## process

### general:

process & files

bash as a process

linux kernel

functions & commands

(API)

system calls

wrapper

kernel

kernel activities

user/groups process/threads files/directories

### terminology:

pid

every process

unique at any point of time

ppid

parent

init

first process to run

(or systemd)

mostly pid = 1

### files:

modes:-

r : read

w : write

x : execute

training.xlsx

r everyone

w training team

x prem

rwx rw- r--

prem training everyone

user group others

111 110 100

7 6 4

110 100 110

6 4 6

umask

max: rw- rw- rw-

6 6 6

2 2 2

4 4 4

umask 222

7 7 7

2 2 2

5 5 5

r-x r-x r-x

7 5 5

type:

- regular files

d directory (folder)

p pipes (fifos)

b block driver

c character driver

s sockets

l links

inode number:

unique identifier

uniqueness within file system (different partitions)



file desc

(temporary number for every file opened in the memory = RAM)

lowest available number

process opens:

stdin 0

stdout 1

stderr 2

### system calls:

getpid()

getppid()

open()

close()

write() (printf uses this)

read() (scanf, fread etc use this)

create()

pipe (popen uses this)

msgget()

## states:

R running

S sleep

wait

T suspended

stop

I (kernel thread, process, service)

D uninterruptible

Z zombie

W paging

X dead (won't be seen by ps)

+ foreground to display

l multi threaded

probably CLONE\_THREAD

s session leader

< high priority

L memory lock

jobs:

process in background of shell:

jobs [1] [2]

fg

man pages:

1 commands (executables)

2 system calls

3 functions

4 special files (/dev)

5

6 games, entertainment

7 signals, protocols, miscellaneous

8 system based commands (mount etc)

## signals:

kill : used to send signals

-9 (force)

signal:

handle in 3 ways:

SIG\_IGN ignore

SIG\_DFL default behaviour

handler custom handler

can not be handled:

SIGKILL

SIGSTOP

keyboard signals:

ctrl + C SIGINT

ctrl + Z SIGSTOP

ctrl + \ SIGQUIT

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| signal | terminate ? | keyboard? | handle? | core dump? |
| SIGTERM | yes | no | yes | no |
| SIGINT | yes | ctrl + C | yes | no |
| SIGKILL | yes | no | no | no |
| SIGQUIT | yes | ctrl + \ | yes | yes |
| SIGSTOP | no (stops) | ctrl + Z | no | no |
| SIGSEGV | yes | no | yes | yes |
|  |  |  |  |  |

user defined signals:

SIGUSR1

## process Memory:

1. code
2. data
   1. stack
   2. heap
   3. static
3. code (text)
4. static
5. stack
6. heap

## fork:

return value:

0 to child

child’s pid to parent

## IPCs:

pipes

fifos

### pipes:

unidirectional

pipe(pd)

pd[0] read end

pd[1] write end

data once read is deleted

lseek wont work

related process can use it for communication

## file systems

ext4

procfs

VFS virtual file system

/proc

tasks (assignments):

1. kill () to send a signal to another process
2. kill yourself
3. explore alarm (kill yourself)
   1. handle the signal
4. on response from a signal, terminate your process in 10 seconds
5. three children from 1 parent (no grand children)
6. check if parent waits for any child
7. use wait in parent & make it wait
   1. does wait() wait for one child or all children
   2. wait() waits for first child to get created or first child to exit
8. create
9. file copy using system calls
10. are signals & signal handlers shared between parent & child
11. are open files & file desc shared between parent & child
12. are the internal cursors also shared ?

parked for future discussion :

what is the difference between ext4 or VFS etc and Root FS?

hidden chars in text

select()