使用setsockopt来控制connect超时

原来我们实现connect()超时基本上都使用unix网络编程一书的非阻塞方式（connect_nonb），今天在网上看到一篇文章，觉得很有意思，**转载**如下：

读Linux内核源码的时候偶然发现其connect的超时参数竟然和用SO_SNDTIMO操作的参数一致:
File: net/ipv4/af_inet.c

```
559     timeo = sock_sndtimeo(sk, flags & O_NONBLOCK);
560
561     if ((1 << sk->sk_state) & (TCPF_SYN_SENT | TCPF_SYN_RECV)) {
562         /* Error code is set above */
563         if (!timeo || !inet_wait_for_connect(sk, timeo))
564             goto out;
565
566         err = sock_intr_errno(timeo);
567         if (signal_pending(current))
568             goto out;
569     }
```

这意味着：**在Linux平台下，可以通过在connect之前设置SO_SNDTIMO来达到控制连接超时的目的。**简单的写了份测试代码:

[cpp:showcolumns] view plaincopy

```
·····10·····20·····30·····40·····50·····60·····70·····80·····90·····100·····110·····120·
```

```cpp
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <errno.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>

int main(int argc, char *argv[])
{
        int fd;
        struct sockaddr_in addr;
        struct timeval timeo = {3, 0};
        socklen_t len = sizeof(timeo);

         fd = socket(AF_INET, SOCK_STREAM, 0);
         if (argc == 4)
                 timeo.tv_sec = atoi(argv[3]);

         setsockopt(fd, SOL_SOCKET, SO_SNDTIMEO, &timeo, len);
         addr.sin_family = AF_INET;
         addr.sin_addr.s_addr = inet_addr(argv[1]);
         addr.sin_port = htons(atoi(argv[2]));

        if (connect(fd, (struct sockaddr*)&addr, sizeof(addr)) == -1) {
                if (errno == EINPROGRESS) {
                        fprintf(stderr, "timeout/n");
                        return -1;
                }
                perror("connect");
                return 0;
        }
        printf("connected/n");

        return 0;
}
```

使用 ./cmd ip地址 端口 超时秒数
(测试的ip和端口必须是不存在的，或者是ip的机器是死掉的，才会出现，否则机器存在而端口不存在会立即返回的)