Signals

Signals

- Can be termed as a way of Inter-process communication, but often it is not categorized as IPC (why!?)
- Processes can send each other “signals”, that is indicators of an event and receiver can “act” on the receipt of the signal
  - Integers are used to denote/signify each event

Let’s see signals in action using Linux command line

- **Use of “kill” command**
  - **Does not mean “kill” literally every time**
  - **“kill” means send signal!**

Signals

- Signals are used in UNIX systems to notify a process that a particular event has occurred.
- Signal handling
  - Synchronous and asynchronous
- A signal handler (a function) is used to process signals
  - Signal is generated by particular event (asynchronous like segfault or synchronous like “kill” system call)
  - Signal is delivered to a process
  - Then, signal is “handled” by the handler

Signals

- **More about signals**
  - **Different signals are typically identified as different numbers**
  - **Operating systems provide system calls like kill() and signal() to enable processes to deliver and handle signals**
  - **sigaction()/signal() - is used by a process to specify a “signal handler” – a code that should run on receiving a signal**
  - **kill() is used by a process to send another process a signal**
  - **There are restrictions on which process can send which signal to other processes**

Signals

- **Actions**
  - **Term Default action is to terminate the process.**
  - **Ign Default action is to ignore the signal.**
  - **Core Default action is to terminate the process and dump core (see core(5)).**
  - **Stop Default action is to stop the process.**
  - **Cont Default action is to continue the process if it is currently stopped.**

Demo

- Let’s see a demo of signals with respect to processes
- Let’s see signal.h
  - /usr/include/signal.h
  - /usr/include/asm-generic/signal.h
  - /usr/include/linux/signal.h
  - /usr/include/sys/signal.h
  - /usr/include/x86_64-linux-gnu/asm/signal.h
  - /usr/include/x86_64-linux-gnu/sys/signal.h
- man 7 signal
- Important signals: SIGKILL, SIGUSR1, SIGSEGV, SIGALRM, SIGCLD, SIGINT, SIGPIPE, ...

Signal handling by OS

Process 12323 {

signal(19, abcd);

}

OS: sys_signal {

Note down that process 12323 should handle signal number 19 with function abcd

}

Process P1 {

kill (12323, 19) ;

}

OS: sys_kill {

Note down in PCB of process 12323 that signal number 19 is pending for you.

}

When process 12323 is scheduled, at that time the OS will check for pending signals, and invoke the appropriate signal handler for a pending signal.

Sigaction: POSIX

```
#include <signal.h>

typedef void (*sighandler_t)(int);

sighandler_t signal(int signum, sighandler_t handler);
```

Threads and Signals

- **Signal handling Options:**
  - **Deliver the signal to the thread to which the signal applies**
  - **Deliver the signal to every thread in the process**
  - **Deliver the signal to certain threads in the process**
  - **Assign a specific thread to receive all signals for the process**

The “errno”
And error conventions