# RDMA

RDMA（Remote Direct Memory Access），通俗的说可以看成是远程的DMA技术，为了解决网络传输中服务器端数据处理的延迟而产生的。RDMA允许用户态的应用程序直接读取或写入远程内存，而无内核干预和内存拷贝发生。直接在内存和内存之间进行资料互传的技术，在数据传输的过程中，不需要和操作系统Kernel做沟通，完全实现了Kernel Bypass。

起初，只应用在高性能计算领域，最近，由于在大规模分布式系统和数据中心中网络瓶颈越来越突出，逐渐走进越来越多人的视野。

RDMA那么好，但是为什么没有普及呢？个人认为，有两方面原因。

1. 因为价格太贵；
2. 其缺乏像TCP/IP协议的灵活性和更高层的抽象。

在使用RDMA进行文件传输的时候可以设置传输块的大小，比如我们为了让文件传输的更快，可以将传输块设置的很大，由于RDMA可以通过特殊的网卡将文件数据从一台机器直接发送到另一台机器，不需要唤起操作系统，也不需要从内核态向用户态拷贝，所以传输的速度非常快。但是由于设置的块很大，对于小文件来说，这样的开销又很大。所以我设计的文件传输程序分布两部分，一部分是利用RDMA的c API开发的服务端和客户端，专门用来传输大文件，另一部分是使用Python的FTP模块，对目录和小文件进行传输。并且这样做可以同时利用RDMA网卡和以太网网卡。

# SIMFS

传统文件系统，比如ext2到ext4，NTFS等等一些基于磁盘存储的文件系统，在进行文件读写时，首先通过VFS（虚拟文件系统）找到文件对应的驱动程序，然后通过读取磁盘的超级块等信息找到文件在磁盘中具体的位置。接下来，文件在磁盘中的数据要copy到内存中（后面还有cache和寄存器）。

这样一层一层的数据copy让文件读写效率变得很低。于是有了基于内存的文件系统。传统基于内存的文件系统，比如PMFS、NOVA、Ext4-dax等，他们都是利用mmap函数，通过管理文件mmap结构体维护文件系统，用户程序读写内存文件系统中的文件时，还是要利用VFS，调用内存文件系统的驱动程序，把文件数据从用户态拷贝到内核态。

SIMFS是参考的内存页表进行设计的，将文件按照内存4级页表的方式进行管理，文件页表和内存页表的结构完全一致，所以在读取文件内容的时候，利用计算机本身的MMU硬件来读取文件页表，实现非常快速的文件访问。通过这种设计方法，SIMFS文件系统可以达到接近内存的带宽。

# 参考

https://blog.csdn.net/u011459120/article/details/78469098