对一个对端已经关闭的socket调用两次write, 第二次将会生成SIGPIPE信号, 该信号默认结束进程, 用gdb执行程序, 退出时提示"Broken pipe".

当服务器close一个连接时，若client端接着发数据。根据TCP协议的规定，会收到一个RST响应，client再往这个服务器发送数据时，系统会发出一个SIGPIPE信号给进程，告诉进程这个连接已经断开了，不要再写了。

根据信号的默认处理规则SIGPIPE信号的默认执行动作是terminate(终止、退出),所以client会退出

为了避免进程退出, 可以捕获SIGPIPE信号, 或者忽略它, 给它设置SIG_IGN信号处理函数:

```
signal(SIGPIPE, SIG_IGN);
```

这样, 第二次调用write方法时, 会返回-1, 同时errno置为SIGPIPE. 程序便能知道对端已经关闭.

忽略SIGPIPE信号的方法

```
struct sigaction sa;
sa.sa_handler = SIG_IGN;//设定接受到指定信号后的动作为忽略
sa.sa_flags = 0;
if (sigemptyset(&sa.sa_mask) == -1 ||     //初始化信号集为空
sigaction(SIGPIPE, &sa, 0) == -1) {     //屏蔽SIGPIPE信号
perror("failed to ignore SIGPIPE; sigaction");
exit(EXIT_FAILURE);
}
```

pthread线程里如何屏蔽SIGPIPE异常

http://bbs2.chinaunix.net/viewthread.php?tid=985166&extra=&page=1

在pthread中, 可能会遇到Program received signal SIGPIPE, Broken pipe的问题, 解决方法是每一个线程启动之前时, 先执行下面代码:

```
#ifndef WIN32
sigset_t signal_mask;
```

```
sigemptyset (&signal_mask);
sigaddset (&signal_mask, SIGPIPE);
int rc = pthread_sigmask (SIG_BLOCK, &signal_mask, NULL);
if (rc != 0) {
printf("block sigpipe error\n");
}
#endif
```