linux—select详解

收藏

linux—select详解

select系统调用时用来让我们的程序监视多个文件句柄的状态变化的。程序会停在select这里等待，直到被监视的文件句柄有一个或多个发生了状态改变。

关于文件句柄，其实就是一个整数，通过socket函数的声明就明白了：

int socket(int domain, int type, int protocol);

我们最熟悉的句柄是0、1、2三个，0是标准输入，1是标准输出，2是标准错误输出。0、1、2是整数表示的，对应的FILE *结构的表示就是stdin、stdout、stderr。

继续上面的select，就是用来监视某个或某些句柄的状态变化的。select函数原型如下：

int select (int nfds, fd_set *readfds, fd_set *writefds, fd_set *exceptfds, struct timeval *timeout);

函数的最后一个参数timeout是一个超时时间值。其类型是struct timeval *，即一个struct timeval结构的变量的指针，所以我们在程序里要声明一个struct timeval tv;然后把变量tv的地址&tv传递给select函数。struct timeval结构如下：

```
struct timeval
{
    long tv_sec;  //seconds
    long tv_usec; //microseconds
};
```

第2、3、4三个参数是一样的类型;fd_set *,即我们在程序里要申请几个fd_set类型的变量，比如rdfds，wtfds，exfds，然后把这个变量的地址&rdfds,&wtfds,&exfds传递给select函数。这三个参数都是一个句柄的集合，第一个rdfds是用来保存这样的句柄的:当句柄的状态变成可读时系统就告诉select函数返回，同理第二个函数是指向有句柄状态变成可写时系统就会告诉select函数返回，同理第三个参数exfds是特殊情况，即句柄上有特殊情况发生时系统会告诉select函数返回。特殊情况比如对方通过一个socket句柄发来了紧急数据。如果我们程序里只想检测某个socket是否有数据可读，我们可以这样：

```
fd_set  rdfds;
struct timeval tv;
int ret;
FD_ZERO(&rdfds);
FD_SET(socket, &rdfds);
tv.tv_sec = 1;
tv.tv_uses = 500;
ret = select (socket + 1, %rdfds, NULL, NULL, &tv);
if(ret < 0) perror (“select”);
else if (ret = = 0) printf(“time out”);
else {
    printf(“ret = %d\n”,ret);
    if(FD_ISSET(socket, &rdfds)){
  /* 读取socket句柄里的数据 */
recv( );
```

```
}
}
```

注意select函数的第一个参数，是所有加入集合的句柄值的最大那个那个值还要加1.比如我们创建了3个句柄;

```
int sa, sb, sc;
sa = socket(……);
connect (sa,….);

sb = socket(….);
connect (sb,…);

sc = socket(….);
connect(sc,…);

FD_SET(sa, &rdfds);
FD_SET(sb, &rdfds);
FD_SET(sc, &rdfds);
```

在使用select函数之前，一定要找到3个句柄中的最大值是哪个，我们一般定义一个变量来保存最大值，取得最大socket值如下:

```
int maxfd = 0;
if(sa > maxfd) maxfd = sa;
if(sb > maxfd) maxfd = sb;
if(sc > maxfd) maxfd = sc;
```

然后调用select函数:

```
ret = select (maxfd+1, &rdfds, NULL, NULL,&tv);
```

同样的道理，如果我们是检测用户是否按了键盘进行输入，我们就应该把标准输入0这个句柄放到select里来检测，如下:

```
FD_ZERO(&rdfds);
FD_SET(0, &rdfds);
tv.tv_sec = 1;
tv.tv_usec = 0;
ret = select (1, &rdfds,NULL,NULL,&tv);
if(ret < 0) perror(“select”);
else if (ret = = 0) printf (“time out\n”);
else{
    scanf(“%s”,buf);
}
```

发表于 @ 2009年04月07日 22:58:00 | 评论(0) | 举报 | 收藏

旧一篇: 内存中堆和栈 | 新一篇: HP大中华区总裁孙振耀退休感言

发表评论

表 情:

评论内容:

用 户 名: 匿名用户 登录 注册

验 证 码: GBZU 重新获得验证码

发表评论