

CET.jpg

COLLEGE OF ENGINEERING, TRIVANDRUM

EXPERIMENT 3

Network Programming Lab

Author:
Alan Anto

Registration Number :
TVE16CS09

February 10, 2019

Contents

1	Process and Thread	2
1.1	AIM	2
1.2	Theory	2
1.3	Algorithm	3
1.4	Program	3
1.5	Output	4
1.6	Result	4

1 Process and Thread

1.1 AIM

Familiarizing and understanding the use and process and thread.

1.2 Theory

An executing instance of a program is called a process. Each process provides the resources needed to execute a program. A thread is a subset of the process. A thread is a path of execution within a process. A process can contain multiple threads. A thread is also known as lightweight process. The idea is to achieve parallelism by dividing a process into multiple threads. A thread shares information like data segment, code segment, files etc. with its peer threads while it contains its own registers, stack, counter etc.

BASIS FOR COMPARISON	PROCESS	THREAD
Basic	Program in execution.	Lightweight process or part of it.
Memory sharing	Completely isolated and do not share memory.	Shares memory with each other.
Resource consumption	More	Less
Efficiency	Less efficient as compared to the process in the context of communication.	Enhances efficiency in the context of communication.
Time required for creation	More	Less
Context switching time	Takes more time.	Consumes less time.
Uncertain termination	Results in loss of process.	A thread can be reclaimed.

1.3 Algorithm

Algorithm 1 Algorithm for creating N threads

```

1 START
2 Let NUMTHREAD=10
3 Procedure *MyFunction(void * tid)
4 Begin
5     Print("Thread_:_" tid)
6 End MyFunction
7 create pthread thread
8 f=fork() //create fork
9 IF (f==0)
10     PRINT "child_:_"pid"
11     Begin For i<NUMTHREAD
12         call pthread_create(&thread,NULL,MyFunction,tid)
13         i++
14     END FOR
15 ELSE
16     PRINT "parent_:_"pid"
17     Begin For i<NUMTHREAD
18         call pthread_create(&thread,NULL,MyFunction,tid)
19         i++
20     END FOR
21 ENDIF
22 STOP

```

1.4 Program

The following program was written based on the algorithm.

```

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

void *threadfunc(void *var)
{
    printf("Thread from function %d \n ",var);
    sleep(1);
    printf("Exiting from function %d \n",var);
}

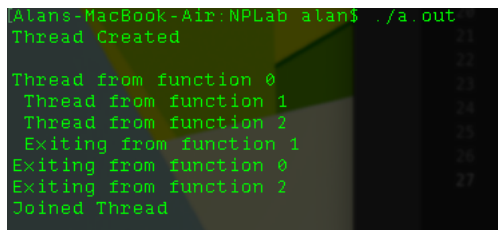
int main()
{

```

```
pthread_t id;  
printf("Thread Created \n \n");  
for(int i=0;i<3;i++)  
{  
pthread_create(&id, NULL, threadfunc, i);  
}  
pthread_join(id, NULL);  
printf("Joined Thread \n \n ");  
  
exit(0);  
  
}
```

1.5 Output

The following output was obtained on running the program



```
[Alans-MacBook-Air:NPLab alans$ ./a.out  
Thread Created 21  
Thread from function 0 22  
Thread from function 1 23  
Thread from function 2 24  
Exiting from function 1 25  
Exiting from function 0 26  
Exiting from function 2 27  
Joined Thread
```

1.6 Result

The familiarization of processes and thread was completed and the program executed successfully.