如可启动外部可执行程序？这里我们通过 [exec 函数族](http://baike.baidu.com/link?url=-GZLZkU-S3Wlp4w3WiVqfvG-PuxjYY79kKkq_5pEK1TYnbwcylVGYWTtnokWbFyG1a1pXT20SV9V_2EZROCJg_)实现。

[exec 函数族](http://baike.baidu.com/link?url=-GZLZkU-S3Wlp4w3WiVqfvG-PuxjYY79kKkq_5pEK1TYnbwcylVGYWTtnokWbFyG1a1pXT20SV9V_2EZROCJg_)，顾名思义，就是一簇函数，在 Linux 中，并不存在 exec() 函数，exec 指的是一组函数，一共有 6 个：

#include <unistd.h>

int execl(const char \*path, const char \*arg, ...);

int execlp(const char \*file, const char \*arg, ...);

int execle(const char \*path, const char \*arg, ..., char \* const envp[]);

int execv(const char \*path, char \*const argv[]);

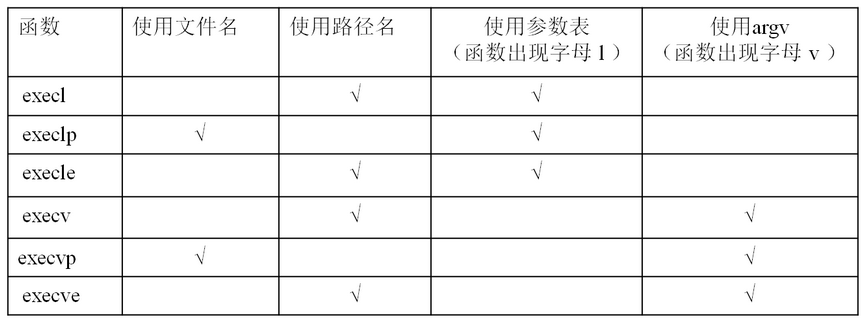
int execvp(const char \*file, char \*const argv[]);

int execve(const char \*path, char \*const argv[], char \*const envp[]);

其中只有 execve() 是真正意义上的系统调用，其它都是在此基础上经过包装的库函数。

exec 函数族提供了六种**在进程中启动另一个程序的方法**。exec 函数族的作用是根据指定的文件名或目录名找到可执行文件，并创建新进程，取代当前进程

[exec 函数族](http://baike.baidu.com/link?url=-GZLZkU-S3Wlp4w3WiVqfvG-PuxjYY79kKkq_5pEK1TYnbwcylVGYWTtnokWbFyG1a1pXT20SV9V_2EZROCJg_)的 6 个函数看起来似乎很复杂，但实际上无论是作用还是用法都非常相似，只有很微小的差别。



l(list)：参数地址列表，以空指针结尾。

v(vector)：存有各参数地址的指针数组的地址。

p(path)：按 PATH 环境变量指定的目录搜索可执行文件。

e(environment)：存有环境变量字符串地址的指针数组的地址。

exec 函数族与一般的函数不同，exec 函数族中的函数执行成功后不会返回，而且，**exec 函数族下面的代码执行不到**。只有调用失败了，它们才会返回 -1，失败后从原程序的调用点接着往下执行。

excel代码：

#include <stdio.h>

#include <unistd.h>

int main(int argc, char \*argv[])

{

    printf("before exec\n\n");

    /\* /bin/ls：外部程序，这里是/bin目录的 ls 可执行程序，必须带上路径（相对或绝对）

       ls：没有意义，如果需要给这个外部程序传参，这里必须要写上字符串，至于字符串内容任意

       -a，-l，-h：给外部程序 ls 传的参数

       NULL：这个必须写上，代表给外部程序 ls 传参结束

    \*/

    execl("/bin/ls", "ls", "-a", "-l", "-h", NULL);

    // 如果 execl() 执行成功，下面执行不到，因为当前进程已经被执行的 ls 替换了

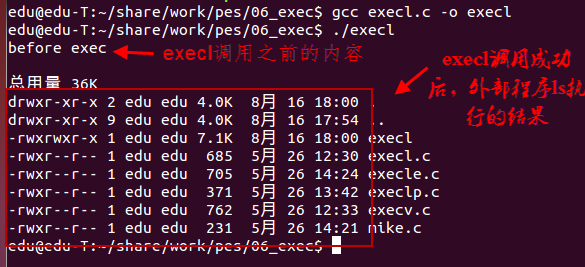
    perror("execl");

    printf("after exec\n\n");

    return 0;

}

运行结果：



**execv()示例代码：**

execv() 和 execl() 的用法基本是一样的，无非将列表传参，改为用指针数组。

#include <stdio.h>

#include <unistd.h>

int main(int argc, char \*argv[])

{

    // execv() 和 execl() 的用法基本是一样的，无非将列表传参，改为用指针数组

    // execl("/bin/ls", "ls", "-a", "-l", "-h", NULL);

    /\* 指针数组

       ls：没有意义，如果需要给这个外部程序传参，这里必须要写上字符串，至于字符串内容任意

       -a，-l，-h：给外部程序 ls 传的参数

       NULL：这个必须写上，代表给外部程序 ls 传参结束

    \*/

    char \*arg[]={"ls", "-a", "-l", "-h", NULL};

    // /bin/ls：外部程序，这里是/bin目录的 ls 可执行程序，必须带上路径（相对或绝对）

    // arg： 上面定义的指针数组地址

    execv("/bin/ls", arg);

    perror("execv");

    return 0;

}