# linux系统编程

## 计算机系统组成

|  |
| --- |
|  |

## 系统编程与应用编程

|  |
| --- |
| 系统编程：  在操作系统之上利用系统调用、C库对系统资源进行访问。如apache、gcc、gdb  应用编程：  在更高层次的编程接口或者库之上构建应用程序。如android程序、iphone程序、QT程序设计、MFC程序设计 |
| 系统资源：  处理器  输入输出  进程管理  内存  设备  定时器  进程间通信  网络 |
| 所有操作系统都提供多种服务的入口点，由此程序向系统内核请求服务。这些入口点称之为系统调用（system call） |
| 系统调用与C库关系（C库做到了缓冲，减少调用系统调用，效率更高）  系统调用与C库从形式上看都是C函数  C库函数有些是调用系统来实现的。比如说malloc,free调用brk，printf调用write系统，有些不需要调用系统来完成，比如abs,strcpy,atoi，因为不需要内核服务 |
| 内核处理系统调用  每个系统调用被赋予一个系统调用号  在i386平台上，执行一个系统调用是通过int 0x80指令完成的  eax存放系统调用号  ebx,ecx,edx,esi,edi存储系统调用参数，对于超过5个参数的系统调用，用一个寄存器指向用户空间的存储所有的系统调用参数的缓冲 |
|  |

## I/O章节

### I/O含义

|  |
| --- |
| 输入/输出是主存与外部设备之间拷贝数据的过程  设备 --> 内存 （输入操作）  内存 --> 设备 （输出操作） |
| 高级I/O  ANSI C提供的标准I/O库称为高级I/O，通常也称为带缓冲的I/O |
| 低级I/O  通常也称为不带缓冲I/O |

### 文件描述符与文件指针转换

|  |
| --- |
| fileno: 将文件指针转换为文件描述符 |
| fdopen: 将描述符转为文件指针 |

### 文件系统调用

|  |
| --- |
| open: |
| close: |
| creat: |
| read: |
| write: |

### 打开文件的方式

|  |  |
| --- | --- |
| 打开方式 | 描述 |
| O\_RDONLY |  |
| O\_WRONLY |  |
| O\_RDWR |  |
| O\_APPEND |  |
| O\_CREAT |  |
| O\_EXCL | 如果已经置O\_CREAT且文件已经存在，则强制open()失败 |
| O\_TRNUC |  |

### 访问文件的权限

|  |  |
| --- | --- |
| 打开方式 | 描述 |
| S\_IRUSR |  |
| S\_IWUSR |  |
| S\_IXUSR |  |
| S\_IRWXU | S\_IXUSR | S\_IWUSR | S\_IRUSR |
| S\_IRGRP |  |
| S\_IWGRP |  |
| S\_IXGRP |  |
| S\_IRWXG | S\_IXGRP | S\_IWGRP | S\_IRGRP |
| S\_IROTH |  |
| S\_IWOTH |  |
| S\_IXOTH |  |
| S\_IRWXO | S\_IXOTH | S\_IWOTH | S\_IROTH |

### 查看系统的限制

|  |
| --- |
| ulimit -a  core file size (blocks, -c) 0  data seg size (kbytes, -d) unlimited  scheduling priority (-e) 0  file size (blocks, -f) unlimited  pending signals (-i) 7336  max locked memory (kbytes, -l) unlimited  max memory size (kbytes, -m) unlimited  open files (-n) 65536  pipe size (512 bytes, -p) 8  POSIX message queues (bytes, -q) 819200  real-time priority (-r) 95  stack size (kbytes, -s) 8192  cpu time (seconds, -t) unlimited  max user processes (-u) 7336  virtual memory (kbytes, -v) unlimited  file locks (-x) unlimited |
| 查看系统最多能打开多少个文件 cat /proc/sys/fs/file-max |