Sub Report – details of server

by GaoZhen
no 3150104390

1. The concept of IOCP(Completion port)

For the reason that we decide to design a Online chatting room, how to design a robust server to carry more clients' requests is really a serious problem. As the team member who suppose to code the part of server, I decide to use IOCP(Completion port) to fulfill these requirements.

First of all, the reason why it is called "completion" port is that the system will notify us after the network I/O operation is "completed". That is to say, when we receive the notification from the system, the network operation has actually been completed, that is, when the system notifies us, it is not that there is data coming from the network. Instead, the data from the network has been received; or the client's access request has been connected to the system, and so on. We just need to deal with the following things.

Generally speaking, using the completion port only follows the following steps:

1. Call the CreateIoCompletionPort () function to create a completion port.

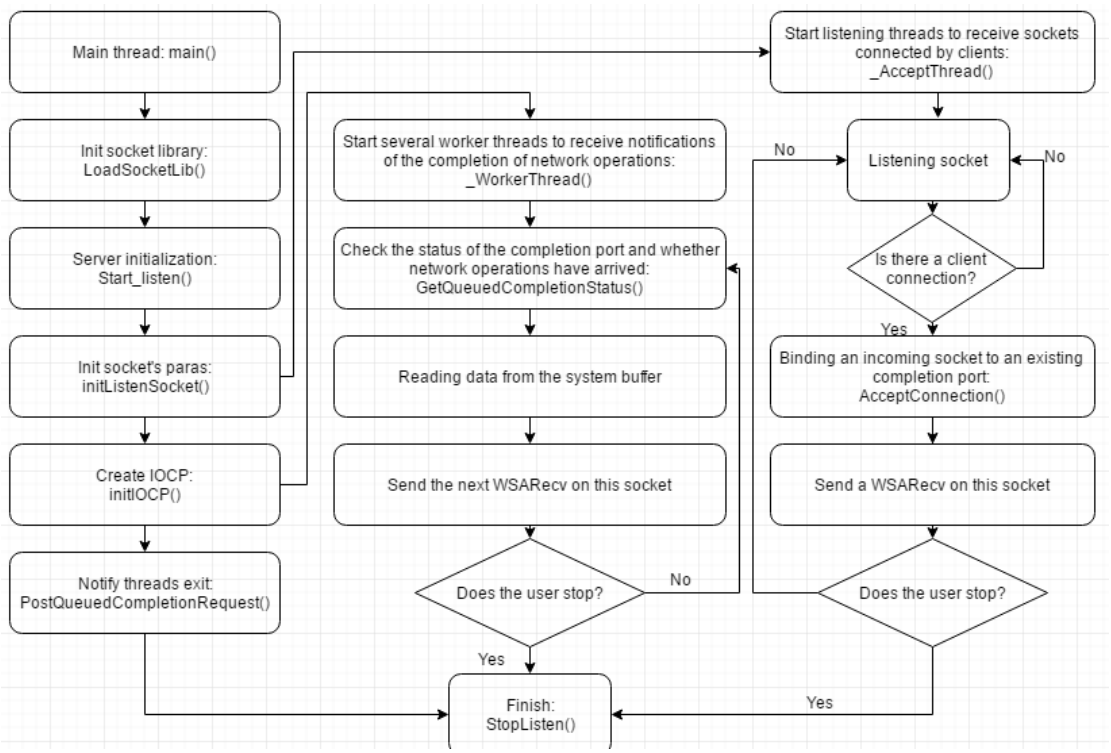
2. According to how many processors are in the system, how many worker threads are created.

3. Here's how to receive the connected Socket connection. There are two ways to implement it: first, like other programming models, you need to start a separate thread for accept client connection requests, and second, asynchronous AcceptEx () requests with better performance.

4. Whenever there is a client connection, we still have to call the CreateIoCompletionPort () function. Instead of creating a new completion port, we bind the newly connected Socket (the so-called device handle) to the current completion port.

5. For example, after the client is connected, we can submit a network request on this Socket, such as WSARecv (), and then the system will help us to perform the operation of receiving data obediently. We can rest assured that we can do other things.

6. At this point, the worker threads we prepared in advance can't be idle. The worker we built before will be busy. We need to call the GetQueuedCompletionStatus () function separately to see if there are network communication requests in the queue of the scan completion port (such as reading data, sending data, etc.). Once there is, take the request back from the completion port queue, continue to execute the processing code in this thread, after processing, we continue to deliver the next network communication request, such a cycle.



IOCP flowchart

2. Test in Online chatting room

I try to test the server part in order to check its capacity. In my several test cases, I create 20,000 clients to send request at the same time. Only 3.82% of the CPU is occupied by Server program with completion port, and the peak value of the whole operation process is not more than 4%.

On the contrary, Client programs with multiple concurrent threads occupy 11.53% of the CPU, even several times more than Server programs...

3. Summary

Unfortunately, we finally choose other method to design server, but I upload the detail of IOCP on Github.

IOCP is really a well-designed model of server, in my opinion. If we successfully using it in our Online chatting room program, I believe we can get a better performance.