一、网络协议

1、tcp/ip

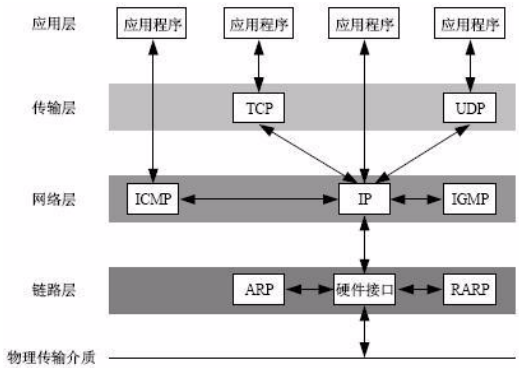
1.2概念

TCP/IP（Transmission Control Protocol/Internet Protocol）是一种可靠的网络数据传输控制协议。定义了主机如何连入因特网以及数据如何在他们之间传输的标准。

TCP/IP协议参考模型把所有TCP/IP系列协议归类到四个抽象层中；

每一个抽象层建立在低一层提供的服务上，并且为高一层提供服务

网络中的五层架构：

	**ICMP:**控制报文协议**IGMP:internet**组管理协议**ARP:**地址解析协议**RARP:**反向地址转化协议

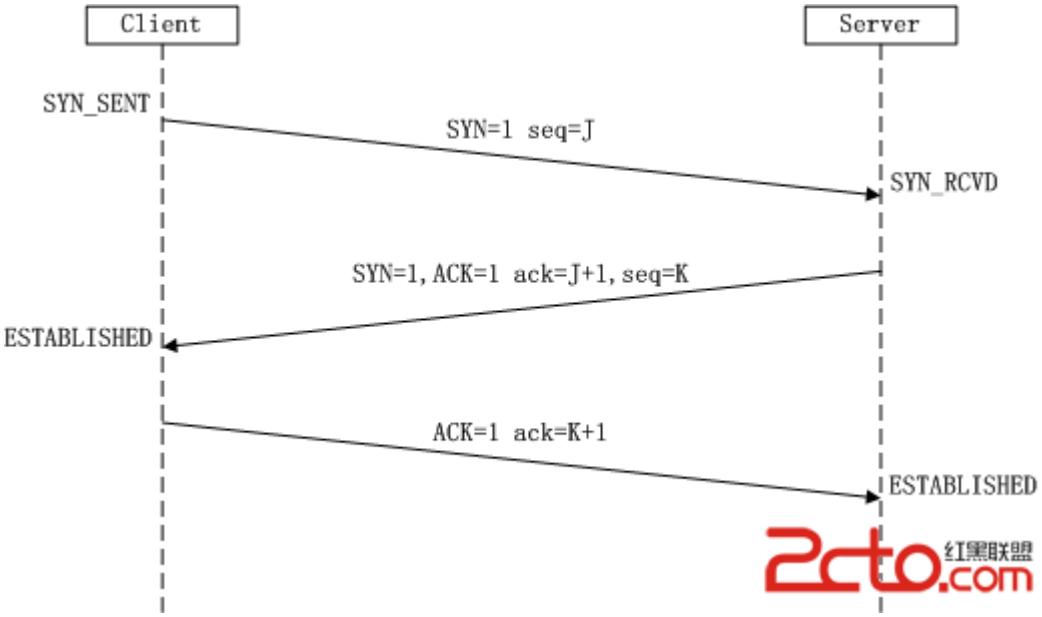
OIS模型：

OSI模型（开放式系统互联通信参考模型），它是由国际标准化组织提出的，试图使各种计算机在世界范围内互联为网络的标准框架

OSI模型多了表达层、会话层

1.2三次握手

所谓三次握手（Three-Way Handshake）即建立TCP连接，就是指建立一个TCP连接时，需要客户端和服务端总共发送3个包以确认连接的建立



位码即tcp标志位,有6种标示:SYN(synchronous建立联机) ACK(acknowledgement 确认) PSH(push传送) FIN(finish结束) RST(reset重置) URG(urgent紧急)

Sequence number(顺序号码) Acknowledge number(确认号码)

第一次握手：主机A发送位码为syn＝1,随机产生seq number=1234567的数据包到服务器，主机B由SYN=1知道，A要求建立联机；

第二次握手：主机B收到请求后要确认联机信息，向A发送ack number=(主机A的seq+1),syn=1,ack=1,随机产生seq=7654321的包

第三次握手：主机A收到后检查ack number是否正确，即第一次发送的seq number+1,以及位码ack是否为1，若正确，主机A会再发送ack number=(主机B的seq+1),ack=1，主机B收到后确认seq值与ack=1则连接建立成功。

完成三次握手，主机A与主机B开始传送数据。

在TCP/IP协议中，TCP协议提供可靠的连接服务，采用三次握手建立一个连接。 第一次握手：建立连接时，客户端发送syn包(syn=j)到服务器，并进入SYN_SEND状态，等待服务器确认； 第二次握手：服务器收到syn包，必须确认客户的SYN（ack=j+1），同时自己也发送一个SYN包（syn=k），即SYN+ACK包，此时服务器进入SYN_RECV状态；

第三次握手：客户端收到服务器的SYN＋ACK包，向服务器发送确认包ACK(ack=k+1)，此包发送完毕，客户端和服务器进入ESTABLISHED状态，完成三次握手。完成三次握手，客户端与服务器开始传送数据.

实例:

IP 192.168.1.116.3337 > 192.168.1.123.7788: SYN 3626544836:3626544836

IP 192.168.1.123.7788 > 192.168.1.116.3337: SYN 1739326486:1739326486 ack 3626544837

IP 192.168.1.116.3337 > 192.168.1.123.7788: ack 1739326487,ack 1

第一次握手：192.168.1.116发送位码syn＝1,随机产生seq number=3626544836的数据包到192.168.1.123,192.168.1.123由SYN=1知道192.168.1.116要求建立联机;

第二次握手：192.168.1.123收到请求后要确认联机信息，向192.168.1.116发送ack number=3626544837,syn=1,ack=1,随机产生seq=1739326486的包;

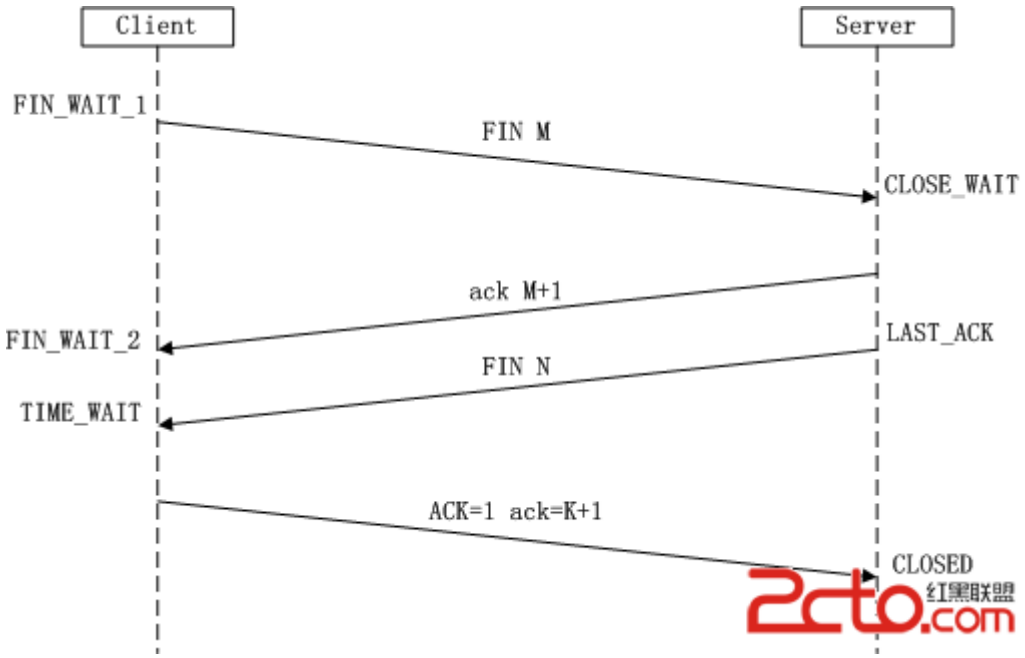
第三次握手：192.168.1.116收到后检查ack number是否正确，即第一次发送的seq number+1,以及位码ack是否为1，若正确，192.168.1.116会再发送ack number=1739326487,ack=1，192.168.1.123收到后确认seq=seq+1,ack=1则连接建立成功。

总结：

（1）第一次握手：Client将标志位SYN置为1，随机产生一个值seq=J，并将该数据包发送给Server，Client进入SYN_SENT状态，等待Server确认。``（2）第二次握手：Server收到数据包后由标志位SYN=1知道Client请求建立连接，Server将标志位SYN和ACK都置为1，ack=J+1，随机产生一个值seq=K，``并将该数据包发送给Client以确认连接请求，Server进入SYN_RCVD状态。``（3）第三次握手：Client收到确认后，检查ack是否为J+1，ACK是否为1，如果正确则将标志位ACK置为1，ack=K+1，并将该数据包发送给Server，``Server检查ack是否为K+1，ACK是否为1，如果正确则连接建立成功，Client和Server进入ESTABLISHED状态，完成三次握手，随后Client与Server之间可以开始传输数据了。``SYN攻击：`` 在三次握手过程中，Server发送SYN-ACK之后，收到Client的ACK之前的TCP连接称为半连接（half-open connect），此时Server处于SYN_RCVD状态，当收到ACK后，``Server转入ESTABLISHED状态。SYN攻击就是Client在短时间内伪造大量不存在的IP地址，并向Server不断地发送SYN包，Server回复确认包，并等待Client的确认，``由于源地址是不存在的，因此，Server需要不断重发直至超时，这些伪造的SYN包将产时间占用未连接队列，导致正常的SYN请求因为队列满而被丢弃，从而引起网络堵``塞甚至系统瘫痪。SYN攻击时一种典型的DDOS攻击，检测SYN攻击的方式非常简单，即当Server上有大量半连接状态且源IP地址是随机的，则可以断定遭到SYN攻击了，``使用如下命令可以让之现行：`` #netstat -nap | grep SYN_RECV

1.3四次挥手

三次握手耳熟能详，四次挥手估计就听得比较少了，所谓四次挥手（Four-Way Wavehand）即终止TCP连接，就是指断开一个TCP连接时，需要客户端和服务端总共发送4个包以确认连接的断开



由于TCP连接时全双工的，因此，每个方向都必须要单独进行关闭，这一原则是当一方完成数据发送任务后，发送一个FIN来终止这一方向的连接，收到一个FIN只是意味着这一方向上没有数据流动了，即不会再收到数据了，但是在这个TCP连接上仍然能够发送数据，直到这一方向也发送了FIN。首先进行关闭的一方将执行主动关闭，而另一方则执行被动关闭，上图描述的即是如此。

（1）第一次挥手：Client发送一个FIN，用来关闭Client到Server的数据传送，Client进入FIN_WAIT_1状态。

（2）第二次挥手：Server收到FIN后，发送一个ACK给Client，确认序号为收到序号+1（与SYN相同，一个FIN占用一个序号），Server进入CLOSE_WAIT状态。

（3）第三次挥手：Server发送一个FIN，用来关闭Server到Client的数据传送，Server进入LAST_ACK状态。

（4）第四次挥手：Client收到FIN后，Client进入TIME_WAIT状态，接着发送一个ACK给Server，确认序号为收到序号+1，Server进入CLOSED状态，完成四次挥手。

2、tcp与udp

TCP是一种面向连接的保证可靠传输的协议。通过TCP协议传输，得到的是一个顺序的无差错的数据流。发送方和接收方的成对的两个socket之间必须建立连接，以便在TCP协议的基础上进行通信，当一个socket（通常都是server socket）等待建立连接时，另一个socket可以要求进行连接，一旦这两个socket连接起来，它们就可以进行双向数据传输，双方都可以进行发送或接收操作。

UDP是一种面向无连接的协议，每个数据报都是一个独立的信息，包括完整的源地址或目的地址，它在网络上以任何可能的路径传往目的地，因此能否到达目的地，到达目的地的时间以及内容的正确性都是不能被保证的。

TCP与UDP区别：

TCP特点：

1、TCP是面向连接的协议，通过三次握手建立连接，通讯完成时要拆除连接，由于TCP是面向连接协议，所以只能用于点对点的通讯。而且建立连接也需要消耗时间和开销。

2、TCP传输数据无大小限制，进行大数据传输。

3、TCP是一个可靠的协议，它能保证接收方能够完整正确地接收到发送方发送的全部数据。

UDP特点：

1、UDP是面向无连接的通讯协议，UDP数据包括目的端口号和源端口号信息，由于通讯不需要连接，所以可以实现广播发送。

2、UDP传输数据时有大小限制，每个被传输的数据报必须限定在64KB之内。

3、UDP是一个不可靠的协议，发送方所发送的数据报并不一定以相同的次序到达接收方。

TCP与UDP应用：

1、TCP在网络通信上有极强的生命力，例如远程连接（Telnet）和文件传输（FTP）都需要不定长度的数据被可靠地传输。但是可靠的传输是要付出代价的，对数据内容正确性的检验必然占用计算机的处理时间和网络的带宽，因此TCP传输的效率不如UDP高。

2，UDP操作简单，而且仅需要较少的监护，因此通常用于局域网高可靠性的分散系统中client/server应用程序。例如视频会议系统，并不要求音频视频数据绝对的正确，只要保证连贯性就可以了，这种情况下显然使用UDP会更合理一些。

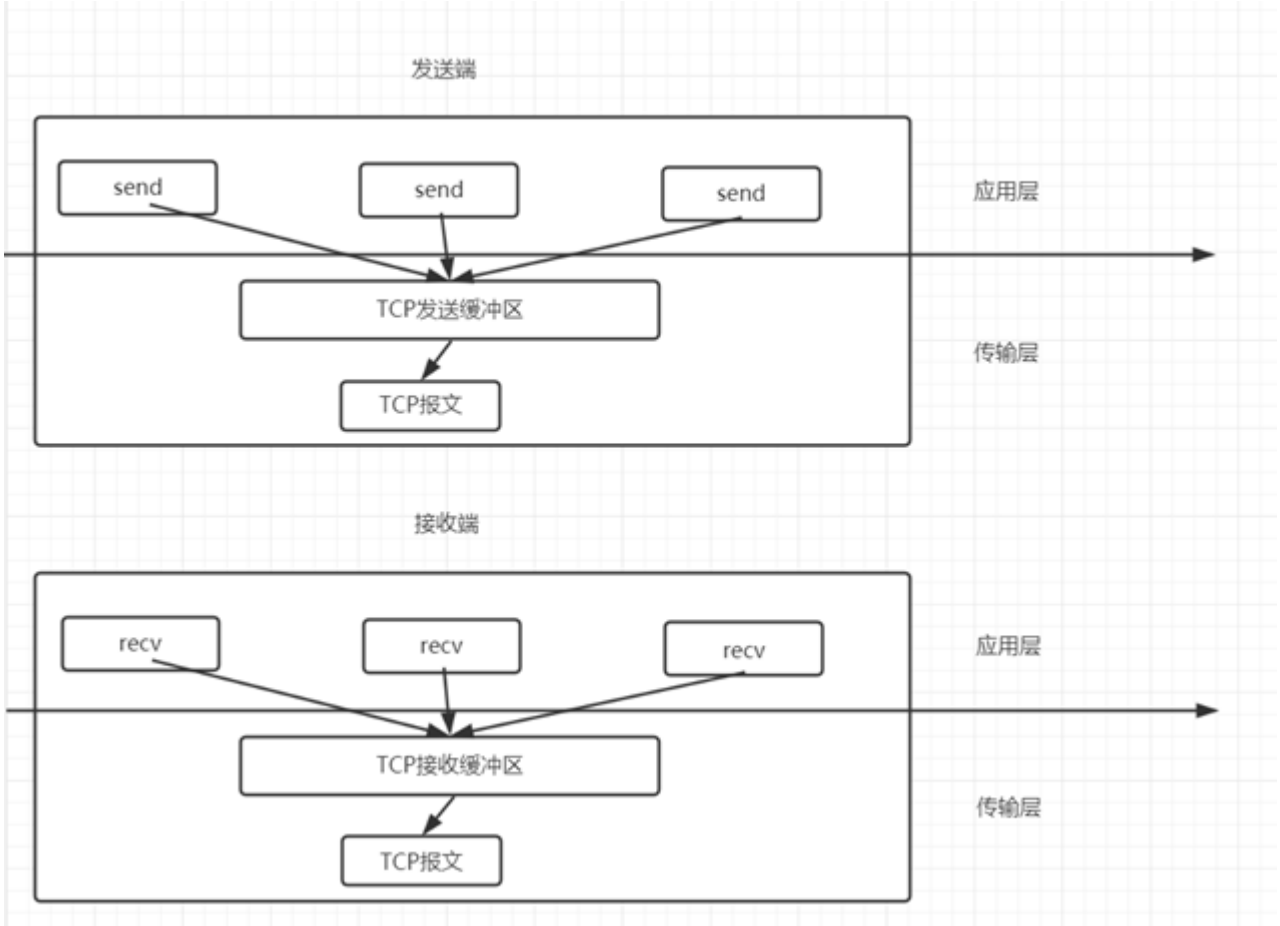
3、tcp通信原理

首先，对于TCP通信来说，每个TCP Socket的内核中都有一个发送缓冲区和一个接收缓冲区，TCP的全双工的工作模式及TCP的滑动窗口就是依赖于这两个独立的Buffer和该Buffer的填充状态。

接收缓冲区把数据缓存到内核，若应用进程一直没有调用Socket的read方法进行读取，那么该数据会一直被缓存在接收缓冲区内。不管进程是否读取Socket，对端发来的数据都会经过内核接收并缓存到Socket的内核接收缓冲区。

read所要做的工作，就是把内核接收缓冲区中的数据复制到应用层用户的Buffer里。

进程调用Socket的send发送数据的时候，一般情况下是讲数据从应用层用户的Buffer里复制到Socket的内核发送缓冲区，然后send就会在上层返回。换句话说，send返回时，数据不一定会被发送到对端。

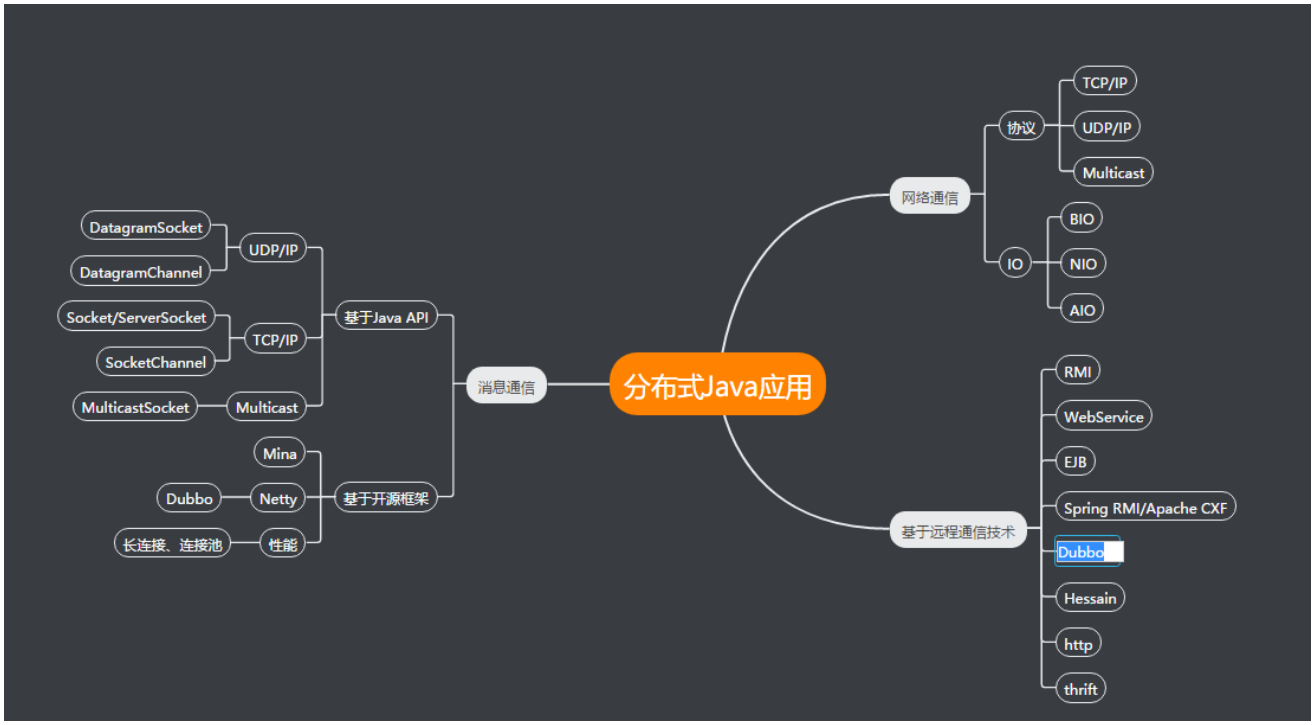


4、滑动窗口

发送方和接收方都会维护一个数据帧的序列，这个序列被称作窗口。发送方的窗口大小由接收方确认，目的是控制发送速度，以免接收方的缓存不够大导致溢出，同时控制流量也可以避免网络拥塞。

下面图中的4,5,6号数据帧已经被发送出去，但是未收到关联的ACK，7,8,9帧则是等待发送。可以看出发送端的窗口大小为6，这是由接受端告知的（事实上必须考虑拥塞窗口cwnd，这里暂且考虑cwnd>rwnd）。此时如果发送端收到4号ACK，则窗口的左边缘向右收缩，窗口的右边缘则向右扩展，此时窗口就向前“滑动了”，即数据帧10也可以被发送

5、通信技术整理



二、晦涩难懂的非阻塞

明白了Socket读写数据的底层原理，我们就很容易理解“阻塞模式”：对于读取Socket数据的过程而言，如果接收缓冲区为空，则调用Socket的read方法的线程会阻塞，直到有数据进入接收缓冲区；而对于写数据到Socket中的线程来说，如果待发送的数据长度大于发送缓冲区空余长度，则会阻塞在write方法上，等待发送缓冲区的报文被发送到网络上，然后继续发送下一段数据，循环上述过程直到数据都被写入到发送缓冲区为止

从前面分析的过程来看，传统的Socket阻塞模式直接导致每个Socket都必须绑定一个线程来操作数据，参与通信的任意一方如果处理数据的速度较慢，会直接拖累到另一方，导致另一方的线程不得不浪费大量的时间在I/O等待上，所以这就是Socket阻塞模式的“缺陷”。但是这种模式在少量的TCP连接通信的情况下，双方都可以快速的传输数据，这个时候的性能是最高的。

三、Multcast

四、代码实现