

preprocessor Quine

endianess

```
# define MAX(x, y) ((x) < (y)) ? (y) : (x)
```

non c'è il ; alla fine

MAX(3, 5) + 2

verrebbe eseguito

altrimenti:

MAX(3, 5) + 2

do {

≡

} while (0)

≡

lui lascia solo la sequenza

senza do { ... } while (0);

deve essere sulla stessa riga

```
printf("TABLE "
```

table A

\ 2 cap-  
# X)

"A"

"hello" "word"

considera le stringhe  
COSTANTI

```
printf (
    " Usage : \n "
    " options : \n "
    "      -c      \n " );
```

table ## X

constituzione 2 stringhe e fare un unico  
identificatore + un parametro

typeof (X) che tipo è la variabile X?

GCC (C - .03) funziona solo per

Debian pedante NON NE CALDEGGIA L'USO.  
NON NE ACCETTA NESSUNA quindi NON LA USATE MA VI PIÙ.

usa getline invece che gets

Never use this function

Perche syscall rispetto chiamate di funzione

(3 mod)

Sicurezza

Kernel  
processo

processo utente < suo codice  
memoria concessa

tutto il resto syscall

aritmetico  
logiche

accesso mediato a syscall

ovattato nel suo guscio

Il sistema non in pericolo

Kernel può uccidere il processo

modo protetto

Funzionalità syscall

gestione processi  
(exit, fork)

gestione memoria  
(exec, brk)

gestione File  
(open, write, close) struttura

ls -s /dev/tty 1

-l /dev/vda 1

gestione filesystem (mkdir, getdir entry)

gestione utenti who am i group  
cambia identità  
permessi di root

I/O core file  
periferiche

gestione errori  
↳ segnali

directory file  
di naming

gestione segnali

errori, concorrenza  
sincrona

IPC

inter  
process  
communication

attesa eventi

disposizione file

select poll epoll emulatore di terminale  
eventi e similari

gestione tempo

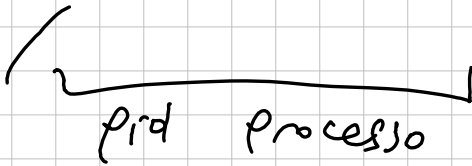
networking

Miscellanea

MISC

esempio come sistema

pid

cd /proc /   
pid processo

ls -l fd

ps 2x | grep "commando"  
tcpip

networking => can  
Car area network

Il "catalogo" delle syscall.

getppid

parent process identifier

( )

chiamata

syscall per chiamare

corsivo

obsolete

6 voci / parametri

comando

syscall

programma  
che lancia

istruzioni in un programma C

man 108

man 2

# strace

man 2 syscall

number syscall -- NR\_write

vi /usr/include/x86\_64-linux-gnu/asm  
/unistd-64.h

int rv = syscall(\_\_NR\_getcwd, NULL, 0)

echo errno

#include <errno.h>

vi /usr/include/x86\_64-linux-gnu/asm/errno.h



## Fork

execve è l'unica syscall

le altre exec del man 3

"sistemone" i parametri e chiamano  
tutte execve (unica syscall)

- L long path, tanti parametri

esempio `exec("/bin/ls", "ls", "-l", "/", NULL);`

`my args = { "ls", "-l", "/", NULL };`

`exec(..., my args, ...)`

- P path, il path del file/cmd lanciato viene cercato anche nel \$PATH del sistema
- E environment

```
void clear 1() { printf("clear 1\n"); }
```

```
void clear 2() { printf("clear 2\n"); }
```

```
main() { atexit(clear 1); atexit(clear 2);
```

```
    exit(42) // chiama atexit... *
```

```
    _exit(42) // non chiama le atexit
```

```
    return 42 // fa lo stesso di exit }
```

\* il programma stampa clear 2\n clear 1\n

questo perché se A deve svolgere cose prima di uscire e B usa A e a sua volta svolge altre cose dovremmo chiudere in ordine prima B poi A se no avremmo chiuso A quando B ne aveva ancora bisogno.

quindi se C usa B, B usa A, A avremo le chiusure in ordine di C, B, A.

E nell'esempio atexit viene "normalmente" chiamato in ordine inverso all'ordine nel codice.

mickey mouse



10 Novembre

syscall il kernel che si occupa.

getline (&buf, &len, stdin)

buf all'oscuro / più lena  
dinamicamente  
indirizzo attuale del buffer  
lunghezza attuale  
del buffer  
(non della linea)

abbiamo visto:

fork  
exec

gestione processi:  
getpid, getppid, waitpid

crea processi:

fork()

≠

esegui programmi:

execve(...)

↓  
gestione memoria  
non crea processi.

stato non precedente e butta via  
senza esecuzione e prosegue

Fork

clonata con memoria SEPARATA

execve      È l'operazione successiva  
Se e solo se fallisce.

execve(...);

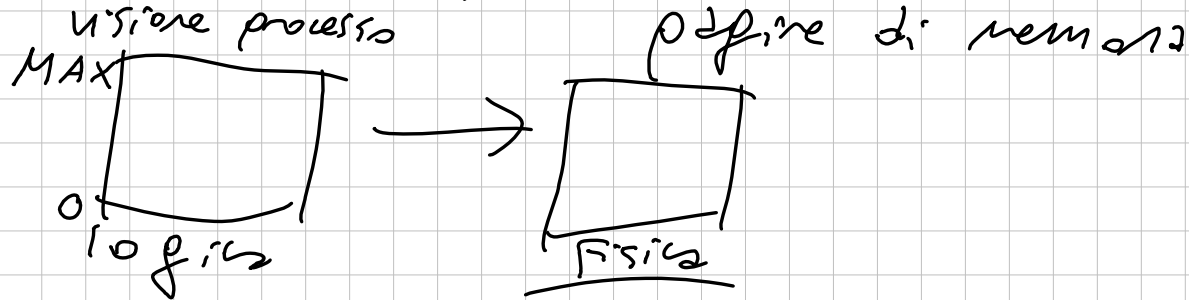
printf("execve ha fallito");

exit(); altrimenti il figlio e il padre  
farebbero getline...

"minish.c" 20221020

# brk

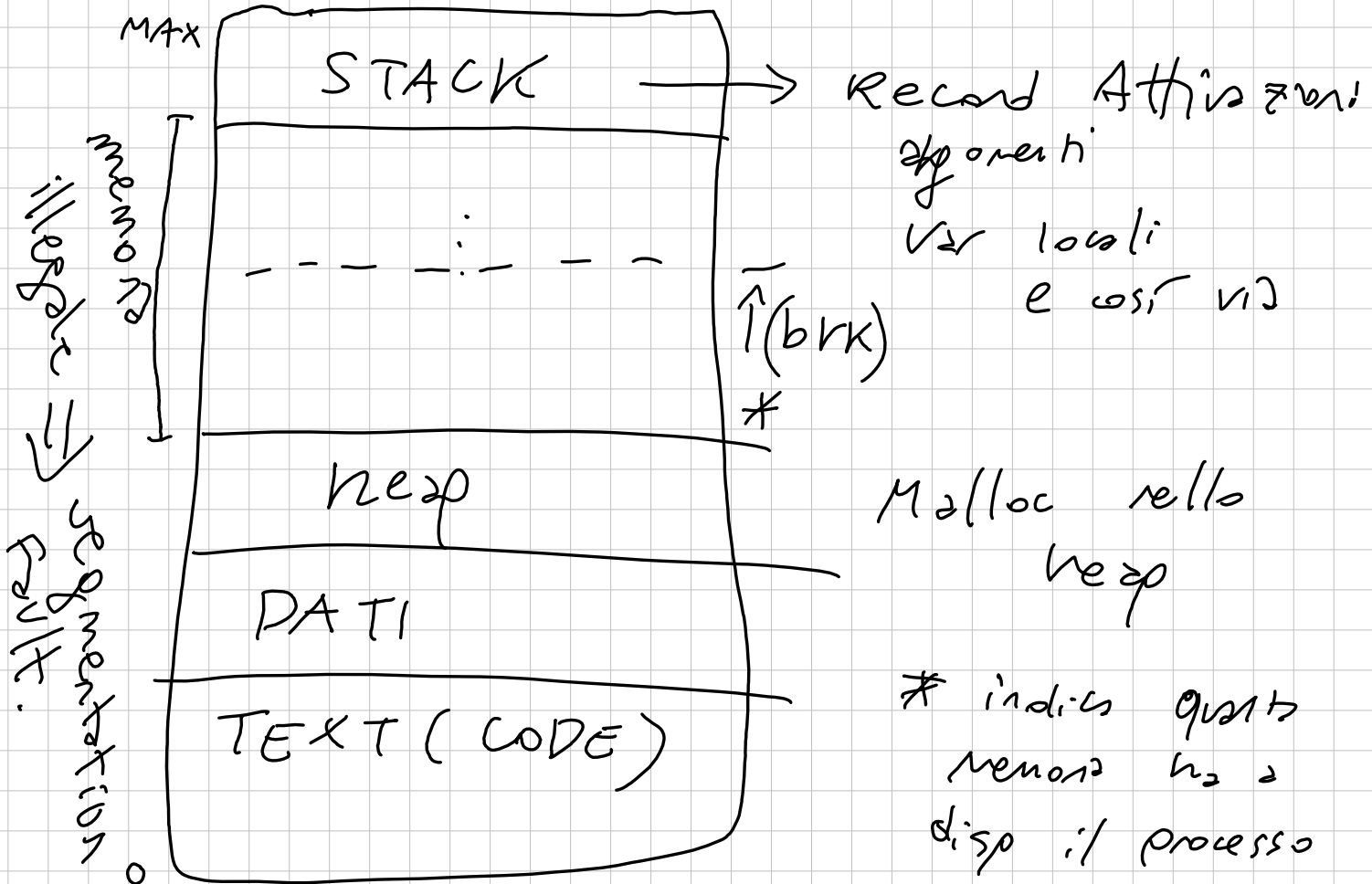
## (Memory Management Unit)



visione logica      visione fisica  
della memoria

\* brk richiede altra memoria  
quando non gli basta più lo spazio.

# Memoria logica del processo



brk sposta gli indirizzi "legali"

Fork "copy"

copy on write

non viene copiato niente

condividiamo la memoria con un flag

che se uno prova a modificare  
qualcosa viene copiato.

Finché leggono solo è condivisa.

MMU

→ padre  
→ figlio

"bloccetto" modificato

sdoppiata la pagina che "modifica"  
la variabile uno dei due

2022-11-10 oggi

"lib c.p.c."

non è una syscall altrimenti  
Fopen(3) qui avremmo  
un 2 non  
un 3!

MAN —

Fgetc(3), Fputc(3), Fclose(3)

vi prova "sempre caro mi fu quest'emo colle"

operat      trace      ./libcp  
read      write



cos'è un file?

(XFILE)  
film

pratica

avvocato

aperto una

pratica

— cercare in archivio

chiudi la

pratica

apertura

"adesso vogliamo aprire  
in lettura/scrittura una pratica"

tutte le operazioni a monte fatte  
una volta per tutte.

(altrimenti verrebbe fatto a ogni carattere)

inode

file system

vnode

aperto

info di: ~~il~~ <sup>con</sup>cesso  
processi che lo stanno  
accendo

quando l'ultimo processo chiude  
salvato in un campo del  
vnode

Il vnode chiuso e inode salvato  
opportunamente.

definite in  
#include <fcntl.h>

accendi questo bit e ... e ...

O\_WRONLY / O\_CREAT / O\_TRUNC

"`fcntl.h`"

0666

rw-

dd if=/dev/urandom of=longFile bs=1024 count=1024

time ./libcp longFile longFile 2 → 0.137 sec

time ./scp longFile longFile 2 4096 → 0.036 sec

1 → 43.267 sec

disk duplicate

copia veloce serve per duplicare  
dischi.

---

casi uso concomitanti

↓  
pthread

(clone)  
main usata  
direttamente

pread  
pwrite

man pread

offset

lseek

leggere

turnare da  
capo

n'leggere

NO open-close costoso

Si usa la LSEEK

offset  $\left\{ \begin{array}{l} \text{SEEK-SET} \quad \text{da capo} \\ \text{SEEK-CUR} \quad \text{pos. corrente} \\ \text{SEEK-END} \quad \text{fine file} \end{array} \right.$

DUP — stanza

duplicate

stdin

file

stdout

man dup

descriptor

stderr

dup2 dup3

se new è aperto chiudi:

dup 2 è atomico

dup 3 ha anche un flag

"red out.c"      redirectione

cat

ls -l /proc/PROCESSO/fd  
CAT

--- 0 → stdin

--- 1 → stdout

--- 2 → stderr

File descriptor

contrassegno per individuare  
la pratica aperta

open restituisce un numero

quel numero identifica quella pratica aperta.

"redsh.c"

./!!

ctrl + R

cerca a ritroso

tutti i comandi

che cominciano con...

ls | sort -r  
systemcall pipe  
man pipe

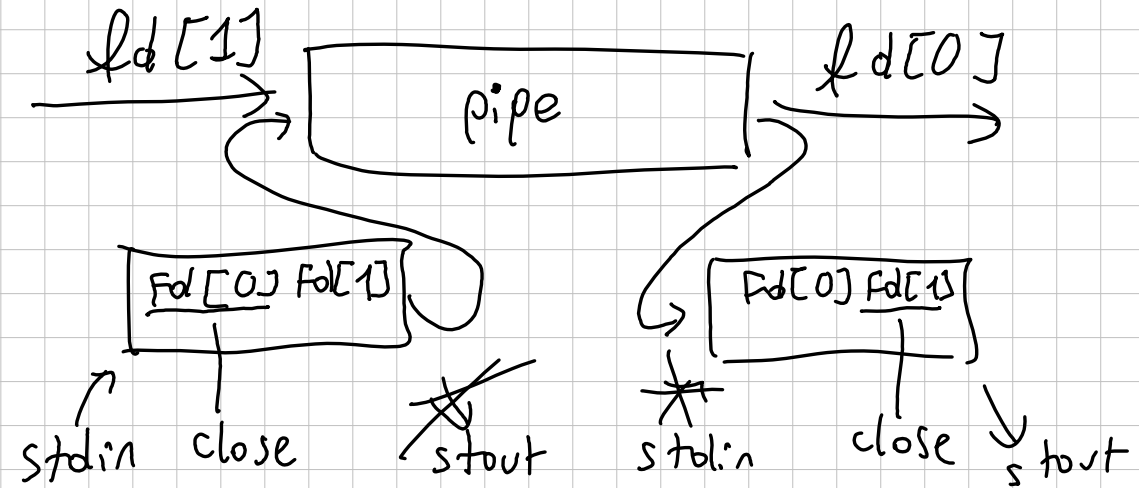


p1.c " pipe

"red pipe.c"  
come fa la shell  
a fare pipe

bounded buf

wait aspetta qualsiasi figlio  
waitpid aspetta un figlio specifico.



pipe devono avere un ancestor comune

pipe tra processi non parenti  
dobbiamo dare un nome alla pipe  
name pipe e usare il filesystem

mkfifo mynamedpipe

ls > mynamedpipe

(e su un altro shell)

sort -r mynamedpipe

STRACE myFifo

mknod crea un file speciale

Un nome associato a un significato  
⇒ NON HA DATI





(creat) open e poi si falliva la creat.



esiste per motivi storici

syscall

parametri

CREAT = OPEN + O\_CREAT / O\_TRUNC / O\_WRONLY

MKNOD File privo di contenuto  
senza aprirlo

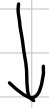
File vuoto

o altre cose per cui  
serve avere un nome

FIFO, SOCKET UNIT

DEVICE / caratteri  
          / blocchi

PIPE



due descrittori  
comunicazione  
tra processi  
parenti

NAMED PIPE



nome

processi  
completamente  
indipendenti

se comando = syscall

il comando chiama la syscall

S\_IFREG S\_IFCHR ... (vedi 'man 2 mknod')

crea un inode e ci dà un nome  
attaccandolo nell'albero in un punto,

device si aprono come se fossero file  
hanno I/O directory /dev per i device  
ls -l /dev

disk vda non possono essere  
vda 1 2 credit né in  
b scrittura (rovinano tutto)  
b 1 né in lettura (privacy)

tutti i device hanno 2 numeri  
(le directory il numero indica la dim.)

una scheda audio

come si danno queste info (alza il volume...)  
ai device? esiste una chiamata:

IOCTL Input output control

IOCTL fa tutto quello che non  
può fare la read... write

fcntl  $\xrightarrow{\text{locking}}$  F\_SETFL  
file control flock

CLOEXEC  
all'atto dell'  
exec viene  
chiuso.  
(se ha questo flag acceso)

FILE  $\xrightarrow{\text{open}}$  FILESYSTEM

pratica

astrazione per catalogare file

fd intero  
file descriptor

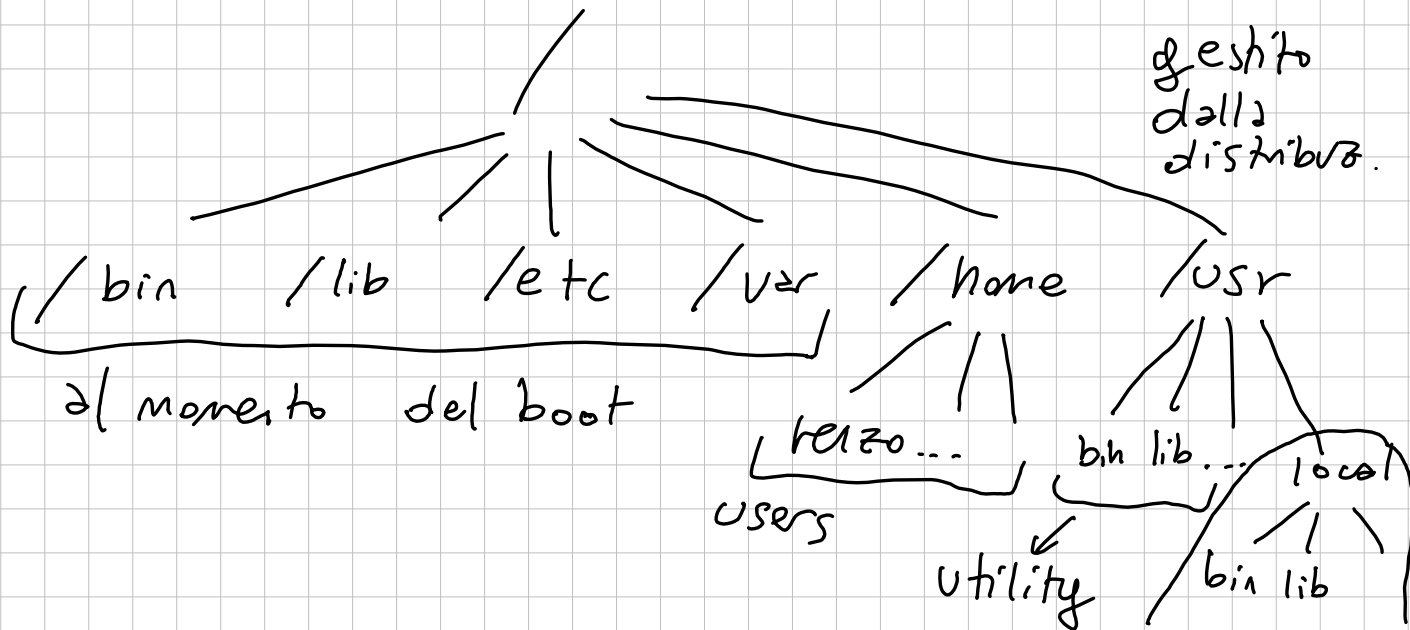
gestione struttura a grafo  
diretto aciclico gestire file,  
permessi di accesso

path

Foglie in comune

ALBERO

ls  
ls /usr



# STAT

man 2 stat

"fa vedere cosa c'è sul inode"

struct stat { ... }

Vnode quando aperto  
"aggiornato in tempo  
reale"

↑  
leggi i commenti su man 2 stat

vi st0.c

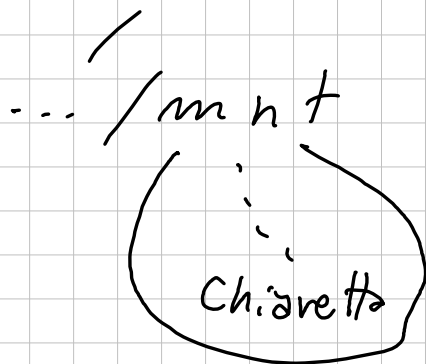
device driver [0, 5]

/dev è in [0, 5]

Id of file [Fe, 1] che nominano

MOUNT

..... /mnt  
chiavetta contenuto "appeso" in /mnt



tutto ciò  
che è  
visibile è in /

UMOUNT

DF

MKFS.EXT2 IMAGE

TRUNCATE -s 100K IMAGE



LINK — simbolici  
fisici

Finto file che  
rappresenta un pathname

ln -s /etc/hostname nickname

↑  
simbolico

l- prefisso

stat link simbolici  
sono trasparenti

ls -stat voglio vedere il  
Symbolic link!

(inode separato)

man readlink per leggere  
a cosa punta un symbolic link

prefisso F- file già aperto

man 2 stat

stat, Fstat, lstat



syscall  
senza  
prefissi



File  
già aperto  
usando  
File descriptor



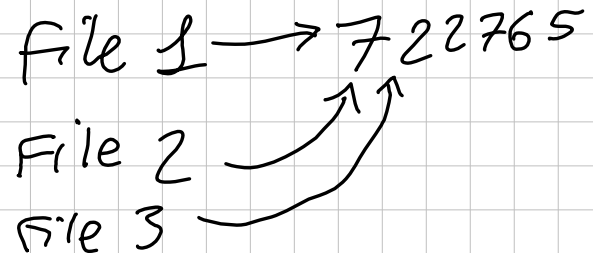
Symbolic  
link

In unix il nome del file non  
è un attributo del file

una directory è una lista di  
[nome, inode]

il link fisico in un'altra  
directory ha nome e lo  
stesso inode dell'altra...

Un file cessa di  
esistere quando non  
c'è nessun nome valido  
per riferirlo



e non c'è nessun processo che  
lo ha aperto

non è una copia sono più nomi  
per lo stesso file

ma 2 UNLINK  
usata da rm

ma 2 LINK  
(LN è comando)  
crea un link  
chiamato da LN

ma 2 SYMLINK

usata da

ln -s

il link simbolico  
punta a un path  
è una scorciatoia

il link fisico non lo possiamo  
farlo tra partizioni diverse.

perché il nome dell'inode  
è riferito a quella partizione.

Un'altra partizione usa quel  
numero per altre cose...

Un link simbolico si può fare  
anche su partizioni diverse  
perché è solo un "indicazione"

RENAME

senza

opio

per atomicità

Stesso effetto

o d:

ln	---
rm	---
<hr/>	
link	---
unlink	---

comando

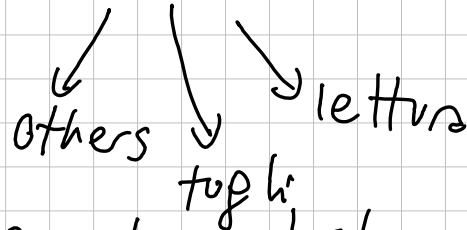
syscall

MKDIR comando

RMDIR cancella dir a patto che  
path + permessi d: <sup>sia vuoto.</sup> accesso

man 2 MKDIR

CHMOD o-r l+s I.o



oppure prendo i bit rwx rwx rwx  
e li penso come ottale.  
di solito si usa tipo ...

CHMOD 777 CNOFILE >  
||| ||| |||

per le directory

i permessi di esecuzione corrisponde

→ farci cd dentro.

UMASK

meccanismo che serve  
per non creare  
troppi permessi (0777)

UMASK

permessi di default

0022

UMASK 0

chmod dopo

"cosciente"

UMASK 077



stat

lezare - - - -

mkFifo

man poll

man select

2022 12 15  
SIGNAL ANS C signal handling

Kill manda un segnale

signal dichiara il gestore

concorrenza  
man 2 signal  
man 7 signal

SIGINT Ctrl + C da tastiera  
Term termina

SIGSEGV segmentation fault

Core la sua il core dumped

file immagine memoria al momento uccisione

SIGCHLD child stopped or terminated  
Ign ignored

I segnali non portano valore  
sincronizzazione

SIGKILL (9) non è ridefinibile  
15 per process murder

Posix standard posix che lo ha definito

grep -R SIGSEGV /usr/include

vi /usr/include/asm-generic/signal.h

man sig action

signal  
old

Sig action

meno importante

segnalato nuovamente  
più importante

sa (3, &new, NULL)

setto nuovo gestore

sa (3, NULL, &old)

situaz. attuale

sa (3, &new, &pre)

sa (3, &pre, NULL)

ripristino situazione  
precedente

processo perματοςo perματοςo.c

" se gli mandate un segnale  
lui Vi avverte. "

reentrant

→

sigproc mask

sigsuspend

hang up

demon:

Berkley socket — rete

socket bind connect python

f sono tutte system call

accept

spachi:  
"light"

s = socket (AF\_INET, SOCK\_STREAM, 0)

bind (i, \_\_\_\_\_)

listen (

For (i;)

conn = accept ( )

Fork o pthread

thread che gestisce la conn connessa

100 thread che rispondono contemporaneamente

socket  
connect

man

socket  
bind  
accept  
connect

NOTE:

man 3 shm\_open

crea un'area di memoria condivisa

File descriptor

↳ accedere contenuti da più processi

↳ punto a un'area di memoria  
mmap

man

sem\_init

man

eventfd

SIGFPE Floating point exception  
o divisione per zero.

man  $\neq$  notify

se qualcuno crea un file in ...

se succedono determinate cose  
nel file system

segnalo File info d. cosa è successo

---

p - poll  
select

segnali: POSIX

e - poll

specifico di: linux

necessario

file descriptor

---

Resteggiamento

saturnali  
speranza per il futuro

Isaac Newton  
Galileo