Linux信号和阻塞Linux认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/641/2021_2022_Linux_E4_BF_A1_E5_8F_c103_641391.htm 1. 信号掩码被阻塞的信号集 每个进程都有一个用来描述哪些信号传送来将被阻塞的信号集，如果某种信号在某个进程的阻塞信号集中，则传送到该进程的此种信号将会被阻塞。当前被进程阻塞的信号集也叫信号掩码，类型为sigset_t。每个进程都有自己的信号掩码，且创建子进程时，子进程会继承父进程的信号掩码。 2. 信号阻塞和忽略的区别 阻塞的概念与忽略信号是不同的：操作系统在信号被进程解除阻塞之前不会将信号传递出去，被阻塞的信号也不会影响进程的行为，信号只是暂时被阻止传递；当进程忽略一个信号时，信号会被传递出去，但进程将信号丢弃。 3. 信号集的操作 信号集可以由以下几个函数操作： int sigemptyset(sigset_t *set). //清空信号集 int sigfillset(sigset_t *set). //将所有信号填充进set中 int sigaddset(sigset_t *set, int signum). //往set中添加信号signum int sigdelset(sigset_t *set, int signum). //从set中移除信号signum int sigismember(const sigset_t *set, int signum). //判断signnum是不是包含在set中，在返回1，不在返回0 初始化往往可以用sigemptyset（）将信号集清空，再用sigaddset（）向信号集中添加信号；或者可以使用sigfillset（）将所有信号添加到信号集，再用sigdelset（）将某信号从中删除掉。 4. sigprocmask（）介绍 可以使用函数sigprocmask（）来检查或者修改进程的信号掩码。函数信息如下： #include gt. int sigprocmask (int how, const sigset_t *restrict set, sigset_t *restrict old). 参数how 是一个整数，说明信号掩码的修改方式

：SIG_BLOCK --- 将set指向的信号集中的信号添加到当前阻塞信号集中；SIG_UNBLOCK --- 从当前阻塞信号集中移除set指向的信号集中的信号；SIG_SETMASK --- 指定set所指向的信号集为当前阻塞信号集。 此外，如果参数set 为NULL, 说明不需要修改，如果old 为NULL，sigprocmask会将修改之前的信号集放在*old 之中返回。 5.sigaction（）回顾 在前面有用过sigaction（）函数： include gt. int sigaction(int signum,const struct sigaction *act, const struct sigaction *oldact). 该函数是用于注册一个信号处理函数。参数结构体sigaction与函数同名，具体信息如下： struct sigaction { void (*sa_handler)(int). //老类型的信号处理函数指针 void (*sa_sigaction)(int, siginfo_t *, void *).//新类型的信号处理函数指针 sigset_t sa_mask. //将要被阻塞的信号集合 int sa_flags. //信号处理方式掩码 void (*sa_restorer)(void). //保留 } 5.1 sa_handler：一个函数指针，用于指向原型为void handler(int)的信号处理函数地址（老类型的信号处理函数）；5.2 sa_sigaction：也是一个函数指针，用于指向原型为： void handler(int （新类型的信号处理函数）；三个参数的含义为： iSignNum：传入的信号 pSignInfo：与该信号相关的一些信息，它是个结构体 pReserved：保留，现没用 5.3 sa_handler和sa_sigaction只应该有一个生效，如果想采用老的信号处理机制，就应该让sa_handler指向正确的信号处理函数；否则应该让sa_sigaction指向正确的信号处理函数，并且让字段sa_flags包含SA_SIGINFO选项。 5.4 sa_mask是一个包含信号集合的结构体，该结构体内的信号表示在进行信号处理时，将要被阻塞的信号。该信号集可以用前面标题3提到的5个函数来进行操作。 5.5 字段sa_flags是一组掩码的合成值

，指示信号处理时所应该采取的一些行为，各掩码的含义为：（1）SA_RESETHAND ---处理完毕要捕捉的信号后，将自动撤消信号处理函数的注册，即必须再重新注册信号处理函数，才能继续处理接下来产生的信号。（2）SA_NODEFER ---在处理信号时，如果又发生了其它的信号，则立即进入其它信号的处理，等其它信号处理完毕后，再继续处理当前的信号，即递规地处理。如果sa_flags包含了该掩码，则结构体sigaction的sa_mask将无效；（3）SA_RESTART--- 如果在发生信号时，程序正阻塞在某个系统调用，例如调用read()函数，则在处理完毕信号后，接着从阻塞的系统返回。该掩码符合普通的程序处理流程，所以一般来说，应该设置该掩码，否则信号处理完后，阻塞的系统调用将会返回失败；（4）SA_SIGINFO ---指示结构体的信号处理函数指针是哪个有效，如果sa_flags包含该掩码，则sa_sigactiion指针有效，否则是sa_handler指针有效。 需要注意的是：函数sigprocmask是全程阻塞，在sigprocmask中设置了阻塞集合后，被阻塞的信号将不能再被信号处理函数捕捉，直到重新设置阻塞信号集合。而在sigaction（）注册信号处理函数时，选择阻塞的信号集只是在处理捕捉的信号时，才对指定的其他信号进行阻塞。