Lecture # 7 CS347 23.8.2023

Recap: signals & scheduling mechanism. (policy).

the system call interface.

- open, close, fork, read, write, exec, exit, wait, dup, pipe, waitpid, kill,

- system call is an OS mechanism to request for OS services / functionality.

(i)

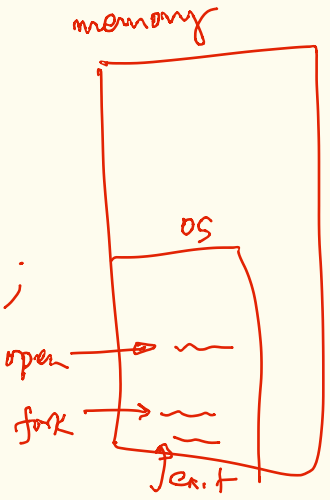
- system call interface is the list/ collection of all these calls.

(ii) ABI vs API

source / object files

main()
{
eat icecream();

binary
(executable)

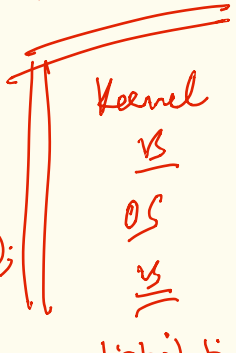


(iii) a few things need to be in place for system calls (the mechanism)

- how to invoke a system call?
- h/w mechanism to switch privilege levels?
- context save & restore
- mechanism to handle arguments/ret. value
- specify the system call #.
- where in binary is required functionality?

fd = open(filename, RDWR, 0666);

pid = fork();

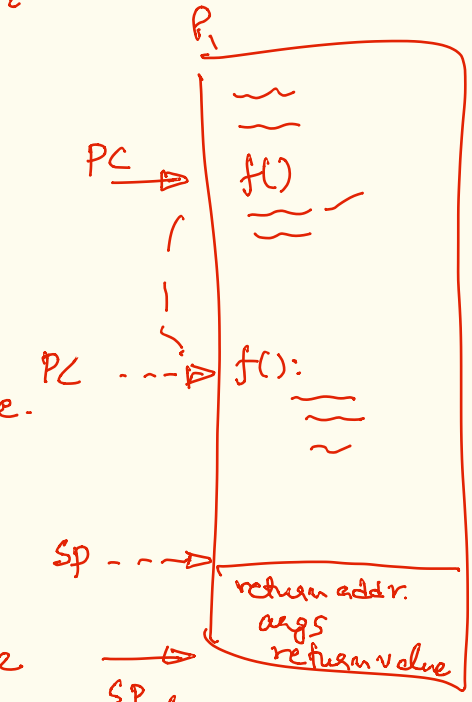


(iv) all system calls are interrupts!

interrupt
- hardware — external IO
- software — failures & exceptions (implicit)
  - (explicit) int 0x80 x86

* explicit s/w interrupt (eg. int 0x80)
  - s/w CPU to highest privilege level
  - save context of running process (all regs of CPU) in the kernel stack.
  - s/w to kernel stack
  - jmp to the interrupt handler

(x86) iret
- restores context of CPU regs.
- changes PL to user mode
- jmps to PC in user mode code.



(v) interrupt handler

ISR - interrupt service routine

- where is the interrupt descriptor table? (IDT)