

Linux进程、线程和调度(2)

讲解时间：9月13日、9月15日、9月19日、9月22日晚20点
宋宝华 <21cnbao@gmail.com>

报名直播或者录播：

<http://edu.csdn.net/course/detail/5995>

扫描二维码报名



麦当劳喜欢您来，喜欢您再来



扫描关注
Linuxer



9.15 第二次课大纲

- 1.fork、vfork、clone
- 2.写时拷贝技术
- 3.Linux线程的实现本质
- 4.进程0和进程1
- 5.进程的睡眠和等待队列
- 6.孤儿进程的托孤，SUBREAPER

练习题

- 1.fork、vfork、Copy-on-Write例子
- 2.life-period例子，实验体会托孤
- 3.pthread_create例子，strace它
- 4.彻底看懂等待队列的案例

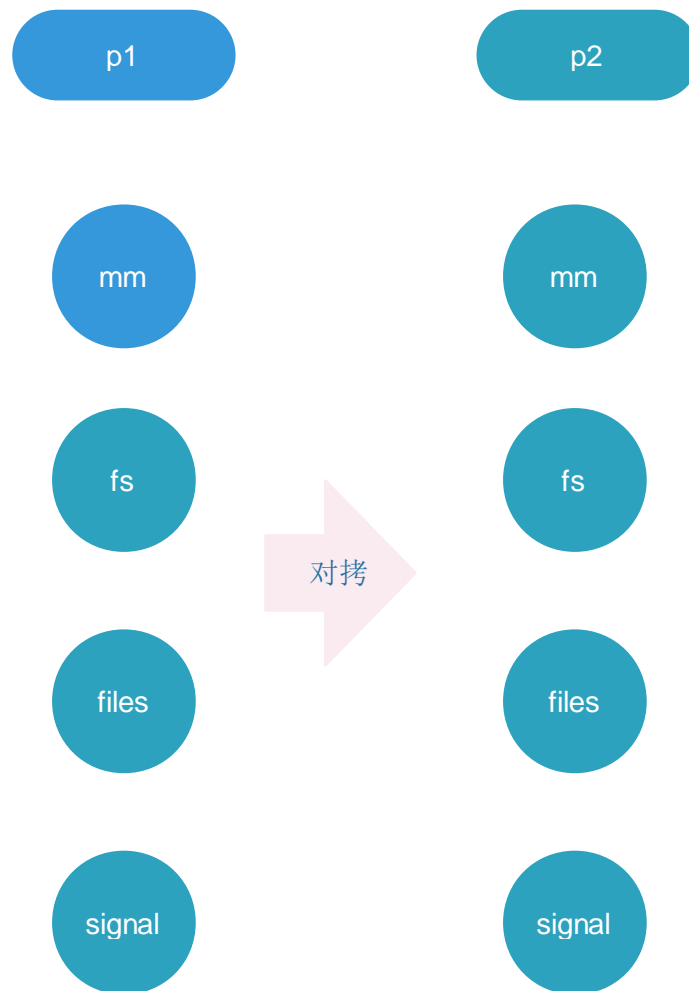
fork

- fork()

1. SIGCHLD

执行一个copy，但是任何修改都造成分裂，如：

chroot, open, 写memory, mmap, sigaction....



Copy-on-write

最开始

virt1	phy1	原则上R+W
-------	------	--------

fork后

virt1	phy1	RD-ONLY
-------	------	---------

virt1	phy1	RD-ONLY
-------	------	---------

write后

virt1	phy1	原则上R+W
-------	------	--------



virt1	phy2	原则上R+W
-------	------	--------

Mmu-less Linux

无copy-on-write,没有fork

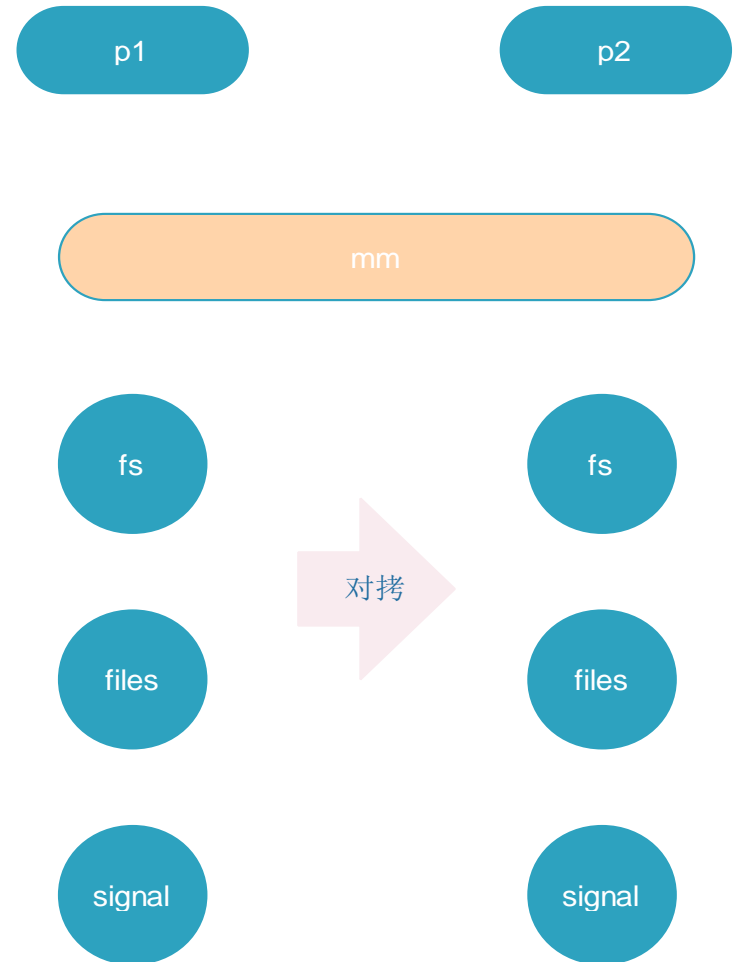
使用vfork:父进程阻塞直到子进程

1. exit

2. exec

vfork

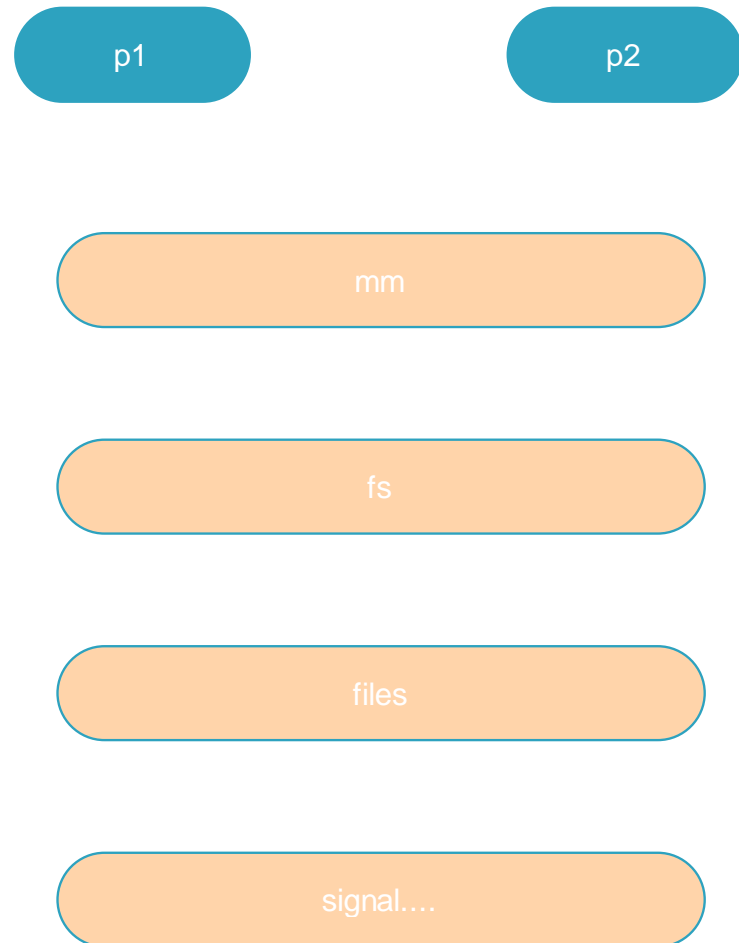
- vfork()
 1. CLONE_VM
 2. CLONE_VFORK
 3. SIGCHLD



pthread_create-> clone

- clone()
 1. CLONE_VM
 2. CLONE_FS
 3. CLONE_FILES
 4. CLONE_SIGHAND
 5. CLONE_THREAD

共享资源，可调度

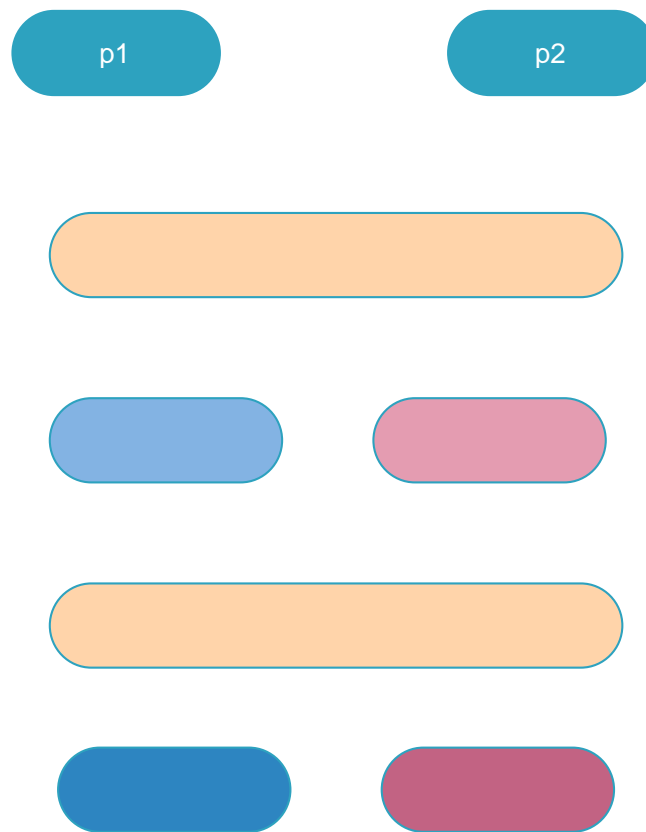


进程、线程与“人妖”

- clone

如果我们只clone一部分资源呢？

进程？
线程？
人妖？



妖有了仁慈的心,就不再是妖,是人妖

PID 和 TGID

`pthread_create`

P1

P2

P3

P4

top -H: 线程视角
内核里的PID

PID1

PID2

PID3

PID4

TGID

PID1

PID1

PID1

PID1

top: 进程视角

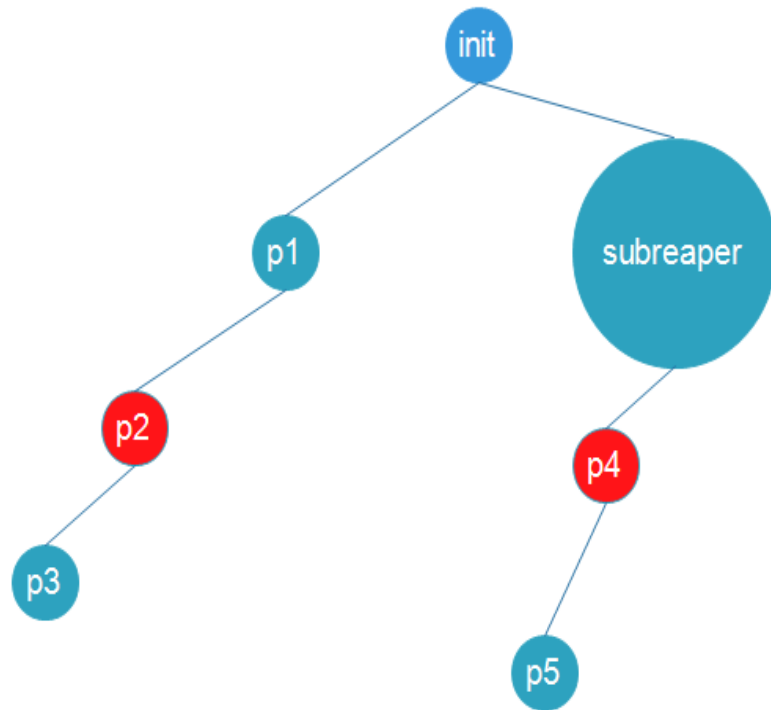
`getpid()`

SUBREAPER 与 托 孤

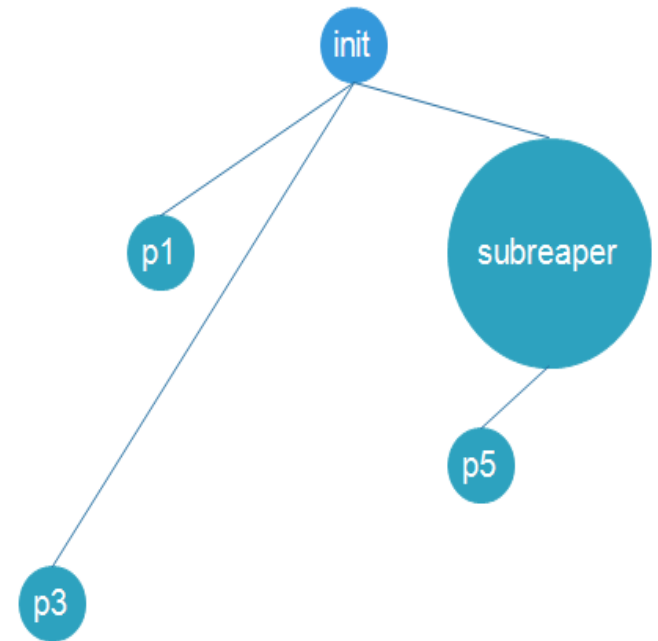
```
/* Become reaper of our children */
if (prctl(PR_SET_CHILD_SUBREAPER, 1) < 0) {
    log_warning("Failed to make us a subreaper: %m");
    if (errno == EINVAL)
        log_info("Perhaps the kernel version is too old (<
3.4?));
}
```

PR_SET_CHILD_SUBREAPER 是 Linux 3.4 加入的新特性。把它设置为非零值，当前进程就会变成 subreaper，会像 1 号进程那样收养孤儿进程了。

init vs. SUBREAPER

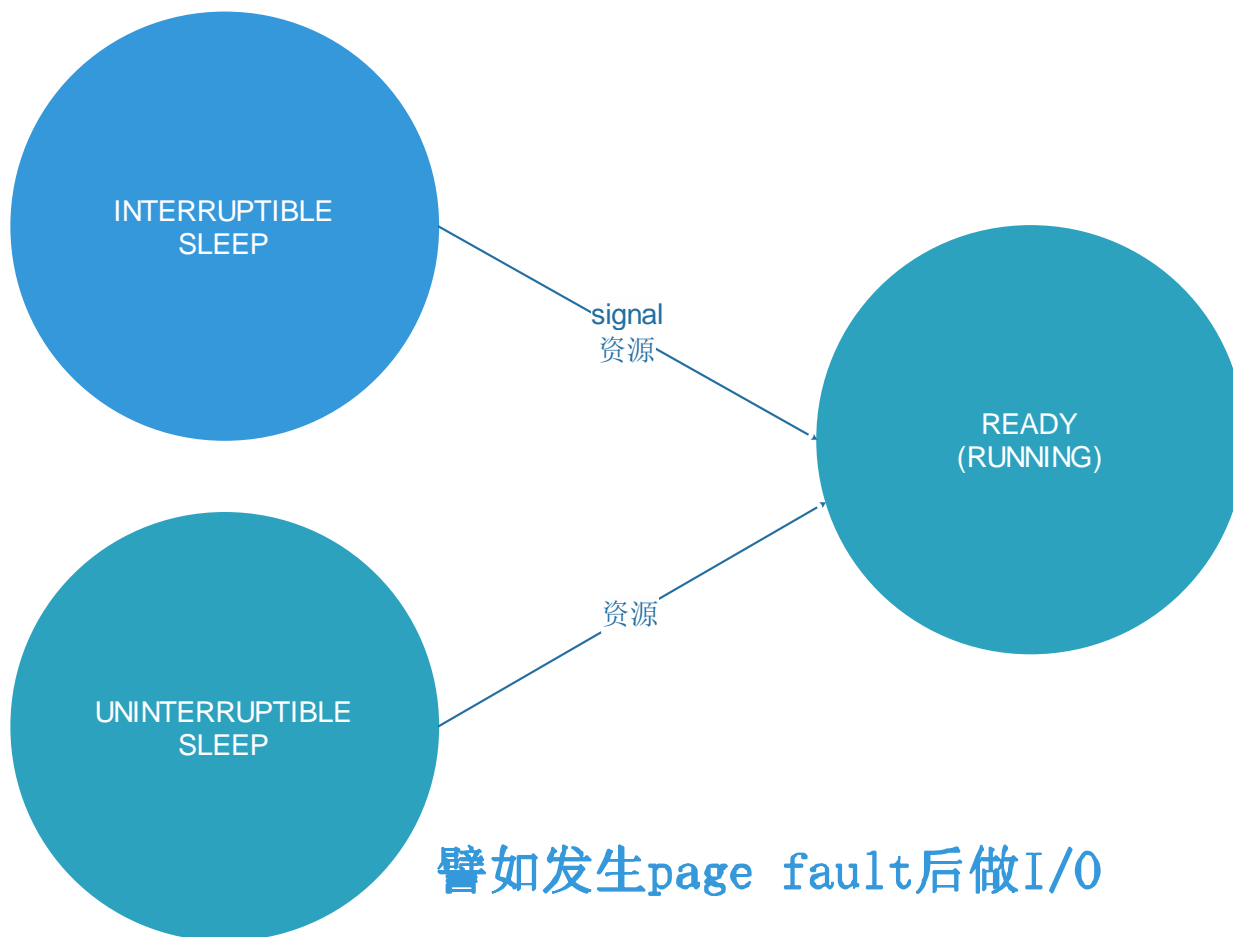


p2和p4死



睡眠

深度睡眠 vs. 浅度睡眠



wait queue

```
static ssize_t globalfifo_read(struct file *filp, char __user *buf,
                              size_t count, loff_t *ppos)
{
    int ret;
    struct globalfifo_dev *dev = container_of(filp->private_data,
        struct globalfifo_dev, miscdev);

    DECLARE_WAITQUEUE(wait, current);

    mutex_lock(&dev->mutex);
    add_wait_queue(&dev->r_wait, &wait);

    while (dev->current_len == 0) {
        if (filp->f_flags & O_NONBLOCK) {
            ret = -EAGAIN;
            goto out;
        }
        __set_current_state(TASK_INTERRUPTIBLE);
        mutex_unlock(&dev->mutex);

        schedule();
        if (signal_pending(current)) {
            ret = -ERESTARTSYS;
            goto out2;
        }

        mutex_lock(&dev->mutex);
    }

    if (count > dev->current_len)
        count = dev->current_len;

    if (copy_to_user(buf, dev->mem, count)) {
        ret = -EFAULT;
        goto out;
    } else {
        memcpy(dev->mem, dev->mem + count, dev->current_len - count);
        dev->current_len -= count;
        printk(KERN_INFO "read %d bytes(s), current_len: %d\n", count,
```

进程0和1

进程0

也是IDLE进程

父进程

进程1

```
baohua@baohua-VirtualBox:/$ pstree
init--ModemManager--2*[{ModemManager}]
    --NetworkManager--dhclient
    --NetworkManager--dnsmasq
    --NetworkManager--3*[{NetworkManager}]
    --VGAAuthService
    --accounts-daemon--2*[{accounts-daemon}]
    --acpid
    --at-spi-bus-laun--dbus-daemon
    --at-spi-bus-laun--3*[{at-spi-bus-laun}]
    --at-spi2-registr--{at-spi2-registr}
    --atd
    --atop
    --avahi-daemon--avahi-daemon
    --bluetoothd
    --colord--2*[{colord}]
    --cron
    --cups-browsed
```

课 程 练 习 源 码

<https://github.com/21cnbao/process-courses>

更早课程

- 《Linux总线、设备、驱动模型》录播：
<http://edu.csdn.net/course/detail/5329>
- 深入探究Linux的设备树
<http://edu.csdn.net/course/detail/5627>
- Linux进程、线程和调度
<http://edu.csdn.net/course/detail/5995>

谢谢！