National University of Computer and Emerging Sciences

Operating System Lab – 07

Lab Manual

Contents

Objective....2

What are Threads?....2

Basic System Calls Related to Multithread Programming....2

'pthread_create()' System Call....3

'pthread_join()' System Call....3

Example 1: *Two Threads displaying two strings "Hello" and "How are you?" independent of each other*....4

Example 2: *Create a function message() that takes threadid as argument and prints the message with thread id. There should be atleast four independent threads*....5

Attributes in Threads....5

System Calls related to Attributes of Threads....6

'pthread_attr_init()' System Call....6

'pthread_attr_setdetachstate()' System Call....6

'pthread_attr_destroy()' System Call....6

Example 3: *Create a detached thread for a function infoThread()*....7

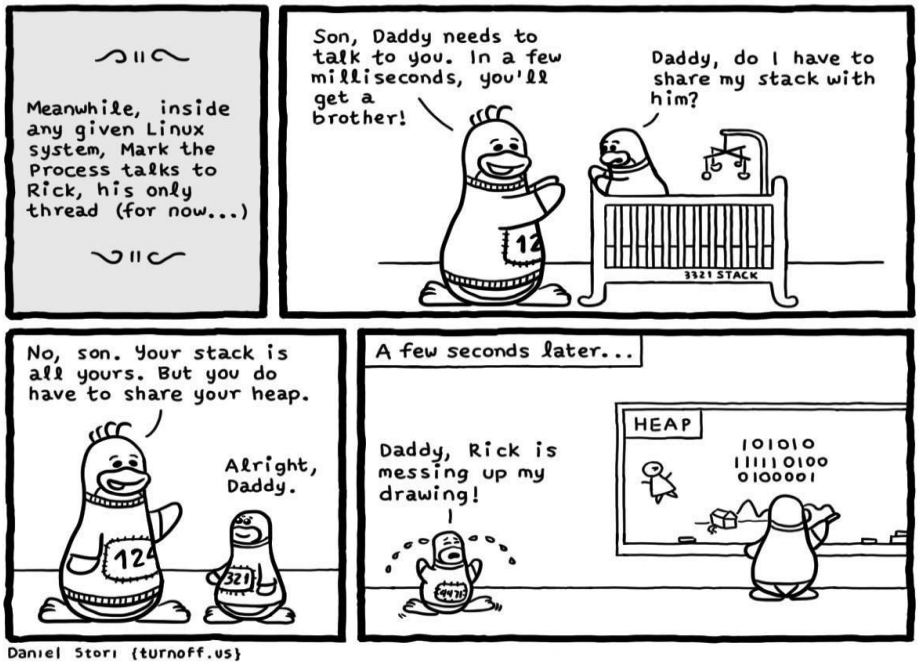
Objective

In this lab you are introduce to multithreaded programming using **pthread_create** system call. We will learn how create multithreads and how to join them .First we will understand system calls related to multithreaded programming then we will move towards multithreaded programming. Our main objective are

- Thread creation in Linux
- Joining of thread in Linux
- Initializing thread attributes
- Setting Attribute detach state
- Destroying attribute

What are Threads?

Threads are often described as **light-weight processes**. They can work like two or more processes sharing the same address space i.e. *they will work independently like processes but can share the same global variables*. They are mostly used when two tasks can be done independently without depending much on each other.



Basic System Calls Related to Multithread Programming

The following are two basic system calls related to multithreaded programming however, there are many system call available.

S.NO	System Call	Description
1	Pthread_create()	For creating threads
2	Pthread_join()	Wait of thread termination

'pthread_create()' System Call

This system call is used to create new thread, a syntax is given below

```
#include<pthread.h>
int pthread_create(

pthread_t *threaded,              //id of thread
const pthread_attr_t *attr,       //attributes of thread
void *(*start_routine) (void*),   //function that is to assign
void *arg                         //arguments that have to pass to thread function

);
```

Return Values:

If successful it return 0 otherwise it generates a nonzero number.

thread : is a pthread_t variable, pthread_t is a data type that holds information about threads. Each thread requires one pthread_t variable.

attr : is a variable of type pthread_attr_t, if specified it holds details about the thread, like scheduling policy, stack size etc. If we specify NULL the thread runs with default parameters.

start_routine : is the function the thread executes. The function needs to have a void* pointer as argument and must return a void* pointer (void* can be interpreted as a pointer to anything).

arg : is a void pointer which points to any data type.

'pthread_join()' System Call

This system call waits for the thread specified by thread to terminate. A syntax is shown below:

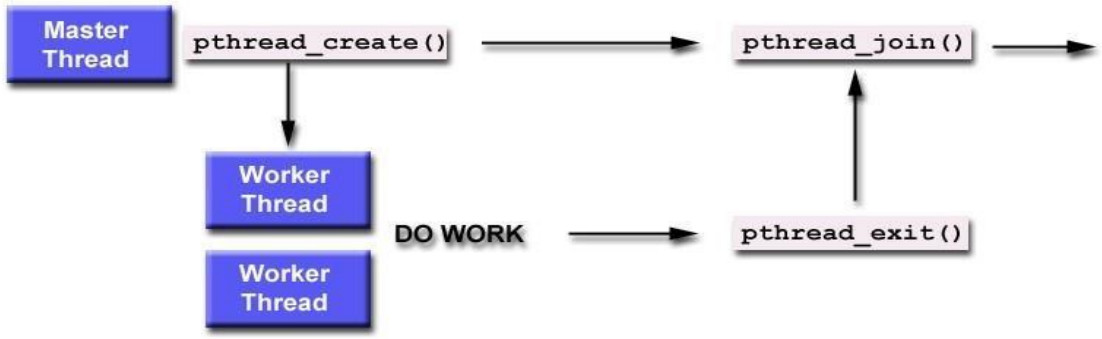
```
Int pthread_join (

Pthread_t threaded,     //id of thread which have to join
void **retval           //return status of thread

);
```

Return Values:

If successful it return 0 otherwise it generates a nonzero number.



Example 1: *Two Threads displaying two strings "Hello" and "How are you?" independent of each other*

- Create a new file thread.c with .c extension using any editor
- Type the following code.

```
#include <stdio.h>
#include <pthread.h>
#include <stdlib.h>
void * thread1()
{
while(1){
printf("Hello!!\n");
}
}
void * thread2()
{
while(1){
printf("How are you?\n");
}
}
int main()
{
int status;
pthread_t tid1,tid2;
pthread_create(&tid1,NULL,thread1,NULL);
pthread_create(&tid2,NULL,thread2,NULL);
pthread_join(tid1,NULL);
pthread_join(tid2,NULL);
return 0;}
```

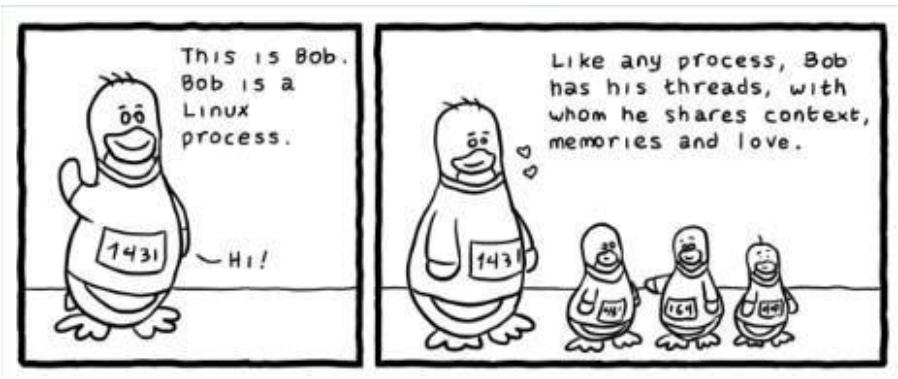
- Save and exit.
- To compile it type the following command on terminal.

```
gcc –o thread thread.c -lpthread
```

- Run it by using following command.

```
./thread
```

The –lpthread at the end to link the pthread library.



Example 2: *Create a function message() that takes threadid as argument and prints the message with thread id. There should be atleast four independent threads*

- Create a new file msgthreads.c with .c extension using any editor
- Type the following code.

```
#include <stdio.h>
#include <pthread.h>
#include <stdlib.h>
#define NUM_THREADS 4
#define MSG “Hello from message”
void *message(void *threadid) {
        printf(“msgthreads [INFO] Message: %s \t Thread ID: %ld \n”, MSG, (long) *threadid);
}
int main() {
        pthread_t threads[NUM_THREADS];
        int rc;
        long t;
for(t=0;t<NUM_THREADS;t++) {
        printf ("IN:main creadting thread %ld\n", t);
        rc = pthread_create(&threads[t],0, message,(void *)t);
}
pthread_join(threads[0],0);
pthread_join(threads[1],0);
pthread_join(threads[2],0);
pthread_join(threads[3],0);
return 0;
}
```

- Save and exit.
- To compile it type the following command on terminal.

```
gcc –o msgthreads msgthreads.c -lpthread
```

- Run it by using following command.

```
./msgthreads
```

Note: remove pthread_join system call and then observe the changes

Attributes in Threads

Previously we passed a NULL in place of thread attribute however, we may place thread attributes that uses default attributes of threads. However, we may create and customize a thread attribute object to specify other values for the attribute. Thread attributes are thread characteristics that affect the behavior of the thread.

System Calls related to Attributes of Threads

The following are the system calls related to threads' attribute.

S.NO	System Call	Description
1	pthread_attr_init()	Initializes a thread attributes object
2	pthread_attr_setdetachstate()	Controls detach state of a thread
3	pthread_attr_destroy()	Destroys attribute objects

'pthread_attr_init()' System Call

This initializes a thread attributes object attr with the default value. The syntax is shown below:

```
int pthread_attr_init(pthread attr t *attr)
```

Return Values:

If successful completion, it will return a 0 otherwise, an error number is returned to indicate the error.

'pthread_attr_setdetachstate()' System Call

The detachstate attribute controls whether the thread is created in a detached state.

```
int pthread_attr_setdetachstate(pthread_attr_t *attr, int detachstate)
```

PTHEAD_CREATE_DETACHED

Thread state is detached means it cannot be joined with other threads.

PTHREAD_CREATE_JOINABLE

Thread state is joinable means it can be joined with other threads

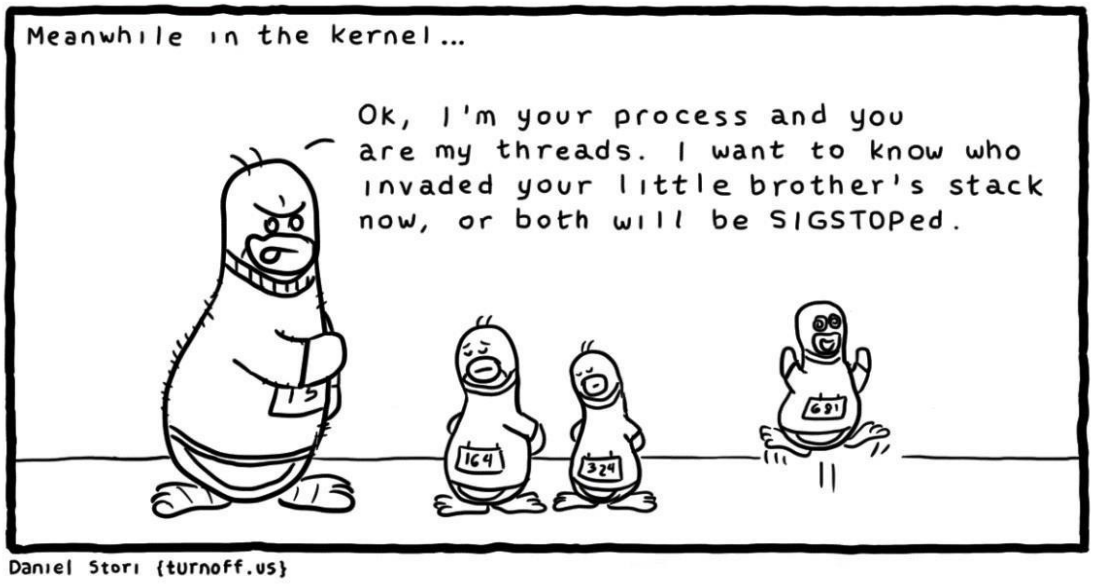
'pthread_attr_destroy()' System Call

When a thread attributes objects is no longer required, it should be destroyed using this system call.

```
int pthread_attr_destroy(pthread_attr_t *attr)
```

Return Values:

If successful completion, it will return a 0 otherwise, an error number is returned to indicate the error.



Example 3: *Create a detached thread for a function infoThread()*

- Create a new file detachthread.c with .c extension using any editor
- Type the following code.

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

void *theThread(void *parm) {
        printf("Entered the thread\n"); return NULL;
}

int main(int argc, char **argv) {
        pthread_attr_t  attr;
        pthread_t       thread;

        printf("Create a default thread attributes object\n");
        pthread_attr_init(&attr);
        printf("Set the detach state thread attribute\n");
        pthread_attr_setdetachstate(&attr,PTHREAD_CREATE_DETACHED);

        printf("Create a thread using the new attributes\n"); pthread_create(&thread, &attr, theThread, NULL);
        printf("Destroy thread attributes object\n"); pthread_attr_destroy(&attr);
int rc;
rc = pthread_join(thread, NULL);

        printf("Join now fails because the detach state attribute was changed\n pthread_join returns non zero
value %d",rc);

printf("Main completed\n");
return 0;
}
```

- Save and exit.
- To compile it type the following command on terminal.

```
gcc –o detachthread –pthread detachthread.c
```

- Run it by using following command.

```
./detachthread
```

You can get much information about these attributes and more information about system calls related to thread attributes: follow the links below

- https://docs.oracle.com/cd/E19455-01/806-5257/6je9h032j/index.html
- http://www.cs.cmu.edu/afs/cs/academic/class/15492-f07/www/pthreads.html
- https://vcansimplify.wordpress.com/2013/03/08/pthread-tutorial-simplified/

Lab Activity

1. Write a program which make 4 threads. Each thread will print one table out of [5678] up to 1000.
2. Procom has 4 volunteers on their front desk.
    - Volunteer 1 manages On day registration
    - Volunteer 2 handles announcements
    - Volunteer 3 handles sponsors
    - Volunteer 4 resolve queries of participants

    Implement this system using pthread for 100 participants.