# Topic 1: Basic Concept

## Overview

### Guidance

* Chapter: Operating-System-Concepts.pdf 1, 2.2, 2.3; Unix-Network-Programming-Vol.2 Part 1
* Spend time: 2 days
* Learn suggestion: Take a general understanding of the operating system
* Key points: POSIX, System Call, Signal, Timer, File Descriptor
  + What's the OS? Why do we need OS?
  + The history of Unix, What's the relationship between Linux and Unix?
  + What's the fundamental function of OS?
  + Try to get help from the Linux help manual
  + Write some programs for each type of system call using the function in page 68.

### Practice

None

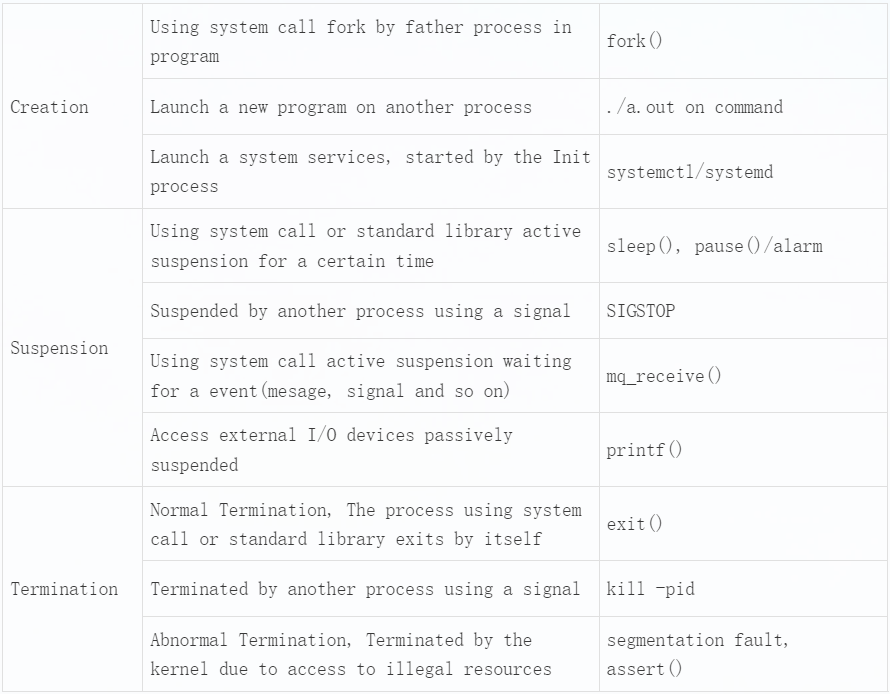
## Process

### Guidance

* Chapter: Operating-Systems-Internals-and-Design-Principles.pdf 3
* Spend time: 1 day
* Learn suggestion: To know about PCB, Process state, Process creation/termination
* Key points:
  + Using command ps/top to check the information of process, Try to explore what options can be used
  + How to capture the different process state?
  + The memory layout of Process address space

### Practice

1. How many approaches can a process be created, suspended, or terminated? Illustration



1. What's the meaning of the zombie and orphan process? Create a zombie and orphan process separately and check it by command ps

Orphan process: When a parent process exits while one or more child processes are still running, they become orphan processes. Orphan processes are adopted by the init process (process id 1), and the init process collects status on them.

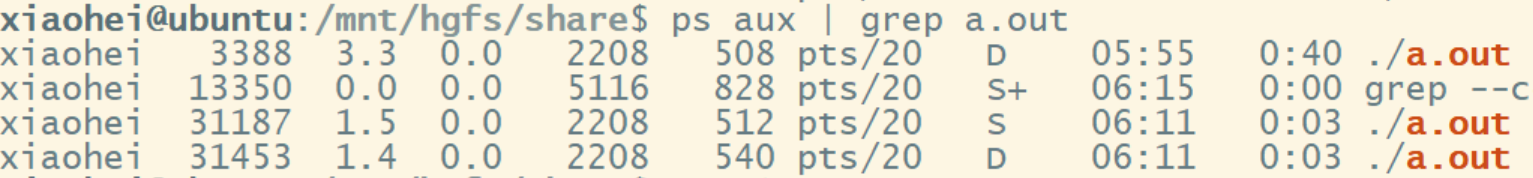
Zombie process: A process creates a child process using fork. If the child process exits and the parent process does not call wait or waitpid to get the state information of the child process, the process descriptor of the child process is still saved in the system. This process is called a dead process.



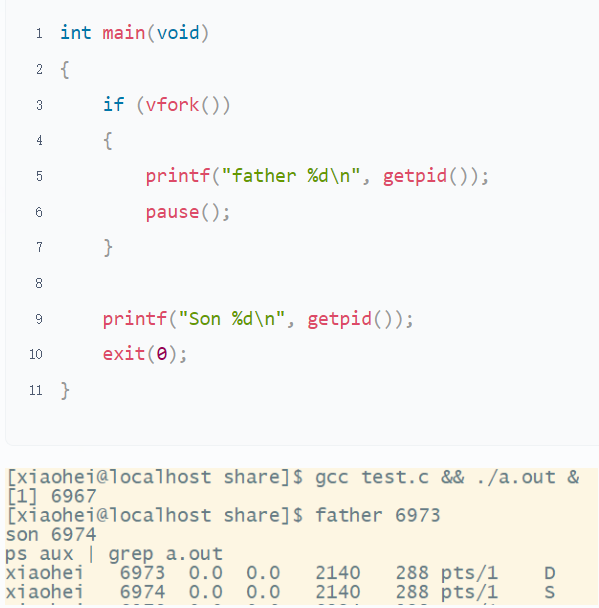


1. Try to get the state TASK\_UNINTERRUPTIBLE(D) of process in linux
   1. Prolongs the waiting time of a process in the task queue





* 1. vfork(Not uninterruptible)



* 1. Using kmod to lock kernel processes.(Leave it for you to explore)

## Thread

### Guidance

* Chapter: Operating-System-Concepts.pdf 4
* Spend time: 1 day
* Learn suggestion: Learn to compare with process and coroutine
* Key points:
  + Difference between process and thread
  + Difference between ULT and KLT
  + How many ways to implement the ULT?

### Practice

1. For multiple threads in the same process, which resources are shared and which are unique?
2. What are the advantages and disadvantages of multi-process(only one thread in each process) and multi-thread programming?
3. What are reentrant and non-reentrant functions? What are thread-safe and thread-unsafe functions? Illustration

## IPC

### Guidance

* Chapter: Unix-Network-Programming-Vol.2.pdf Part 2-5; Advanced-Programming-in-the-UNIX-Environment chapter 17
* Spend time: 2 days
* Learn suggestion: Compare and understand the characteristics of communication modes between different processes
* Key points:
  + How many ways to communicate between different processes?
  + The life cycle of the IPC object
  + What is different between Posix and System V IPC?
  + Can these ways be used for inter-thread communication?
  + What are the characteristics of each IPC?

### Practice

1. According to your understanding, try to divide all IPCs into different categories, and the reason
2. To summarise, what are the characteristics of each IPC?
3. Using fork/exec to simulate executive "ping 127.0.0.1 -c 3" and get the result
4. Write a program to implement the synchronisation and mutual exclusion by semaphore or lock
5. Follow the client/server model: client sends a filename to the server, then the server responds its content to the client

Requirements:

* Return the state of server to client when the specified file can't be opened correctly
* Follow the TLV format to organise data
* Don't mix the file operation and socket operation together. Separate them