OSI七层网络协议

1. **OSI七层网络协议**  
   OSI是Open System Interconnect的缩写，意为开放式系统互联。



OSI七层模式简单通俗理解

这个模型学了好多次，总是记不住。今天又看了一遍，发现用历史推演的角度去看问题会更有逻辑，更好记。本文不一定严谨，可能有错漏，主要是抛砖引玉，帮助记性不好的人。总体来说，OSI模型是从底层往上层发展出来的。

这个模型推出的最开始，是是因为美国人有两台机器之间进行通信的需求。

需求1：(物理层)

科学家要解决的第一个问题是，两个硬件之间怎么通信。具体就是一台发些比特流，然后另一台能收到。

于是，科学家发明了物理层：

主要定义物理设备标准，如网线的接口类型、光纤的接口类型、各种传输介质的传输速率等。它的主要作用是传输比特流(就是由1、0转化为电流强弱来进行传输，到达目的地后在转化为1、0，也就是我们常说的数模转换与模数转换)。这一层的数据叫做比特。

物理层，主要解决不同设备之间的连接和传输数据；通过媒介传输比特，如网线的接口类型、光纤的接口类型、各种传输介质的传输速率等；传输是比特流

需求2：

现在通过电线我能发数据流了，但是，我还希望通过无线电波，通过其它介质来传输。然后我还要保证传输过去的比特流是正确的，要有纠错功能。

于是，发明了数据链路层：

定义了如何让格式化数据以进行传输，以及如何让控制对物理介质的访问。这一层通常还提供错误检测和纠正，以确保数据的可靠传输。

定义操作通信连接的程序；  
封装数据包为数据帧；  
监测和纠正数据包传输错误

在物理层提供比特流服务的基础上，建立相邻结点之间的数据链路，通过差错控制提供数据帧(Frame)在信道上无差错的传输，并进行各电路上的动作系列。数据链路层在不可靠的物理介质上提供可靠的传输。该层的作用包括：物理地址寻址、数据的成帧、流量控制、数据的检错、重发等。在这一层，数据的单位称为帧(frame)。数据链路层协议的代表包括：SDLC、HDLC、PPP、STP、帧中继等。

数据链路层： 不同设备之间的比特流的解析和纠错；比特流封装成帧（Frame）, 数据的成帧、物理地址寻址、流量控制、数据的检错（奇偶校验，LRC校验）、重发等。补充：**每一帧（也**叫“以太网数据包”**）分成两个个部分：标头（Head）和数据（Data）。“标头”包含数据包的一些说明项，比如发送者、接受者、数据类型等等。**

需求3：

 传输层只是解决了打包的问题。但是如果我有多台计算机，怎么找到我要发的那台？或者，A要给F发信息，中间要经过B，C，D,E，但是中间还有好多节点如K.J.Z.Y。我怎么选择最佳路径？这就是路由要做的事。

于是，发明了网络层。即路由器，交换价那些具有寻址功能的设备所实现的功能。这一层定义的是IP地址，通过IP地址寻址。所以产生了IP协议。

定义网络设备间如何传输数据；  
根据唯一的网络设备地址路由数据包；  
提供流和拥塞控制以防止网络资源 的损耗  
在 计算机网络中进行通信的两个计算机之间可能会经过很多个数据链路，也可能还要经过很多通信子网。网络层的任务就是选择合适的网间路由和交换结点， 确保数据及时传送。网络层将数据链路层提供的帧组成数据包，包中封装有网络层包头，其中含有逻辑地址信息- -源站点和目的站点地址的网络地址。如 果你在谈论一个IP地址，那么你是在处理第3层的问题，这是“数据包”问题，而不是第2层的“帧”。IP是第3层问题的一部分，此外还有一些路由协议和地 址解析协议(ARP)。有关路由的一切事情都在这第3层处理。地址解析和路由是3层的重要目的。网络层还可以实现拥塞控制、网际互连等功能。在这一层，数据的单位称为数据包(packet)。网络层协议的代表包括：IP、IPX、RIP、OSPF等。

网络层：产生原因：网络中有很多的数据链路，网络中有很多的地址， 网络层将数据链路层提供的帧组成数据包，

怎么解决：网络层包头，其中含有逻辑地址信息- -源站点和目的站点地址的网络地址。选择合适的网间路由和交换结点， 确保数据及时传送。

需求4：

现在我能发正确的发比特流数据到另一台计算机了，但是当我发大量数据时候，可能需要好长时间，例如一个视频格式的，网络会中断好多次（事实上，即使有了物理层和数据链路层，网络还是经常中断，只是中断的时间是毫秒级别的）。

那么，我还须要保证传输大量文件时的准确性。于是，我要对发出去的数据进行封装。就像发快递一样，一个个地发。

于是，先发明了传输层（传输层在OSI模型中，是在网络层上面）

例如TCP，是用于发大量数据的，我发了1万个包出去，另一台电脑就要告诉我是否接受到了1万个包，如果缺了3个包，就告诉我是第1001，234，8888个包丢了，那我再发一次。这样，就能保证对方把这个视频完整接收了。

例如UDP，是用于发送少量数据的。我发20个包出去，一般不会丢包，所以，我不管你收到多少个。在多人互动游戏，也经常用UDP协议，因为一般都是简单的信息，而且有广播的需求。如果用TCP，效率就很低，因为它会不停地告诉主机我收到了20个包，或者我收到了18个包，再发我两个！如果同时有1万台计算机都这样做，那么用TCP反而会降低效率，还不如用UDP，主机发出去就算了，丢几个包你就卡一下，算了，下次再发包你再更新。

传输层：产生原因：

1. 网络中的种[通信子网](https://baike.baidu.com/item/%E9%80%9A%E4%BF%A1%E5%AD%90%E7%BD%91)在性能上存在着很大差异（的[吞吐量](https://baike.baidu.com/item/%E5%90%9E%E5%90%90%E9%87%8F" \t "_blank),[传输速率](https://baike.baidu.com/item/%E4%BC%A0%E8%BE%93%E9%80%9F%E7%8E%87),数据延迟通信费用各不相同）[会话层](https://baike.baidu.com/item/%E4%BC%9A%E8%AF%9D%E5%B1%82)来说,却要求有一性能恒定的接口

功能：

共同

1. 传输层是为应用进程之间提供端到端的逻辑通信（网络层是为主机之间提供逻辑通信），两个主机进行通信实际上就是两个主机中的应用进程相互通信。应用进程之间的通信又称为端到端的通信。（一台主机中ip地址只有一个，通过端口区分不同的应用程序）
2. 传输层要对收到的报文进行差错检测。
3. 传输层需要两种不同的运输协议，即面向连接的用户数据报协议TCP（Transmission Control Protocol）和无连接的传输控制协议UDP（User Datagram Protocol）。

不同

（1）为会话提供可靠的,无误的[数据传输](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E4%BC%A0%E8%BE%93" \t "_blank). 通过: 传输层的服务一般要经历传输连接建立阶段,[数据传送](https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E4%BC%A0%E9%80%81)阶段,传输连接释放阶段3个阶段

（2）传输层一个很重要的功能就是复用和分用。应用层不同进程的报文通过不同端口向下交到传输层，再往下就共用网络层提供的服务。

需求5：

现在我们已经保证给正确的计算机，发送正确的封装过后的信息了。但是用户级别的体验好不好？难道我每次都要调用TCP去打包，然后调用IP协议去找路由，自己去发？当然不行，所以我们要建立一个自动收发包，自动寻址的功能。

于是，发明了会话层。会话层的作用就是建立和管理应用程序之间的通信。保证应用程序自动收发包和寻址了

会话层：

解决的问题：保证应用程序自动收发包和寻址了

功能：建立和管理应用程序之间的通信。

需求6：

现在我能保证应用程序自动收发包和寻址了。但是我要用Linux给window发包，两个系统语法不一致，就像安装包一样，exe是不能在linux下用的，shell在window下也是不能直接运行的。于是需要表示层（presentation），帮我们解决不同系统之间的通信语法问题。

表示层就专门负责这些有关网络中计算机信息表示方式的问题。表示层负责在不同的数据格式之间进行转换操作，以实现不同计算机系统间的信息交换。 两台计算机之间的信息交换除了编码外，还包括数组、浮点数、记录、图像、声音等多种数据结构，表示层用抽象的方式来定义交换中使用的数据结构，并且在计算 机内部表示法和网络的标准表示法之间进行转换。

表示层还负责数据的加密，以在数据的传输过程对其进行保护。数据在发送端被加密，在接收端解密。使用加密密钥来对数据进行加密和解密。表示层还负责文件的压缩，通过算法来压缩文件的大小，降低传输费用。

表示层：

产生的原因：

（1）不同的系统之间通信语法是不一样的，数据类型也不一样（数组、浮点数、记录、图像、声音等多种数据结构）

功能：

1. 负责数据格式的转换，转换为所在系统的类型
2. 压缩与解压缩，加密与解密。保证数据的安全性

需求7：

 应用层是网络体系中最高的一层，也是唯一面向用户的一层，应用层将为用户提供常用的应用程序，并实现网络服务的各种功能。常用的电子邮件、上网浏览等网络服务，都是应用层的程序。

应用层主要是面对用户访问网络的。主要有一些应用程序，如：DNS,FTP,E-mail，Telnet，HTTP

应用层：1.

补充：

Socket：

这不是一个协议，而是一个通信模型。其实它最初是伯克利加州分校软件研究所，简称BSD发明的，主要用来一台电脑的两个进程间通信，然后把它用到了两台电脑的进程间通信。所以，可以把它简单理解为进程间通信，不是什么高级的东西。主要做的事情不就是：

A发包：发请求包给某个已经绑定的端口（所以我们经常会访问这样的地址182.13.15.16:1235，1235就是端口）；收到B的允许；然后正式发送；发送完了，告诉B要断开链接；收到断开允许，马上断开，然后发送已经断开信息给B。

B收包：绑定端口和IP；然后在这个端口监听；接收到A的请求，发允许给A，并做好接收准备，主要就是清理缓存等待接收新数据；然后正式接收；接受到断开请求，允许断开；确认断开后，继续监听其它请求。

可见，Socket其实就是I/O操作。Socket并不仅限于网络通信。在网络通信中，它涵盖了网络层、传输层、会话层、表示层、应用层——其实这都不需要记，因为Socket通信时候用到了IP和端口，仅这两个就表明了它用到了网络层和传输层；而且它无视多台电脑通信的系统差别，所以它涉及了表示层；一般Socket都是基于一个应用程序的，所以会涉及到会话层和应用层。