

第 004 讲 2 进程优先级与调度策略实战（课堂笔记）

一、基础知识

1、Linux 内核当中有 3 种调度策略：

SCHED_OTHER 分时调度策略；

SCHED_FIFO 实时调度策略，先到先服务；

SCHED_RR 实时调度策略，时间片轮转。

备注：如果有相同优先级的实时进程（根据优先级计算的调度权值是一样的）已经准备好，FIFO 时必须等待该进程主动放弃之后才可以运行这个优先级相同的任务。而 RR 可以每个任务都执行一段时间。

2、获取线程设置的最高和最低优先级函数如下：

`int sched_get_priority_max(int policy);` // 获取实时优先级的最大值

`int sched_get_priority_min(int policy);` // 获取实时优先级的最小值

SCHED_OTHER 它不支持优先级使用，而 SCHED_RR/SCHED_FIFO 支持优先级使用，它们分析为 1-99，数值越大优先级越高。

实时调度策略（SCHED_FIFO/SCHED_RR）优先级最大值为 99；

普通调度策略（SCHED_NORMAL/SCHED_BATCH/SCHED_IDLE），始终返回 0，即普通任务调度的函数。

3、设置和获取优先级 2 个主要核心函数：

`int pthread_attr_setschedparam(pthread_attr_t *attr, const struct sched_param *param);` // 创建线程优先级

`int pthread_attr_getschedparam(pthread_attr_t *attr, const struct sched_param *param);` // 获取线程优先级

`struct sched_param`

{

`int sched_priority;` // 所有设定的线程优先级

};

`param.sched_priority=11;` // 设置优先级

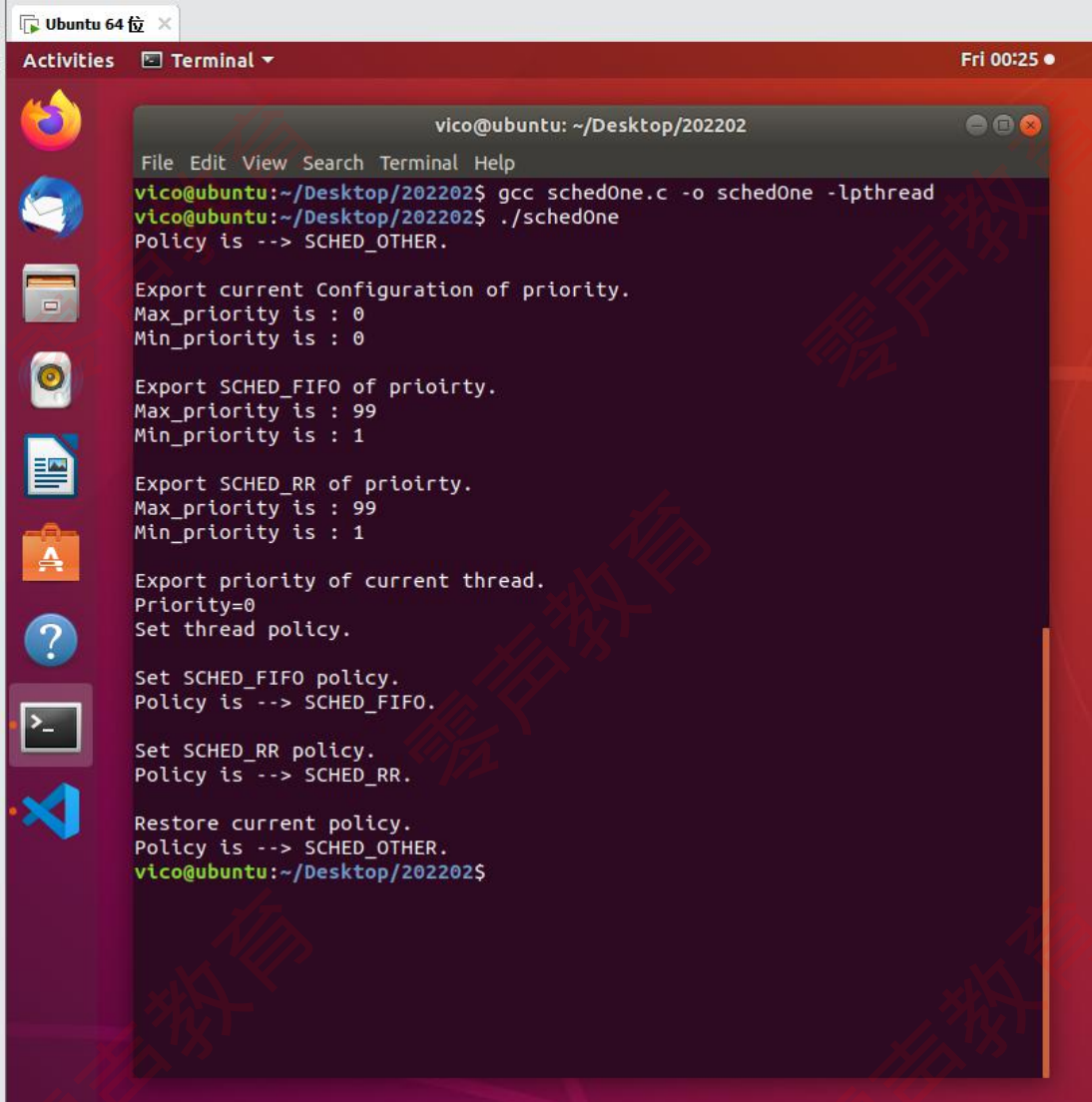
当操作系统创建线程时，默认线程是 SCHED_OTHER，我们也可以通过改变调度策略，使用如下函数：

`int pthread_attr_setschedpolicy(pthread_attr_t *attr, int policy);` // 设置线程调度策略

二、基础案例分析

1、操作系统所支持优先级测试程序分析，具体源码在文件夹工程包里面。

运行结果如下：



```
Ubuntu 64 位 x
Activities Terminal Fri 00:25
vico@ubuntu: ~/Desktop/202202
File Edit View Search Terminal Help
vico@ubuntu:~/Desktop/202202$ gcc schedOne.c -o schedOne -lpthread
vico@ubuntu:~/Desktop/202202$ ./schedOne
Policy is --> SCHED_OTHER.

Export current Configuration of priority.
Max_priority is : 0
Min_priority is : 0

Export SCHED_FIFO of priority.
Max_priority is : 99
Min_priority is : 1

Export SCHED_RR of priority.
Max_priority is : 99
Min_priority is : 1

Export priority of current thread.
Priority=0
Set thread policy.

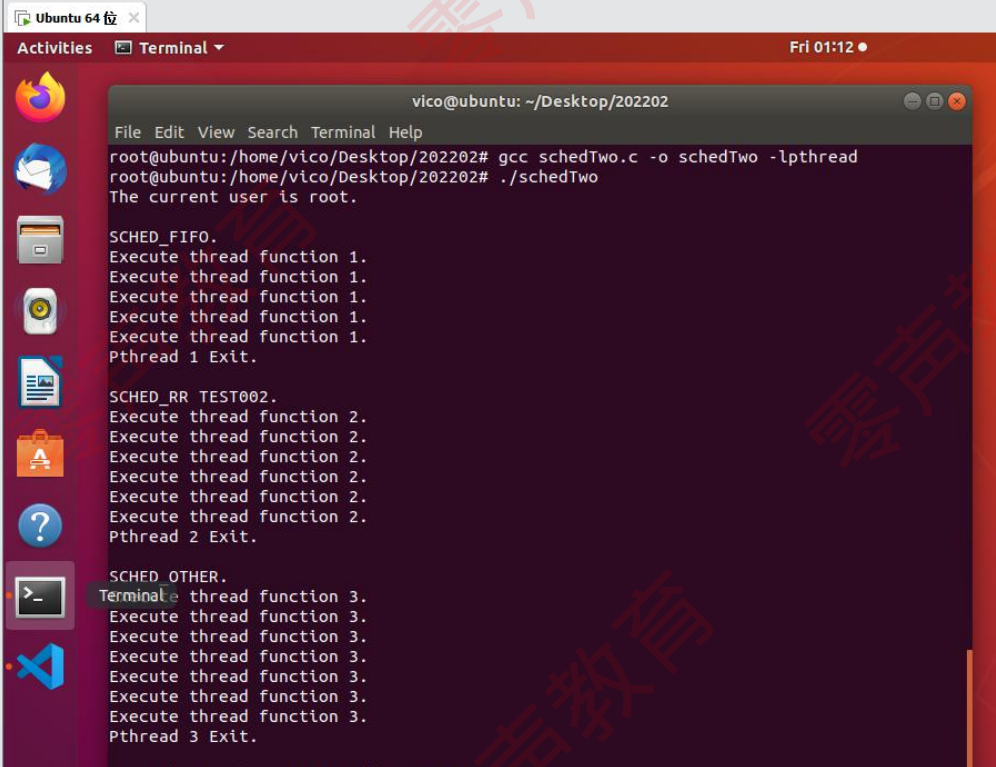
Set SCHED_FIFO policy.
Policy is --> SCHED_FIFO.

Set SCHED_RR policy.
Policy is --> SCHED_RR.

Restore current policy.
Policy is --> SCHED_OTHER.
vico@ubuntu:~/Desktop/202202$
```

2、简单线程调度策略，我们创建三个线程，默认创建的线程它的调度策略为 **SCHED_OTHER**，另外两个线程调度策略为 **SCHED_RR/FIFO**。具体源码在文件夹工程包里面

运行结果 1（管理员权限）：



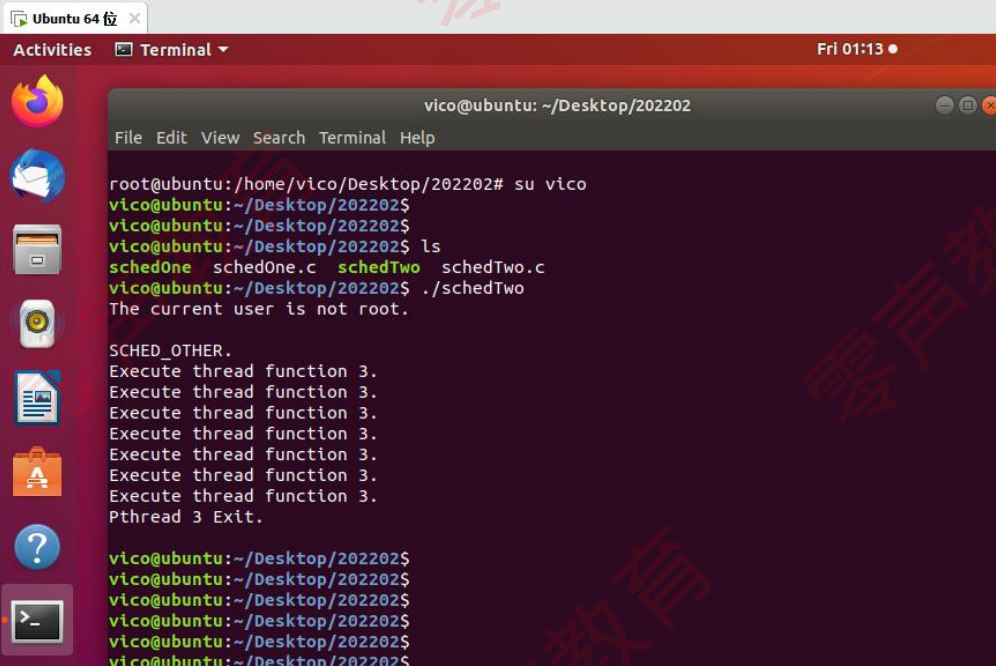
```
vico@ubuntu: ~/Desktop/202202
File Edit View Search Terminal Help
root@ubuntu:/home/vico/Desktop/202202# gcc schedTwo.c -o schedTwo -lpthread
root@ubuntu:/home/vico/Desktop/202202# ./schedTwo
The current user is root.

SCHED_FIFO.
Execute thread function 1.
Execute thread function 1.
Execute thread function 1.
Execute thread function 1.
Execute thread function 1.
Pthread 1 Exit.

SCHED_RR TEST002.
Execute thread function 2.
Execute thread function 2.
Execute thread function 2.
Execute thread function 2.
Execute thread function 2.
Pthread 2 Exit.

SCHED_OTHER.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Pthread 3 Exit.
```

运行结果 2（普通用户）：



```
vico@ubuntu: ~/Desktop/202202
File Edit View Search Terminal Help
root@ubuntu:/home/vico/Desktop/202202# su vico
vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$ ls
schedOne schedOne.c schedTwo schedTwo.c
vico@ubuntu:~/Desktop/202202$ ./schedTwo
The current user is not root.

SCHED_OTHER.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Execute thread function 3.
Pthread 3 Exit.

vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$
vico@ubuntu:~/Desktop/202202$
```