In order for a program to communicate over the network it needs to create a socket. The socket is created using the sys\_connect() system call and takes as parameters the remote address, and both the local and remote ports needed in order to communicate over the network. The connect\_system() call then returns a file descriptor that represents the created socket in the program. All active sockets are maintained in a singly linked list, this list is used to find the appropriate socket object associated with a particular file descriptor when handling system calls such as read() or write().

When the E1000 receives a packet from the network it creates an interrupt and notifies the kernel about the new packet. The network driver interrupt handler then reads the received packet and queues it in a kernel space buffer container. When the user program issues a read call, it passes the socket’s file descriptor and a container buffer, the kernel then checks if the passed socket has any received packets waiting in its queue. If there are no packets in the socket's queue the kernel sleeps the program until the socket associated with the passed file descriptor receives a packet.

If there are packets in the socket's queue the kernel reads the packets one by one and copies the data from the kernel space buffer to the program buffer that was passed in the read() call and updates the que. The program can then process the data from the buffer as needed.

When writing to a socket the program prepares a buffer containing the payload data as well as any information for the packet headers. The program then issues a sys\_write() system call, passing as an argument the file descriptor of the socket to be written to, as well as the address of the buffer containing the data the program needs to send.

The sys\_write() function retrieves the data from the program buffer and copies the packet's payload and header information to a new buffer inside the kernel space. This buffer is then added to the transmit queue of the E1000 device using e1000\_transmit(). The E1000 device takes the data payload and sends it over the network to the appropriate remote address and port number.