虚拟存储器提供了三个重要的能力：

1. 将主存看成时一个存储在磁盘上的地址空间的高速缓存，在主存中只保存活动区域，并根据需要在磁盘和主存来回
   传送数据，通过这种方式高效的使用主存
2. 它为每个进程提供一致的地址空间，从而简化了存储器管理
3. 保护了每个进程的地址空间不被其他进程破坏。

9.1 物理和虚拟寻址

物理地址：计算机系统的主存被组织成一个由M个连续的字节大小的单元组成的数据，每个字节都有一个唯一的物理地址。

使用虚拟地址时，CPU通过生成一个虚拟地址来访问主存。
需要进行地址翻译。

9.2 地址空间

地址空间是一个非负整数地址的有序集合
物理地址空间
虚拟地址空间，一个地址空间的大小是由表示最大地址所需要的位数来描述

9.3 虚拟存储器作为缓存工具（？？？）

虚拟存储器(VM)系统通过将虚拟存储器分割为虚拟页的大小固定的块来处理这个问题
每个虚拟页的大小为2^p字节，类似的，物理存储器被分割为物理页，大小也为2^p字节.

虚拟页面的集合都分为三个不相交的子集：
未分配的：VM系统还未分配的页。未分配的块没有任何数据和它们相关联，因此也就不占用任何磁盘空间
缓存的：当前缓存在物理存储器中的已分配页
为缓存的：没有缓存在物理存储器中的已分配页。

9.3.2 页表

假设为单独页表。
页表用于将虚拟页映射到物理页。

概念：
页表条目：由一个有效位和一个n位的地址字段组成。
若设置了有效位，那么地址字段就表示相应物理页的起始地址，这个物理页中缓存了该虚拟页

若没有设置有效位，那么一个空地址表示这个虚拟页还未被分配。否则，这个地址就指向该虚拟页在磁盘中的起始位置。

9.3.3 页命中

9.3.4 缺页

9.3.5 分配页面

9.4 虚拟存储器作为存储器管理工具

简化了存储器管理，并提供了一种自然的保护存储器的方法。

操作系统为每个进程提供了一个独立的页表，也就是一个独立的虚拟地址空间。
注意：多个虚拟页面可以映射到同一个共享物理页面上。

VM简化了链接和加载、代码和数据共享，以及应用程序的存储器分配。

- 简化链接：独立的地址空间允许每个进程的存储器映像使用相同的基本格式。
- 简化加载：将数据段和代码段加载到一个新创建的进程中，linux加载器分配虚拟页的一个连续的片，从地址0x08048000处开始或者从
  0x400000处开始，把这些虚拟页标记为无效的（即未被缓存的），将页表条目指向目标文件中适当的位置。加载器从不实际拷贝任何数据从磁盘到存储器。
  之后按需页面调度。
- 简化共享：当进程需要共享数据和代码的时候，操作系统通过将不同的进程中适当的虚拟页面映射到相同的物理页面，
  从而安排多个进程共享这部分代码的一个拷贝。
- 简化存储器分配：当一个运行在用户进程中的程序要求额外的堆空间时，操作系统分配一个适当数字个连续的虚拟存储器页面，
  并将它们映射到物理存储器中任意位置的k个任意的物理页面。由于页表工作的方式，操作系统没有必要分配k个连续的物理存储器页面。页面可以随机地分散在物理存储器中。

9.5 虚拟存储器作为存储器保护地工具

因为每次CPU生成一个地址时，地址翻译硬件都会读一个PTE，所以通过在PTE上添加一些额外地许可位来控制虚拟页面内容的访问。

9.6 地址翻译

CPU中一个控制寄存器，页表基址寄存器指向当前页表。
n位的虚拟地址包含两个部分：一个p位的虚拟页面偏移(VPO)和一个(n-p)位的虚拟页号(VPN)。

存储器管理单元（MMU）利用VPN来选择适当的PTE.例如VPN0选择PTE0，VPN1选择PTE1。
将页表条目中的物理页号和虚拟地址中的VPO串联起来，就得到相应的物理地址。

执行步骤：

1. 处理器生成一个虚拟地址，并把它传递给MMU
2. MMU生成PTE地址，并从高速缓存/主存请求得到它
3. 高速缓存/主存向MMU返回PTE.
4. MMU构造物理地址，并把它传送给高速缓存或主存
5. 高速缓存/主存返回所请求的数据字给处理器

缺页：
123步同上述步骤
4. PTE中的有效位时零，所以MMU触发了一次异常，传递CPU中的控制到操作系统内核中的缺页异常处理程序。
5. 缺页处理程序确定出物理存储器中的牺牲页，如果该页面修改了，则换出到磁盘
6. 缺页处理程序页面调入新的页面，并更新存储器中PTE
7. 缺页处理程序返回到原来的进程，再次执行导致缺页的指令。

9.6.1 结合高速缓存和虚拟存储器

9.6.3 多级页表（后面再仔细看看）

9.8 存储器映射

linux通过将一个虚拟存储器区域与一个磁盘上对象关联起来，以初始化这个虚拟存储器区域的内容，这个过程称为存储器映射。

虚拟存储区域可以映射到的两种对象：

1. 文件系统中的普通文件：一个区域可以映射到一个普通磁盘文件的连续部分，例如一个可执行文件。如果区域比文件区要大，那么就用零来填充这个区域的余下部分。
2. 匿名文件：一个区域也可以映射到一个匿名文件。匿名文件由内核创建，包含的全部是二进制零。CPU第一次引用这样一个区域内的虚拟页面时，内核就在物理存储器中找到一个合适的页面，如果该页面被修改过，就将这个页面换出来，用二进制零覆盖牺牲页面并更新页表，将这个页面标记为时驻留在存储器中的。
   注意在磁盘和存储器之间并没有实际数据的传送。因此，映射到匿名文件的区域中的页面有时也叫请求二进制零的页。

9.8.1 再看共享对象

- 进程对共享对象所对应的区域的任何写操作，对那些也把这个共享对象映射到它们虚拟存储器的其他进程而言也是可见的。而且这些变化也会反映在磁盘上的原始对象中。
- 进程对一个映射到私有对象的区域做的改变，对于其他进程不可见，对着区域所作的任何写操作都不会反映在磁盘上的对象中。

共享区域：一个映射到共享对象的虚拟存储器区域叫做共享区域。

共享对象：略
私有对象：使用写时拷贝
一开始其他进程共享同一份物理拷贝，对于每个映射私有对象的进程，相应私有区域的页表条目标记为只读。
当有一个进程需要对私有对象的某个页面进行写的时候，会触发一个保护故障。它会在物理存储器中创建这个页面的新拷贝，并修改该进程对应页面的页表，并且其中的标记改为可写。

再看fork()函数

再看execve()函数

9.9 动态存储器分配