

-pero si hubieramos creado usuario y contrasena deberiamos ponerlo

-y para crear el dashboar ponemos:



VARIABLES EN GRAFANA:

$varname

-este formato es facil de usar pero no te deja usarlo en medio de una expresion

${varname}

-estopermite usar en medio de expresiones

${varname:<format>}

-esta es la mejor forma de usar la variable casteando el tipo de la variable

-puede ser:

-csv , json , lucene , pipe , raw , text

ENVIAR RESULTADOS DE QUERY A OTRAS GRAFICAS

-esto para ahorrar uso de la red para los querys

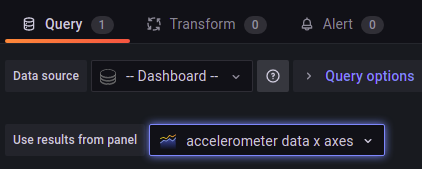
-reduce el tiempo de carga, puesto que tendria que cargar los datos de todos lo querys de cada panel

-para usarlo:

1.creamos un nuevo panel

2.en datasource seleccionamos “--Dashboard--”

3.en use result from panel seleccionamos el panel fuente

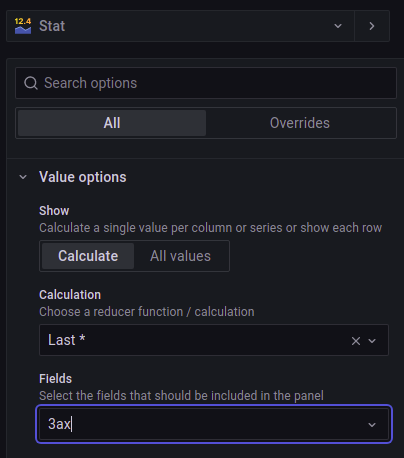


SELECCIONAR DATOS PARA MOSTRAR

-en la columna derecha:

-”calculation” :sirve para selccionar si mostrar min , max , first , last , mean , total , count (numero de valores) , etc

-”fields” : sirve para filtrar un dato de todos los que te llegan



**GITHUB**

svn checkout https://github.com/abelvillca/breathdx/trunk/data

-para descargar solo una folder de github

-el link:

-link del ropositorio + /trunk + folder\_name

**MOSQUITTO**

-este es un servidor mqtt

-cuando se instala mosquitto en linux este empieza a ejecutarse un servidor (broker) como servicio

mosquito\_sub -h 192.168.0.10 -p 1883 -t ‘topico’ -v

-h es la ip del broker

-p es el puerto del broker por defecto el servicio de mosquitto escucha en 1883

-t es para el topico entre comillas simples

-v es para que se muestre todo lo que se recive

moquitto\_pub -h 192.168.0.10 -p 1883 -t ‘test’ -m ‘mensaje’

-el publicador tiene los mismos argumentos que el subscriptor

-m es para el mensaje a enviar

**JUPYTER**

jupyter nbconvert --to html notebook.ipynb

-esto es para convertir el notebook en html

**CONVERTIR AUDIO**

ffmpeg -i audio.m4a audio.wav

-con este comando se puede convertir los audios a .wav

**MANEJO DE ERRORES**

**-**cuando al hacer update sale error del servidor,anda a actualizaciones y cambia la fuente de las actualizaciones

**GIT HUB**

-para utentificarse en github ahora tenemos que usar tokens

boton superior derecho > settings > developer settings > Personal access tokens >Generate new token

-ponemos el nombre al token

-ponemos tiempo de expiracion o no expiracion

-seleccionamos los permisos que tiene este token

-ponemos crear token

-depues nos data el pasword del token **COPIARLO Y GUARDARLO** poq no se mostrara nunaca mas

-una vez creado el token el commit se realiza de igual forma, pero ahor aen pasword de autentificacion pornemos el toke

-si presionamos la tecla de punto “.” en github nos abre un visualstudio online

**DOCKER**

-docker se basa en muchos concptos de linux

-si se quiere compartir un contenedor se tiene que hacer un “build” y enviar la nueva imagen

-**en docker hub**: aquí podemos encontrar imagenes de distintas aplicaciones para instalar

-se puede descargar hasta la imagen de ubuntu

-esta imagen solo puede ejecutar un proceso de ubuntu

COMANDOS

docker images

-esto nos muestra las imagenes que tenemos isntaladas en la compu

-muestra 5 columnas

-repository: es el nombre de la iamgen

-tag : la version de la imagene que es

docker search ubuntu

-esto nos da la lista de las distintas imagenes que hay de ubuntu en docker hub

-nos da una descripcion de la imagen

-esto nos dice la puntuacion

-nos dice si es una imagen oficial

docker pull ubuntu

-este sirve para instalar las imagenes desde docker hubieramos

-en lugar de ubuntu ponemos el nombre q salga en el search

ALGUNAS IMAGENES

-

**APRENDIDO DESPUES**

lsmod

-esto muestra los modulos isntalados (esto es como los driver de windows

modprobe –r <nombre de modulo>

-es para quitar los modulos

dmesg

-esto muestra los mensajes a lso dispositivos del sistema, si se levantaron bien o mal

**NGINX**

-este es un web server optimizado

-sirve archivos estaticos de manera eficiente

-nginx maneja mejor el envio de socketio que gunicorn

**APUNTES DEL LIBRO**

ADVANCED LINUX PROGRAMMING (Mark Mitchell, Jeffrey Oldham, Alex Samuel)

**GNU MAKE**

**https://www.gnu.org/software/make/manual/html\_node/Catalogue-of-Rules.html**

**-**aqui puedo ver el manual de reglas de make

**-**cuando instalamos “build-essential” se instalas make

**-**ngu make sirve para automatizar la compilacion y la construccion de ejecutables desde los ficheros c/c++ (tambn sirve para js)

-cuando se realiza cambios en alguno de los ficheros fuente y se ejecuta el make, este reconoce los ficheros que cambiaron y solo compila dichos ficheros

-esto lo hace utilizando las fechas de modificacion

-si el target tiene una ficha mas vieja que la dependencias, se ejcuara de nuevo

-lo utili de **make** es que una vez compilado el codigo, este elimina los ficheros intermediarios que se crean y no se usaran mas

-make es util cuandose tiene un programa con librerias, las cuales ejecutarlas colo con el compilador se tendria que aumentar muchos flags, lo que lo haria largo y dificil de escribir

-por lo general el Makefile imprime los comandos de las reglas cuando las ejecuta

-si no queremos que se imprima los comandos, poner @ (arroba) al inicio del comando

PARTES DEL MAKEFILE

-rule : es al conjunto de los siguientes partes

-targets : es el file resultante de la regla, ya se el .o .cpp o el ejecutable

-dependencias: son todos lo ficheros con las dpendencias del target

**-**es importante que para usar las dependencias, dichas dependencias existan

-commands : son las lineas que comoponen la regla

ESTRUCTURA

target : dependecies

commands

objetivo : dependecias

instrucciones

CREAR UN MAKEFILE

-el fichero creado **siempre** tiene que llamarse de lamisma forma “Makefile”,puesto que cuando ejecutemos el “make” buscara directamente el Makefile

-**se puede ver un ejemplo en dentro de ./programas\_c++/ejem\_make/**

EJECUTAR EL MAKEFILE

make programa

-se pone el comando make y la regla en especifico que queremos que se ejecute

CONVENCIONES

-es muy recomendado tener una regla llamada “clean”

-no hay necesidad que tenga dependecias

-poner el target “clean” como dependencia en el target “all” para que clean se ejecute

-esta es para borrar los ficheros .o

clean:

rm -f programa \*.o

-poner la regla principal ( la que devuelve el ejecutable) en primer lugar

-esto poque si ejecutamos “make” sin ningun parametro este ejecutara la primera regla del Makefile

-poner como primer regla una llamada “all” y como dependecia “programa”

all: programa clean

VARIABLES EN MAKE

-se recomienda poner el nombre de las variables con mayusculas

-se las puede declarar en la terminal como variable local de linux o como variable en el Makefile

-por lo general se usa las variables para guardar:

-nombres de los ficheros

-los flags que se usaran

OBJS = main.o salida.o calculadora.o

-asi se declara la variable

${OBJS}

-asi se usa las variables

TIPOS DE VARIABLES

-variables de expansion recursiva

-variables de expansion simple

-variables predefinidas

x = Hola

y = ${x} Adios #variable recursiva, en y se guarda la direccion de memoria de x

z := ${x} Adios #variable simple, en z se guarda el valor de x

x = Hello

result;

y = Hello Adios

z = Hola Adios

VARIABLES PREDEFINIDAS EN LAS REGLAS

[https://www.gnu.org/software/make/manual/html\_node/Automatic-Variables.html#Automatic-Variables](https://www.gnu.org/software/make/manual/html_node/Automatic-Variables.html" \l "Automatic-Variables)

-aqui se puede ver todas las variables predefinidas

$?

-esto nos devuelve un array con todas las dependencias que se pusieron en la regla en la que se esta ejecutando

$<

-esto devuelve la primera varaible de las dependencias de la regla en la que se esta ejecutando

$@

-esto devuelve el nombre del target que se esta ejecutando en la regla

REGLAS IMPLICITAS

-cuando tenemos una regla de compilacion (cuando entrega el .o) podemos obviar el comando y solo poner el target y las dependencias

-de esta manera GNU make ya sabe como ejecutar esto

-cuando make detecta un target con una extencion quereconoce, pero no tiene ningun comando, make ya sabe que compilador usar

PATRONES EN MAKE

-se usa % (porcentaje) reeplaza a uno o mas caracteres

-ver el ejemplo en ./programas\_c++/ejem\_make/Makefile

MULTIPLES MAKEFILE

include src/Makefile

-esto hace que se ejecute el makefile de ese directorios

-esto se usa cuando el programa es tan complejo que se usan varios makefiles

EXTRAS EN MAKE

-a veces make se queda corto cuando hay muchos .o

-averiguar sobre **automake y cmake**

**LINUX DEBUGGER** (**gdb)**

-es el debugger mas usado en linux

-primero para habilitar el debugger al momento de compilar aumentamos el flag “-g”

gcc -g programa.c

-o al momento de usar el makefile ejecutarlo pasando esto como parametro

make CFLAGS=-g

-el flag “-g” se lo puede agregar al momento del enlazamiento o de la compilacion

-para correr el debuger usando el resultado dicho anteriormente ponemos

gdb programa

gdb programa.o

-una vez hecho entrara en modo debuggeo y aparecera “(gdb)” como prompt

break fichero.c:5

-esto pone un break point en la linea 5 del fichero.c

break funcion\_name

-esto pondra un breakpoint cuando inicie dicha funcion

clear funcion\_name

clear fichero.c:5

-de esta manera limpiamos los breakpoints

set args 20

-en caso de que el programa reciva argumentos para correr, se pasan asi

layout next

-esto lansara el layout de la ejecucion

run

-esto empezara a ejecutar el programa

next

n

-esto pasa a la siguiente linea, se peude poner next o “n”

print name\_var

-con esto imprimira el valor de la variable “name\_var” en es momento

print name\_array@10

-para imprimir arrays se pone @(arroba) despues del nombre y despues el tamano del array

-esto solo nos dara las direcciones de momeria de cada posicion

print \*name\_array@10

-poniendo el \*(asterisco) imprimimos los valores

watch name\_var

-esto nos avisara cada vez que esta variable cambie de valor sin necesidad de pedirselo explicitamente

step

-con esto pasamos de funcion en funcion, **tener cuidado con eso**

**-terner cuidado en el orden de ejecucion de las funciones para no perderse**

refresh

-cuando falla algo para que se reinicie todo el debugger

q

-poniendo “q” sale del debugger

ANOTACIONES DEL DEBUGGER

-en caso de que se quede en un loop, se puede darle ctrl+c para interrumpir la ejecucion del programa **no del debugger**

**-**con las flechas o la rueda del raton se peude scrollear en el display codigo

**APUNTES DE C++**

**NOTAS PARA MI**

-se comenta con //

-se comenta bloques con /\* \*/

-para poder intercambiar valores de variables en c++ se tiene que utilizar unas variables auxiliares.

-para hacer print

%d integer format specififier

%f float format

%c character

%s string

%u unsigned integer

%id long int format

-para inisializar variables de objetos se los tiene que hacer con llaves

-una buena prueba de inicion es el **programa0.cpp,** ahi hay:

-como usar una funcion deontro de la misma clase

nombre\_clase::nombre\_funcion();

-como usar condiciones

-**como usar tuplas**

#include <tuple> // tenemos que importar clase

std::tuple<int,float,char> mitupla; //asi se declara la tupla

mitupla = std::make\_tuple(10, 2.6, ‘a’); //asi se define la tupla;

int miint;

float mifloat;

char michar;

std::tie(miint, mifloat, michar) = mitupla; //asi se separa la tupla

-como usar strings

string variable = “hola rodri”;

printf(“la variable es %s\n”,variable.c\_str()); //asi se imprime una string

-para trabajar el numero de decimales con los que se trabajara

std::cout.precision(5); //asi definimos el numero de decimales que imprimira //esto esta en <iostream>

std::cout << std::fixed; //eso dice que a partir de ahora imprima con se definio en precision

std::cout << std::scientific; //esto dice volver a notacion cientifica

std::cout << hex << 255 <<endl; //poniendo el **hex** hacemos que imprima en hexadecimal

std::cout << dec << 0x10 << endl;//poniendo **dec** se imprime en decimal

std::cout << oct << 255 << endl; //poniendo **oct** imprime en octal

std::cout << setbase(3) << 255 << endl; //poniendo **setbase()** especificamos que se imprima en base 3 en esta caso

std::cout << square << 25 << endl; //asi se puede elevar al cuadrado el 25 sobre la linea de impresion

-se vera c++ version 11 en adelante

-la clave de c++ es como maneja la memoria

-uso de punteros

-**el uso de punteros es mejor respuesta a mover/copiar EN TERMINOS DE RENDIMIENTO DE SOFTWARE**

**-**un namespace es un contenedor o entorno abstracto creado para contener aun agrupacion logica de entidades unicas

-las entidades definidas dentro de un namespace solo estan definidos a ese namespace

TIPOS DE OBJETOS

-primitivos:

-integer

-floating

-void

**-**(int, char, float, double)

**-**para poder guardar una cadena de caracteres, se la debe poner como puntero

char\* cadena = “hola rodrigo”;

printf(cadena);

**-**para asignar un carácter se lo puede hacer desde el codigo ascii

char carácter = 58;

-**se puede asignar un char a un int** y viceversa

-se tomara su codigo ascii como int

-bool

-esto es igual que en python 0(cero) es falso y 1(uno) es verdadero

-derivados

-array

-punteros

-referencias

-definidos por el usuario

-estructuras

-uniones

-clases

-enumeraciones

**CASTEAR VARIABLES**

char carácter = ‘X’; //aqui esta guardado como caracter

int entero = carácter; //aqui esta guardado como codigo ascii

cout << “carácter << static\_cast<char>(entero) <<endl; //asi se castea las variables

-para castear a otro tipo ponemos:

static\_cast<float>(variable)

TIPOS DE CLASES

**clases POD**

**-**estos son las estructuras de datos, solo pueden tener solo datos

-en c++ se los declara con **struct o class**

**-**struct los declara como publico

-class los declara como privado

-pero se lo puede poner en publico con palabra reservada **public:**

-se puede poner funciones dentro de las estructuras

struct Hucha {

int moneda;

void add\_moneda(){

moneda ++;

}

};

class Books {

public:

char titulo;

char autor;

}

**ENCAPSULACION**

-es tener en una clase variables y fuciones

-se puede esconder las variables dentro de las clases

**CONTROLES DE ACCESO**

-public : cualquiera accede

-private : solo de la misma clase peuden acceder

-protected : solo tienen acceso los miembros de la clase **y tambien los que heredan**

**CONSTRUCTOR**

**-**no es obligatorio usarlo

**-**son metodos que no retornan nada

-su nombre siempre tiene que coinsidir con los nombres de la clase de la cual forma parte

-dentro del consructor se puede

-inisializar variables

-poner funciones

-cuando se incicializa un objeto tambien se ejecuta el constructo

-una clase puede tener mas de un constructor todos con el mismo nombre

**DESTRUCTOR**

**-no es obligao**

**-**por defecto este atributo es llamado por el compilador el los llama en el momento

-se lo escribe igual que el constructor pero con una seja antes del nombre

-no resibe ningun argumento ni devolver argumento

por lo general se lo usa para:

-liberar identificadores de archivos

-limpiar sockets de red

-liberar objetos dinamicos

~destructo(void){

printf(“objeto destruido”);

}

**PUNTEROS**

**-**un buen ejemplo de punteros esta en **programa6.cpp y en programa5.cpp**

-encargados de almacenar las direcciones en memoria de los objetos

-el puntero tiene que ser el mismo tipo que la variable

int puntos{}; //inisializamos una variable con 0

int\* punto\_direcc{&puntos} ; //iniciazliamos un puntero con la direccion de memoria de puntos

int\* punto\_direcc = &puntos; //esta es otra forma de inicializar puntetos

-asi se declara el puntero y **se tiene que inicializar** si o si

-para asi

-los punteros **siempore tiene que estar inicizliado**

printf(“&puntos: %p\n” , puntos\_direcc);

-para imprimir un puntero el especificador de formato es **%p**

OPERADOR DE DIRECCION (&)

-con este operador podemos tener la direccion de memoria **DEL OBJETO** al cual se antepone

-se lo usa mas con los punteros como se ve arriba

OPERADOR DE DESREFERENCIA (\*)

-antepuesto al un puntero nos devuelve **EL VALOR** de memoria de la direccion del puntero

-si asignamos valor con este operador la variable (espacio de memoria) de la direccion del punter sera el que cambie

\*puntos\_direcc = 123

-con esta linea **la variable puntos** sera la que tenga el valor 123 y el puntero **puntos\_direcc** se mantedra sin cambios

OPERADOR FLECHA

-cumple dos tareas

-desreferencia el puntero (la misma funcion que el \*

-accede a los atributos del objeto al cual apunta el puntero especificado

sruct estructura{

int var1{};

void funcion(){

printf(‘hola’);

}

}

int main(){

estructura estructura\_var;

estructura\* estuctura\_ptr = &estructura\_var

printf(“%d\n” , estructura\_ptr→var1);

return 0;

}

**REFERENCIAS**

**-**tanto las referencias como los punteros son utilizados para evitar la copia de grandes estructuras de datos

-todo lo que se cambie en la referencia se cambia en el original

-es como un alias

-es solo un nombre alternativo a las variables

int x = 10;

int& ref = x; // asi es que se crean las referencias

**limitacions:**

**-**las referencias **siempre** tiene que ser inicializada cuando se la crea

**-**no podemos tener referenccias con NULL.

-una vez inicializada la referencias no puede cambiarse el objeto al que referencia

**DIFERENCIAS ENTRE PUTNERO Y REFERENCIAS**

-la referencia solo cuando se lo crea, los punteros se los puede reasignar cuantas veces quiera

-las referenicas **siempre** apuntan a un objeto, los punteros pueden a puntar a NULL

-no se puede obtener direccion de memoria de las referencias, de los punteros si

-no se puede hacer aritmetica con referencias, con los punteros si

**PUNTERO THIS**

-this es un punteroque apunta al objeto donde se este ejecutando en ese instante

-se puede ver un buen ejemplo en **programa7.cpp**

**CICLO DE VIDA DE UNA VARIABLE**

VARIABLES DE DURACION AUTOMATICA

**-variable local** ( variable automatica) : esta variable se limita en vida tanto como en acceso se limita a la funcion la cual la creo,solo dentro de la funcion se la puede utilziar

VARIABLES DE DURACION ESTATICA

-**variable global :** para que sea una variable global se la tiene que declarar fuera de todas las funciones (en raiz), esta variable puede accederse y cambiarse en cualquier parte del codigo

-**variable estatica local:** solo puede ser accedida dentro de la funcion donde fue creada, su ciclo de vida empieza cuando se declara su funcion y muere **cuando el programa termina**

**-**se la declara poniendo un static delante del tipo

static int variable = 0;

-osea que cuando se ejecuta el objeto mas de una vez y se altera la variable estatica todas esas veces, la variable guardara todos los cambios hasta que termine el programa

-esta es como una variable global peroq ue solo se la puede acceder dentro la funciones

-ejemplo de esta variable esta en **programa7.cpp**

-para poder acceder a una variable estatica loca desde afuera del objeto se lo hace desde la instanacia y con dos : (::).

-ejemplo de esto en **programa8.cpp**

**-variable estatica global:** esta varaible se la declara fuera de todas las clases (en raiz) su siclo de vida es igual que la variable esta

**-varaible extern**:esta veriable permite comunicar con oros programas

-solo se pone la palabra **extern** antes del tipo de variable

VARIABLES DE DURACION DINAMICA

-se asignan y desasignan a peticion

-el programador determina el timepo de vida

-se asigna con “new” y se desasigna con “delete”

int\* mi\_ptr = new int;

int\* mi\_ptr = new int{32};

delete mi\_ptr

-cuando se elimina este obejto **se tiene que limpiar el espacio liberado**

-el comando **new** crea un objeto del tipo especificado y **devuelve un putnero** al objeto

-**este sistema de asignacion dinamica de memoria en tiempo de ejecucion que decuelve un putnero ofrece una optimizacion de rendimiento indiscutible**

**-**es por eso que los motores de creacin de software mas potentes lo usan

-unreal envigne

-amazon laber gear

-**es importante aprener bien como c++ gestiona el tiempo de vida de los objetos**

**THREADS EN C++ (CONCURRECIA)**

thread\_local int variable{“nombe”};

-asi se crea un thread

-esto significa la capacidad la capacidad de manejar variaos procesos al mismo tiempo

-esto se puede lograr de dos moneras:

-multitareas : con un solo nucleo ejecutar varias tareas

-procesamiento paralelo : usando multiples mucleos

-**hay que tener cuidado:**

**-**con las variables estaticas globales : por que cambiar el valor en un hilo **cambiara el valor en todos lo hilos**

**-**para evitarlo se da a cada hilo su propia copia de a variable

static thread local int variable=200;//asi se tiene que declara la variable estatica local

thread local int variable = 200;// poniendo asi tambien es lo mismo

**no esta terminado >> leer mas**

**CONDICIONALES**

-los operadores de condicion es igual que en python

IF

if (condicion){

instrucciones;

}else{

instrucciones;

}

SWITCH

switch(selector){

case 1:

instrucción;

break;

case 2:

instrucción;

break;

default:

instrucción;

}

FOR

for (int contador = 1 ; contador <= 10 ; contador ++){

instrucciones;

}

**-**en el primer lugar es el valor incio

-en el segundo lugar se pone como final del conteo

-en el tercer lugar se pone el paso que tendra el contador

-mirar un buen ejemplo de uso de for con punteros en **programa6.cpp**

WHILE

while(condicion){

instrucciones;

}

-while en c++ funciona igual que en python

DO WHILE

do{

intrucciones;

}while( condicion );

-primero se ejecuta las instrucciones **despues** se evalua la condicion

**POLIMORFISMO**

-en tiempo de compilacion

-en tiempo de ejecucion : este es cuando se ejecuta el programa

-se lo logra usando:

-interfaces

-composicion

-objetos

-herencias

**CLASE ABSTRACTA E INTERFACES**

**-**no se puede crear objetos de una clase abstracta

-el objetivo de estas clases es servir de base para las **clases concretas**

-las clases concretas son las que **heredan y definen**

-las clases abstractas tienen **metodos virtuales puros**

-los metodos vituales son metodos que se declara en la clase abstracta pero de la define en la clese concreta si o si

-**los metodo virtuales NO pueden ser globales ni estaticas**

-un metodo virtual puro se lo declara con la palabra reservada **virtual e igualando a cero**

-las interfaces son lo mismo que la abstracta

-En resumen las clases abstractas son solo las declaraciones de los metodos y variables base, pero vacias, entonces la clase concreta la hereda y define los metodos y variables

-un ejemplo de esto esta en **programa11.cpp**

class base{

int nLados;

public:

void setNumero(int n) { nLados = n; }//en la clase base se peude tener clases concretas

virtual coid mostrar() = 0; /asi se declara los metodos virtuales

OVERRIDE

-este comando se usa para evitar problemas en tiempo de ejecucion

-este comando compara que funcion definida en la clase concreta se efectivamente un metodo virtual de la clase base

=ejemplo de esto se puede ver en **programa11.cpp**

**LIBRERIAS**

#include <iomanip>//setw(),getline()

cout<<setw(15)<<variable<<setw(10)<<variable2<<endl;

-esta funcion es para imprimir en fromato de columna

-setw pone 15 espacios en donde se lo especifica

string var1

getline(cin,var1); //esto es lo mismo que cin>>var1;

while ( var1.length() == 0 ){

getline(cin,var1);

}

-se usa la funcion getline poq cuando metemos texto con espacios cin solo recive la primera palabra

-**cuidado :** cuando se tenga un “cin” u otro “getline” antes de un getline, puede que detecte el “enter” de la funcion previa como suya, entonces guardara vacio en var1

-para solucionar esto poner el ciclo como en el ejemplo para corroborar que guarde bien la variable

-un buen ejemplo del uso de esto esta en **programa12.cpp**

#include <math.h>

int numero = 2;

int potencia = pow(numero , 2); //esto eleva al cuadrado numero, sep uede poner al cubo

#include <iostream>

-esta es la forma de importar librerias

-iostream es la librería con la cual podemos utilizar las entradas y salidas etandar

using namespace std;

-si no ponemos esto tendriamos que imprimir asi

std::cout<<’hello world’<<endl;

-pero poniendo esta linea solo seria:

cout<<’helloworld’<<endl;

-endl esto solo es para hacer un salto de linea es como (\n)

int variable;

cin>>variable;

-para recibir variables del teclado tenemos que declarar la variable antes de usarla

cout.prescision(2);

-con esto configuramos el numero de decimales se imprimira en pantalla

**COMPILAR Y EJECUTAR C++**

g++ -c nombre.cpp

-esto nos crea el fichero compilado del programa llamado “nombre.o”

-este archivo es un fichero .objeto

g++ -o nombre\_exe nombre.cpp

-con esto **enlazamos** el fichero fuente y nos devuelve un ejecutable, podemos pasarle el .cpp o el .o como entradas

-esto se hace asi poq en caso de q el programa dependa de carios ficheros, solo se recompila el necesario y el resto directo se usa su fichero compialdo previamente

-cuando trabajamos con muchos ficheros lo mejor es trabajar con **Makefile,** esto hara mas facil la compilacion del programa final

g++ es el compilador en linux

-o es el flag de salida del ejecutable

nombre\_exe es el nombre del ejecutable que resultara de este comando

nombre.cpp es el fichero del programa de c++

**C++ MODERNO**

-se conoce como c++ moderno a las versiones de c++ mayores a la 11

#include <cstdio>

int variable = 34;

int main(){

printf(‘hola mundo”);

return 0;

}