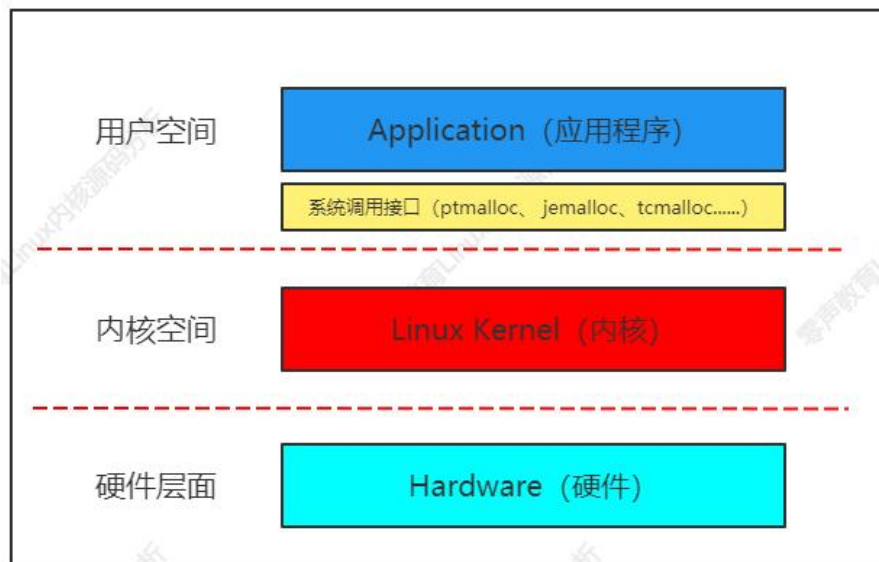




## 第0002讲 进程原理及系统调用



零声学院讲师: Vico老师



一、进程/进程生命周期

二、task\_struct数据结构分析

三、进程优先级/系统调用



# 一、进程/进程生命周期

## 1、进程

操作系统作为硬件的使用层，提供使用硬件资源的能力，进程作为操作系统使用层，提供使用操作系统抽象出的资源层的能力。

进程：是指计算机中已运行的程序。进程本身不是基本的运行单位，而是线程的容器。程序本身只是指令、数据及其组织形式的描述，进程才是程序（那些指令和数据）的真正运行实例。

Linux内核把进程叫做任务（task），进程的虚拟地址空间可分为用户虚拟地址空间和内核虚拟地址空间，所有进程共享内核虚拟地址空间，每个进程有独立的用户虚拟地址空间。



Linux通过：ps命令用于输出当前系统的进程状态。显示瞬间进程的状态，并不是动态连续；  
如果我们想对进程进行实时监控就top命令。

```
vico@localhost:~  
File Edit View Search Terminal Help  
[vico@localhost ~]$ ps aux  
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND  
root         1   0.6   0.0 175744 14052 ?        Ss   00:33   0:02 /usr/lib/systemd/systemd  
root         2   0.0   0.0      0      0 ?        S    00:33   0:00 [kthreadd]  
root         3   0.0   0.0      0      0 ?        I<   00:33   0:00 [rcu_gp]  
root         4   0.0   0.0      0      0 ?        I<   00:33   0:00 [rcu_par_gp]  
root         5   0.0   0.0      0      0 ?        I    00:33   0:00 [kworker/0:0-ata_sff]  
root         6   0.0   0.0      0      0 ?        I<   00:33   0:00 [kworker/0:0H-events_high]  
root         9   0.0   0.0      0      0 ?        I<   00:33   0:00 [mm_percpu_wq]  
root        10   0.0   0.0      0      0 ?        S    00:33   0:00 [ksoftirqd/0]  
root        11   0.0   0.0      0      0 ?        R    00:33   0:00 [rcu_sched]  
root        12   0.0   0.0      0      0 ?        S    00:33   0:00 [migration/0]  
root        13   0.0   0.0      0      0 ?        S    00:33   0:00 [watchdog/0]  
root        14   0.0   0.0      0      0 ?        S    00:33   0:00 [cpuhp/0]  
root        16   0.0   0.0      0      0 ?        S    00:33   0:00 [kdevtmpfs]  
root        17   0.0   0.0      0      0 ?        I<   00:33   0:00 [netns]  
root        18   0.0   0.0      0      0 ?        S    00:33   0:00 [rcu_tasks_trace]  
root        19   0.0   0.0      0      0 ?        S    00:33   0:00 [rcu_tasks_rude_]  
root        20   0.0   0.0      0      0 ?        S    00:33   0:00 [kauditd]  
root        21   0.0   0.0      0      0 ?        S    00:33   0:00 [khungtaskd]
```



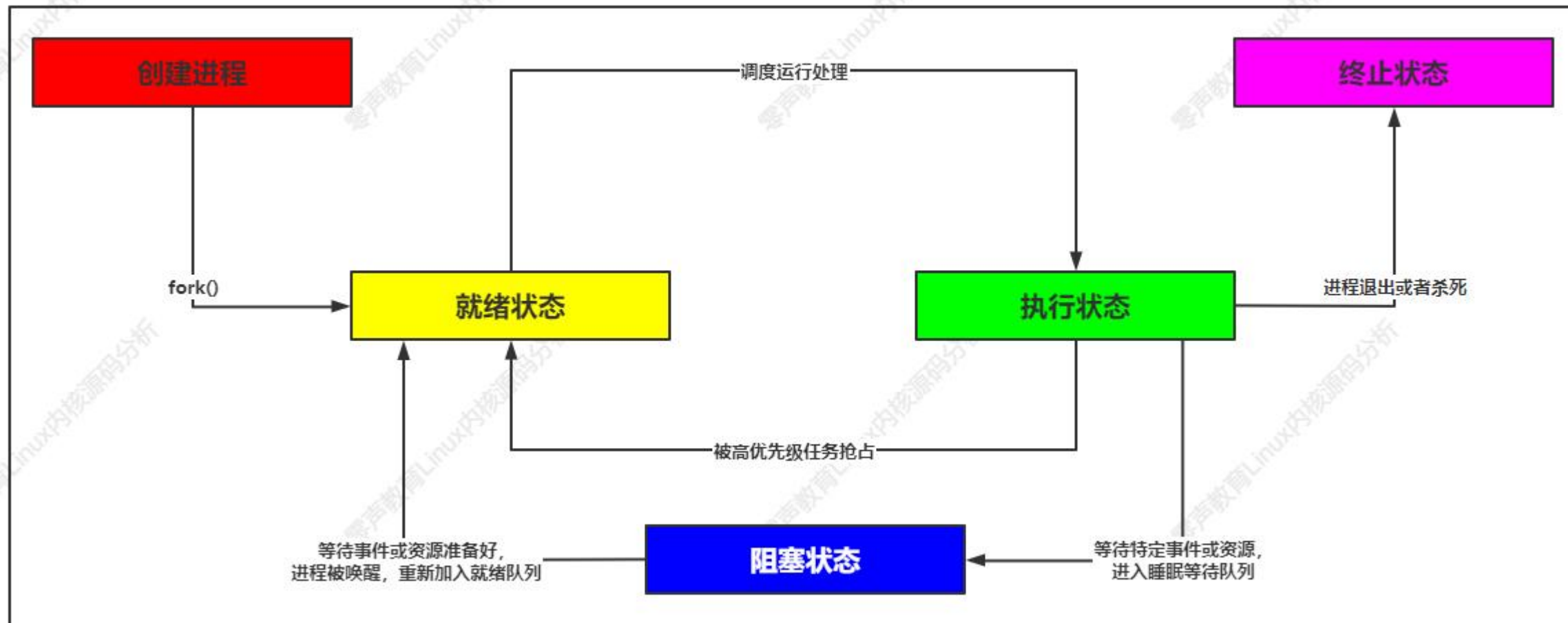
## 2、进程生命周期

Linux操作系统属于多任务操作系统，系统中的每个进程能够分时复用CPU时间片，通过有效的进程调度策略实现多任务并行执行。而进程在被CPU调度运行，等待CPU资源分配以及等待外部事件时会属于不同的状态。进程状态如下：

- 创建状态：创建新进程；
- 就绪状态：进程获取可以运作所有资源及准备相关条件；
- 执行状态：进程正在CPU中执行操作；
- 阻塞状态：进程因等待某些资源而被跳出CPU；
- 终止状态：进程消亡。



## Linux内核进程状态间关系如下:





## 二、task\_struct数据结构分析

进程是操作系统调度的一个实体，需要对进程所必须资源做一个抽象化，此抽象化为进程控制块（PCB，Process Control BLock），在Linux内核里面采用task\_struct结构体来描述进程控制块。Linux内核涉及进程和程序的所有算法都围绕名为task\_struct的数据结构而建立操作。具体Linux内核源码task\_struct结构体核心成员如下：

```
C sched.h
include > linux > C sched.h > task_struct

629
630 struct task_struct {
631 #ifdef CONFIG_THREAD_INFO_IN_TASK
632     /*
633      * For reasons of header soup (see current_thread_info()), this
634      * must be the first element of task_struct.
635      */
636     struct thread_info    thread_info;
637 #endif
```





## 三、进程优先级/系统调用

### 1、进程优先级

限期进程的优先级比实时进程要高，实时进程的优先级比普通进程要高。

- 限期进程的优先级是-1;
- 实时进程的优先级1-99，优先级数值越大，表示优先级越高;
- 普通进程的静态优先级为：100-139，优先级数值越小，表示优先级越高，可通过修改nice值改变普通进程的优先级，优先级等于120加上nice值。

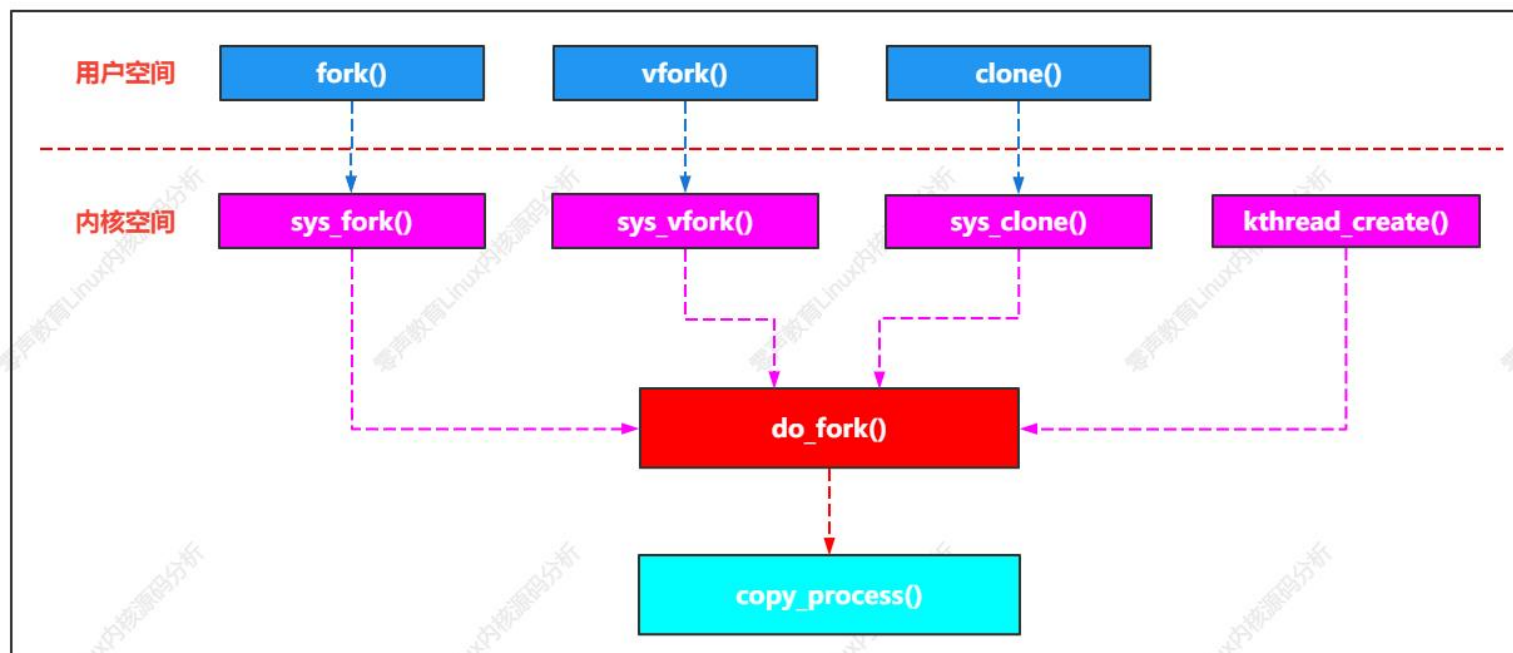
```
include > linux > sched.h > task_struct > prio
676     int          prio;
677     int          static_prio;
678     int          normal_prio;
679     unsigned int  rt_priority;
```





## 2、系统调用

当运行应用程序的时候，调用fork()/vfork()/clone()函数就是系统调用。系统调用就是应用程序如何进入内核空间执行任务，程序使用系统调用执行一系列的操作：比如创建进程、文件IO等等。具体如下图所示：





### 3、内核线程

内核线程是直接由内核本身启动的进程。内核线程实际上是将内核函数委托给独立的进程，与系统中其他进程“并行”执行（实际上，也并行于内核自身的执行）。内核线程经常称之为（内核）守护进程。它们用于执行下列任务。

- 周期性地将修改的内存页与页来源块设备同步（例如，使用mmap的文件映射）；
- 如果内存页很少使用，则写入交换区；
- 管理延时动作（deferred action）；
- 实现文件系统的事务日志。



## 4、退出进程

退出进程有两种方式：一种是调用exit()系统调用或从某个程序主函数返回；另一个方式为被接收到杀死信号或者异常时被终止。

```
kernel > C exit.c > ...  
866  
867 |  
868 SYSCALL_DEFINE1(exit, int, error_code)  
869 {  
870 | do_exit((error_code&0xff)<<8);  
871 }  
872
```



办学宗旨：一切只为渴望更优秀的你

办学愿景：让技术简单易懂