Linux进程间通信：管道及有名管道（2） 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_E8_BF_9B_E7_A8_c103_144877.htm 1.3管道的读写规则： 管道两端可分别用描述字fd[0]以及fd[1]来描述，需要注意的是，管道的两端是固定了任务的。即一端只能用于读，由描述字fd[0]表示，称其为管道读端；另一端则只能用于写，由描述字fd[1]来表示，称其为管道写端。如果试图从管道写端读取数据，或者向管道读端写入数据都将导致错误发生。一般文件的I/O函数都可以用于管道，如close、read、write等等。 从管道中读取数据： 如果管道的写端不存在，则认为已经读到了数据的末尾，读函数返回的读出字节数为0；当管道的写端存在时，如果请求的字节数目大于PIPE_BUF，则返回管道中现有的数据字节数，如果请求的字节数目不大于PIPE_BUF，则返回管道中现有数据字节数（此时，管道中数据量小于请求的数据量）；或者返回请求的字节数（此时，管道中数据量不小于请求的数据量）。注：（PIPE_BUF在include/linux/limits.h中定义，不同的内核版本可能会有所不同。Posix.1要求PIPE_BUF至少为512字节，red hat 7.2中为4096）。 关于管道的读规则验证： * readtest.c *#include #include #include main(){int pipe_fd[2].pid_t pid.char r_buf[100].char w_buf[4].char* p_wbuf.int r_num.int cmd.memset(r_buf,0,sizeof(r_buf)).memset(w_buf,0,sizeof(r_buf)).p_wbuf=w_buf.if(pipe(pipe_fd)0){close(pipe_fd[0]).//readstrcpy(w_buf,"111").if(write(pipe_fd[1],w_buf,4)!=-1)printf("parent write over\n").close(pipe_fd[1]).//writeprintf("parent close fd[1]

over\n").sleep(10).}}程序输出结果：* parent write over* parent close fd[1] over* read num is 4 the data read from the pipe is 111附加结论：管道写端关闭后，写入的数据将一直存在，直到读出为止。100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com