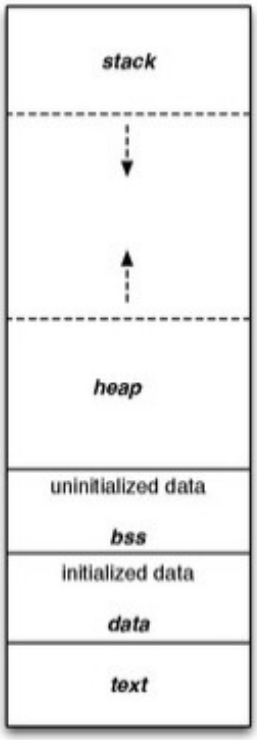
PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA
SECCIÓN INFORMÁTICA

SISTEMAS OPERATIVOS
ESPACIO DE DIRECCIONES VIRTUALES DE UN PROCESO

El espacio de direcciones virtuales es una abstracción del sistema operativo que le permite asignar espacio de memoria a un proceso para que pueda disponer de ella sin las complicaciones de administrar la memoria real (física). Desde el punto de vista de un proceso, éste tiene un espacio de memoria exclusivamente para él, por supuesto, este espacio es virtual y no existe. Solo las páginas que se está usando se mantienen en la memoria real.

Todo el espacio de direcciones se distribuye en segmentos, tal como se muestra en el gráfico:



Esquema típico de un programa en memoria
Tomado de https://en.wikipedia.org/wiki/Data_segment

Texto

Este segmento contiene el código objeto del programa y es el que ejecuta el procesador. Es de tamaño fijo y es de solo lectura.

Data

En este segmento se encuentran las variables globales y estáticas que son inicializadas. A su vez este segmento se divide en dos, una parte es de solo lectura, mientras que la otra es de lectura y escritura.

BSS

En este segmento se encuentran las variables globales y estáticas que han sido declaradas pero no han sido inicializadas. Acerca del origen del nombre usted puede encontrar información en https://en.wikipedia.org/wiki/.bss

Heap

A esta área también se le conoce como memoria dinámica y es la que administra las funciones *malloc*, *calloc*, *realloc* y *free*. Estas a su vez emplean las llamadas al sistema *sbrk* y *brk*. Esta área es compartida por todos los hilos, librerías compartidas y módulos cargados dinámicamente por un proceso. Esta área crece hacia las direcciones más altas.

Stack

En este segmento se encuentran los argumentos de la línea de comando, las variables del entorno, las variables automáticas (es decir las variables declaradas dentro de una función incluyendo los parámetros) y las direcciones de retorno de las funciones.

/proc

Según el manual, es un seudo sistema de archivos que contiene información de los procesos. En el directorio */proc* se encuentran directorios que corresponden a los *pids* de los procesos. Nosotros estaremos interesados en dos de ellos: */proc/[pid]/maps* y */proc/[pid]/mem*. El primero muestra las direcciones de inicio y fin de cada segmento (que en ese momento se encuentran en la memoria real), así como los permisos sobre esa región. El segundo contiene la memoria virtual y puede ser accedida mediante las llamadas al sistema: open(), read(), y write() (si tiene permiso de escritura). Hay que aclarar que solo pueden ser accedidas aquellas direcciones que se encuentran en la memoria física, de lo contrario obtendrá un error de entrada/salida.

Objetivo del laboratorio

El objetivo de este laboratorio es comprobar el contenido de cada segmento del programa en memoria, leyendo directamente de su memoria o modificándola en caso que los permisos lo permitieran. Para este fin crearemos un programa en C que contenga algunas declaraciones y después accederemos a la memoria para comprobar que se encuentran en el segmento indicado.

Experimento 1

Escribiremos dos programas en C, el primero (*exp1.c*) solicitará memoria dinámica y almacenaremos una cadena en dicho espacio. Se imprime por pantalla el pide del proceso, la dirección de la variable dinámica y el contenido al que apunta. Luego se hace una lectura del teclado, para detener la ejecución del programa.

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(void)
{
    char *cad;

    cad = (char *)malloc(20);
    strcpy(cad,"Sistemas Electricos");
    printf(" pid:%d \n dirección:%p \n contenido:%s\n",getpid(),cad,cad);
    getchar();
    printf("----------------------------\n");
    printf("pid:%d \n dirección:%p \n contenido:%s\n",getpid(),cad,cad);
    return 0;
}
```

Al ejecutarlo, tenemos la siguiente salida (en la terminal no presione tecla alguna)

```
alulab@minix:~/Documents$ ./exp1
 pid:198821
 dirección:0x55f0b654d2a0
 contenido:Sistemas Electricos
```

Abra otra terminal, buscaremos en */proc/198821/maps* y confirmaremos que se encuentra en el área del *heap*

```
alulab@minix:~/Documents$ cat /proc/198821/maps
55f0b6002000-55f0b6003000 r--p 00000000 fc:04 7340482                    /home/alulab/Documents/exp1
55f0b6003000-55f0b6004000 r-xp 00001000 fc:04 7340482                    /home/alulab/Documents/exp1
55f0b6004000-55f0b6005000 r--p 00002000 fc:04 7340482                    /home/alulab/Documents/exp1
55f0b6005000-55f0b6006000 r--p 00002000 fc:04 7340482                    /home/alulab/Documents/exp1
55f0b6006000-55f0b6007000 rw-p 00003000 fc:04 7340482                    /home/alulab/Documents/exp1
55f0b654d000-55f0b656e000 rw-p 00000000 00:00 0                          [heap]
7f3f40c34000-7f3f40c59000 r--p 00000000 fc:01 4195653                    /lib/x86_64-linux-gnu/libc-2.31.so
7f3f40c59000-7f3f40dd1000 r-xp 00025000 fc:01 4195653                    /lib/x86_64-linux-gnu/libc-2.31.so
7f3f40dd1000-7f3f40e1b000 r--p 0019d000 fc:01 4195653                    /lib/x86_64-linux-gnu/libc-2.31.so
7f3f40e1b000-7f3f40e1c000 ---p 001e7000 fc:01 4195653                    /lib/x86_64-linux-gnu/libc-2.31.so
7f3f40e1c000-7f3f40e1f000 r--p 001e7000 fc:01 4195653                    /lib/x86_64-linux-gnu/libc-2.31.so
7f3f40e1f000-7f3f40e22000 rw-p 001ea000 fc:01 4195653                    /lib/x86_64-linux-gnu/libc-2.31.so
7f3f40e22000-7f3f40e28000 rw-p 00000000 00:00 0
7f3f40e3c000-7f3f40e3d000 r--p 00000000 fc:01 4195649                    /lib/x86_64-linux-gnu/ld-2.31.so
7f3f40e3d000-7f3f40e60000 r-xp 00001000 fc:01 4195649                    /lib/x86_64-linux-gnu/ld-2.31.so
7f3f40e60000-7f3f40e68000 r--p 00024000 fc:01 4195649                    /lib/x86_64-linux-gnu/ld-2.31.so
7f3f40e69000-7f3f40e6a000 r--p 0002c000 fc:01 4195649                    /lib/x86_64-linux-gnu/ld-2.31.so
7f3f40e6a000-7f3f40e6b000 rw-p 0002d000 fc:01 4195649                    /lib/x86_64-linux-gnu/ld-2.31.so
7f3f40e6b000-7f3f40e6c000 rw-p 00000000 00:00 0
7ffe5fcaf000-7ffe5fcd0000 rw-p 00000000 00:00 0                          [stack]
7ffe5fd6b000-7ffe5fd6e000 r--p 00000000 00:00 0                          [vvar]
7ffe5fd6e000-7ffe5fd6f000 r-xp 00000000 00:00 0                          [vdso]
ffffffffff600000-ffffffffff601000 --xp 00000000 00:00 0                  [vsyscall]
```

Como podemos comprobar el contenido se encuentra en el *heap.*

Ahora escribiremos un programa que accederá a este segmento y cambiara el contenido. No cierre la primera terminal donde el primer programa (*exp1*) se ejecutó, tampoco presione tecla alguna si el foco está en él.

Escriba el siguiente programa (*modifica.c*):

```c
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main(int narg, char *argv[])
{  int fd;
   long offset;
   char path[20];
   char newcad[]="Sistemas Operativos" ;

   sprintf(path,"/proc/%s/mem",argv[1]);
   offset = strtol(argv[2],NULL,16);

   if((fd=open(path,O_RDWR)) < 0)
       perror("Error al abrir /proc/[pid]/mem");
   lseek(fd,offset,SEEK_SET);
   write(fd,newcad,sizeof(newcad));
   close(fd);
   return 0;
}
```

Compile el programa. Ejecútelo como administrador (sudo). Proporcione el *pid* y la dirección de la variable dinámica

```
osboxes@minix:/home/alulab/Documents$ sudo ./modifica 198821 0x55f0b654d2a0
osboxes@minix:/home/alulab/Documents$
```

Ahora regrese a la primera terminal y presione <Enter>. El contenido de la variable dinámica ha cambiado.

```
alulab@minix:~/Documents$ ./exp1
 pid:198821
 dirección:0x55f0b654d2a0
 contenido:Sistemas Electricos

----------------------------
pid:198821
 dirección:0x55f0b654d2a0
 contenido:Sistemas Operativos
alulab@minix:~/Documents$
```

TAREA

1) Escriba un programa que acepte dos argumento por la línea de ordenes. El primero corresponde al *pid* del proceso y el segundo corresponde a una dirección. El programa debe de verificar en */proc/[pid]/maps* que esa dirección corresponde a un segmento de memoria. En caso que sea así debe de imprimir la línea (de *maps*) que contiene información acerca del segmento de memoria, en caso contrario debe emitir un mensaje de error.

2) Escriba un programa que contenga dos variables globales, una inicializada y otra sin inicializar. Imprima las direcciones de ambas variables. Con el programa de la pregunta 1) verifique que esas variables se encuentran en el área de data y bss.

3) Escriba un programa que imprima el contenido de la pila del programa *exp1* presentado como ejemplo.

4) Experimente con diferentes estructuras, para determinar dónde se almacenan sus direcciones y sus contenidos.