1. 网络编程通常涉及大量 I/O 操作，这些操作会阻塞线程并导致 CPU 空闲。通过多线程可以通过重叠等待时间提升效率。例如当一个线程因 I/O 阻塞时，其他线程可以继续使用 CPU 处理任务，从而提高整体吞吐量。此外，还可借助时间片轮转实现并发来更高效地复用 TCP 连接等资源。

2.（1）socket()不会阻塞。socket()仅创建套接字并分配文件描述符，不涉及任何网络或 I/O 操作。即使在内存耗尽的情况下也不会阻塞进程。

（2）listen()不会阻塞。listen()仅设置套接字为监听状态，并指定连接请求队列的最大长度。它本身不等待连接，只是配置套接字属性。

（3）connect()可能阻塞，因为 connect()会等待三次握手完成。若对端未响应或网络延迟高，可能长时间阻塞。

（4）accept()可能阻塞。在阻塞模式下，accept()会一直等待，直到队列中有新的连接到达。

（5）read()可能阻塞。在阻塞模式下，read()会等待数据到达或连接关闭。

（6）write()可能阻塞。在阻塞模式下，write()会等待缓冲区有足够空间写入数据。