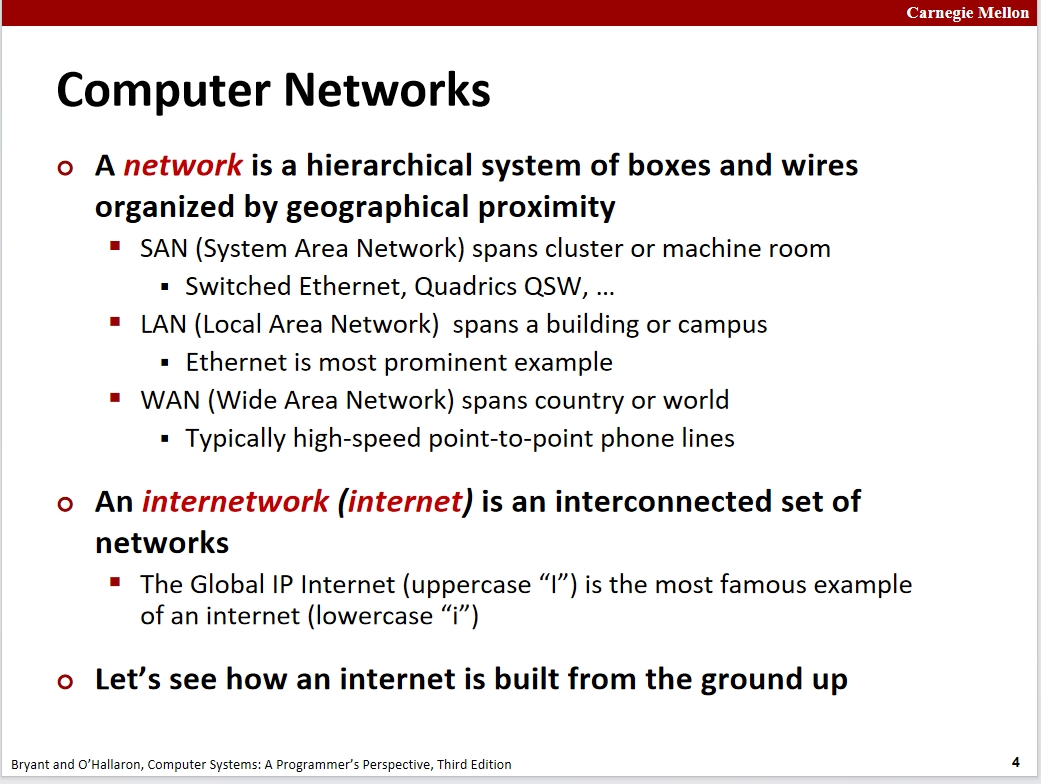


在unix系统中，将网络抽象为一个文件，当我们通过网络发送一些东西时，可以视为是对文件进行了写操作。当我们通过网络获取一些东西时，可以视为是从文件里读取了一些数据。



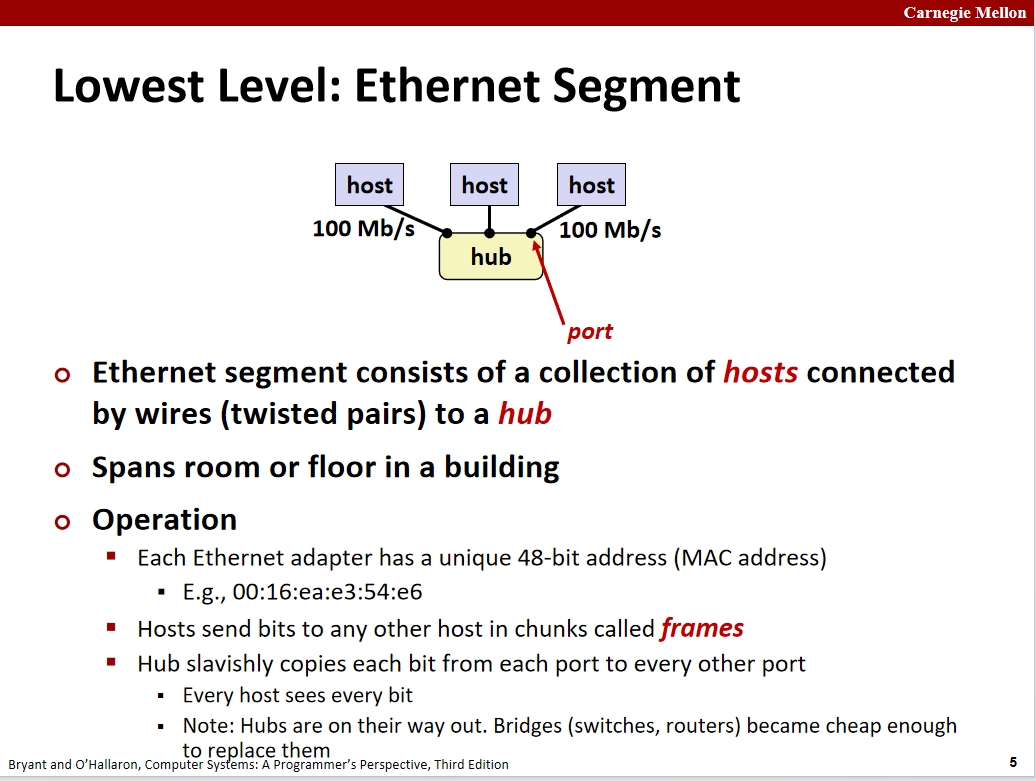
对于网络，其实是可以视为一个一些被称为主机的系统的集合。网络上的主机能够通过某种通信结构相连接沟通。

根据网络连接的范围，可以简单的进行一些划分。

局域网，这种一般是小区域内的。可以是一个公司家庭等的。

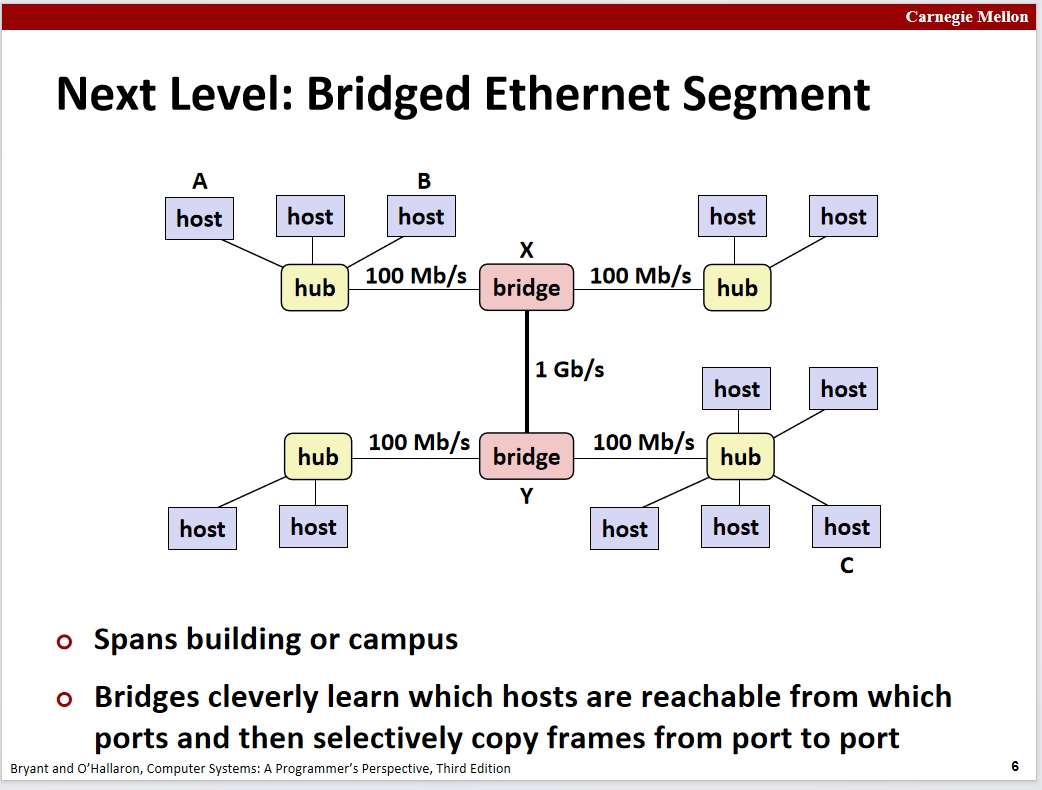
广域网，这种的范围相对来说更大，通常能够覆盖一个城市或者更大的区域。就比如说一个校园一般都有一个覆盖自己的相对来说可能比较小的局域网。

一般来说，我们使用的Internet来描述互联网时，指的是最大的因特网，即那个覆盖全球的网络。小写的internet指的是互联网的**概念**，而不是具体存在的东西。



虽然现在而言集线器(hub)并不是什么值得讨论的东西，但是还是可以讨论下。首先集线器通过物理层面上的线进行多个设备间的链接。当一个设备想要向外发送信息时，它会将集线器发送这些数据，然后集线器会将这些数据发送给所有连接上这个集线器的设备，但是一般而言并不是所有设备都需要接收这些数据。所以，在主机层面会选择是否接受数据。对于不接受这些数据的设备，会直接忽略这些数据。

所以，也不难看到，这种集线器的缺点就是它的传输模式是一种广播式的。是很浪费时间和空间的。而且，对于集线器来说，每一个时间就只能有一串数据在传输，这就导致其实很容易造成数据的拥堵。



接下来看一个现在最常用的网络设备**路由器**。

路由器相对于集线器来说，它的一个最明显区别就是它不在是一种广播的传播信息的方式。相反，它通过一定的学习，知道每个位置对应哪些主机。当一个主机想要相对应的主机发送数据时。不会再像集线器一样进行广播，而是进行一种定向，将数据传送到对应的主机。

在路由器架构中，还是存在集线器的，不过这个层次是位于主机之上，路由器之下的，这个集线器负责将一个区域里的设备连接起来，减少路由器的连接负担。但是这种架构中集线器不会再进行广播了。

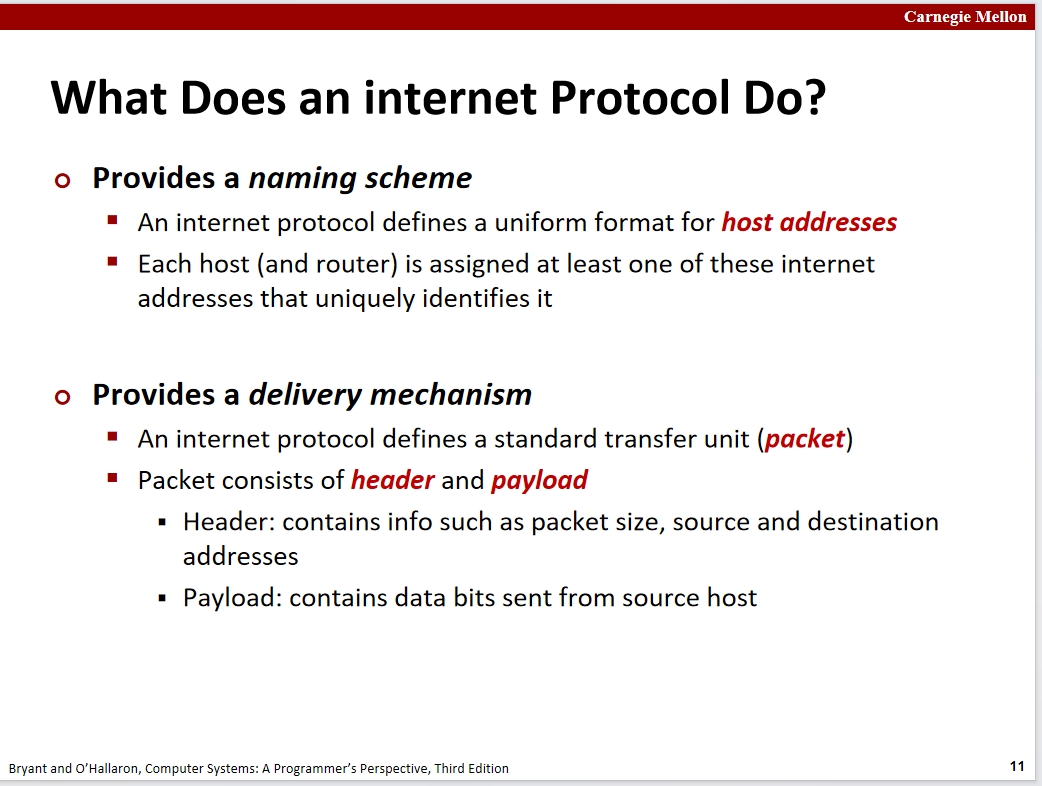
由于现代中集线器已经基本被交换机淘汰，基于一些原因，我准备直接使用交换机来替换hub的位置，具体实现可能有点差异，等我学完计网再说吧。

在上图中的架构中，交换机连接了一个有限区域内的主机设备。一个交换机可以视为一个局域网的代表。而路由器负责将多个交换机连接起来。同时，路由器之间也存在桥进行连接。通过这种从小到大的连接来实现网络的构建。

在这种架构中，交换机与路由器间的通信速率通常能有100M以上，而桥与桥之间的通信能够达到1G/s.这种通信效率的分级的原因跟计算机的效率分层设计也有相似之处。

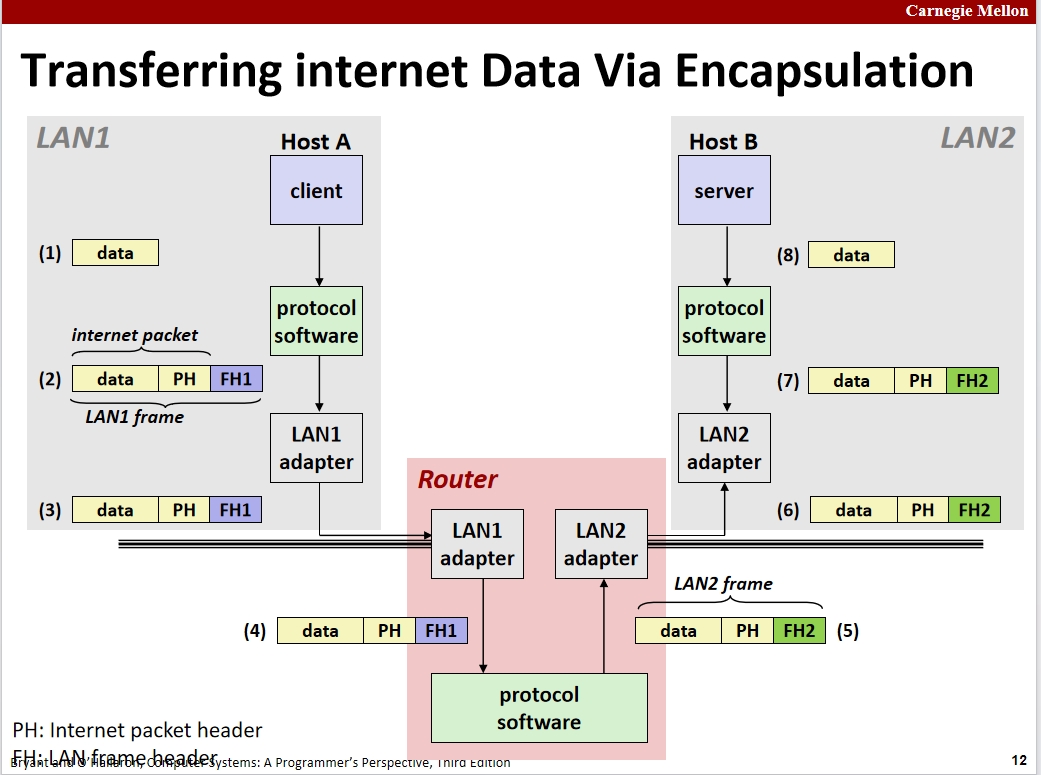
在互联网中，想要实现多个局域网间的通信，必须得有同一的通信格式，这里就需要协议来进行各个方面的规定了。就比如规定报文得是什么格式，得以什么形式被发送。如果发送失败得怎么回复他们等等。

接下来浅谈一下协议规定的内容



首先，协议中规定了一些命名规则。就比如我想向一个确定的主机发送信息，我需要一定的信息来确定这个主机是谁。然后是确保路由系统或者说网络系统能有可靠的方式将我们发送的数据准确的发送到对应的主机。

就比如传输的数据。这个一般会被包装为一个或多个数据包，这个数据包通常为1000或者2000字节，然后发送。当我们传输的数据比较大时，会对这个数据分割，形成多个小包进行一定次序的发送。所有数据的传输发生在数据包这一层。



分析下传输数据的场景，当我们想通过主机A向主机B发送一串报文时，实际上主机A会将这个报文包装成一个或多个数据包。数据包包含一部分数据。

数据包通信时指在数据包上附加一个包头，包头包含有确切的路由信息，也就是目标主机的信息。数据包在不同的传输阶段会有不同数量的包头，而且这些包头使用的协议不是完全相同的。其实也好理解，毕竟每个包头只能支持一个阶段的通信，就比如一个邮票只能支持你送到一定的区域一般。

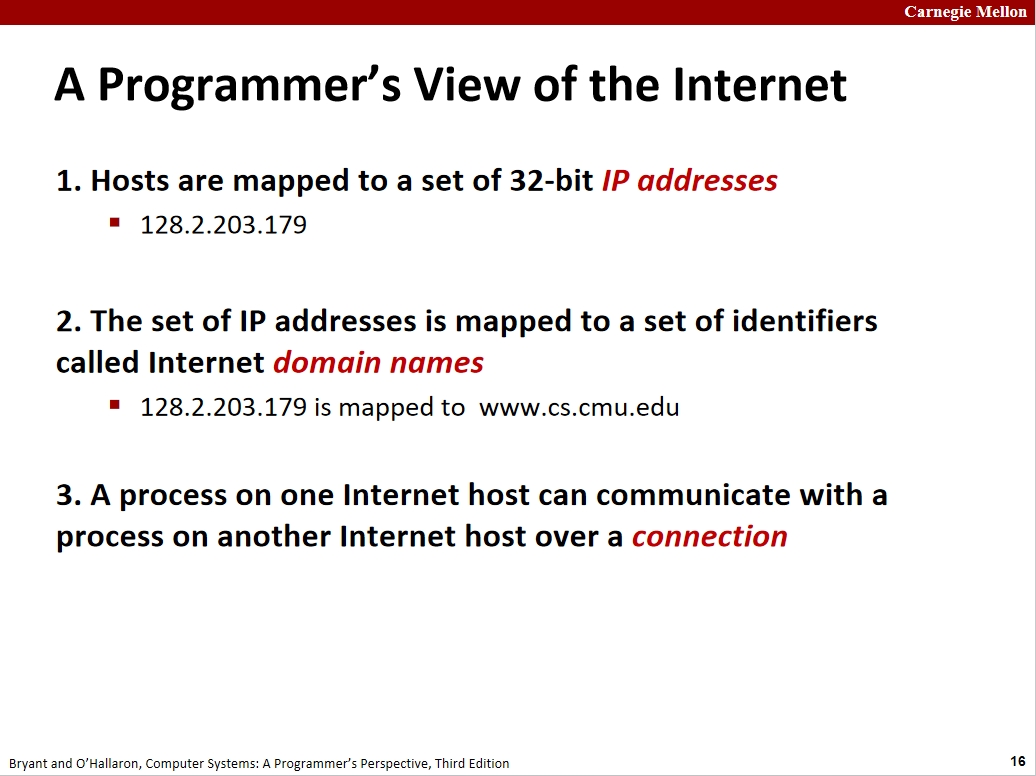
当我想要在电脑上发送一条报文的时候，这些步骤都有电脑上的软件和硬件给我们准备好了。它们会给这个数据包附上一个描述网络路由的包头和一个描述主机路由的包头。(就比如你的家庭住址)这个就是数据包上的PH(包含于Internet packet)再之后，本地局域网还会对这个数据包进行包装，再在上面附上一个FH1，表示这个数据包是在这个局域网内发送的。或者说，这个是指在本地网络内发送的特定信息的帧头。这些包装都是在本地的计算机进行完成的。

接下来，计算机的网卡或者网络适配器等硬件在确认这个数据包包含上面的信息后将这条报文发送出去(电磁波)，最后这个报文所代表的数据包会被接收端的网络适配器所接受，一般都是这个主机所在的局域网的上一层设备。  
 在这个数据包被网络适配器接收之后，还会通过一定的程序(硬件或者软件)传输到另一个网络适配器中。请记住，无论是那个适配器，在接受到一个数据包时，都会去替换帧头FH成当前局域网对应的帧头。

再之后，这个对应帧头的数据包会被剥离，最后所有的主机都可以看到也只会看到数据包的数据部分。

对于整个传输过程，当数据包发送到一个转发器时，会通过对应的表进行查找下一跳所需要跳往的地址，每个路由器或者转发设备中只储存着下一步所能到达的区域。而在这种设计下想要保证数据包的准确到达，就必须对于地址有着严格的规定，就比如，有着严格的层级前缀，可以通过跳转到每个对应层级再进行下一层级的递归查找进行寻址。

总而言之，这种在数据包前加上包头的操作叫做[**封装**]。



接下来简单看一下IPV4(第四代互联网协议)。

首先，我们常说的IP地址本质上是一个32位地址，这对TCP协议也适用。IP地址的格式通常被称为[点分十进制]格式。就比如196.128.2之类。这类地址由4个10进制数组成。这四个数都应该在0到255之间。这是因为这里面的每一个数都只能由一个字节来标识。这些IP地址被以一种网络字节序的顺序被传递和储存，这是一种大端顺序。

需要知道的是，这个IP地址的不同段的字节都代表了不同层次的信息。就比如地址128.2.203.179.这里的128.2代表以这个开头的IP地址都位于CMU中，一般来说，这种情况下的后俩位是具体的地址，而这些会有2^16个IP地址可供使用。

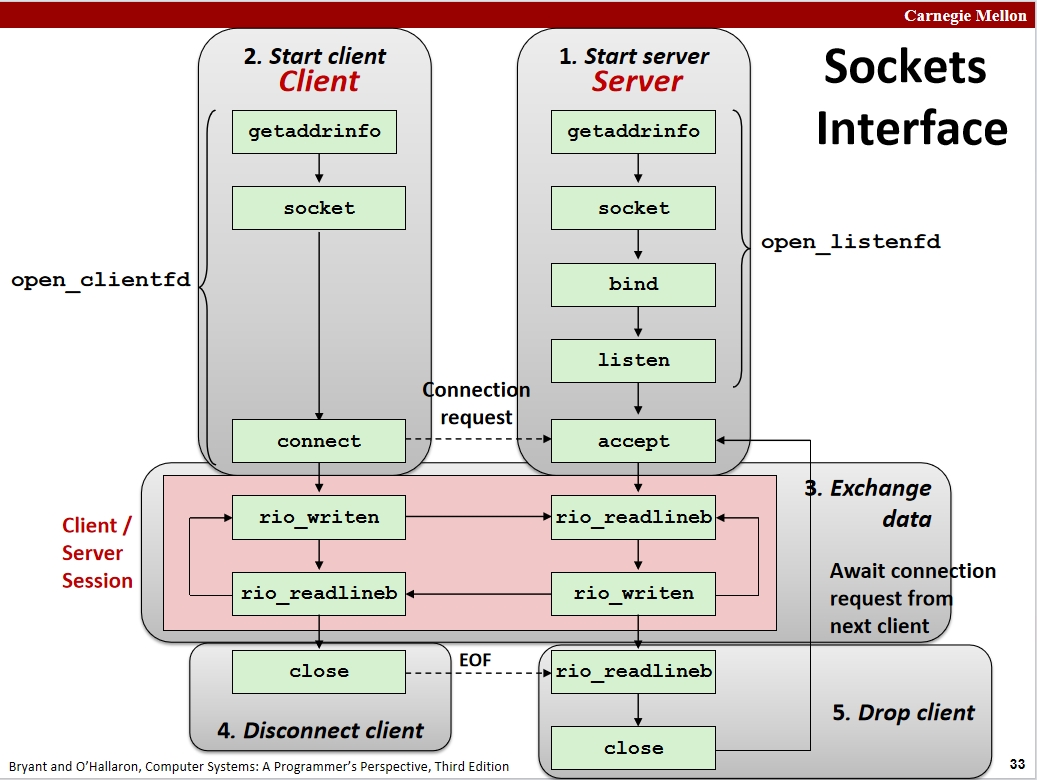
除IPV4之外，还有我国大力推广的IPV6协议地址。这个协议中是一个128位地址，能支持更多的IP地址。IPV4早就被分配完力。

域名与地址之间存在多种映射方式，可以是一对一的，一对多的，多对一的，域名也可以是一个完全不映射任何IP地址的。

接下来看一下设备间的连接。现在来说，主机间通信应用的主要方式为TCP。这是一种**面向连接**的协议。一个主机如果创建了一个链接另一个主机的通道。它们之间可以通过发送任意长度的字节流来实现相互间的通信。所以说连接就是俩台主机间的交流。连接的每一段都是一个socket（套接字）。

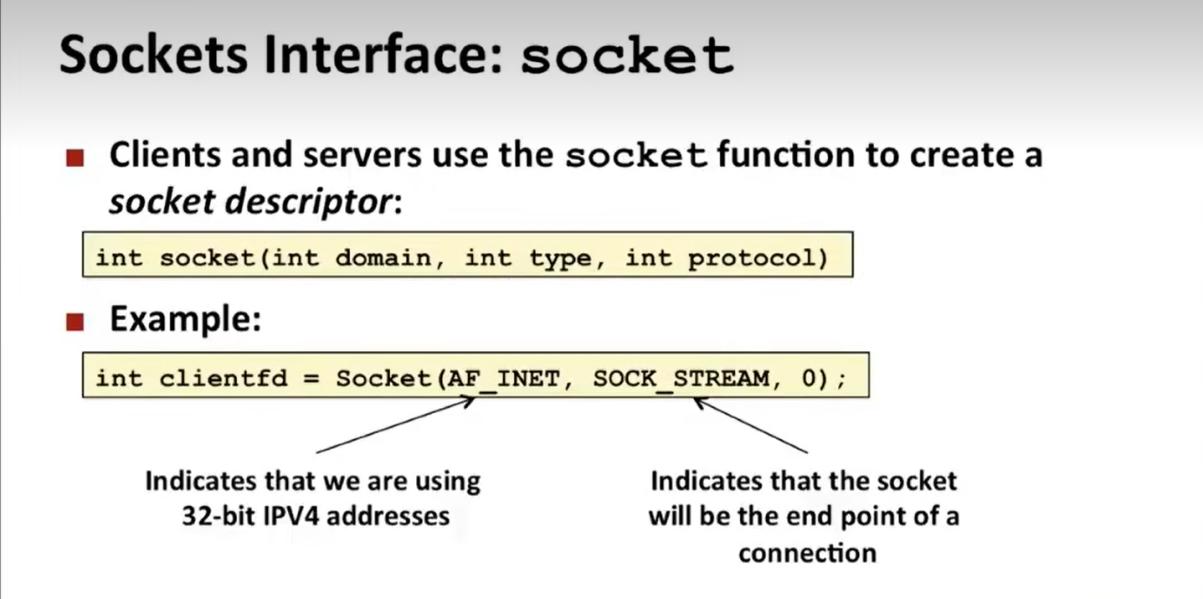
套接字包括了ip地址和端口号。对于每个连接都有一个16位的数字。这个是端口号。端口号是一个主机提供服务的类别的区分。包括但不限于ssh服务，html服务等等。同样的，客户机上也有一定的端口号用于标识当前正在通信的设备。

在客户端上，端口是被动态分配的，只会在一定时间内被分配，一般是指有链接需要用到这些端口时。



接下来看一下用户-客户端模型

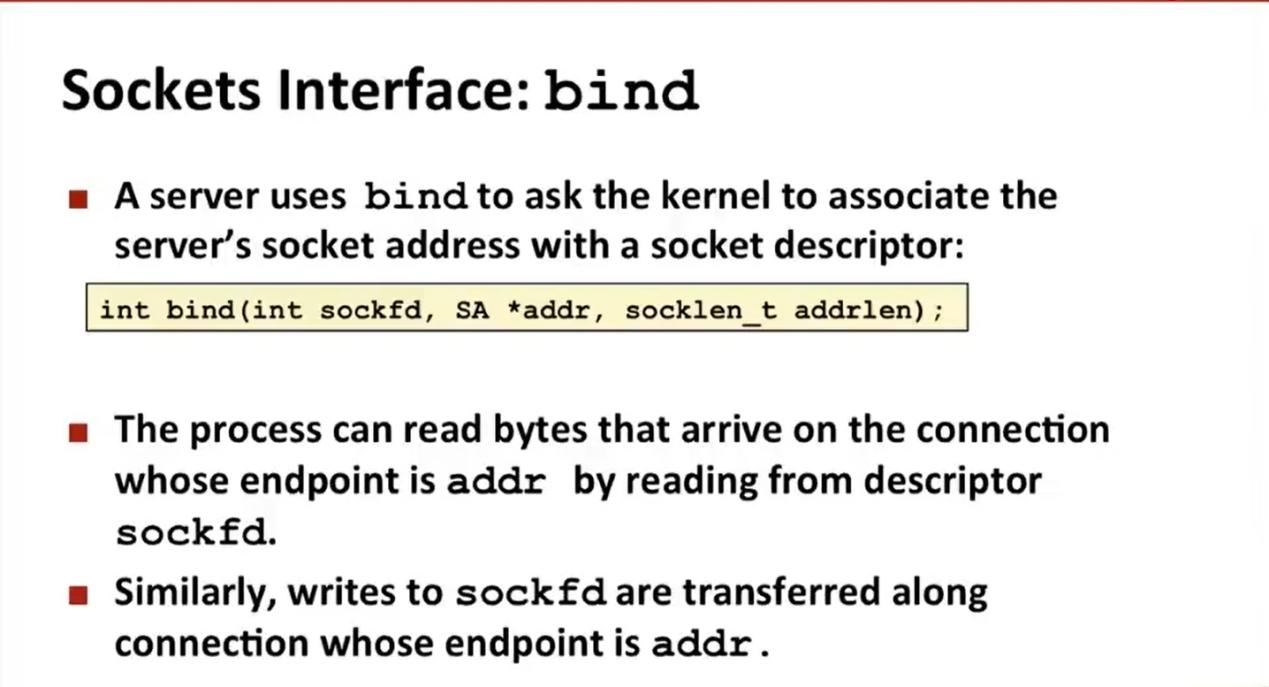
无论在用户端还是服务器方面，所要做的第一件事都是调用socket函数来建立连接。前面已经知道，套接字可以抽象为俩个相互连接的端点。



先来看一下socket函数，调用socket函数并不会调用任何操作系统的功能，也不会发送任何信息到网络。就只是单纯的创建一个套接字socket，只在程序内部起作用。

这个函数有三个参数，第一个参数会是各种各样的宏，包括在各种.h中。就比如这里的AF\_INET代表的是这里使用的是一个32位的IPV4地址。第二个参数SOCK\_STRAM指的是这个是一个TCP连接，可以说这个参数是指定了传输层使用的协议。第三个参数结合前面俩个参数能够实现对具体使用的协议的使用，使用0使得函数自动根据前面俩个参数来选择对应的协议TCP/UDP等。

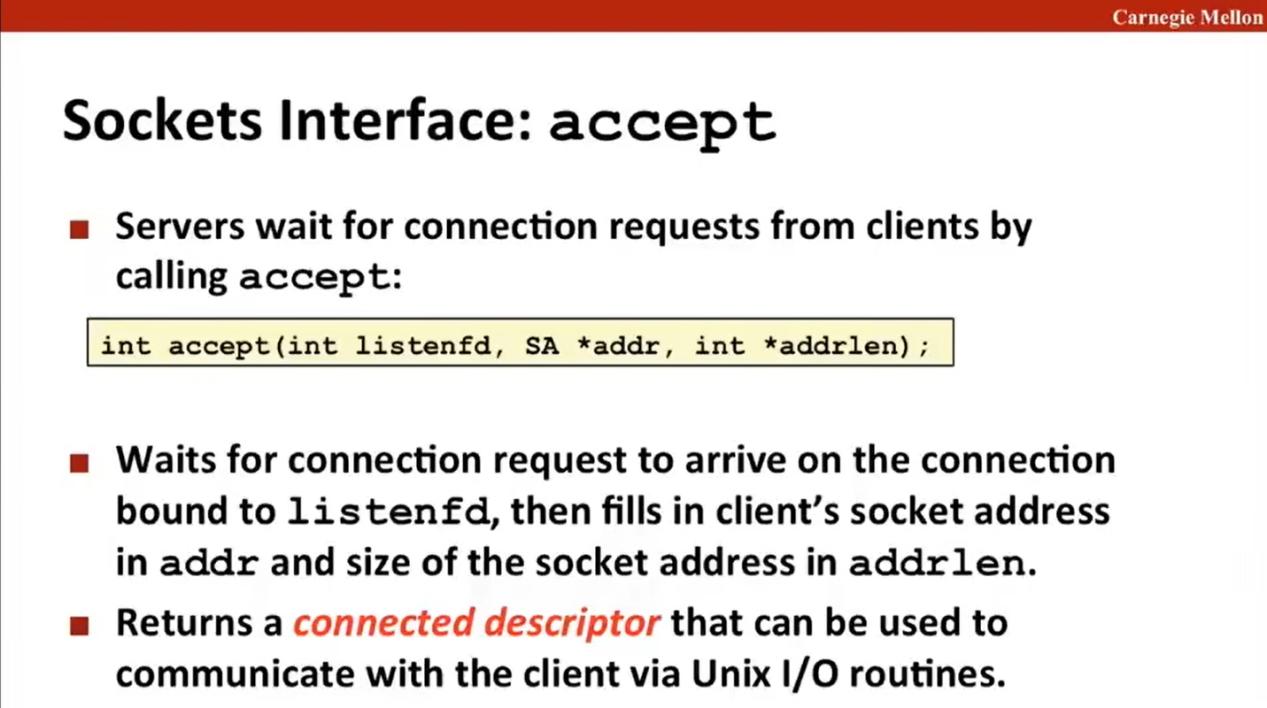
这里还需要注意下这个函数的返回值，可以看到是一个整形，其实这个整形是我们之前所学到的文本描述符。对应的一个具体的文件。



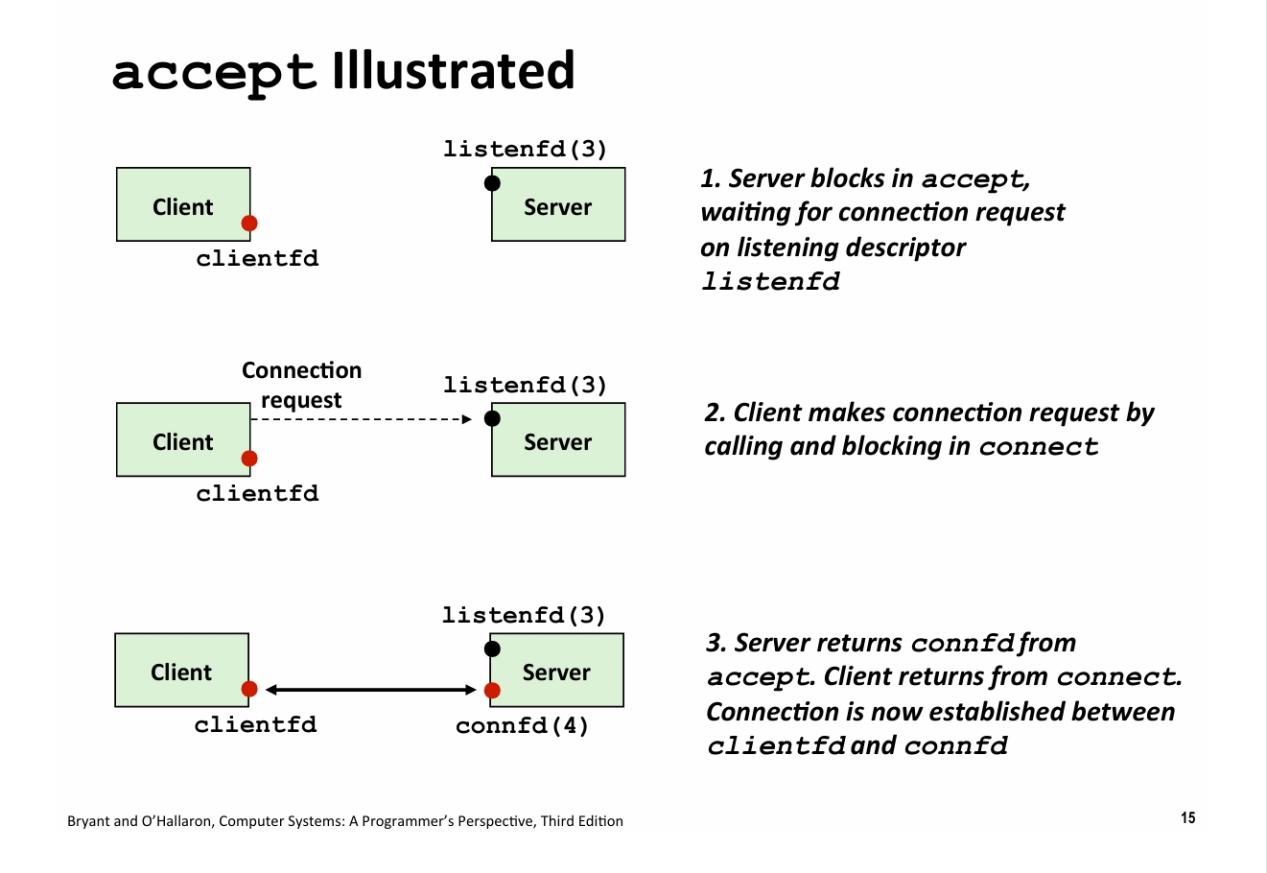
接下来看一下服务器端的一个函数bind，注意，这个函数是一个内核函数，也就是说它其实是一个系统调用。

函数的第一个函数需要一个socket函数的返回值fd来指定对应的文件，接下来还需要提供一个网络套接字的地址。除此之外，还需要一个标识这个套接字长度的地址。

使用socket函数和bind函数后，还需要调用一个listen函数来让套接字转向一种监听的状态。再之后就需要一个accept函数了。



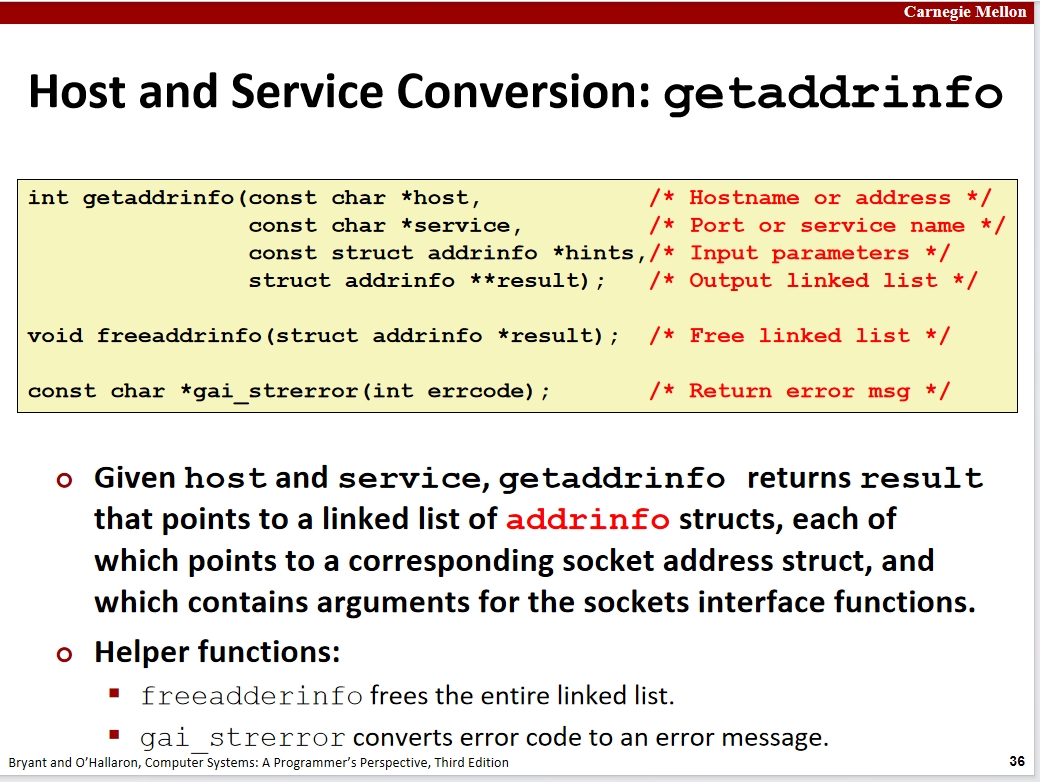
对于这个函数，与我们前面的socket函数很相似，最后的返回值也是一个文本标识符，不过这个不是跟我们服务器端的文件所一致的文件了，而是一个新的文件。在之后的程序中，服务器会通过这个文件与对应的进行信息交流。



接下来看一下客户端与服务器方面的交流。

首先，服务器方面是会一直活跃着一个用来监听的文件的就比如以3为文本描述符的文件，当服务器方面的监听函数监听到对应的客户端的请求的时候，会去创建特定的单独对应这个客户端的文本文件并分配对应的文本描述符。就比如4。

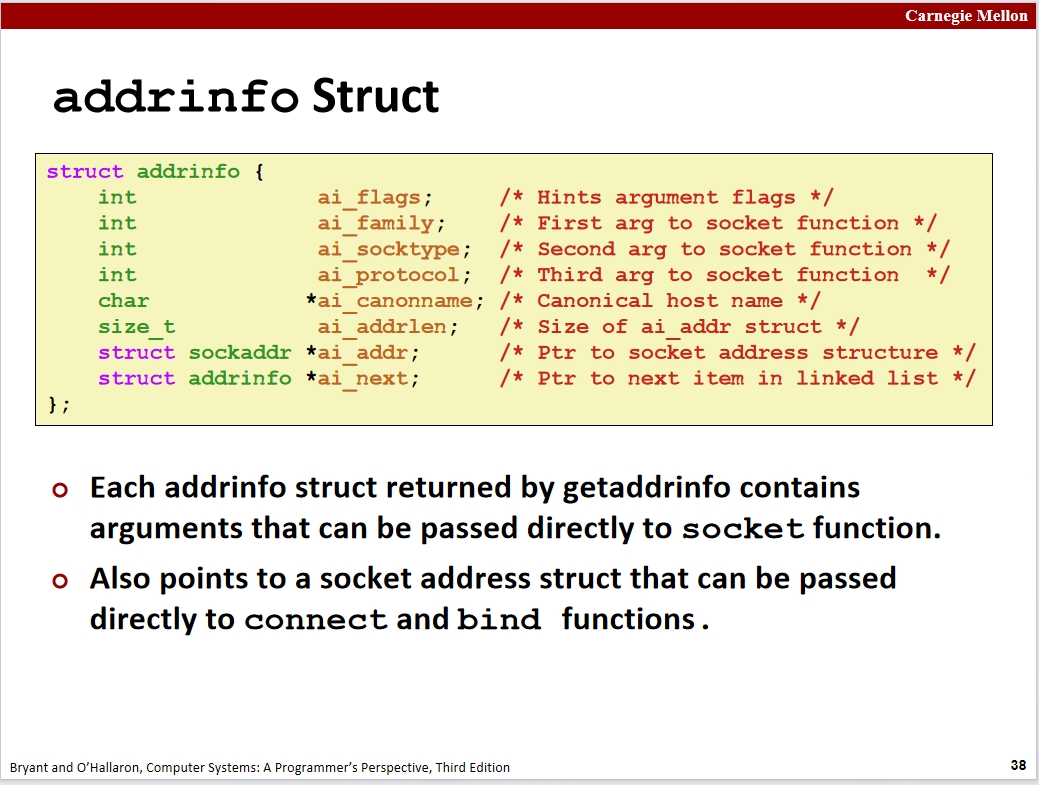
之后，这个文本描述符所对应的文件能够单独监听这个客户端发来的请求并处理。在这种架构下，服务器能够在同一时间区执行多个客户端发来的请求。



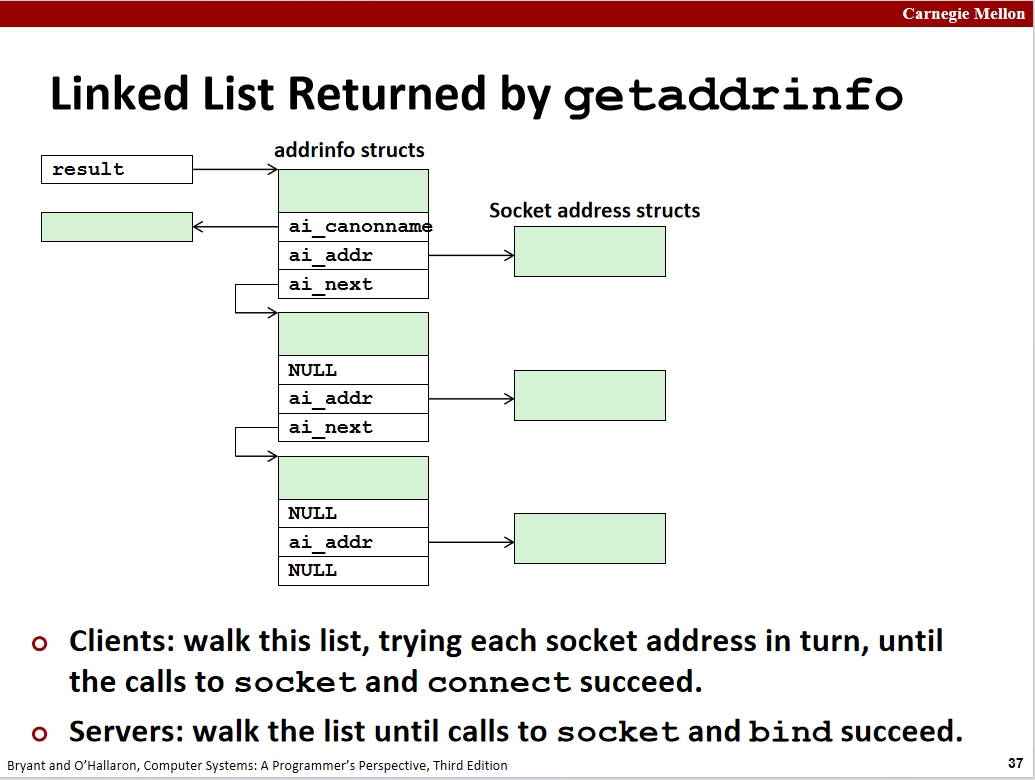
看一个重要函数，getaddrinfo

这个函数的第一个参数接收一个地址，这里需要的是本地的域名或者ip地址。第二个参数跟第一个参数一样，可以是类似于www.bilibili.com等域名或者直接的ip地址。第三个参数开始有意思起来，传递的是一个结构体指针。这个结构体指针指向的其实是一个结构体链表的第一个元素。结构体如下图。

简单来看的话这个结构体储存的就是每个唯一的域名或者IP地址所对应的信息。



接下来看一下这里的结构体链表



我们需要先知道这个链表到底是一个什么样的功能。首先，一个链表对应的其实是一个域名所对应的信息，就比如(www.bilibili.com)这个链表储存的就是一个域名所对应的信息，就比如ip地址，链表中的一个小块只能对应一个ip地址。且在链表中的所有元素中，只有一个元素包含着这个链表所代表的域名。当然，链表的元素远远不止上面那个图中的几个，而是包含着结构体中完整的信息。

也就是说，对于这个函数，我们只需要提供对应的域名(IP地址)以及我们需要的服务名(对应名称或者分配的数字)就可以构建起我们需要的整个结构体的信息，而且我们可以通过我们传进去的result指针进行访问。

对于这个函数，我现在反正由于没有上手我其实是不了解这个函数的使用环境的。但是，对于这个函数，好像是可以简化我们获取一些重要的网络信息的步骤，就比如，你想要获取一个网站的IP地址，这通常需要一个繁琐的步骤，而这个函数帮我们封装了这些步骤，只需要我们提供一些信息就可以获取IP地址和其他一些信息。接着我们就可以利用这些信息进行后续的操作。但是也就仅止于此了。等之后计网时再深入吧。