Linux系统环境进程间通信：信号灯（1） PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/144/2021_2022_Linux_E7_B3_BB_E7_BB_c103_144778.htm 一、信号灯概述 信号灯与其他进程间通信方式不大相同，它主要提供对进程间共享资源访问控制机制。相当于内存中的标志，进程可以根据它判定是否能够访问某些共享资源，同时，进程也可以修改该标志。除了用于访问控制外，还可用于进程同步。信号灯有以下两种类型： 二值信号灯：最简单的信号灯形式，信号灯的值只能取0或1，类似于互斥锁。 注：二值信号灯能够实现互斥锁的功能，但两者的关注内容不同。信号灯强调共享资源，只要共享资源可用，其他进程同样可以修改信号灯的值；互斥锁更强调进程，占用资源的进程使用完资源后，必须由进程本身来解锁。 计算信号灯：信号灯的值可以取任意非负值（当然受内核本身的约束）。 二、Linux信号灯 linux对信号灯的支持状况与消息队列一样，在red had 8.0发行版本中支持的是系统V的信号灯。因此，本文将主要介绍系统V信号灯及其相应API。在没有声明的情况下，以下讨论中指的都是系统V信号灯。 注意，通常所说的系统V信号灯指的是计数信号灯集。 三、信号灯与内核 1、系统V信号灯是随内核持续的，只有在内核重起或者显示删除一个信号灯集时，该信号灯集才会真正被删除。因此系统中记录信号灯的数据结构（struct ipc_ids sem_ids）位于内核中，系统中的所有信号灯都可以在结构sem_ids中找到访问入口。 2、下图说明了内核与信号灯是怎样建立起联系的： 其中：struct ipc_ids sem_ids是内核中记录信号灯的全局数据结构；描述一个具体的信号灯及其相

关信息。 其中，struct sem结构如下： struct sem{int semval. // current valueint sempid // pid of last operation}从上图可以看出，全局数据结构struct ipc_ids sem_ids可以访问到struct kern_ipc_perm的第一个成员：struct kern_ipc_perm；而每个struct kern_ipc_perm能够与具体的信号灯对应起来且蛭诟媒峁怪校幸桓ey_t类型成员key，而key则唯一确定一个信号灯集；同时，结构struct kern_ipc_perm的最后一个成员sem_nsems确定了该信号灯在信号灯集中的顺序，这样内核就能够记录每个信号灯的信息了。kern_ipc_perm结构参见《Linux环境进程间通信（三）：消息队列》。struct sem_array见附录1。