SO

Els processos

* Procés
  + Què és
    - Programa en execució – instància d’un programa
  + Bloc de control de Processos
    - Llista dels processos que s’executen. Per a cada un d’ells guarda:
      * En quina part de la mem. Física estan
      * Quin és el fitxer executable associat
      * Quin és l’usuari que l’executa
      * El seus privilegis
  + No hi ha paral·lelisme: s’executa un únic procés en un instant de temps
  + En sistemes multiprocessador sí, però és segueix la mateixa idea
  + Interfície de programació
    - Gestió de processos i comunicació amb dispositius d’E/S:
      * Un procés pot crear un altre procés?
      * La funcionalitat està implementada a nivell de nucli i els processos hi poden accedir mitjançant crides a sistema.
    - Seguretat:
      * Antigament: CPU ho controlava tot
      * Actualment: processos creen i gestionen altres processos.
        + En comptes de crear un programa déu, es creen programes petits i especialitzats
  + Gestió de processos:
    - En crear un nou...:
      * Crear i inicialitzar el Bloc de Control de Processos
      * Crear i inicialitzar l’espai d’adreces de memòria
      * Carregar el programa a memòria
      * Avisar al nucli de que hi ha un nou procés
      * Preparar el nucli per executar el main
    - Hi ha processos que poden crear altres processos – Jerarquia de processos:
    - Procés pare – Procés Fill, què pot controlar?
      * Els privilegis
      * Temps d’execució
      * Memòria màxima
      * Prioritat sobre altres processos
      * On s’envia el que s’imprimeix
      * D’on rep
    - A LINUX
      * Fork(): copia completa del pare per crear el nou procés
        + El procés fill creat és independent del procés pare.
      * Exec: família de funcions que reemplacen la imatge del procés amb la especificada als arguments. No es crea cap procés nou
      * Llavors perquè dos funcions?
        + Amb fork hereta el context del pare
        + Després de fork i abans d’exec es pot establir un nou context
        + Fork no té arguments i exec en té 2
      * Wait: per esperar el resultat del fill. Si no es posa, s’executen en parel·lel
  + Entrada – Sortida
    - Antigament: interfície de programació específica per cada dispositiu. Calia actualitzar la interfície per un nou dispositiu
    - Ara: mateixa interfície per escriure i llegir a disc que per escriure i llegir dades de la xarxa
    - A UNIX, la interfície entrada – Sortida és:
      * Uniformitat: tots els dispositius utilitzen el mateix conjunt de crides
      * Obrir abans d’utilitzar: s’ha de open primer. Llavors el SO pot veure permisos, etc. En obrir un dispositiu, el SO retorna un descriptor de fitxer associat al procés que es pot fer servir per llegir/escriure al dispositiu
      * Orientat a bytes: s’accedeix al dispositiu mitjançant vectors de bytes
      * Lectures amb buffer al nucli: totes les dades que el nucli llegeix es guarden a un buffer seu. Aixó permet una única crida a sistema: read.
      * Escriptures amb buffer al nucli: totes les dades que s’han descriure es guarden a un buffer del nucli. Una única crida: write.
      * Tancament explícit: quan un procés ha acabat de fer servir un dispositiu es crida close, alliberant tots els recursos.
  + Comunicació interprocés
    - En crear-se un procés es creen 3 fitxers per defecte:
      * Descriptor 0: entrada estandar (teclat)
      * Descriptor 1: sortida estandar (terminal)
      * Descriptor 2: error (terminal)
    - Canonada
      * Crea un buffer al SO que gestiona la comunicació: totes les dades passen per allà
      * 2 descriptors de fitxers: de lectura i d’escriptura
      * Si en llegir no hi ha dades, espera
      * Si en escriure el buffer es ple, espera