Daniel Murga

1. When using the read write method, the smaller the buffer size, the more calls are needed to read and write the entire file because the buffer can only hold a small amount of data before it is full. Since more read and write system calls must be used when the buffer size is small, even though they may be fast operations, this leads to many unnecessary and time consuming operations. Once the buffer size is large enough to dismiss the amount of read and write calls, the bottleneck becomes I/O from the disk. Since both read and write are used when copying the file in the read-write method, the data is copied once to the user space then copied again back to the kernel space when writing to the destination, whereas the mmap method only maps the data once to the user process' logical address space and then pointed to by the destination pointer.
2. When supplying a buffer size for the read and write operations, if the buffer does not divide the size of the source file evenly, the last bytes to be copied will not fill the array completely and will contain incorrect or empty data. The newly copied file will be a multiple of the buffer size, while the mmap file will be exactly the same amount of data that was previously mapped from the source file. If the code were to check for the end of the source file and only perform the read system call on relevant data without exceeding the end address, the file sizes would be the same.