



MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY

Santosh, Tangail-1902

LAB REPORT

Lab Report No : 03
Lab Report name : How to view threads of a process on linux and thread program .
Course Title : Operating System Lab
Course Code : ICT-3110
Date of Performance : 20-08-2020
Date of Submission : 28-20-2020

Submitted by,

Student Name : Tanvir Ahmed

Student ID : IT-18043

Session : 2017-18

3rd Year 1st semester

Dept. of ICT

Submitted to,

Nazrul Islam

Assistant Professor

Dept. of ICT,

MBSTU.

Lab Report No – 03

Lab Name- How to view threads of a process on linux and thread program.

Objectives:

- i. What is Thread.
- ii. Types of Thread
- iii. Implementation of Thread

Theory : A thread is a **flow of execution** through the process code, with its own program counter that keeps track of which instruction to execute next, system registers which hold its current working variables, and a stack which contains the execution history.

A thread shares with its peer threads few information like code segment, data segment and open files. When one thread alters a code segment memory item, all other threads see that.

A thread is also called a **lightweight process**. Threads provide a way to improve application performance through parallelism. Threads represent a software approach to improving performance of operating system by reducing the overhead thread is equivalent to a classical process.

What are the differences between process and thread?

Threads are not independent of one other like processes as a result threads shares with other threads their code section, data section and OS resources like open files and signals. But, like process, a thread has its own program counter (PC), a register set, and a stack space.

Types of Thread:

Threads are implemented in following two ways –

- **User Level Threads** – User managed threads.
- **Kernel Level Threads** – Operating System managed threads acting on kernel, an operating system core.

Multithreading Models:

Some operating system provide a combined user level thread and Kernel level thread facility. Solaris is a good example of this combined approach. In a combined system, multiple threads within the same application can run in parallel on multiple processors and a blocking system call need not block the entire process.

Multithreading models are three types

- Many to many relationship.
- Many to one relationship.
- One to one relationship.

Corresponding Code:

```

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

#include<unistd.h> pthread_t
tid[2];

void* doSomething(void *arg)
{
    unsigned long i = 0;

    pthread_t id = pthread_self();

    if(pthread_equal(id,tid[0]))
    {
        printf("\n First thread processing\n");
    }
    else
    {
        printf("\n Second thread processing\n");
    }

    for(i=0; i<(0xFFFFFFFF);i++);

    return NULL;
}

int main(void)
{
    int i = 0;
    int err;

    while(i < 2)
    {
        err = pthread_create(&(tid[i]), NULL, &doSomething, NULL);
        if (err != 0)
            printf("\ncan't create thread :[%s]", strerror(err));
        else
            printf("\n Thread created successfully\n");

        i++;
    }

    sleep(5);
    return 0;
}

```

Output:

```
tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx: ~/CodePractice/C_programming
File Edit View Search Terminal Help
(base) tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx:~/CodePractice/C_programming$ gcc 05_multithreading.c -lpthread
(base) tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx:~/CodePractice/C_programming$ ./a.out

Thread created successfully

First thread processing

Thread created successfully

Second thread processing
(base) tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx:~/CodePractice/C_programming$
```

Thread in command line:

Here are several ways to show threads for a process on Linux.

1: PS

In ps command, "-T" option enables thread views. The following command list all threads created by a process with <pid>

```
tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx: ~
File Edit View Search Terminal Help
(base) tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx:~$ ps -T -p 1128
  PID   SPID TTY          TIME CMD
  1128   1128 tty7        00:19:02 Xorg
  1128   1287 tty7        00:00:00 Xorg:disk$0
  1128   1288 tty7        00:00:00 Xorg:disk$1
  1128   1289 tty7        00:00:00 Xorg:disk$2
  1128   1290 tty7        00:00:00 Xorg:disk$3
  1128   1328 tty7        00:00:00 Xorg:disk$0
  1128   1329 tty7        00:00:00 Xorg:disk$1
  1128   1330 tty7        00:00:00 Xorg:disk$2
  1128   1331 tty7        00:00:00 Xorg:disk$3
  1128   1359 tty7        00:01:15 InputThread
(base) tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx:~$
```

The "SID" column represents thread IDs, and "CMD" column shows thread names.

2: Top:

The top command can show a real-time view of individual threads. To enable thread views in the top output, invoke top with "-H" option. This will list all Linux threads.

```

tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx: ~/CodePractice/C_programming
File Edit View Search Terminal Help
top - 00:13:49 up 5:26, 1 user, load average: 0.32, 0.34, 0.59
Tasks: 283 total, 1 running, 282 sleeping, 0 stopped, 0 zombie
%Cpu(s): 3.8 us, 1.7 sy, 0.0 ni, 94.0 id, 0.3 wa, 0.0 hi, 0.2 si, 0.0 st
MiB Mem : 7847.3 total, 1294.0 free, 2506.6 used, 4046.7 buff/cache
MiB Swap: 3906.0 total, 3904.7 free, 1.3 used. 4412.3 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
 1832 tanvir    20   0 3825796 186200 93812 S   9.6   2.3   23:33.82 cinnamon
 1128 root      20   0 1246584 191476 137548 S   7.0   2.4   20:06.73 Xorg
 1886 tanvir    20   0 1039612 70468  46716 S   4.3   0.9    0:07.01 nemo-desktop
26552 tanvir    20   0 391528  40400  32412 S   4.3   0.5    0:00.13 gnome-screensho
 1286 mysql     20   0 2070104 356296 32912 S   1.3   4.4   3:30.38 mysqld
   11 root      20   0      0      0      0 I   0.3   0.0    0:28.88 rcu_sched
  904 root      20   0 410092  19728  16396 S   0.3   0.2    0:26.75 NetworkManager
 1581 tanvir    20   0   8376   5308   3608 S   0.3   0.1    0:06.14 dbus-daemon
 3213 tanvir    20   0 979964 272668 150960 S   0.3   3.4   24:38.97 chrome

```

To restrict the top output to a particular process <pid> and check all threads running inside the process: then we use `$ top -H -p <pid>`

3: Htop:

A more user-friendly way to view threads per process is via htop, an ncurses-based interactive process viewer. This program allows you to monitor individual threads in tree views.

```

tanvir@tanvir-HP-Pavilion-Laptop-15-cc1xx: ~
File Edit View Search Terminal Help

 1  [|||||] 18.1% 5 [||] 2.0%
 2  [|||] 4.5% 6 [||] 2.0%
 3  [|||] 6.5% 7 [||||] 5.9%
 4  [|||] 6.5% 8 [||] 1.3%
Mem[|||||||||||||||||||||||||||||||||]3.07G/7.66G Tasks: 145, 518 thr; 1 running
Swp[|] 1.26M/3.81G Load average: 0.84 0.48 0.57
Uptime: 05:32:42

  PID USER      PRI  NI   VIRT   RES   SHR  S  CPU%  MEM%   TIME+  Command
 1128 root      20   0 1228M  195M  142M S 12.5  2.5 20:29.46 /usr/lib/xorg/Xorg -core :0 -seat s
26829 tanvir    20   0 382M  39784 31824 S 9.9  0.5 0:00.30 /usr/bin/gnome-screenshot --gapplic
 1832 tanvir    20   0 3737M  181M 93988 S 7.9  2.3 23:52.67 cinnamon --replace
26781 tanvir    20   0 11452  4740  3388 R 2.6  0.1 0:00.79 htop
 1359 root      20   0 1228M  195M  142M S 2.0  2.5 1:16.20 /usr/lib/xorg/Xorg -core :0 -seat s
 1286 mysql     20   0 2021M  347M 32912 S 1.3  4.4 3:34.95 /usr/sbin/mysqld
 1304 mysql     20   0 2021M  347M 32912 S 1.3  4.4 1:57.85 /usr/sbin/mysqld
25370 tanvir    20   0 21.7G  373M  199M S 0.7  4.8 1:17.75 /usr/lib/libreoffice/program/soffic
20042 tanvir    20   0 4683M  233M  122M S 0.7  3.0 0:33.07 /opt/google/chrome/chrome --type=re
 1333 mysql     20   0 2021M  347M 32912 S 0.7  4.4 0:13.08 /usr/sbin/mysqld
23090 tanvir    20   0 4739M  282M  126M S 0.7  3.6 2:04.28 /opt/google/chrome/chrome --type=re
26761 tanvir    20   0 451M  39288 29968 S 0.0  0.5 0:01.14 /usr/libexec/gnome-terminal-server
 1306 mysql     20   0 2021M  347M 32912 S 0.0  4.4 0:10.86 /usr/sbin/mysqld
 1672 tanvir    20   0 159M  7080  6312 S 0.0  0.1 0:11.16 /usr/libexec/at-spi2-registryd --us

F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice F8Nice + F9Kill F10Quit

```

Conclusion : Thread is a very basic elements of process management system. Each program has a number of process in it and each process has a number of threads in it.

In this lab experiment I had to write a code in C++ that created thread and showed how they works. In the output we can see the thread id when they created. These threads also worked simultaneously with the main thread. By this way any program can be more faster with multithreading.

Here I have to include a special library file named **pthread** that is mandatory for process thread creation by user. We can also do this kind of program for multithreading in any other languages also but this is a simple demo.

For looking for process we use many linux commands lik ps, top , htop etc. All of these are very helpful and well organized for showing process detail in CLI. Processes are identified by there PID or process id number. By these described command we can find processes info by these pid numbers. Here also shows the process users and many other detail.

There are many other commands to manage processes but regular user doesn't handle with thread. But they internally use thread via many applications.