

`mmap` 函数要求内核创建一个新的虚拟内存区域，最好是从地址 `start` 开始的一个区域，并将文件描述符 `fd` 指定的对象的一个连续的片(chunk)映射到这个新的区域。连续的对象片大小为 `length` 字节，从距文件开始处偏移量为 `offset` 字节的地方开始。`start` 地址仅仅是一个暗示，通常被定义为 `NULL`。为了我们的目的，我们总是假设起始地址为 `NULL`。图 9-32 描述了这些参数的意义。

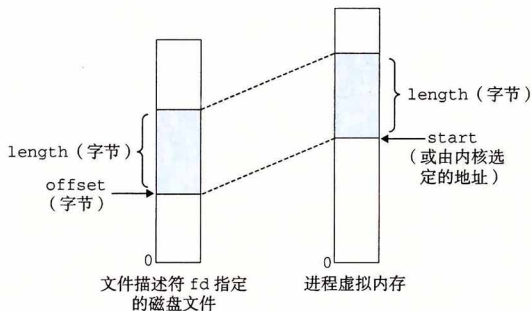


图 9-32 `mmap` 参数的可视化解释

参数 `prot` 包含描述新映射的虚拟内存区域的访问权限位(即在相应区域结构中的 `vm_prot` 位)。

- `PROT_EXEC`: 这个区域内的页面由可以被 CPU 执行的指令组成。
- `PROT_READ`: 这个区域内的页面可读。
- `PROT_WRITE`: 这个区域内的页面可写。
- `PROT_NONE`: 这个区域内的页面不能被访问。

参数 `flags` 由描述被映射对象类型的位组成。如果设置了 `MAP_ANON` 标记位，那么被映射的对象就是一个匿名对象，而相应的虚拟页面是请求二进制零的。`MAP_PRIVATE` 表示被映射的对象是一个私有的、写时复制的对象，而 `MAP_SHARED` 表示是一个共享对象。例如

```
bufp = Mmap(NULL, size, PROT_READ, MAP_PRIVATE|MAP_ANON, 0, 0);
```

让内核创建一个新的包含 `size` 字节的只读、私有、请求二进制零的虚拟内存区域。如果调用成功，那么 `bufp` 包含新区域的地址。


`munmap` 函数删除虚拟内存的区域：

```
#include <unistd.h>
#include <sys/mman.h>

int munmap(void *start, size_t length);
```

返回：若成功则为 0，若出错则为 -1。

`munmap` 函数删除从虚拟地址 `start` 开始的，由接下来 `length` 字节组成的区域。接下来对已删除区域的引用会导致段错误。

 **练习题 9.5** 编写一个 C 程序 `mmapcopy.c`，使用 `mmap` 将一个任意大小的磁盘文件复制到 `stdout`。输入文件的名字必须作为一个命令行参数来传递。