在介绍scoket之前先来看了解一下网络协议方面的知识

TCP/IP OSI

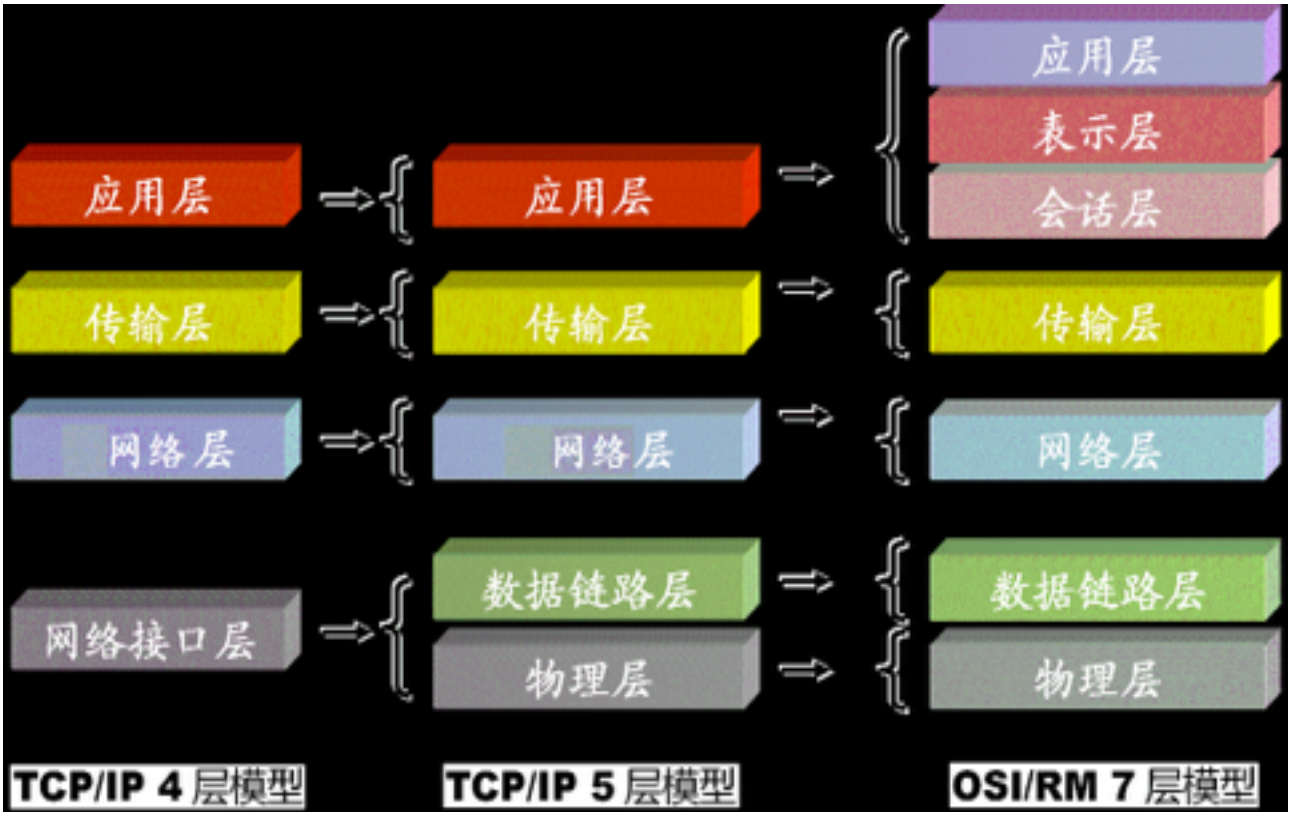
要想理解socket首先得熟悉一下TCP/IP协议族， TCP/IP（Transmission Control Protocol/Internet Protocol）即传输控制协议/网间协议，定义了主机如何连入因特网及数据如何再它们之间传输的标准，

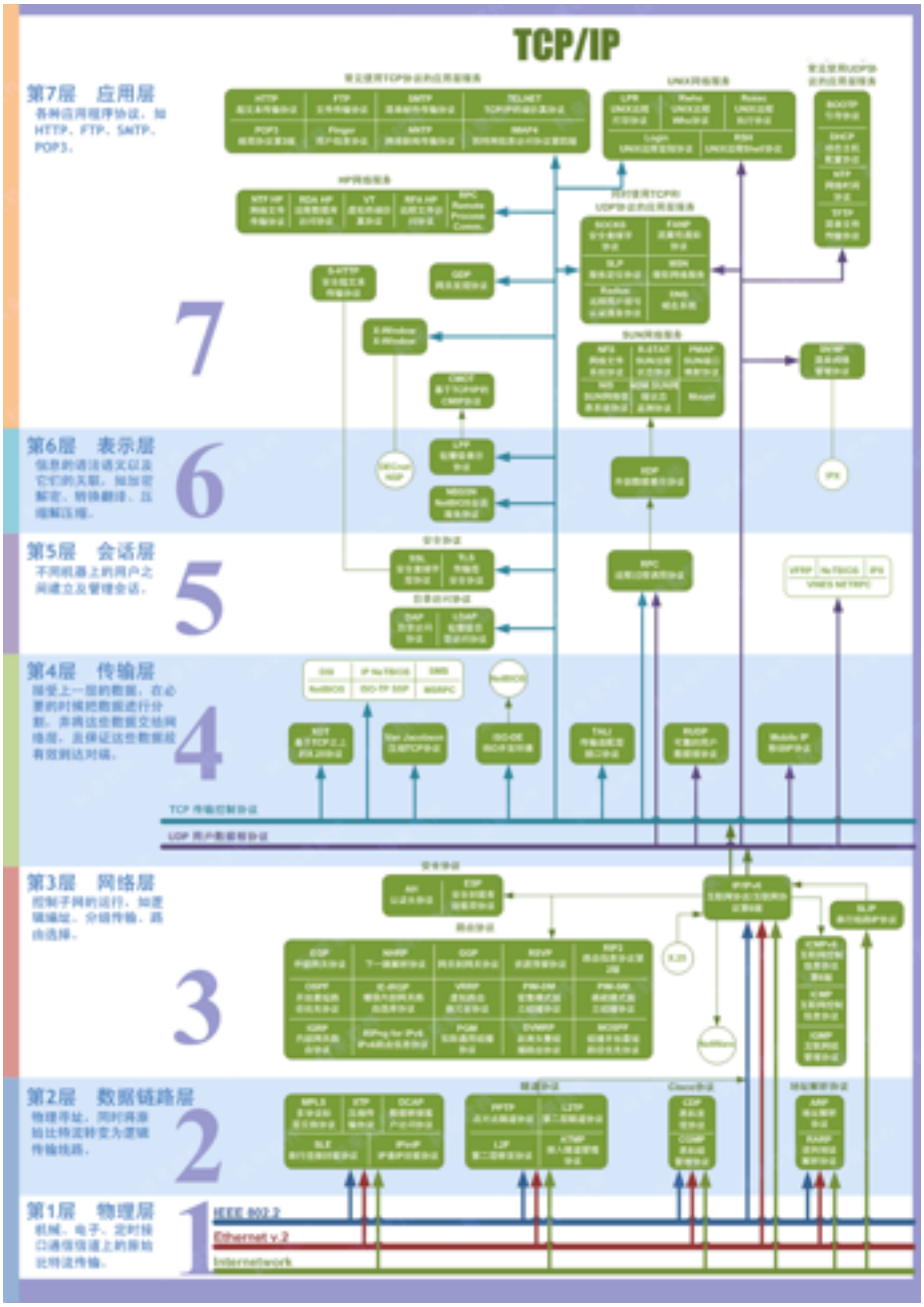
从字面意思来看TCP/IP是TCP和IP协议的合称，但实际上TCP/IP协议是指因特网整个TCP/IP协议族。不同于ISO模型的七个分层，TCP/IP协议参考模型把所有的TCP/IP系列协议归类到四个抽象层中（也有说5层）

OSI/RM（Open System Interconnection/Reference Model） **开放系统互连参考模型**

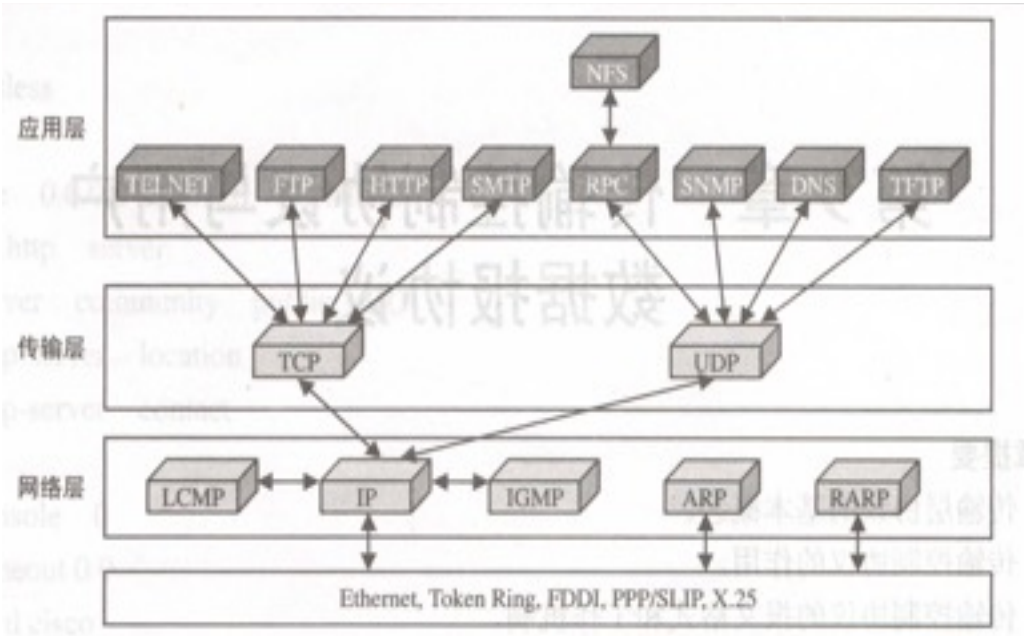
OSI 七层模型通过七个层次化的结构模型使不同的系统不同的网络之间实现可靠的通讯，因此其最主要的功能就是帮助不同类型的主机实现数据传输 。

TCP/IP模型和OSI七层模型的对比如下图：





每一层的协议分布如下图：



必须明确的是，TCP/IP不是一个单独的协议，而是一个协议栈，或者说是多个子协议的集合

TCP/UDP

传输层有两种不同的传输协议：即面向连接的 传输控制协议TCP (Transmission Control Protocol)和无连接的 用户数据报协议UDP(User Datagram Protocol)。

TCP(传输控制协议)：

1)提供IP环境下的数据可靠传输(一台计算机发出的字节流会无差错的发往网络上的其他计算机，而且计算机A接收数据包的时候，也会向计算机B回发数据包，这也会产生部分通信量)，有效流控，全双工操作(数据在两个方向上能同时传递)，多路复用服务，是面向连接，端到端的传输;TCP提供超时重发，丢弃重复数据，检验数据，流量控制等功能，保证数据能从一端传到另一端。
2)面向连接：正式通信前必须要与对方建立连接。事先为所发送的数据开辟出连接好的通道，然后再进行数据发送，像打电话。
3)TCP支持的应用协议：Telnet(远程登录)、FTP(文件传输协议)、SMTP(简单邮件传输协议)。TCP用于传输数据量大，可靠性要求高的应用。

UDP(用户数据报协议，User Data Protocol)

1)面向非连接的(正式通信前不必与对方建立连接，不管对方状态就直接发送，像短信，QQ)，不能提供可靠性、流控、差错恢复功能。UDP用于一次只传送少量数据，可靠性要求低、传输经济等应用。
2) UDP支持的应用协议：NFS(网络文件系统)、SNMP(简单网络管理系统)、DNS(主域名称系统)、TFTP(通用文件传输协议)等。

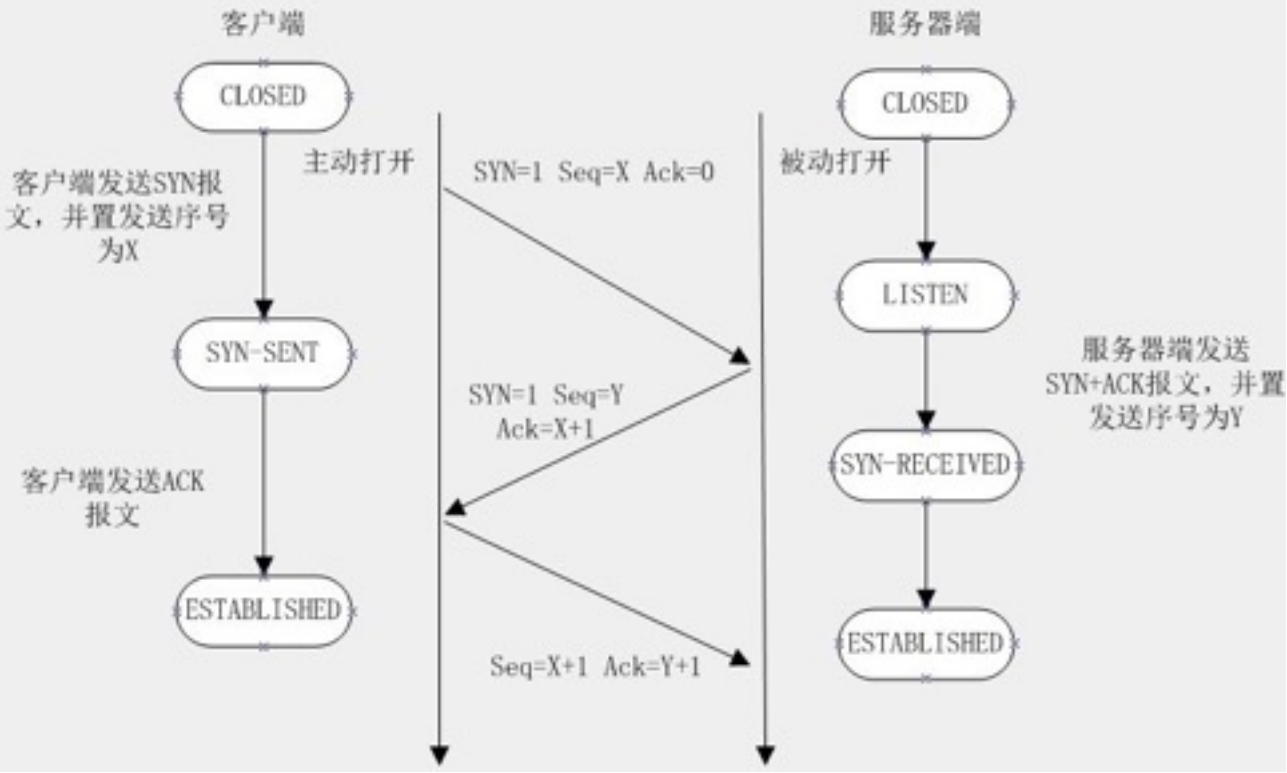
总结：

TCP：面向连接、传输可靠(保证数据正确性,保证数据顺序)、用于传输大量数据(流模式)、速度慢，建立连接需要开销较多(时间，系统资源)。
UDP：面向非连接、传输不可靠、用于传输少量数据(数据包模式)、速度快。

TCP与UDP在socket编程中的区别
http://blog.chinaunix.net/uid-26421509-id-3814684.html

TCP连接的三次握手

第一次握手：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。
握手过程中传送的包里不包含数据，三次握手完毕后，客户端与服务器才正式开始传送数据。理想状态下，TCP连接一旦建立，在通信双方中的任何一方主动关闭连接之前，TCP 连接都将被一直保持下去。断开连接时服务器和客户端均可以主动发起断开TCP连接的请求，断开过程需要经过“四次握手”（过程就不细写了，就是服务器和客户端交互，最终确定断开）



接下来详细介绍一下什么是scoket,以及scoket的工作原理

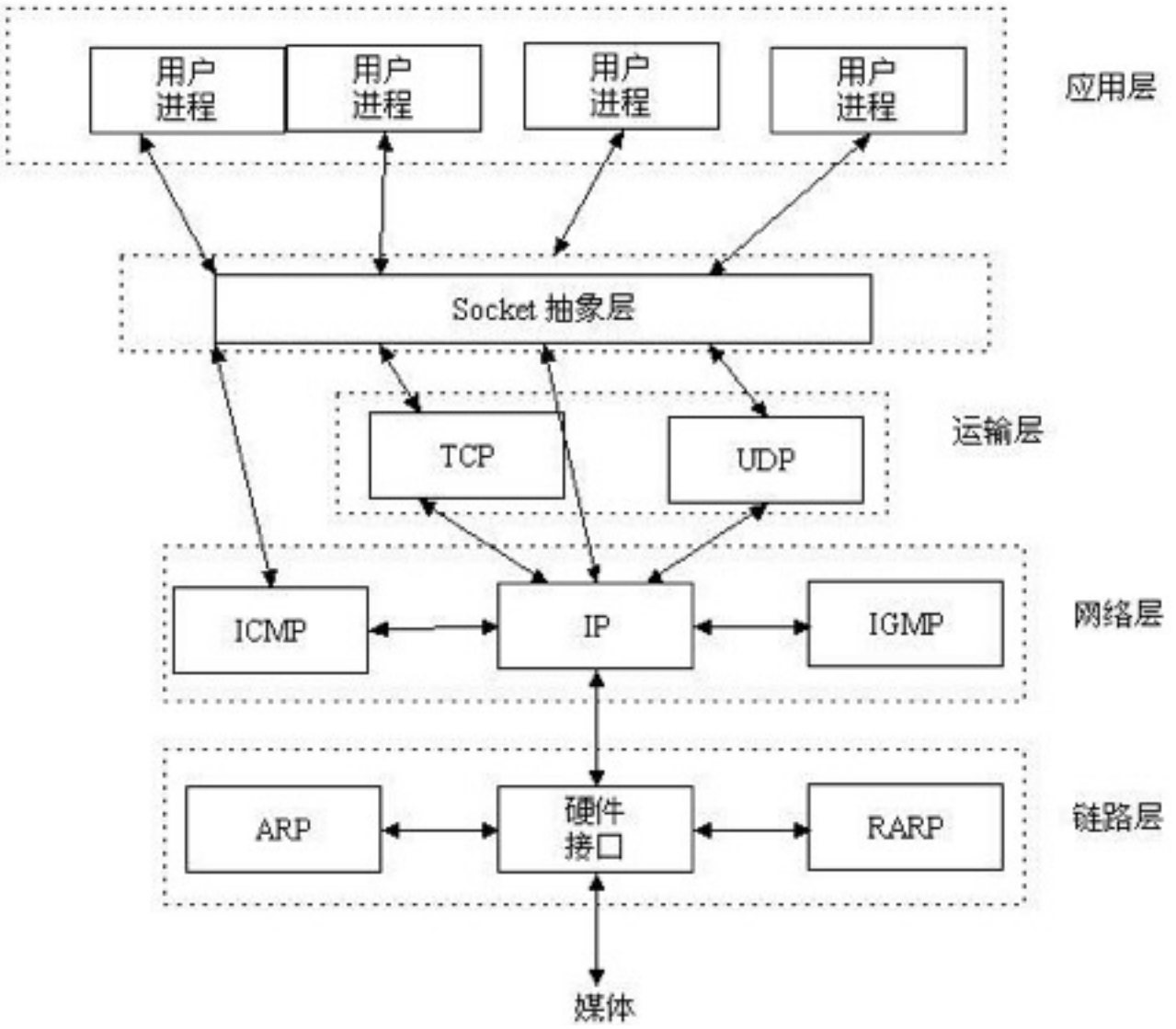
网络进程间如何通讯?

本地的进程间通信（IPC）有很多种方式，但可以总结为下面4类：
1、消息传递（管道、FIFO、消息队列）
2、同步（互斥量、条件变量、读写锁、文件和写记录锁、信号量）
3、共享内存（匿名的和具名的）
4、远程过程调用（Solaris门和Sun RPC）

网络层的“**ip地址**”可以唯一标识网络中的主机，而传输层的“**协议+端口**”可以唯一标识主机中的应用程序（进程）。这样利用三元组（ip地址，协议，端口）就可以标识网络的进程了，网络中的进程通信就可以利用这个标志与其它进程进行交互。
使用TCP/IP协议的应用程序通常采用应用编程接口：UNIX BSD的套接字（socket）和UNIX System V的TLI（已经被淘汰），来实现网络进程之间的通信。就目前而言，几乎所有的应用程序都是采用socket，而现在又是网络时代，网络中进程通信是无处不在，这就是我为什么说“一切皆socket”。

什么是scoket?

我们经常把socket翻译为套接字，socket是在应用层和传输层之间的一个抽象层，它把TCP/IP层复杂的操作抽象为几个简单的接口供应用层调用已实现进程在网络中通信。Socket是相当于发动机，提供了网络通信的能力。



套接字

多个TCP连接或多个应用程序进程可能需要通过同一个TCP协议端口传输数据。为了区别不同的应用程序进程和连接，许多计算机操作系统为应用程序与TCP/IP协议交互提供了称为套接字(Socket)的接口。

常用的TCP/IP协议的3种套接字类型：

流套接字（SOCK_STREAM）：

流套接字用于提供面向连接、可靠的数据传输服务。该服务将保证数据能够实现无差错、无重复发送，并按顺序接收。

流套接字之所以能够实现可靠的数据服务，原因在于其使用了传输控制协议，即TCP（The Transmission Control Protocol）协议。

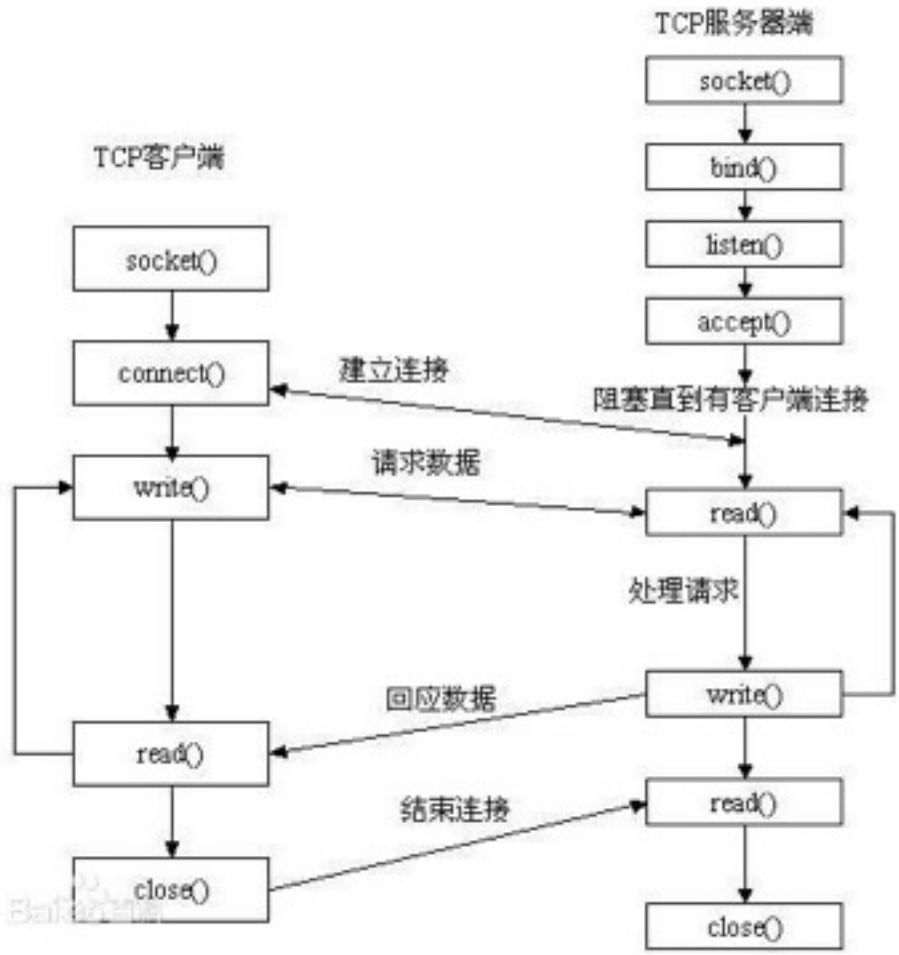
数据包套接字（SOCK_DGRAM）：

数据包套接字提供了一种无连接的服务。该服务并不能保证数据传输的可靠性，数据有可能在传输过程中丢失或出现数据重复，且无法保证顺序地接收到数据。
数据包套接字使用UDP（User Datagram Protocol）协议进行数据的传输。由于数据包套接字不能保证数据传输的可靠性，对于有可能出现的数据丢失情况，需要在程序中做相应的处理。
原始套接字（SOCK_RAW）：
原始套接字与标准套接字（标准套接字指的是前面介绍的流套接字和数据包套接字）的区别在于：
原始套接字可以读写内核没有处理的IP数据包，
而流套接字只能读取TCP协议的数据，数据包套接字只能读取UDP协议的数据。因此，如果要访问其他协议发送数据必须使用原始套接字。
scoket套接字 http://blog.csdn.net/hguisu/article/details/7445768/
套接字和scoket编程http://blog.csdn.net/guihaijinfen/article/details/8446128

Socket是一个针对TCP和UDP编程的接口，你可以借助它建立TCP连接。Socket是对TCP/IP协议的封装，Socket本身并不是协议，而是一个调用接口（API），通过Socket，我们才能使用TCP/IP协议。在设计模式中，Socket其实就是一个门面模式，它把复杂的TCP/IP协议族隐藏在Socket接口后面，Socket的出现只是使得程序员更方便地使用TCP/IP协议栈而已，是对TCP/IP协议的抽象，从而形成了我们知道的一些最基本的函数接口。对用户来说，一组简单的接口就是全部，让Socket去组织数据，以符合指定的协议。

通过下面一个图来看一下客户端和服务器端通讯的流程：

建立Socket连接至少需要一对套接字，其中一个运行于客户端，称为ClientSocket ，另一个运行于服务器端，称为ServerSocket 。
套接字之间的连接过程分为三个步骤：服务器监听，客户端请求，连接确认。
1。服务器监听：服务器端套接字并不定位具体的客户端套接字，而是处于等待连接的状态，实时监控网络状态，等待客户端的连接请求。
2。客户端请求：指客户端的套接字提出连接请求，要连接的目标是服务器端的套接字。为此，客户端的套接字必须首先描述它要连接的服务器的套接字，指出服务器端套接字的地址和端口号，然后就向服务器端套接字提出连接请求。
3。连接确认：当服务器端套接字监听到或者说接收到客户端套接字的连接请求时，就响应客户端套接字的请求，建立一个新的线程，把服务器端套接字的描述发给客户端，一旦客户端确认了此描述，双方就正式建立连接。而服务器端套接字继续处于监听状态，继续接收其他客户端套接字的连接请求。



先从服务器端说起。服务器端先初始化Socket，然后与端口绑定(bind)，对端口进行监听(listen)，调用accept阻塞，等待客户端连接。在这时如果有个客户端初始化一个Socket，然后连接服务器(connect)，如果连接成功，这时客户端与服务器端的连接就建立了。客户端发送数据请求，服务器端接收请求并处理请求，然后把回应数据发送给客户端，客户端读取数据，最后关闭连接，一次交互结束。

常见端口号：

21：　　FTP服务

23：　　Telnet服务

25：　　SMTP邮件发送服务

53：　　DNS服务

67/68：DHCP请求/服务

80：　　Web服务

110：　　POP3邮件收件服务

135：　　RPC服务（DCOM组件服务）

137、138、139、445：文件共享服务

1024：　动态端口的起始号

1080：　代理软件典型的端口号

1433：　SQL Server服务