PROGRAMACIÓN Y REDES

Diferecnia entre dir ip ficsico y lógico, depende de donde la mires, si estas mirando dir ip + puertos(fisicas), estamos en la cap de aplicacion

Dir ip lógica, las que eliges tu,

Ventajas e incivenientes entre los mensajes de texto y mensajes bianrios

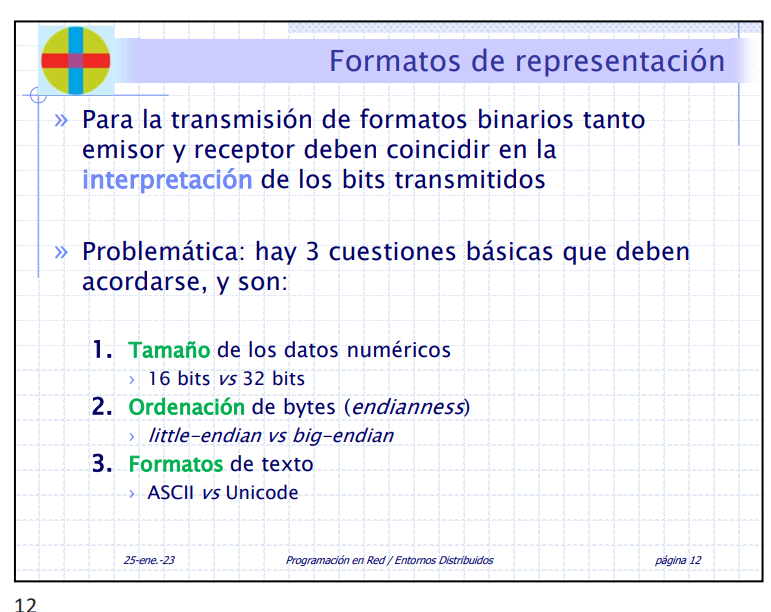
Mensaje texto: incveniente: teienes que llegar a un acuerdo si el texto es valido o invalido, inconveniente,como sabes si el texto enviado es correcto(defones una sintxis(estructura), con una gramática, mediante parsers de texto )

Ventajas,

Mensajes binarios: inconvenientes,

Ventjas, nos protege si el canal no es perfecto, da igual escribir mas

Como mínimo para poder intercambiar datos



Formatos de texto en utf-8

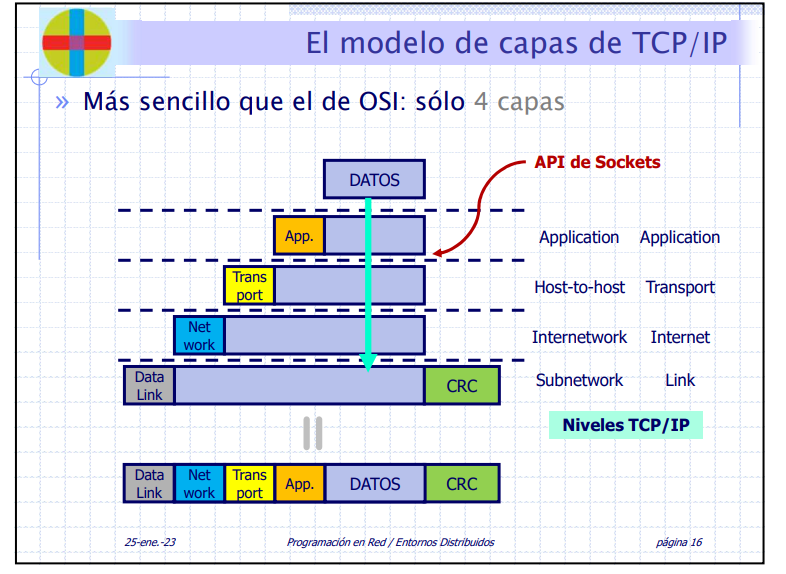
Endianess. Si vas a intercambar datos pro tcp/ip le intresa saber tu dieccion tu dir ip esta formada por 4 octetos de bytes, tienes que definir la endianess, normalmente, se escoge el big endian, ya que en los 80 los procesadoes tenían big endian(menos intel), tenemos una red definida como big endinan formada por host que utilizan Little endian,

Si vas transitir caracteres te teien q ponerte de acuerdo con el formto8unicode, utf-8..) y si vas a transimit n binario tiens ue vponerte de acuerdo en la endianess



Vas bajando y se añade cabecera en función de la info que quiera proporcionar cada nivel tcp/ip

Transporte: la cabecera que se añade tiene dentro info acerca del puerto,



Subnetwork: la trama,

Las rayas son interfaces, la primera es la interfaz de usuario,

Puerto: te permite dsitinguir proceso de una maquina, una máquina tine normalmente 220 porceso o así,

Para subir a la capa de aerrba necesitas undriver para enviar y recibir, para integrarlo dentro de tcp/ip,

La interfaz entre interntenetwork y subnetwork,Lo que hace es conectar con varias tecnologias

IP, encaminamiento,

Garantia de ip, bestefffort, no da ninguna, perdida de fallos, duplicación, que lleguen en desorden

Udp, pone y quitas los puertos, es un mux de paquetes ip, funciona bien cundo las coss a transmitir son sencniallas

Tcp, conmutación de cirtuatis, conexión viertual entre los extremos, (ventna deslizante ), reordena paquetes,

Red ip privada, se utilizan para wu no pongas la ip publica, ya que si la pones significa qu tu casa es internte, es decir, qu si quieres acceder al mismo recruso de internte que has establecido como tu dir ip no podras

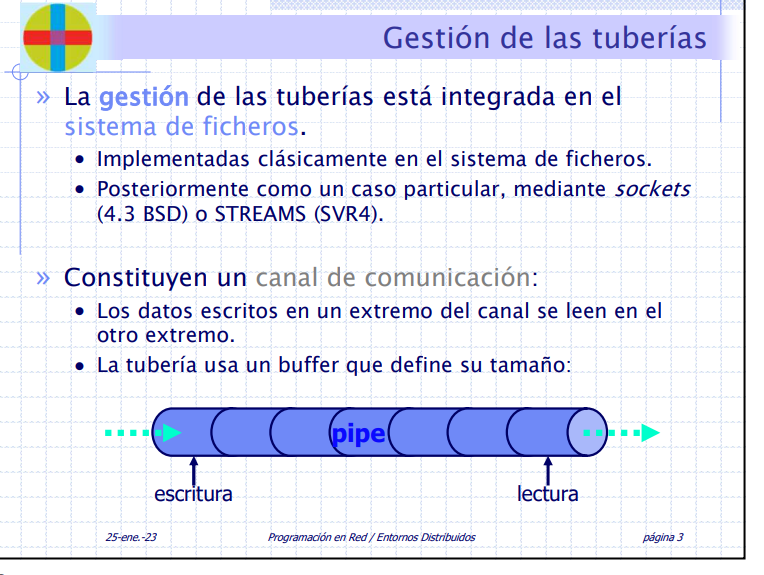
TUBERÍAS Y FIFOS

No existe ningún unix que no tengan tuberías, salvo versiones anteriores a 1973,

Sistema de memoria virtual, traduce las direcciones d memoria, en unix lammeoria es privada, no se pued utilizar para transmitir datos, àra ello se crearon las tuberías, es una ola de tipo de fifo, además permite la transmisión fiable(garantia de q llegan ls datos, si metes u5 numeros transi¡mite 5 numeros), chorro de bytes(que es continuado, no tines forma de sber todo lo q hay, si tinees una tubería, si metes 3 cosas, el otro extremo solo tine que leer una vez, no 3, )

Halfduplex(solo puedes leer o escribir, no puedes hacer ñas 2 cosas a la vez)

Presentran un tamaño máximo fijo para el buffer(zona de almacenamiento temporal, q luego se vorra todo elt iempo),coges un cacho de memoria, y esta memoria es de una tubería, cuando quieras enviar algo lo que haces es acceder al kernel, es un cacho de memroia que tiene el kernel, escirber dentro del kernel con llamadas al sistema, y luego lo lees con otra llamada al sistema,



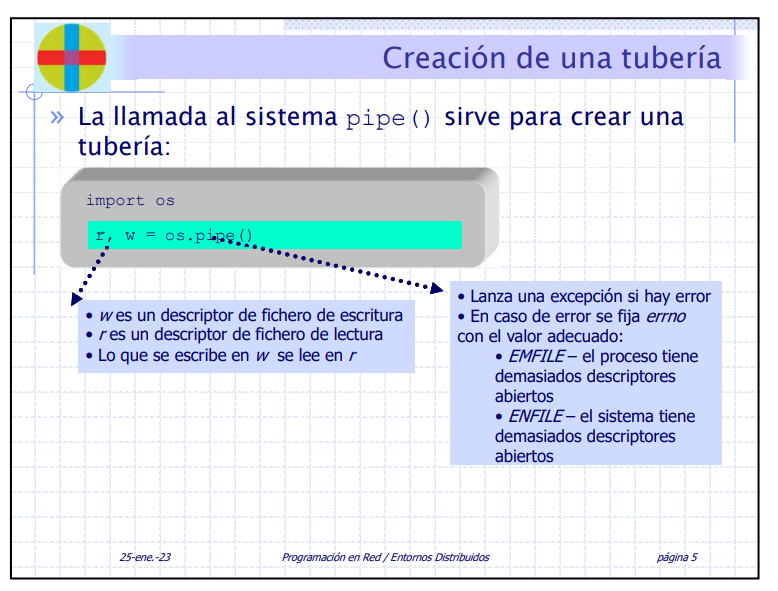
Siq ueires enviar un mensaje a alguien en una tubería de 7 bytes, es una implementación de un buffer memoria circular que se escribe en un lado y se lee en otro, en una tiberia se escibe y se lel igual que en fichero en unix(con lamamdas al sistema, open(si el open funciona, el proceso tiene un file descriptor), read write(se escribe bytes,),close)

Unix. Abstracciones(una pa guardar info, el fichero(hay 3, ficheros, directorios(guardan nombres de otros ficheros y sirven para llegar a otros ficheros) y especiales(representa cosas de E/S(un usb, el ratón , el teclado..) dispositivos modo bloque y modo carácter, diferencia(bloque, debajo tiene la block buffer cache)))) y otra para hacer cosas(el proceso)

Cahe. Esta cacheando algunos sectorrs de los discos(estos de guardan dentro de la zona de memoria del kernel), los mas utilizados, y los guarda en la RAM, cando tenia que ller muchos los guardaba,

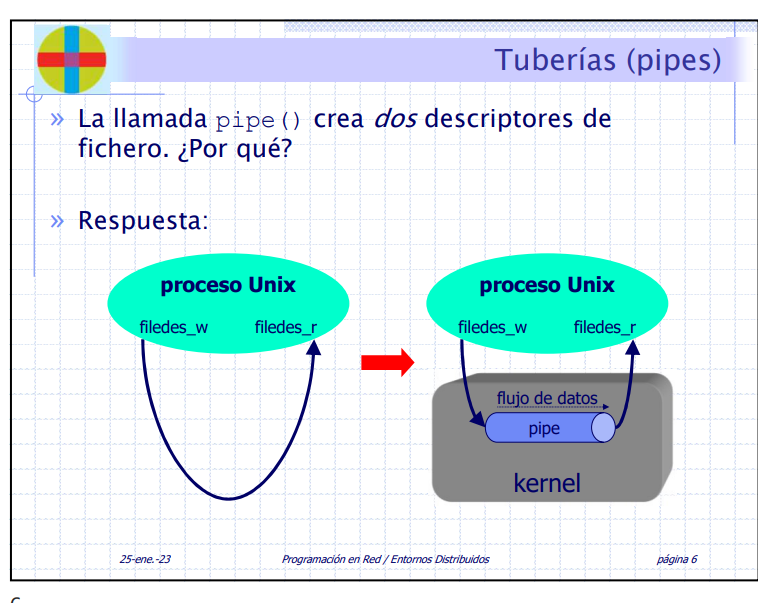
» La lectura/escritura se realiza mediante un chorro de bytes sin ninguna estructura. ∙ La lectura de los datos es independiente de la escritura. ∙ Permite leer de una vez datos escritos en varias ocasiones., lo mismo que un fichero, pero hora con tuberías,

Cuando hac una llamda al sistema se lo idce al kernel, porque tu no tinees accesoa esas cosas, y el kernel le dice que esperas, en un estado, en concreto, dormido(hasta quetermone de escribir,), y el kernel lyego lo despierta hace lo que le ha mandado, y pasa a la cola de los activos, el read funciona igual, y con una tubería, lo mismo, si ocupa el tamañode la tubería, utilizas otra llamada al sistema para poder seguir escribiendo en dicha tubería auqn no tenga almacenamiento, le divces que se duerma, la lectura es indenediente d ela escritura, se llee y escribe todo lo que s epuede,



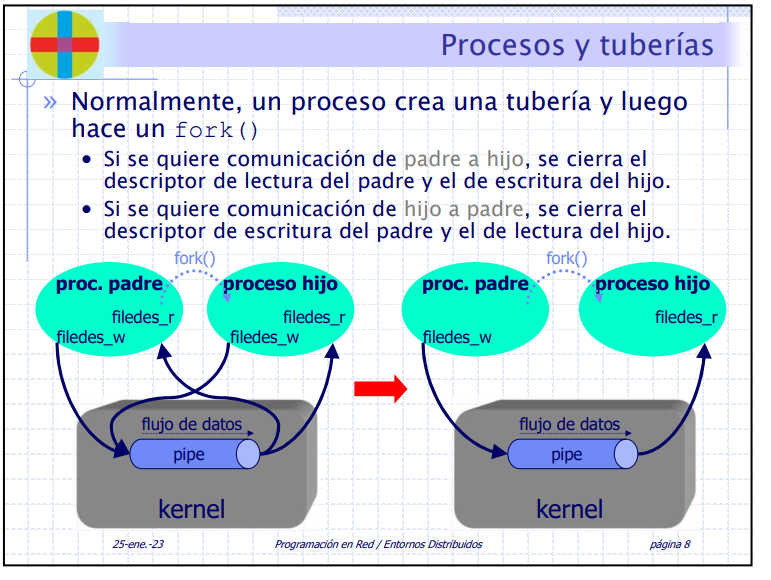
Es una llamada al kernel, pipe es una llamda al sistema, la escritura en unix con Python se hace importando os, la sointaxis es universal, es deir, es como se accede al kernel.,

r, w = os.pipe(), con esto creas una tubería, si hay un errro, ale errno, esto depende del so,

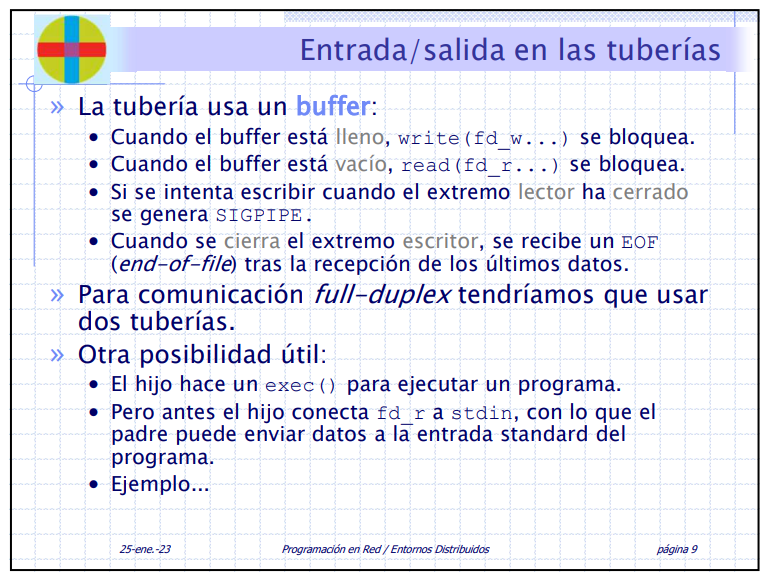


En el kernel se crea una ziona de meoria, unmextremo del pipe se cnecta a kazoan de scritura y otro al de lectura, conecta el mismo proceso unix el cual tinene dos descriptores de fichero,

Si solo tienesun descriptor e inntentas meter cosas a una tubería,



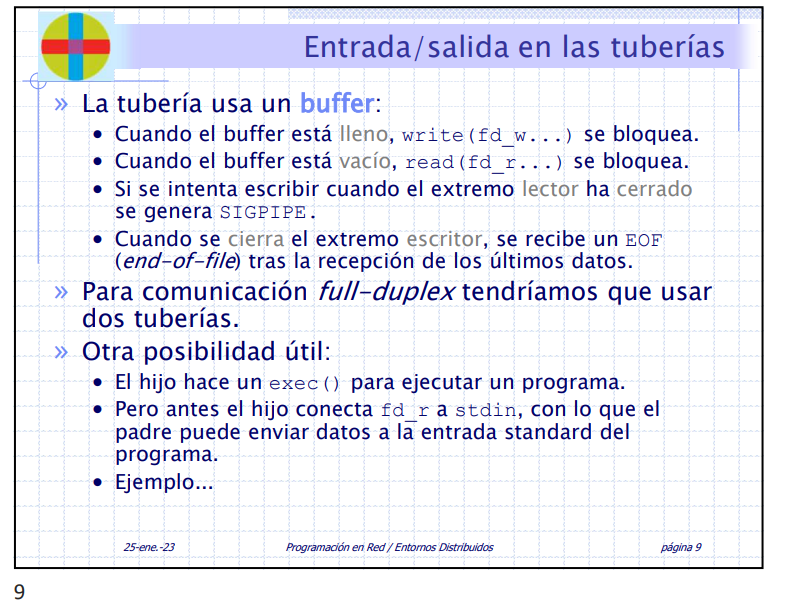
Las tuberías solo funcioan entre procesos distintos si ambos tienen un padre común, en la diapo, hace un close de los de la izq, para comunicar la escritura del padre con la lectura del hijo,



Condición final de lectura, cuando te devuelve si ya no queda nada por leer 0, como ha 2 porocesos, cuando el escritor hace un close, el read ya sabe que ya ha termiando porque e el extremo han devuelto un 0, si ocurre al revés, es decir, que el extremo del lector ha cerrado antes, se genera un SIGPIPE, cuando lo recibe dicho proceso se muere,



El tamaño de la stuberias, 512 bytes(es el sector de un disco),

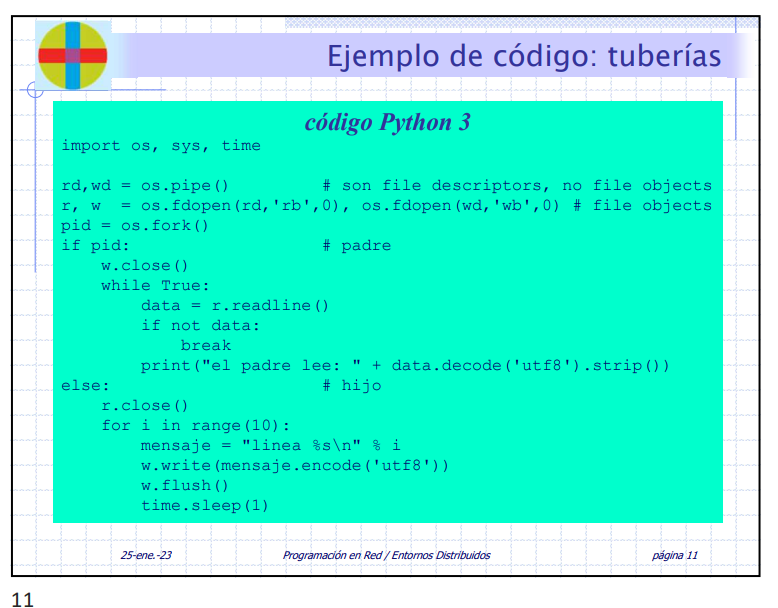


A que velocidad va, vas a la velocidad mas lenta, lastuberias son autosincronizantes, es mejor que le buffer sea pequeño, para adpatr la velocidad de los exrteemos,

Who(busca porceso) | sort(lls ordena) | lpr(imprime), es decir, imprime la lista ordenada de los procesos del sistema,

Coencta la saido de who con la entrada de sort,

Riesgode seguridad, las tuberías solo los puedes utilizar tu y tus hijos, entonces son inviolables,



Una tubería se maneja como un fichero, hay un proceso hijo que escribe 10 lienas cada segundo y el padre las lee,

Limitaciones, solo puedes esto entre parientes(es la mas restrictiva ), es unidireccional, el tamaño del buffer es fijo,(cuanto mas grande sea, es peor, ya que se puede dar el caso de que el otro poceso no lo ela y estes peridnedo tiempo)

Vmos a hacer que las tuberías parezcan ficheros, una vez que el pipeline()FIFO parezca un fichero, lo abres, ecribes y lo cierras, lo único que tineen que saber es como se comprarte el fichero

Pregunta abierta: Las FIFOs son menos seguras que las tuberías clásicas. ¿Por qué?

Le pones permisos al fichero par que no se meta cualquier proceso, el sistema de permisos de unix es un sistema cuyo objetivo es la protección, diferencia entre seguridad(limitar los procesos chungos borres ficheros) y protección(evitar que nos hagamos daño inadvertidamente), no es recomendable utilizar un usuario root, ya que no le afecta el sistema de protección, ya que dicho sistema evita que te hagas daño a ti mismo, es mejor usar un usuario normal.

El sistema de protecciond eunix no es un sistema de seguridad,

Mkfifo(), es mkfifo f crear un fifo, p d pipeline, no tiene tamaño inicialmente,

Si haces cat < f, estas activando el proceso lector, se ha quedado pillado,

Cat > f para escribir ,

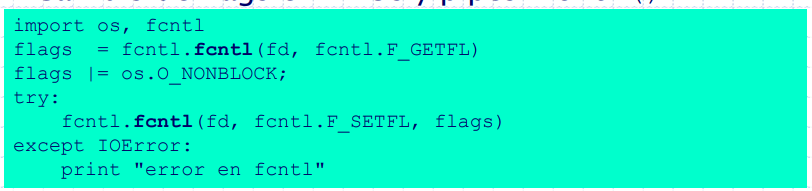
Control +d , no matar ni dormir el proceso,

Abres dos terminales,

Si ejecutas tres termianles con el mismo f, va a ver balanceo de carga(no exactamente), es porque es una cola, cuando le envia la escritura a uno, se sale de la cola y va ael otro poces, por eso lo va alternando, depende de como este la planificación de la cola, es insensato hacerlo, si escribes dos ficheros en un pipe, lo va escribir a cachos, esos cachos son el tamaño de la tubería, el tamaño no aaumento a media que vas escribiendo, por la p del pirncipio d ellos permisos

Ls -lai f, te da el inodo,

Utilizar la llamda mknod,



Le añadimos un flag llamada modo no bloqueante(0\_NONOBLOCK), si escribes a alguien q no lee te duermes, ano ser que pongas el flag y el write te devuelve un error ya que no esta preparado,

La constante PIPE\_BUF define el número máximo de caracteres que se pueden escribir en una FIFO atómicamente, es decir, lo cachitos que puedes escribir, si pones esto y un nonblock, no puedes escribir ya que si escribes 2k y solo entran 1k, te da error,



Como contestas peticines de lo clientes?,

Hace falta otra fifo un fifo petciones y otra fifo respuestas, es decir, habría que quedar una fifo de crpuestas d cada cliente, y esta fifo la crea el cliente antes de realizar la petición

Cuantas llamadas al sistema hacen falta para montar esto??,

Mkfifo f

2 open, uno para leer y otro para escribir,

Pregunta abierta: ¿Por qué una FIFO se abre en modo de sólo lectura o escritura pero no en modo lectura/escritura?

Lo que estas escirviendoi se te va avenir encima, porque las fifo son unidireccionales, es decir, conexión boccaculo, si quieres tener una comunicación bidireccionl hcen flta dos fifos