SCC.211 Operating Systems

Session 3, week 7

Dr Andrew Scott

a.scott@lancaster.ac.uk

2. Processes

Diagrams from Silberschatz 10th Ed.

From a program to a process

- Compiler generates executable file
  - Contains
    - Code
    - Global variables
    - Metadata
      - Including what libraries are needed
- Loader takes file and 'loads' into memory
  - Also loads in any libraries required

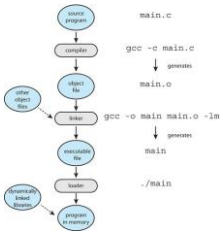


Figure 2.11 The role of the linker and loader.

Process Structure

- Stack — function parameters, local variables
- Heap — dynamically allocated memory
- Data — global variables and initialised data
- Text — executable code

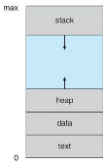
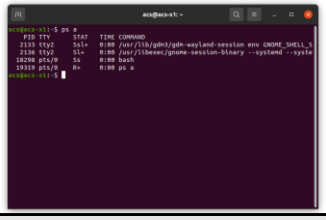


Figure 3.1 Layout of a process in memory

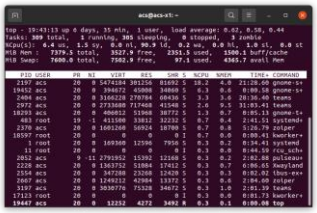
Unix Processes

- Listing my processes: ps



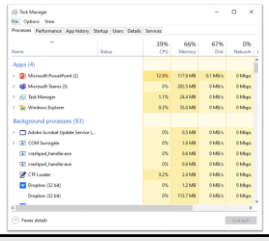
Unix Processes and Resource Usage

- Top command



Windows Processes and Resource Usage

- Task Manager



Processes form a process tree

- 'Family' tree of processes and children, each with unique Process ID
- Each will have/ inherit some permissions to access resources
  - Typically no greater than parent's permissions

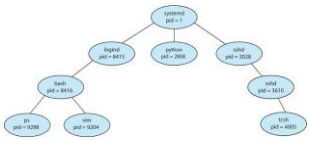
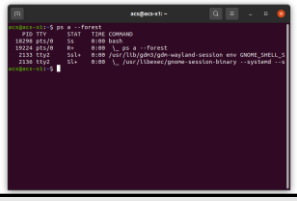


Figure 3.7 A tree of processes on a typical Linux system.

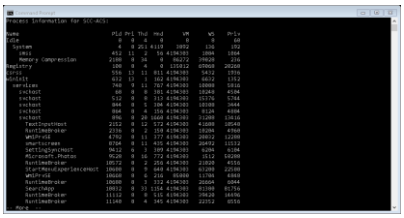
Unix Process Hierarchy example

- ps is a child of bash
- gnone-session-binary a child of gdm-wayland-session



Windows Process Hierarchy example

- pslist utility from Sysinternals.com



Process permissions

- Easiest to understand in Unix, where processes have
  - User ID
    - Permissions inherited from user invoking process
  - Effective User ID
    - Permissions process currently operating with

Commands to change effective ID

Unix

`su` *or* `sudo`

Windows

`runas`

```
$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 59640 Mar 22  2019 /usr/bin/passwd
$
```

s instead of x for user execute bit indicates process should run with root/ admin permissions even if started by regular user. Can also do same for group permissions, i.e. blue x would be an s

Create processes using a system call

- On Unix we use *fork( )* + *exec( )* to invoke a different program
  - Parent and child see different return value
    - Parent ≠ 0
    - Child = 0

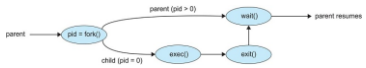


Figure 3.9 Process creation using the fork() system call.

Process defined by its context…

- Held as a Process Control Block (PCB)

  …or sometimes a Task Control Block (TCB)

- Everything OS needs to know about a process *(running program)*

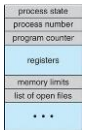


Figure 3.3 Process control block (PCB)

3. Memory Allocation

Includes diagrams from Silberschatz 10th Ed.

Variable sized regions

- Natural way of placing things in/ on a storage medium

  …e.g., a disk, or in RAM

- But look what happens over time…

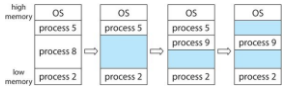


Figure 9.7 Variable partition.

Variable sized regions…

- Consider what happens over time…

  …lots of things in memory, all different sizes

Equal sized regions…

- Consider what happens if…

  we divide storage medium into equal sized regions or *blocks*…

Fixed vs. Variable Sized Regions

- Things to think about when watching next videos…

- Both approaches lead to wasted space
  - But which waste is easiest to manage?
  - What do we do if we have a choice of where to place something?
    - Does it matter?