组长:梅洛瑜71117408

组员:张婧媛71117402，刘钰宵71117405

讨论题目:Why do user use APIs rather than system calls directly?

1. 系统调用:用户空间应用程序和内核提供服务之间的一个接口。程序的运行空间逻辑上是分离的，分为内核空间和用户空间。由于服务实在内核中提供给的，因此无法执行直接调用；用户必须使用一个进程来跨越用户空间与内核之间的界限。用户通过操作系统提供的接口（API）获得内核提供的一些服务。

系统调用处理程序执行下列操作：

* 1. 在内核态栈保持大多数寄存器的内容
  2. 调用名为系统调用服务例程的相应的C函数来处理系统调用
  3. 退出系统调用处理程序：用保存在内核栈中的值加载寄存器，CPU从内核态切换回到用户态

因为内核实现了很多不同的系统调用，因此进程必须传递一个名为系统调用号的参数来是吧所需的系统调用。为了把系统调用号与相应的服务例程关联起来，内核利用了一个系统调用分派表，这个表放在sys\_call\_table数组中，有NR\_syscalls个表项，第n个表项包含系统调用号为n的服务例程的地址

1. APIs: **API**是一些预先定义的函数，目的是提供应用程序与开发人员基于某软件或硬件得以访问一组例程的能力，而又无需访问源码，或理解内部工作机制的细节。API常以c库(libc)的形式提供，c库提供了绝大部分API，每个系统调用在c库中都有对应的封装函数(通常封装函数与系统调用的名称相同)。系统调用与c库函数并不是一一对应的，有些c库函数可能使用多个系统调用来实现，也有可能多个c库函数使用同一个系统调用来实现，也有些c库函数不使用任何系统调用来实现。
2. 系统调用和API调用的区别
3. 系统调用是为了方便应用使用操作系统的接口，而库函数是为了方便人们编写应用程序而引出的，比如你自己编写一个函数其实也可以说就是一个库函数。
4. 系统调用可以理解为内核提供用户系统内核接口，可以认为是某种内核的库函数。
5. 系统调用可以为用户空间提供访问硬件资源的统一接口，以至于应用程序不必去关注具体的硬件访问操作。比如，读写文件时，应用程序不用去管磁盘类型，甚至于不用关心是哪种文件系统。
6. 系统调用可以对系统进行保护，保证系统的稳定和安全。系统调用的存在规定了用户进程进入内核的具体方式，换句话说，用户访问内核的路径是事先规定好的，只能从规定位置进入内核，而不准许肆意跳入内核。有了这样的进入内核的统一访问路径限制才能保证内核的安全。

4. 为什么使用API而不是直接使用系统调用

1)系统调用效率不一定比库函数高

2)API是函数的定义，规定了这个函数的功能，跟内核无直接关系。而系统调用是通过中断向内核发请求，实现内核提供的某些服务。程序员调用的是API（API函数），然后通过与系统调用共同完成函数的功能。因此，API是一个提供给应用程序的接口，一组函数，是与程序员进行直接交互的。系统调用则不与程序员进行交互的，它根据API函数，通过一个软中断机制向内核提交请求，以获取内核服务的接口。

3)并不是所有的API函数都一一对应一个系统调用，有时，一个API函数会需要几个系统调用来共同完成函数的功能，甚至还有一些API函数不需要调用相应的系统调用（因此它所完成的不是内核提供的服务）。不接触内核会更好地保护系统地稳定和安全。

4)使用库函数的好处是隐藏访管指令细节，使系统调用更像函数调用。对用户隐蔽系统调用，使得在改动内核时，不影响应用程序的正确性。

5)库函数调用通常用于应用程序中对一般文件的访问，库函数调用是系统无关的，因此移植性好。

6)系统调用所提供给用户的是直接而纯碎的高级服务，如果想要更加人性化，具有更符合特定情况的功能，那么就要用户自己定义，API把部分系统调用包装起来。比如当我们要用C语言打印一句话的时候，如果没有用到库函数printf，那么我们就需要自己实现就需要调用putc()和write()等这样一些系统函数。显得比较麻烦，系统调用是为了方便使用操作系统的接口，而API可以让人们编程更方便。