Tarea 3: uso de **exec**

Martínez Ostoa Néstor Iván

Arquitectura Cliente Servidor - 2946
Facultad de Ingeniería
UNAM

10 de Marzo del 2021

Ejercicios

1. Realizar el programa `testenv.c` y comentar dentro del mismo programa qué hacen las funciones `putenv()` y `getenv()`. Compilar, ejecutar y explicar los resultados.

```c
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>

int main(void) {
  char envval[] = {"MYPATH=/user/local/someapp/bin"};

  /*
    int putenv(const char *varname);

    putenv se encarga de declarar el valor de una variable de entorno mediante
    la modificacion de alguna existente o la creacion de una nueva. *varname es
    un apuntador a un string en donde el valor de *varname es el valor de la nueva
    variable de entorno.

    En este caso *varname es char envval[]
  */
  if (putenv(envval))
    puts("putenv failed");
  else
    puts("putenv suceeded");

  /*
    char *getenv(const char *varname);

    Regresa un apuntador a un string que contiene el valor especificado en
    *varname en el ambiente actual. Si esta funcion no logra encontrar la variable
    de entorno regresa NULL.
  */
  if (getenv("MYPATH"))
    printf("MYPATH=%s\n", getenv("MYPATH"));
  else
    puts("MYPATH unassigned");

  if (getenv("YOURPATH"))
    printf("YOURPATH=%s\n", getenv("YOURPATH"));
  else
    puts("YOURPATH unassigned");
  return 0;
}
```

Compilación y ejecución

```
gcc testenv.c -o testenv.o
./testenv.o
```

Resultado de ejecución

```
> ./testenv.o
putenv suceeded
MYPATH=/user/local/someapp/bin
YOURPATH unassigned
```

Lo que podemos analizar de esta ejecución es que primero con la función `putenv()` definimos una variable de ambiente llamada `MYPATH`. Posteriormente, con `getenv()` obtenemos el valor de `MYPATH` y finalmente verificamos de nuevo con `getenv()` si se encuentra definida la variable de entorno `YOUPATH` y como esto es falso, el programa imprime `YOURPATH unassigned`

2. Realizar el par de programas `myecho.c` y `execve.c`

   a) Comprueba que puedas compilar y ejecutar `myecho` de manera independiente en la terminal pasándole los argumentos que gustes

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[]) {
  int j;
  for (j = 0; j < argc; j++) {
    printf("argv[%d] %s\n", j, argv[j]);
  }
  return 0;
}
```

Compilación y ejecución

```
> ./myecho.o Ni hao ! Wo jiao Néstor Martínez !
argv[0] ./myecho.o
argv[1] Ni
argv[2] hao
argv[3] !
argv[4] Wo
argv[5] jiao
argv[6] Néstor
argv[7] Martínez
argv[8] !

```

   b) Ejecución desde `execve` y explicar los resultados mostrados

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char *argv[]) {
  char *newargv[] = {NULL, "hello", "world", NULL};
  char *newenviron[] = { NULL };
  if (argc != 2) {
    fprintf(stderr, "Sintaxis: %s <file-to-exec> \n", argv[0]);
    exit(EXIT_FAILURE);
  }
  newargv[0] = argv[1];
  execve(argv[1], newargv, newenviron);
  perror("execve");
  exit(EXIT_FAILURE);
}
```

Compilación y ejecución

```
> cc execve.c -o execve.o
> ./execve.o ./myecho.o
argv[0] ./myecho.o
argv[1] hello
argv[2] world

```

Análisis de resultados

`execve` nos permite mandar a llamar a otros programas. En este caso, queremos mandar a llamar al ejecutable `myecho.o` desde la ejecución de `execve.o`. Dentro de `execve.c` lo que estamos haciendo es verificar que sus argumentos desde la terminal no sean 2 pues si lo son, imprimimos un mensaje

de error. En caso contrario, (caso en el que `execve.c` solo recibe otro argumento) lo que hacemos es mandar a llamar a la función `execve()` y le pasamos lo siguiente:

- Primer argumento: `argv[1]` que es `./myecho.o`. Este primer argumento representa el programa a ejecutar
- Segundo argumento: `newargv` que representa *"hello world"*
- Tercer argumento: `newenviron` que representa un `NULL`

y `execve` se encarga de ejecutar el programa que reciba como primer argumento y pasarle los argumentos recibidos como segundo y tercer parámetro.

Referencias

[1] IBM Knowledge Center *putenv() — Change/Add Environment Variables* revisado el 10 de marzo del 2021 en: `https://www.ibm.com/support/knowledgecenter/en/ssw_ibm_i_72/rtref/putenv.htm`

[2] IBM Knowledge Center *getenv() — Search for Environment Variables* revisado el 10 de marzo del 2021 en: `https://www.ibm.com/support/knowledgecenter/en/ssw_ibm_i_72/rtref/getenv.htm#getenv`