

操作系统第一次书面作业

- 数据科学与计算机学院
- 软工三班
- 米家龙
- 18342075

米家龙 18342075 软工三班

2.2

1. 执行文件。OS 将一个文件的目录或者章节载入到内存并运行。如果该程序是用户程序，那它是不可信任并且需要分配好 CPU 时间

2. I/O。磁盘、磁带等有读装置和事件，及其它设备需在低水平下进行通信。用户指定指令装置和执行要求，OS 将具体要求转换成合理的命令，并且用户程序不能访问其它被授权访问的设备。

3. 文件系统操作，包括创建、修改删除、分配和命名等。上述操作存在许多用户不能执行的细节。文件所使用的磁盘空间会被跟踪。在删除该文件过程中，文件的信息和空间会分别被清除和释放。用户程序在此不会有效实施，不能破坏

4. 信任

5. 通信。系统间交换信息转换成包，送网络控制器中，通过通信媒介传输，并在目的地系统重新组装。该过程中，信息包调整和数据修改会发生。用户程序可能不被授权调用网相关网络装置，并接收其它的数据包。

6. 错误检测。在软件硬件中都会发生。硬件水平下，数据转移需要检查以防止传输过程中不会被破坏。写入媒介时，需要检查以确保写入时不会被改变。在软件水平下，媒介不需不间断检查，而进程的独立会带来数据破坏错误，因此需要一个综合处理程序的存在。而经过该系统处理后，系统中的程序不再需包含已改正错误的代码，只需抛出错误即可。

2.3

1. 寄存器传参

2. 寄存器传参数块的地址。

3. 程序存放/压进堆栈，并由 OS 弹出。

2.4

① 可发出周期性的定时器中断，并监视中断出现时正在执行哪部分代码

② 一个用于记录哪部分代码活跃的记录文件，让与该程序在不同部分花费同样的时间。

③ 在获取到该统计信息后，能推断出相关信息优化 CPU 资源分配

2.5 ① 创建/删除文件

② 创建/删除文件目录文件夹

③ 对多个文件/文件目录的原数据支持

④ 将文件映射到二级存储

⑤ 在非易失媒介上备份文件

2.6

优点：

① 可以像访问文件系统一样访问设备

② 内核大多用文件外连接以处理设备，因此抽象文件代码接口可使添加新设备驱动程序相对容易

③ 因此即有利于以相同方式写入文件访问设备和文件用户程序代码，也可写入支持良好的 API 的驱动程序代码

缺点：

① 可能难以在功能访问 API 的上下文开发设备功能，从而导致功能丧失/性能下降

② ioctl 操作为进程调用设备操作提供通用接口，可以使用

2.7

用途：命令解释器通常从用户/文件中读取命令并执行，方式是将它们变成一个或多个系统调用。

为什么：因为命令解释器可能发生变更。

可能发展：可通过使用 OS 提供的系统调用界面来开发，因为命令解释器功能可由用户程序用系统调用来实现

2.8

① 两种模式：共享内存 和 信息传递

② 内存共享通过系统调用创建，内存共享块在多个进程间建立后进程间通信可通过该块通信，不需要内核协助

③ 内核直接包含进程间通信，因为信息传递包含系统调用，并且影响较小

④ 信息传递可以用作同步机制处理通信间进程间的并行行动，而内存共享无

2.9

1. 保证系统比较轻松地修改。若两者分开，则可使机制不变时改变政策。

2.10

① 来由 OS 将其映射到主机 OS 来提供服务

② 问题：系统调用接口方面是否够用，以便支持较 OS 相关功能

2.18 见下

```
// 在 linux 上写
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>

#define BUFFLENGTH 8192

int main(int argc, char **argv)
{
    if (argc != 3) // 参数不够
    {
        printf("usage: copy src dst\n");
        return -1;
    }
    int src, dst;
    char buffer[BUFFLENGTH];
    src = open(argv[1], O_RDONLY);
    dst = open(argv[2],
                O_RDWR | O_CREAT | O_TRUNC, // 设置权限
                S_IRUSR | S_IWUSR | S_IXUSR); // 设置用户权限

    int tmp;
    while ((tmp = read(src, buffer, BUFFLENGTH)) > 0) // 不断读取
    {
        if (write(dst, buffer, tmp) != tmp) // 写入失败
            printf("ERROR in writing\n");
    }
    if (tmp < 0) // 读取失败
        printf("ERROR in reading\n");

    // 关闭文件
    close(src);
    close(dst);
    return 0;
}
```

米宇龙 18342075 软工3班

3.1: 主要在执行时

- ① 短期调度须经常调用新进程
- ② 中期调度将部分程序移出内存, 在中断位置继续
- ③ 长期调度则可能让进程离开系统时才会唤起, 不频繁调用

3.2

- ① 保存正在运行的进程状态, 主要包括CPU寄存器的值和内存分配, 上下文切换
- ② 恢复进程的状态, 上下文切换, 进行数据刷新和指令缓存等操作。

3.3

- ① PRC 不支持“最多一次”、“至少一次”的语义, 则PRC服务器则不会保证不会多次调用而远程过程, 而远程过程的单次调用可能导致服务器多次退出
- ② 为了支持上述语义, 需要服务器维护客户端状态。如果不能支持, 则该系统只能安全提供不会更改数据或提供时间敏感的远程过程。

5

① 对称和非对称通信

对称通信允许收发者之间有 n 个集合点, 但在阻塞发送时并不需要集合点, 并且消息是同步传递的。因此需要两种能同步形式

② 自动和显式缓冲

自动缓冲提供了无限长的队列, 保证发送者复制消息时不会阻塞, 但缺点是会浪费缓存。可通过明确缓冲区大小来避免浪费。

③ 复制发送和引用发送

复制发送不允许接收者改变参数状态, 而引用发送允许。因此引用发送允许程序员写一个分布式版本的集中的应用程序

④ 固定大小和可变大小消息

固定大小的消息在特定大小缓冲区中可容纳数量已知, 而可变大小消息则不然。

4.3

内核线程页面错误时, 另外的内核线程会转换为使用页错误时间, 对于单一线程进程, 如果发生页错误, 其性能会下降。

4.4 共享: 堆内存和全局变量

如: 寄存器值和栈内存。

- 4.6. ① 如果系统将进程和线程视为实体, 系统代码可简化
- ② 上述情况会使得进程资源限制更加困难, 因为需要线程与进程哪些一致和程序重复计数。

4.7 DC: 5

② P: 0

补充

2.11

- ① 虚拟子系统和存储子系统相耦合, 许多允许对被映射到一个执行进程的虚拟空间
- ② 虚拟子系统通常用存储子系统提供当前不在内存的页
- ③ 虚拟空间中更新的文件在磁盘刷新前缓冲到物理内存。

2.12

- 优点: ① 增加新服务不需要修改内核。
- ② 在用户模式中比在内核模式中更安全和易操作

③ 可能更可靠

相互作用方式: 使用进程间通信机制

缺点: 进程间通信过度联系, 为了保证用户程序和系统服务相互作用会频繁使用操作系统的消息传递。

2.13

- ① 模块化内核方法要求子系统通过创建接口来相互作用
- ② 两者细节相似, 但分层内核要有严格排序的子系统, 从而限制低层只提高层子系统。而模块化内核则彼此之间可以随意提