CC3301: Programación de Software de Sistemas, Primavera 2010
Rodrigo Dueñas Fernández

Linux y C

Instalando Linux y los paquetes necesarios

Hay distintas opciones para realizar las tareas del curso y de cursos futuros. Algunas de estas son:

(a). Instalar con dual boot alguna distribución de Linux como Ubuntu (la más simple y recomendada para principantes), esta se puede encontrar en: http://www.ubuntu.com. Otras distribuciones posibles son: gentoo, debian, fedora, mandriva, etc. Dual Boot consiste en tener instalados ambos sistemas operativos a la vez en el computador, se elige con cual iniciar mediante un gestor de booteo. Para hacerlo se debe crear una partición especial para el sistema operativo *linux* que queremos instalar, de al menos 9 GB para que funcione holgadamente y con el sistema de archivos *ext3* o *ext4* (el que esté disponible). Además se necesita una partición adicional llamada *Partición de intercambio o SWAP* de al menos la mitad de la cantidad de *RAM* disponible en tu sistema (pero nunca más de 1GB). Esto era necesario comúnmente en versiones anteriores, pero al parecer ya no es necesario llevarlo a cabo manualmente, ya que las últimas versiones del instalador de Ubuntu se encargan de todo este proceso de manera mucho más automatizada.

(b). Instalar alguna distribución de Linux en una Máquina Virtual como VirtualBox (OpenSource, descargable en http://www.virtualbox.org) o VMWare (Mi alternativa preferida, pero no es gratuito). Una máquina virtual, a grandes rasgos, consiste en una aplicación que puede emular un sistema operativo como si fuese una aplicación. El proceso de instalación consiste en lo siguiente:

- Instalar la máquina virtual de su elección.
- Descargar una imagen de la distribución de linux de su elección (n archivo ISO o similar).
- Crear una nueva máquina virtual utilizando la imagen descargada, desde ese momento ya se tendrá una máquina virtual funcional. Sin embargo, por lo general, esta máquina virtual aún no posee todos los paquetes necesarios para poder llevar a cabo las tareas de manera satisfactoria. Más adelante se detalla qué paquetes deben instalar y cómo es que se instalan.

(c). Instalar un emulador de ambiente Linux como CYGWIN. Casi siempre tiene problemas y es difícil de configurar para que sea compatible con los objetivos del curso. Definitivamente no recomendable.

(d). Mac OS X (Sistema operativo de los Mac) también es útil pero tiene problemas con semáforos. Así que sólo sirve para el inicio del curso y cuando empecemos con sincronizació es imperante que trabajen bajo un ambiente UNIX.

Luego de que tenemos una distribución de Linux corriendo (de la manera que estimen conveniente) debemos instalar los paquetes necesarios para trabajar con C. Si se tiene instalado aptitude (por defecto en *Ubuntu*) puede instalarse ejecutando los siguientes comandos:

- `sudo apt-get install build-essential`
- `sudo apt-get install make`
- `sudo apt-get install gcc`

  Si sus tareas necesitan `wish` entonces deberán instalar los siguientes paquetes:

- `sudo apt-get install tk`

Compilando en c

Para compilar se utiliza el comando `gcc`. Algunos ejemplos:

- `gcc file.c`

  Compila el archivo file.c y genera un ejecutable de nombre a.out

- `gcc file.c -o executable`

  Compila el archivo file.c y genera un ejecutable de nombre executable

- `gcc file1.c ... fileN.c -o executable`

  Compila en conjunto todos los archivos y genera un ejecutable de nombre executable

  Si se quiere compilar con distintas flags (como ansi, pedantic o Wall) se debe realizar de la siguiente manera:

- `gcc file1.c ... fileN.c -ansi -pendatic -Wall -o executable`

  Si se quiere también agregar archivos binarios (*.o) necesarios para compilar el código basta con agregarlos a la lista

- `gcc file1.c file2.o file3.o ... fileN.c -ansi -pedantic -Wall -o executable`

  Lo que es especialmente necesario para cuando el profesor envíe archivos de prueba de la tarea donde él claramente no quiere dar el código que soluciona la tarea, pero si quiere que ustedes jueguen con el archivo *main* cambiandole ciertos parámetros.
  Para la primera tarea probablemente esté bien trabajar con estas lineas, pero para sus posteriores trabajos tendrán que usar **MakeFile**, así se ahorran el tener que escribir una y otra vez las órdenes y también se podrán automatizar la tediosa tarea de limpieza de archivos binarios o la de compilar desde 0.