计网总结 by 白睿

【因特网的两种描述方法】
1.根据它的硬件和软件组件来描述
2.根据基础设施向分布式应用程序提供的服务来描述

【网络边缘】：主机/端系统（与因特网相连的计算机和其他设备，因为它们位于因特网边缘所以称为端系统，它们容纳运行程序）

【主机两分类】

【客户端】

【服务器】大型数据中心

端系统通过**通信链路**和**分组交换机**连接到一起

【接入网】将端系统物理连接到其边缘路由器的网络。

【边缘路由器】端系统到任何其他远程端系统的路径上的第一台路由器

【接入网三大类】

【家庭接入】：传统拨号、DSL（DSL,电话）、HFC（电缆，电视）、FTTH（光纤到户）、卫星

【数字用户线 DSL】用户的本地电话公司是它的 ISP

1.每个用户的 DSL 调制解调器使用现有的电话线（双绞铜线）与位于电话公司的本地中心局中的数字用户线接入复用器（DSLAM）交换数据

2.**家庭的 DSL 调制解调器**得到数字数据后将其**转换为高音频**，以通过电话线传输给本地中心局。来自许多家庭的模拟信号在 **DSLAM** 处被**转换回数字形式**。

***家庭电话线同时承载了数据 & 传统的电话信号，他们用不同的频率进行编码(**频分复用技术)**。划分为三个信道：【高速下行信道】，【中速上行信道】，【普通的双向电话信道】

***一个电话呼叫和一个因特网连接能够同时共享 DSL 链路。

***用户一侧，一个分配器把到达家庭的数据信号和电话信号分隔开，并将数据信号转发给 DSL 调制解调器

***电话公司一侧，在本地中心局，DSLAM 把数据和电话信号分隔开，并将数据送往因特网。（上千家庭与同一个 DSLAM 相连）

***DSL 为不对称接入：12Mbps 下行&1.8Mbps 上行传输速率 or 55Mbps 下行&15Mbps 下行传输速率

【对比】传统电话线的拨号接入与 DSL 基于相同的模式：家庭的调制解调器经过电话线连接到 ISP 的调制解调器。但拨号接入比 DSL 慢巨多。

【电缆因特网接入/混合光纤同轴系统 HFC】利用有线电视公司现有的有线电视基础设施。住宅从提供有线电视的公司获得了电缆因特网接入。该系统中应用了光线和同轴电缆。

***电缆因特网接入需要特殊的调制解调器**【电缆调制解调器】**，其是一个外部设备，通过一个以太网端口连接到家庭 PC。（**同** DSL 调制解调器）

***电缆头端，电缆调制解调器端接系统（CMTS）将来自许多下行家庭中的电缆调制解调器发送的模拟信号转换回数字形式。（**同** DSL 在 **DSLAM** 处被**转换回数字形式**）

***电缆调制解调器将 HFC 网络划分为下行和上行两个信道。（**异** DSL（3））

***接入不对称，下行信道分配的传输速率比上行信道的高（**同** DSL）

***电缆因特网接入的重要特征：**共享广播媒体**。（由头端发送的每个愤怒向下行经每段链路到每个家庭；每个家庭发送的每个分组经上行信道向头端传输**<头下到家（下行速率才是用户可用的速率，但实际比这个低），家上向头>**。所以如果几个用户同行经下行信道下载一个视频文件，每个用户接受视频文件的实际速率将大大低于电缆总计的下行速率）（若仅有很少用户访问网页，则每个用户都可以以全部的下行速率接收网页，因为用户很少在完全相同的时刻请求网页）

***上行信道也共享，所以需要一个 分布式多路访问协议来协调传输和避免碰撞

【光纤到户 FTTH】从本地中心局直接到家庭提供了一条光纤路径。

【有竞争性的光纤分布方案/体系结构】

1.【直接光纤】从本地中心局到每户设置一根光纤（easiest）

2.从中心局出来的每根光纤实际由许多家庭共享，直到相对接近这些家庭的位置，该光纤才分成每户一根光纤

【主动光纤网络 AON】交换因特网

【被动光纤网络 PON】所有从中心局中的光纤线路端接器 OLT 发送到分配器的分组在分配器处复制

***每个家庭具有一个光纤网络端接器 ONT，它由专门的光纤连接到邻近的分配器。该分配器把一些家庭（<100）集结到一根共享的光纤，该光纤再连接到本地电话和公司的中心局中的光纤线路端接器 OLT，该 OLT 提供了光信号和电信号之间的转换，经过本地电话公司路由器与因特网相连。

***在家庭中，用户将一台家庭路由器与 ONT 相连，并经过这台家庭路由器接入因特网。

【企业接入】：以太网、无线 LAN（Wifi）

【以太网】最流行的接入技术。以太网用户使用双绞铜线与一台以太网交换机相连，以太网交换机或该交换机网络再通过机构路由器与机构的 ISP 这一更大的因特网相连。

【无限 LAN】无线用户从/到一个接入点发送/接收分组，该接入点与企业网连接（可能使用了有线以太网），企业网再与有线因特网相连。一个无线 LAN 用户必须位于接入点的几十米范围内。

【广域无线接入】：4G、LTE

【通信链路】由不同类型的物理媒体组成。不同的链路能够以不同的速率传输数据，链路的**传输速率**以**比特/秒（bit/s or bps）**度量。

【同轴电缆】

【铜线】

【光纤】

【无线电频谱】

*****【总结】*****

HFC：光纤&同轴电缆

DSL：双绞铜线

以太网：双绞铜线

移动接入网：无线电频谱

【网络核心】

【报文】端系统彼此交换报文，报文可以执行控制功能，也可以包含数据。

【分组】当一台端系统要向另一台端系统发送数据时，发送端系统将数据分段，并为每段加

上首部字节，由此形成的信息包为分组。（为了从源端系统向目的端系统发送一个报文，源将长报文划分为较小的数据块，称之为分组）分组通过网络发送到目的端系统，在目的端系统被装配成初始数据。

【分组交换机】从它的一条入通信链路接收到达的分组，并从它的一条出通信链路转发该分组。分组交换机分为两类：路由器&链路层交换机。二者均是向最终目的地转发分组。

【路由器】用于网络核心中

【链路层交换机】用于接入网中。

【路径】从发送端系统到接收端系统，一个分组所经历的一系列通信链路和分组交换机称为通过该网络的路径。

*****分组——卡车，通信链路——公路，分组交换机——交叉口，端系统——建筑物**

【存储转发传输】多数分组交换机在链路的输入端使用存储转发传输机制。即在交换机能够开始向输出链路传输该分组的第一个比特之前，必须接受到整个分组。

【存储转发时延/端到端时延】d=NL/R（L 个分组通过 N 条速率为 R 的链路组成的路径，注意源到目的地之间有 N-1 台路由器）

【输出缓存/输出队列】每台分组交换机有多条链路与之相连。对于每条相连的链路，该分组交换机具有一个输出缓存，用于存储路由器准备发往那条链路的分组。

【排队时延】到达的分组需要传输到某条链路，但该链路正忙于传输其他分组，该到达分组必须在输出缓存中等待。**存储转发时延&排队时延时延取决于网络的拥塞程度**。

【分组丢失/丢包】缓存空间大小有限，一个到达的分组发现该缓存已被其他等待传输的分组完全充满，则**该到达的分组或已经排队的分组之一**会被丢弃。

【转发表】每台路由器有一个转发表，用于将目的地址（或其一部分）映射成为输出链路。

*****每条与分组交换机（路由器/链路分组交换机）相连的链路，该分组交换机都为其提供一个输出缓存**

*****每台路由器都有一个转发表，来映射选择该分组走哪条输出链路(路由器使用分组的目的地址来索引转发表并决定适当的出链路)**

【路由选择协议】因特网用于自动地设置路由器的转发表

【通过网络链路和交换机移动数据的方法】（3）

【电路交换】

【频分复用 FDM】链路的频谱由跨越链路创建的所有连接共享，连接期间链路为每条连接专用一个频段。该频段的宽度为**带宽**

【时分复用 TDM】时间被划分为固定期间的**帧**，每个帧又被划分为固定数量的**时隙。**当网络跨越一条链路创建一条连接时，网络在每个帧中为该连接指定一个时隙，这些时隙专门由该连接单独使用，一个时隙（在每个帧内）可用于传输该连接的数据。

***循环的 TDM 帧中，每条电路被分配相同的专用时隙。

***TMD 一条电路的传输速率=帧速率*一个时隙中的比特数

*****【对比】*****

FDM 每条电路连续地得到部分贷款

TDM 每条电路在时隙中周期性地得到所有带宽

【分组交换】

*****【对比】***电路交换&分组交换**

1.分组交换不适合实时服务，因为它地端到端时延可变且不可预测

但分组交换提供了更好地带宽共享，且比电路交换更简单有效成本低。
2.电路交换不考虑需求，预先分配了传输链路地使用，这使得已分配而并不需要地链路时间未被利用
3.分组交换按需分配链路使用，链路传输能力将在所有需要在链路上传输分组地用户之间逐分组的被共享。

【网络结构】

【因特网服务提供商 ISP】端系统通过 ISP 接入因特网。每个 ISP 自身就是一个由多台分组交换机和多段通信链路组成的网络。各 ISP 为**端系统**提供了各种不同类型的**网络接入**，并为**内容提供者**提供**因特网接入**服务，将 Web 站点和视频服务器**直接连入**因特网。
***每个分组进行传递时必须要经过 ISP
***因特网就是将端系统彼此互联，因此为端系统提供接入的 ISP 也必须互联。
***较低层的 ISP 通过国家的、国际的较高层 ISP 互联。较高层 ISP 由通过高速光纤链路互联的高速路由器组成。所有 ISP 网络它们每个都是独立管理的，运行着 ISP 协议，遵从一定的命名和地址规则。
***不同 ISP 之间通过网络接入点/访问点 IXP/NAP 互联，IXP/NAP 一般互联一、二级的，很少连二、三级的
【网络结构 1】单一的全球传输 ISP（提供商）互联所有接入 ISP（客户）。
【网络结构 2】两层等级结构。数十万接入 ISP（底层）和多个全球传输 ISP（顶层）组成**（！全球传输 ISP 必须是互联的）**
Eg：任何给定区域可能有一个**区域 ISP，**区域中的接入 ISP 与之连接。每个区域 ISP 与**第一层 ISP** 连接。
【网络结构 3】多层等级结构。有多个竞争的第一层 ISP，且在一个区域可能有多个竞争的区域 ISP。每个接入 ISP 向其连接的区域 ISP 支付费用，每个区域 ISP 向他连接的第一层 ISP 支付费用（一个接入 ISP 也可以直接与第一层 ISP 连接付费）。该等级结构的每一层，都有客户-提供商关系。第一层 ISP 位于顶部不向任何人付费。
【网络结构 4】=【3】+存在点 POP、多宿、对等、因特网交换点**（接入 ISP，区域 ISP，第一层 ISP，PoP，多宿，对等，IXP）**

【PoP 存在点】存在于等级结构除了底层（接入 ISP）的所有层次.一个 PoP 是提供商网络中的一台或多台路由器（在相同位置）群组，其中客户 ISP 可以与提供商 ISP 连接。对于要与提供商 PoP 连接的客户网络，它能从第三方电信提供商租用高速链路将它的路由器之一直接连接到位于该 PoP 的一台路由器

【多宿】除了第一层 ISP 的任何 ISP 可以与两个或更多提供商 ISP 连接。当一个 ISP 多宿时，即使提供商之一出现故障其仍然可以继续发送和接受分组。

【对等】客户 ISP 向它们的提供商 ISP 付费以获得全球因特网互联能力。客户 ISP 支付给提供商 ISP 的费用数额反映了它通过提供商交换的通信流量。为了减少这些费用，位于相同等级结构层次的邻近一对 ISP 能够直接将他们的网络连到一起，使它们之间的所有流量经直接连接而不是通过上游的中间 ISP 传输。两个 ISP 对等时无结算。

【因特网交换点 IXP】沿着这些对等 ISP 的相同路线，第三方公司能够创建一个 IXP，其是一个汇合点，多个 ISP 能在这里一起对等。IXP 通常位于一个有自己的交换机的独立建筑物中
【网络结构 5】=【4】+内容提供商网络
内容提供商网络与较低层 ISP 对等，绕过高层 ISP，减少了向顶层 ISP 支付的费用，而

且对其服务最终如何交付给端用户有了更多的控制。

【分组交换网中的时延】

节点总时延=节点处理时延+排队时延+传输时延+传播时延

$$d_{nodal}=d_{proc}+d_{queue}+d_{trans}+d_{prop}$$

【处理时延】检查分组首部和决定将该分组导向何处所需的时间+检查比特级别差错时间等（处理时延对一台路由器的最大吞吐量有重要影响，【最大吞吐量】一台路由器能够转发分组的最大速率）

【排队时延】在队列中，当分组在链路上等待传输时所经历的时间。（特定分组的排队时延长度取决于先期到达的正在排队等待像链路传输的分组数量）

【传输时延】将**所有分组的比特**推向链路所需要的时间（传输时延仅与分组长度 L 和两路由器之间链路的传输速率 R 有关 t=L/R）

【传播时延】一个比特被推向链路，该比特需要向下一个路由器传播，从该链路的起点到下一个路由器传播所需要的时间（传播时延仅与两路由器间距离 d 和链路的物理媒体（传播速率为 s）有关）t=d/s

【注意】

***所有变量均给定，那么所有分组的处理时延、传输时延、传播时延都是相同的。但排队时延可能不同。

***排队时延取决于流量到达该队列的速率，链路的传输速率和到达流量的性质(周期性到达 or 以突发形式到达)

***【流量强度】=La/R（a 为分组到达队列的平均速率，L 为分组由 L 个比特组成，R 为传输速率）

1. La/R>1：比特到达队列的平均速率超过该队列传输出去的速率，则队列会无限增加，排队时延趋向无穷大

2. La/R<=1：到达的流量性质会影响排队时延。

1）周期性到达，即每 L/R 秒到达一个分组，则不会有排队时延

2）突发性到达，若每(L/R)N 秒到达 N 个分组，则传输的第一个分组没有排队时延，第二个分组就有 L/R 秒的排队时延，第 n 个分组就有（n-1）L/R 秒的排队时延。

【丢包】到达的分组发现一个满的队列，则路由器将丢弃该分组（分组丢失的比例随着流量强度的增加而增加

***丢失的分组可能基于端到端的原则重传以确保所有数据最终从源传送到了目的地

【吞吐量】

【瞬时吞吐量】任何时间瞬时主机 B 接收到文件的速率（bps）

【平均吞吐量】主机 B 接收到所有比特的文件用去 T 时间，F/T bps

*** Rs：服务器与路由器之间的链路速率（注入速率）

Rc：路由器与客户之间的链路速率(转发速率)

1）Rs<Rc：

服务器注入的比特顺畅地通过路由器，并以 Rs bps 到达客户

2）Rs>Rc:

比特以 Rc 离开路由器

链路的吞吐量/【瓶颈链路】的传输速率=min{Rc,Rs}

***如果许多数据都通过一条链路流动，即使是一条具有高传输速率的链路也有可能成为瓶颈链路

***一个节点的性能根据**时延 & 丢包的概率 & 端到端吞吐量** 来度量

【协议层次以及服务模型】

【分层协议中每个层次提供服务的方式/一层的服务模型】

每层通过 在该层中执行某些动作 或 使用**直接**下层的服务 来提供服务

【分层的特点】

只要该层对其上面的曾提供相同的服务，并且使用来自下面层次的相同服务，改变该层一个服务的实现时，系统的其余部分保持不变。（**改变服务实现不影响该系统其他组件**）

【协议】协议定义了在两个或多个通信实体之间交换的**报文的格式**和**顺序**，以及报文发送或接收一条报文或其他事件所**采取的动作**。

***因特网中，涉及两个或都多个远程通信实体的所有活动都受协议的制约。端系统、分组交换机和其他因特网部件都要运行一系列协议，这些协议控制因特网中信息的接受和发送。不同的协议完成不同的通信任务

【两台物理连接的计算机中】硬件实现的协议控制了在两块网络接口卡间的“线上”的比特流

【端系统中】拥塞控制协议控制了在发送方和接收方之间传输的分组发送的速率

【路由器中】协议决定了分组从源到目的地的路径

【TCP】传输控制协议

【IP】网际协议，定义了在路由器和端系统之间发送和接收的分组格式。

***因特网的主要协议为 TCP/IP

***【协议层的实现方式】

【应用层&运输层协议】在端系统中通过**软件**实现

【物理层&数据链路层协议】在与给定链路相关联的网络接口卡中实现（**硬件**）

【网络层协议】**软硬件混合**

*****一个第 n 层协议的不同部分常常位于网络组件的各部分中。**

*****【协议分层的优点】概念化&结构化**

*****【协议分层的缺点】一层可能冗余较低层的功能；某层的功能可能需要仅在其他某层才出现的信息**

【协议栈】各层的所有协议。因特网的协议栈（5）：物理层+链路层+网络层+运输层+应用层

【应用层】应用程是网络应用程序及它们的应用层协议存留的地方。**端系统的应用程序间传报文。**

Eg：HTTP（提供 Web 文档的请求和传送）、SMTP（提供电子邮件报文的传送）、FTP（提供两个端系统之间的文件传送）、DNS（域名解析系统，将端系统的名字转换为 IP 地址）

【运输层】**应用程序端点间传应用层报文，叫报文段。**

Eg：TCP（面向连接服务+可靠性传递+流量控制（一段链路的发送方&接收方）+拥塞控制（源主机和目的主机））、UDP（无连接服务，无可靠性，无流量控制，无拥塞控制）

【网络层/IP 层】**主机间传数据报。**（源主机中的运输层协议向网络层递交运输层报文段和目的地址）

Eg：IP（定义了在数据报中的各个字段以及端系统和路由器如何作用于这些字段。IP 只有一

个，所有具有网络层的因特网组件必须运行 IP。)、决定路由的路由选择协议（根据该路由将数据报从源传输到目的地）

【链路层】**节点间传帧**。将整个帧从一个网络元素移动到邻近的网络元素。在每个节点，网络层将数据报下传给链路层，链路层沿着路经将数据包传递给下一个节点，在该下一个节点，链路层将数据报上传给网络层。

【物理层】**节点间按比特传帧。**将链路层移动的帧中的一个个比特从一个节点移动到下一个节点。

PS:【OSI 七层开放系统互连模型】应用层+**表示层（**使通信的应用程序能够解释交换数据的含义：数据压缩+数据加密+数据描述）**+会话层**（提供了数据交换的定界和同步功能：建立检查点+恢复方案的方法）+运输层+网络层+数据链路层+物理层

*****【注意】*****

主机/端系统可以实现所有五个层次

路由器实现物理层+链路层+网络层

链路层交换机实现物理层+链路层

【封装】

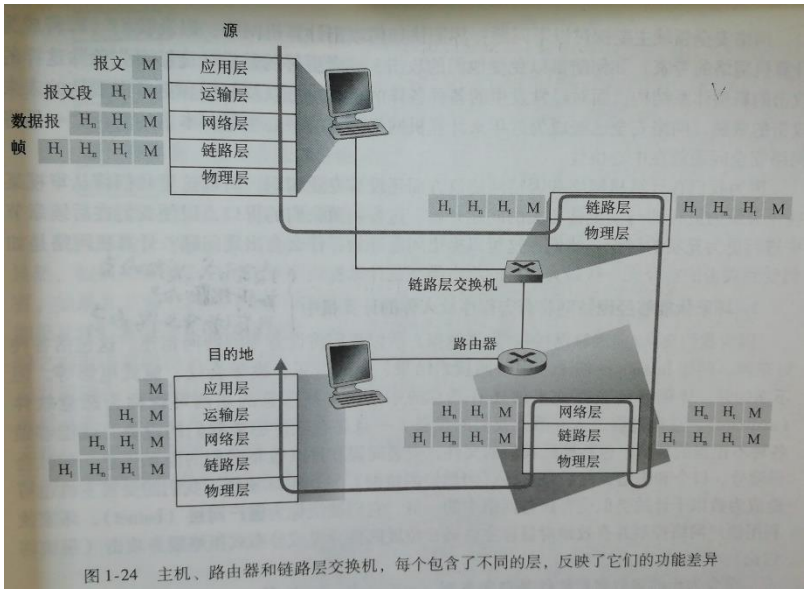

图 1-24　主机、路由器和链路层交换机，每个包含了不同的层，反映了它们的功能差异

【因特网标准】由因特网工程任务组 IETF 研发，IETF 的标准文档为请求评论 RFC。

【分布式应用程序】应用程序涉及多个相互交换数据的端系统

***因特网应用程序运行在端系统上，并不在网络核心中的分组交换机中!

【套接字接口】与因特网相连的端系统提供了一个套接字接口，该接口规范了运行在一个端系统上的程序请求因特网基础设施向运行在另一个端系统上的特定目的地程序交付数据的方式。因特网套接字接口是一套发送程序必须遵循的规则集合，因此因特网能够将数据交付给目的地。

*****【补充】*****

1.　重要的物理介质：

1）端系统/主机/服务器/工作站：凡是运行网络应用程序的均是 host，端系统上一定有五层协议来进行通信

2）通信链路：传输速率=带宽

3）路由器

【区别】

①.传播速率是给定介质固有的

②.传输速率=带宽，其可根据协议改变

2. 拓扑图是某一时段/某一时刻的，因为不论何时均有可能有宕机的路由器，拓扑图由路由器群组组成

3. 因特网由：端系统+路由器+通信链路+协议构成（前三为硬件，后一为软件）

4. 因特网有公有网&私有网，其是网络中的网络，有由逻辑地址进行管理的松散的层次结构，通过分布式应用服务来通信。

5. 物理介质是让比特在线路中传输的，其有两类：有导向介质（在固态介质中传播）、无导向介质（信号发散传播）

6. 【几种物理介质】

1）双绞线（双向传递）：3 类、5 类（带屏蔽 STP、不带屏蔽 UTP）

2）同轴电缆（双向传递）：基带 50Ω、宽带（HFC）75Ω

3）光纤：单模 SMF（快）、多模 NMF

4）无线介质：陆地微波、无限局域网 Wifi、无线广域网 3G、卫星通信（端到端时延 250-270ms）。无线介质受反射、障碍物阻挡、干涉的影响

【对比】

①光纤传输速率最高（协议决定），出错少（因为其是光传输而其他的是脉冲传输）、最便宜但是铺设线路贵，可以防窃听

②同轴电缆传播速率最高，光纤最低

7. 【网络核心三种传数据方式】

1）分组交换（最常用）：按字节流封装成分组、各分组共享网络资源（竞争成功的数据包使用满带宽）、资源按需分配。会产生队列+丢包，所以产生存储转发机制（完整数据包传输使必是经过存储转发的）。路由器会用路由算法算出转发表，分组到路由器时查该表确定将要被发送到的链路是哪条。

2）报文交换

3）电路交换：网络资源划分两方式：频分多路复用 FDM&时分多路复用 TDM（划分后一个线路可供多个用户用，实现了资源分配，但给每个用户的资源仍是被该用户独享，所以不算线路共享，而且每个用户所分到的资源/带宽都会变成 1/n，而且仍然存在线路空闲浪费的情况）。不会出现丢包。

【区别】

①报文交换：一次性把一个文件全传出去、线路可共享

②分组交换：以块为单位传数据，线路共享

③电路交换：传数据前要呼叫，传时独占线路。通信可保证速度，电路不共享，会产生空闲从而浪费线路（电话网络）

8. 端系统=主机=资源子网

路由器构成的网络/通信核心=通信子网

9. 计算机网络的拓扑结构主要是指通信子网的拓扑结构。因特网是不规则的网状拓扑。

10. 【计算机网络的性能】

【速率】=带宽=比特率=传输速率=数据率

【带宽】注意 M=2^{20}

【吞吐量】单位时间内通过某个网络的数据量

【信道利用率】某信道有百分之几的时间是被利用的（有数据通过），信道利用率并非

越高越好。

【总时延】决定因特网的网速。

***分组传输一定是整个分组都在路由器处理完了再一起推到信道。

***一般不考虑节点处理时延

***队列时延一定考虑，其依赖于拥塞程度

***低速链路要考虑传输时延，有线/无线网络中无需考虑传输时延

***传播时延一般也不考虑，而无线通信中要考虑传播时延，而且卫星传输的传播时延巨大

11. 往返时间 RTT 可用户测量网络的拥塞情况

12. Traceroute 程序“提供沿着源到目的地的网络路径的时延测量&分组丢失判断

13. 分组到达的路由器的队列要是满了则后续分组会丢失，丢失的分组可能会被发送方重传（TCP）or not（UDP）。

***注意：路由器只是转发分组，分组丢不丢失、出不出错，全是由端系统重传而不是路由器

14. 【网络协议】是为进行网络中的数据交换而建立的规则、标准或约定。这些规则明确规定了所交换的数据的格式以及有关的同步问题（语法+语义+同步）

【协议分层的好处】1.各层之间独立 2.灵活性好 3. 结构上可分割 4. 易于实现和维护 5.能促进标准化工作

【网络协议的三组成要素】

1.语法：数据与控制信息的结构或格式

2.语义：需要发出何种控制信息，完成何种动作以及做出何种相应

3.同步：事件实现顺序的详细说明

【协议的两种国际标准】

1.法律上的国际标准 OSI 没有得到市场认可

2.TCP/IP 被称为事实上的国际标准，得到广泛应用

15. 【封装】再数据包的头部 or 头部和尾部添加控制信息并封装来保证精准递交&安全通信，封装后叫 PDU（协议数据单元）

***只有数据链路层既加头部（控制信息）又加尾部（循环冗余码）

16. 三种控制信息：地址、差错校验、协议控制信息

17. 三种地址：端口（进程）地址→传输层

逻辑（IP）地址→网络层（可变）

物理（介质访问控制子程）地址→数据链路层（不可变）

***IP 地址通过 ARP 转发表转化成物理地址

***同一局域网/链路/网段中的主机通信无需 IP 地址（网络层），直接利用物理地址寻址来点对点传输即可。

***点对点传输只会改变数据链路层的 Mac（物理）地址，而 IP 地址是不变的

***数据链路层关心的是邮件接收方的人名（物理地址）、网络层关心的是地区（逻辑地址/目的 IP）

18. 【数据链路层 5 功能】

1）将数据分隔成帧

2）增加物理地址

3）流量控制

4）差错控制：数据链路层在尾部增加的循环冗余码

5）访问控制

【区别】

1.数据链路层的流量控制是一段链路两端节点（而不一定是端系统）的流量控制

2.之前运输层 TCP 的流量控制是端系统（发送方&接收方）之间的
*****数据链路层有责任完成点对点（一段链路的两端节点）的递交（到底走哪个路由器、丢没丢都不管，其只管把数据可靠的传给下一个节点），保证传给物理层的数据是无措的**
19. 传输层的分段只在目的接收方重组（因为路由器只负责转发，其没有传输层），每段都包含一个序号
20. 【OSI】：
1）应用层：提供接口以允许访问网络
2）表示层：完成信息语法和语义的定义，提供翻译、加密、数据压缩功能
3）会话层：建立、管理、终止会话
4）传输层：提供可靠的数据端系统到端系统的递交和差错恢复
5）网络层：将数据包从源传到目的主机，提供网络服务
6）数据链路层：将位组织成帧，提供点对点传输
7）物理层：通过介质传输比特流，提供机械和电气规范

应用层：

【研发网络应用程序的核心】写出能够运行在不同端系统和通过网络彼此通信的程序
***网络核心设备并不在应用层&传输层起作用，仅在低三层（网络层、链路层、物理层）起作用。所以应用软件限制在了端系统，促进了大量的网络应用程序的迅速研发和部署。
*****应用程序部署在端系统，向下层传递报文（message）**
*****【区别】*****
1.网络应用程序体系结构：由应用程序研发者设计，规定了如何在各种端系统上组织该应用程序。本质是进程之间的通信（IP+port 标识）
2.网络体系结构：固定的，为应用程序提供了特定的服务集合。

✧ 【3 种应用程序体系结构】
【客户-服务器体系结构】
【服务器】会话开始时等待联系的进程总是打开的主机，服务与来自许多其他客户的请求（往往是配备大量主机的**数据中心**用于创建强大的服务器。
【客户】发起通信（会话开始时发起与其他进程的联系）的进程
***客户的 IP 可以任意改变
【C/S 特点】***C/S 体系结构，客户相互之间不直接通信
***服务器具有固定的、周知的 IP 地址
***客户总能够通过向服务器的 IP 地址发送分组来与其联系
Eg：Web、FTP、Telnet、电子邮件
【对等 P2P 体系结构】
【P2P 特点】***对位于数据中心的专用服务器有最小的（或者没有）依赖。应用程序在间断连接的主机对之间使用直接通信，而不必通过专门的服务器通信。这些主机对被称为**对等方**。
***对等方并不为服务提供商所有，而为用户控制的桌面机和膝上机所有，大多数对等方驻留在家庭、大学、办公室。
*****自扩展性**：在一个 P2P 文件共享应用中，尽管每个对等方都由于**请求文件产生工作负载**，但每个对等方通过向其他对等方**分发文件也为系统增加服务能力**。

所以不需要庞大的服务器基础设施和服务器带宽。

Eg：文件共享(BitTorrent)、对等方协助下载加速器（迅雷）、因特网电话和视频会议（Skype）（流量密集型应用）

*****即使是 P2P，在一次文件共享的过程中，也分 C & S，下载文件的对等方为客户，上载文件的对等方为服务器。**

【混合体系结构】即时讯息应用：服务器被用于跟踪用户的 IP 地址，用户到用户的报文在用户主机之间（无需通过中间服务器）直接发送。

✧ 【进程通信】

【进程通信的 2 种方式】

1. 当多个进程运行在相同的端系统上时，使用**操作系统**提供的**进程间通信机制**相互通信

2. 当进程运行在不同的端系统上时，通过**跨越计算机网络交换报文**相互通信（发送进程生成并向网络中发送报文；接收进程接收这些报文并可能通过回送报文进行相应）

【进程与计算机网络之间的接口】

进程通过**套接字**的软件接口像网络发送报文和从网络接收报文。

【套接字】是同一台主机内应用层与传输层之间的 建立网络应用程序的 可编程接口，称为应用程序和网络之间的应用程序编程接口。

***程序开发者可以控制套接字在应用层端的一切（API），但是除了可以选择运输层协议以及设定几个运输层参数，对该套接字的运输层端几乎没有控制权

【进程寻址】

【标识接受进程的两种信息】**目的主机的 IP 地址（32bit 的量，能唯一标识主机）** & 目的主机中接受进程的标识符，即**目的地端口号（应用特定）**

*****【进程交到目的主机所需的地址】(3)**

IP 地址（确定发送方到接收方走哪个网段）+进程地址（端口地址、符点地址）（确定到目的主机后要把数据交到哪个应用进程）+物理地址（确定在每个网络中传送到下一个目的节点的地址）

✧ 【应用层**需要的**运输服务/运输层协议的通用服务】**(4)** 可靠数据传输、吞吐量、定时、安全性

【可靠数据传输】确保由应用程序的一端发送的数据正确、完全地交付给该应用程序地另一端（电子邮件、文件传输、远程主机访问、Web 文档传输、金融应用）（多媒体应用为**容忍丢失的应用**）

【可用吞吐量】运输层协议能够以某种特定的速率提供确保的可用吞吐量（发送进程能够向接收进程交付比特的速率）（**带宽敏感的应用：**多媒体应用）（**弹性应用**：电子邮件、文件传输、Web 文档传输）

【定时】（交互式实时应用（因特网电话、虚拟环境、电话会议、多方游戏）为了有效性而要求数据交付有严格的时间限制）

【安全性】在发送和接受进程之间提供**机密性**，以防该数据以某种方式在这两个进程之间被观察到。还包括**数据完整性、端点鉴别。**

✧ 【因特网提供的运输服务/两个运输层协议】

【TCP 服务】面向连接服务+可靠数据传输服务

【面向连接服务】在应用层数据报文开始流动**之前**，TCP 让客户和服务器相互交换运输层控制信息（握手过程）。握手后一个 TCP 连接在两个进程的**套接字之间**建立。该连接是全双工的，即连接双方的进程可以在此连接上同时进行报文收发。应用程序结束报文发送时，必须拆除该连接。

【可靠数据传输服务】通信进程能够依靠 TCP，无差错、按适当顺序交付所有发送的数据。当应用程序的一端将字节流传进套接字时，它能够依靠 TCP 将相同的字节流交付给接收方的套接字，没有字节的丢失和冗余。**(字节流→发送方套接字→接收方套接字)**

【拥塞控制机制】发送方和接收方之间的网络出现拥塞时，TCP 的拥塞控制机制会抑制发送进程。并且 TCP 拥塞控制也试图限制每个 TCP 连接，使它们达到公平共享网络带宽的目的。

【区别】

1.流量控制：**接收方**限制发送方发送速率

***流量控制只是在端系统之间的控制，与路由器无关

****接收方来不及处理→流量控制**

2.拥塞控制：**网络核心（路由器）**限制发送方发送速率

***拥塞控制只有路由器有（目前限制网络速率即带宽的主要就是路由器的拥塞控制）

****路由器来不及转发→拥塞控制**

***TCP 就是面向连接的服务，其可靠数据传输、流量控制、拥塞控制均是可选可不选的

【UDP 服务】

UDP 是一种不提供不必要服务的轻量级运输协议，它仅提供最小服务。

UDP 是无连接的，两进程通信前没有握手过程

UDP 提供一种不可靠数据传送服务，不保证发送方发送进 UDP 套接字的报文将到达接收方，而且到达接受进程的报文也可能是乱序的。

UDP 没拥塞控制机制，其发送端可以用它选定的任何速率向网络层注入数据。

****UDP 可做到瞬间实时性：很好地适应了流媒体技术（但注意这个瞬间实时性仅仅是从其发送节点到第一级路由器是瞬间发送地，到下一个路由器就可能出现拥塞、丢包了）**

【因特网运输协议不提供的服务】吞吐量 & 定时保证

****【总结】***

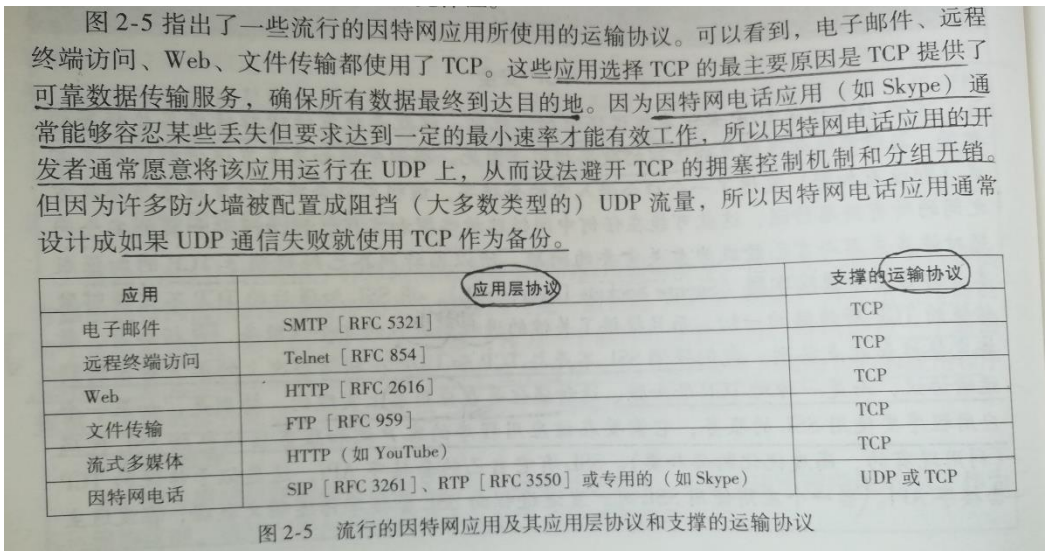

图 2-5 指出了一些流行的因特网应用所使用的运输协议。可以看到，电子邮件、远程终端访问、Web、文件传输都使用了 TCP。这些应用选择 TCP 的最主要原因是 TCP 提供了可靠数据传输服务，确保所有数据最终到达目的地。因为因特网电话应用（如 Skype）通常能够容忍某些丢失但要求达到一定的最小速率才能有效工作，所以因特网电话应用的开发者通常愿意将该应用运行在 UDP 上，从而设法避开 TCP 的拥塞控制机制和分组开销。但因为许多防火墙被配置成阻挡（大多数类型的）UDP 流量，所以因特网电话应用通常设计成如果 UDP 通信失败就使用 TCP 作为备份。

应用	应用层协议	支撑的运输协议
电子邮件	SMTP［RFC 5321］	TCP
远程终端访问	Telnet［RFC 854］	TCP
Web	HTTP［RFC 2616］	TCP
文件传输	FTP［RFC 959］	TCP
流式多媒体	HTTP（如 YouTube）	TCP
因特网电话	SIP［RFC 3261］、RTP［RFC 3550］或专用的（如 Skype）	UDP 或 TCP

图 2-5　流行的因特网应用及其应用层协议和支撑的运输协议

✧ 【应用层协议】

【应用层协议】其定义了运行在不同端系统上的应用程序进程如何相互传递报文

1.交换的报文类型（请求报文 & 响应报文）

2.各种报文类型的语法（报文的各字段）

3.字段的语义（字段中信息的含义）

4.确定一个进程何时以及如何发送报文，对报文进行响应的规则。

*****【区别】***应用层协议是网络应用的一部分**

✧ Web & HTTP

*****【总结】*****

万维网 Web

应用程序体系结构：C/S

应用层协议：HTTP（带流水线的持续连接）（无状态）

运输层协议：TCP

【HTTP（80）超文本传输协议】：Web 的核心，由两个程序实现：客户程序&服务器程序。（客户程序和服务器程序运行在不同的端系统中，通过**交换 HTTP 报文**进行会话。）HTTP 定义了这些**报文的结构**以及**客户和服务器进行报文交换的方式**。/HTTP 定义了 **Web 客户向 Web 服务器请求 Web 页面的方式**以及**服务器向客户传送 Web 页面的方式**

【Web 页面/文档】：由若干对象（HTML 基本文件和引用对象）构成

【对象】一个对象是一个文件，eg：HTML 文件、JPEG 图形、java 小程序、视频片段等，对象均可以通过一个 URL 地址寻址。

*****多数 Web 页面含有一个 HTML 基本文件&几个引用对象。注意一个 HTML 网页也是一个对象。只有把 HTML 网页取回来之后才能再继续串行或并行来取其他引用对象。**

*****打开网页，看到的仅仅是对象的链接，对象并没有存在主机中，点开链接后才发送请求将其载入主机**

*****两主机之间建立好 TCP 连接之后，客户再传路径信息，来 get 文件**

【URL 地址】统一资源定位符。互联网上的每个对象有唯一的 URL，其由两部分组成：存放对象的服务器主机名&对象的路径名

*****HTML 基本文件通过对象的 URL 地址引用页面中的其他对象。**

*****Web 浏览器实现了 HTTP 客户端，Web 服务器（总是打开，有固定 IP）实现了 HTTP 服务器端，它用于存储 Web 对象，每个对象由 URL 寻址。**

【流程】：客户以 80 端口向服务器建立 TCP 连接→发送 HTTP 请求报文→服务器收到请求并封装文件发送响应报文→TCP 连接断开

【无状态协议】：HTTP 服务器不保存客户信息，其只负责向客户发送被请求的文件

【应用程序的两种连接方式】持续连接&非持续连接

【持续连接】:所有请求及其相应经相同的 TCP 连接发送。服务器发送对象后保持 TCP，用于响应后续请求，直到超时 **（HTTP 默认为带流水线的持续连接）**

【流水线】：在建立 TCP 后，客户连续发送请求，而不必等待服务器响应。

【带流水线的持续连接】：N 个对象需 N+1 个 RTT

【非持续连接】：每个请求/响应对是经一个单独的 TCP 连接发送。（每个 TCP 连接只传输一个请求报文和一个响应报文）服务器发送每个对象后立即断开 TCP 连接。（每个对象共 2 次 RTT）

***非持续连接必须为每一个请求的对象建立和维护一个全新的连接。每个这样的连接，在客户和服务器中都要分配 TCP 的缓冲区和保持 TCP 变量，给 Web 服务器带来严重的负担。且每个对象经受两倍的 RTT 交付时延，一个用于创建 TCP,另一个用于请求和接收一个对象。

【往返时间】（从客户请求 HTML 基本文件起到该客户收到整个文件止所花费的时间）一个短分组从客户到服务器然后再返回客户所花费的时间。RTT=分组**传播时延**+路由器&交换机**排队时延+处理时延**

【三次握手】

1.客户向服务器发送一个小 TCP 报文段

2.服务器用一个小 TCP 报文段做出确认和响应

3.客户向服务器返回确认

*****前两步花费一个 RTT，第三次握手确认的同时向该 TCP 连接发送一个 HTTP 请求报文；请求报文到达服务器，服务器在该 TCP 连接上发送 HTML 文件，该 HTTP 请求/响应用去了另一个 RTT。**

【串/并行连接】：配置浏览器以同时与服务器建立多个 TCP 连接

【HTTP 报文类型】(2) 请求报文&响应报文**（所有应用层协议都只有请求&响应这两种消息类型）**

【HTTP 请求报文】：（ASCII 码，换行/r/n）

请求行：GET/POST/PUT/HEAD/DELETE　文件路径 HTTP/1.1

首部行：主机名，是否保持连接，浏览器版本，语言等

空行

实体体（提交的表单）

【HTTP 响应报文】：

状态行：HTTP/1.1 200 OK

首部行：操作系统，发送时间，最后修改时间，对象大小、对象类型等

空行

实体体（返回的对象）

【基于 HTTP 协议的三种工作机制】cookies，Web 缓存，条件 GET 方法

【用户与服务器的交互 Cookie】

【cookie 技术四组件】

1.在 HTTP 相应报文中的一个 cookie 首部行

2.在 HTTP 请求报文中的一个 cookie 首部行

3.在用户端系统中保留有一个 cookie 文件，并由用户的浏览器进行管理

4.位于 Web 站点的一个后端数据库

【过程】服务器第一次响应 HTTP 时在数据库创建用户，并发回 set-cookie 码；浏览器在本地记录 cookie 码，并在后续请求里将其加入至首部行

***cookie 可以用来标识一个用户。用户首次访问一个站点时，可能需要提供一个用户标识，在后继绘画中，浏览器向服务器传递一个 cookie 首部，从而向该服务器标识了用户。因此 cookie 可以在无状态的 HTTP 之上建立一个用户会话层。

【Web 缓存】：代替 web 服务器来满足 HTTP 请求。以**减少响应时间，节省流量，降低费用**

【Web 缓存器/代理服务器 proxy】：能够代表初始 Web 服务器来满足 HTTP 请求的网络实体。Web 缓存器有自己的磁盘存储空间，并在存储空间中保存最近请求过的对象的副本。

【流程】：客户向 web 缓存服务器建立 TCP 连接并发送 HTTP GET 请求→缓存服务器向目标服务器建立 TCP 连接并发送 HTTP 请求→缓存服务器收到响应后，在本地存储一个副本，并向客户发送 HTTP 响应

***Web 缓存器既是服务器又是客户（接受浏览器的请求时是服务器，向初始服务器发出请求并接受响应时是客户）

【条件 GET】允许缓存器证实它的对象是最新的 的 HTTP 协议的一种机制

【流程】缓存服务器向目标服务器发送 GET 请求时，首部行 if-modified-since 标注本地缓存的对象最新修改日期。目标服务器收到后检查若没有产生更新，则返回 304。缓存服务器便可将当前缓存的文件直接发送给客户。

***仅当需要在缓存中取该对象的时候才用 get 检查它是否最新，没被调用的对象不检查

✧ 电子邮件

*****【总结】*****

应用程序体系结构：C/S（该体系结构的核心为 邮件服务器 其既是 C 又是 S）

应用层协议：SMTP（电子邮件的核心）（持续连接）

运输层协议：TCP

【电子邮件三组成部分】用户代理+邮件服务器（电子邮件体系结构的核心）+简单邮件传输协议（SMTP）

【邮箱】在邮件服务器上有邮箱，以管理和维护报文

【SMTP（25）简单邮件传输协议】：SMTP 是因特网电子邮件的核心，使用 TCP 可靠数据传输服务，**从发送方的邮件服务器向接收方的邮件服务器发送报文**。SMTP 有两部分（运行在发送方邮件服务器的客户端&运行在接收方邮件服务器的服务器端）

*****每台邮件服务器上既运行着 SMTP 的客户端也运行着 SMTP 的服务器端**

*****大多数应用层协议都是有 C 和 S 两部分**

*****【HTTP&STMP 对比】*****

1.SMTP 要求每个报文采用 7 比特 ASCII 码格式，若某报文包含了非该格式的字符或二进制数据（图形文件），则该报文必须按照该格式进行编码。HTTP 无该限制。用其传输邮件之前需要将二进制多媒体数据编码位 ASCII 码，并且在使用 SMTP 传输后要求将相应的 ASCII 码邮件解码还原为多媒体数据。

2. SMTP 是推协议（发送邮件服务器把文件推向接收邮件服务器，这个 TCP 连接是由要发送该文件的机器发起的）；HTTP 为拉协议（用户使用 HTTP 从服务器拉取想要的信息，TCP 连接是由想接收文件的机器发起的）

*****所以不能用 SMTP 来取邮件，因为取邮件是一个拉操作，而 SMTP 是推操作**

3.多媒体处理：SMTP 把所有报文对象放在一个报文中；HTTP 把每个对象封装到它自己的 HTTP 响应报文中

（4.使用专有邮件服务器：如果客户端邮件服务器没开，那么发送方邮件服务器会一直尝试与其进行 TCP 连接，所以如果发送失败提示比较慢；使用 Web 邮件服务器，发送方邮件服务器到接收方邮件服务器之间的 TCP 连接只会试一次，如果连接失败直接就提示发送失败。）

【邮件服务器】：缓存待发送（或发送失败）、已接收的邮件

【流程】：A 编写邮件并提供 B 的邮件地址→A 的用户代理使用 SMTP 协议将报文发至 A 的邮件服务器，并存入报文队列→A 的邮件服务器向 B 的邮件服务器建立 TCP 连接→发送 SMTP 报文→B 的邮件服务器将报文分发至 B 的邮箱→B 调用用户代理从邮件服务器中查看邮件

【发送失败】：若 A 的邮件服务器无法与 B 的邮件服务器建立连接，则会给 A 提示

PS“：提示“发送成功”消息后，又退信的可能：

1.发送方邮件服务器队列已满，所以一开始消息进入队列后又被挤出来了

2.用户代理（发送者使用的 PC 等）和发送方邮件服务器之间的 TCP 连接建立失败（注意用户和其用户代理之间的 TCP 连接一定是成功的，因为消息已经到了用户代理才会提示成功）

3.接收方的地址填错了

【SMTP 报文】：helo，mail from，rcpt to，data . ，quit

【邮件报文】：from，to，subject

【MIME】：多媒体邮件扩展协议（因为 SMTP 只能传送 7 位 ASCII 编码的文本文件类型的邮件，所以对其扩展，MIME 可以传的邮件类型包括：文本、图片、音频、视频、应用）（如果一封邮件里包含多个类型，则需要设置边界，并在每种类型开始之前加上该边界以示区分）

【邮件访问/接收协议】**（3）**第三版的邮局协议 POP3（下载并删除，下载并保留），因特网邮件访问协议 IMAP（文件夹/目录管理），Web 浏览器（使用 HTTP 协议收发邮件）

***收邮件必须是人为主动的去**拉**操作，而不是邮件直接给客户的

【对比】

1.SMTP 用来将邮件从**发送方邮件服务器**传输到**接收方邮件服务器**&从**发送方的用户代理**传送到**发送方的邮件服务器**

2.POP3、IMAP 将邮件从**接收方的邮件服务器**传送到**接收方的用户的代理**

3.HTTP 既可以用来发邮件也可以用来收邮件

***SMTP：25 端口

***POP3：110 端口

【POP3 的特点】：

1.需要登陆认证

2.输入的用户名和密码均是明文显示

3.两种事务处理方式：下载并删除，下载并保留

4.会话中不携带状态信息（但会记录哪些用户报文被删除了，因为 dele 时是逻辑删除，quit 退出时才会真正物理删除

【注意】

1.只要是用浏览器收发信，则发送端→邮件服务器&邮件服务器→接收端都是 HTTP 协议；而发送方邮件服务器→接收方邮件服务器之间是 SMTP 协议

2.若是同一邮箱之间发送邮件（比如都是 126 的邮箱），那么无需 SMTP，因为系统管理员可以直接设置让两端邮箱直接彼此发送接收。

3.SOCKET 套接字是应用层的，因为其可选用 TCP/UDP 这两种运输层协议

FTP（了解即可）（20 数据连接、21 控制连接）文件传输协议（明文传输）：TCP

报文形式：ASCII 码

流程：用户以 21 端口与主机建立 TCP 连接→发送用户名、密码和指令→服务器以 20 端口向用户建立 TCP 连接→进行数据传输→关闭 20 端口的连接

带外传送：数据与控制分离（两个端口）

状态保留：服务器在会话期间保留用户状态，始终保持 21 端口 TCP 连接

✧ **DNS**（53）域名解析系统

*****【总结】*****

应用程序体系结构：C/S（因为 DNS 是分布式数据库层次结构，数据库就是服务器，查数据库的就是客户端）

应用层协议：DNS

运输层协议：UDP（因为 UDP 没有握手，可以实现瞬时的快速发送）

【DNS 定义】DNS 是：

1.一个由份曾的 DNS 服务器实现的分布式数据库

2.一个使得主机能够查询分布式数据库的应用层协议。

***DNS 是应用层协议，但它往往是由其他应用层协议所使用，包括 HTTP\SMTP\FTP，用于将用户提供的主机名解析成 IP 地址

【DNS 四服务】

1.进行主机名到 IP 地址的映射

2.主机别名：Web 服务器一个主机只能有一个**规范主机名（唯一特定）**，但可以有多个**别名（可重复、可多个）。一个规范主机名可对应多个别名。**

3.邮件服务器别名：邮件服务器同上**（允许一个公司的邮件服务器和 Web 服务器使用相同的主机名（因为别名可以相同））**

4.负载分配：一个 IP 地址**集合**对应**一个**规范主机名（所以多个 Web 服务器/邮件服务器可有相同的别名）（所以如果同寝室访问同一网站，有可能会在不同的服务器上处理，因为每个服务器都会有一个上限，访问的时候会分流）

【集中式设计的问题】

1.单点故障

2.通信容量

3.远距离的集中式数据库

4.维护

***所以 DNS 是在因特网上实现的分布式数据库。DNS 使用大量服务器，它们以层次方式组织且分布在全世界范围内。

【三种类型的 DNS 服务器/分布式层次数据库】/根 DNS 服务器（13 个）、顶级域服务器（多级）、权威 DNS 服务器

【根 DNS 服务器】提供 TLD 服务器的域名和其对应的 IP 地址

【顶级域 DNS 服务器 TLD】提供权威 DNS 服务器的域名和其对应的 IP 地址

【权威 DNS 服务器】提供能对外提供服务的域名和其对应的 IP 地址（**权威 DNS 是各个网站自己建立的，网站提供的所有服务以及域名&IP 的对应关系均存在权威 DNS 上）**

Eg：

查 www.sina.com 的 IP；

- **在根 DNS 服务器找管理 .com 的顶级域 DNS 服务器**
- **在顶级域 DNS 服务器找管理 sina.com 的权威名称服务器(权威名称服务器安置在网站)**
- **在权威名称服务器找 www.sina.com 的 IP 地址。**
- **(上→下，后→前)**

*****注意实际上 DNS 可能比三层更多，比如如果最后是 com.cn，那么先找管理.cn 的顶级域服务器，然后再找管理.com.cn 的子顶级域服务器**

*****根名称服务器有 13 个**

*****顶级域服务器有两类：负责国家级别（uk/fr/ca/jp）/负责商业领域(com.org/net/edi/gov)**

【本地 DNS 服务器】

1.本地 DNS 服务器不属于服务器层次结构

2.每个 ISP 因特网服务提供商都有一台本地 DNS 服务器**（默认名字服务器）**

3.当主机发出 DNS 请求时，该请求被发往本地 DNS 服务器，其起着**代理**作用。

4.只要是主机的请求都先在本地 DNS 服务器中查找。其会缓存所有查询过的信息。如果本地服务器有，那么就可以越过根名称服务器直接从顶级域名称服务器中找。如果没有那么本地名称服务器会将请求转发到 DNS 服务器层次结构中进行查找。(本地名称服务器既是客户端又是服务器)

【DNS 查询方法】

【递归查询】：用户→本地 proxy

【迭代查询】：本地 proxy→根，本地 proxy→顶级域，本地 proxy→权威

*****DNS 查询=递归查询+迭代查询**

*****从请求主机到本地 DNS 服务器的查询是递归的，其余的查询是迭代的**

【DNS 查询流程】获取主机名→向本地 DNS 服务器发送解析请求→（若没有）本地 DNS 服务器向根 DNS 服务器发送解析请求→本地 DNS 服务器请求顶级域服务器→本地 DNS 请求权威 DNS 服务器→本地 DNS 服务器缓存所有查询记录并返回结果给用户

主机名和 IP 地址的转换：TCP（区域传送）、UDP（域名解析）

主机别名：DNS 提供主机或邮件服务器的别名和规范主机名的映射

负载分配：DNS 为多服务器站点映射不同的 IP 地址，以实现负载平衡

本地 DNS 服务器：（ISP 提供，IP 地址通过 DHCP 告知用户）用于本地 DNS 请求的缓存

【DNS 记录】（name，value，type，ttl）

ttl：当前记录存活的时间

【四种资源管理类型 RR】即 type 四种（决定 name，value 的含义）：

A：主机名→IP 地址

NS：域名→能查到该域主机 IP 地址的权威服务器主机名

CNAME：别名→规范主机名

MX：邮件服务器别名→邮件服务器规范主机名

【重点】

1.根存顶级，顶存权威，权存具体

2.到根名称服务器查的时候，根会返回**两**类记录：**一类是 NS 记录**（得到可以解析.com 的顶级域名称服务器的域名），**一类是 A 记录**（得到 NS 记录中的那个顶级域服务器域名所对应的 IP 地址。往往一个域名会得到两条 IP 地址，因为 IP 地址有两种格式，一种一个）

3.如果是要在本地名称服务器上查询记录，注意第一行域名之后不要写任何东西，因为 nslookup 默认是在本地服务器查询。如果域名后写了东西就代表要从外面查询了。

4.每次查到的信息都只是这一个时刻的记录，不同时刻查到的记录数量或者 IP 地址都有可能不同

Eg：

```
C:\>nslookup -qt=ns cn 198.41.0.4
cn      nameserver = E.DNS.cn
cn      nameserver = B.DNS.cn
cn      nameserver = C.DNS.cn
cn      nameserver = A.DNS.cn
cn      nameserver = D.DNS.cn
cn      nameserver = NS.CERNET.NET
A.DNS.cn        internet address = 203.119.25.1
A.DNS.cn        AAAA IPv6 address = 2001:dc7::1
B.DNS.cn        internet address = 203.119.26.1
C.DNS.cn        internet address = 203.119.27.1
D.DNS.cn        internet address = 203.119.28.1
D.DNS.cn        AAAA IPv6 address = 2001:dc7:1000::1
E.DNS.cn        internet address = 203.119.29.1
NS.CERNET.NET   internet address = 202.112.0.44
```

这个是.cn 后面给定一个根名称服务器的 IP 地址，来说明要在这个根服务器里面查。下面得到的上半部分是 NS 类型的记录，下半部分是 A 类型的记录。然后再 NS 中随意选一个域名进行下一步查找。（如果是在本地查，第一行就写成 nslookup *qt=ns cn）

```
C:\>nslookup -qt=ns edu.cn c.dns.cn
Server:  c.dns.cn
Address:  203.119.27.1

edu.cn  nameserver = dns.edu.cn
edu.cn  nameserver = ns2.cuhk.hk
edu.cn  nameserver = ns2.cernet.net
edu.cn  nameserver = dns2.edu.cn
edu.cn  nameserver = deneb.dfn.de
dns.edu.cn      internet address = 202.112.0.35
ns2.cuhk.hk     internet address = 137.189.6.21
ns2.cernet.net  internet address = 202.112.0.33
ns2.cernet.net  AAAA IPv6 address = 2001:250:c006::2
dns2.edu.cn     internet address = 202.112.0.13
deneb.dfn.de    internet address = 192.76.176.9
```

这个 edu.cn 后面是在上面随便选的一个域名，因为是域名，所以要先解析出 IP 地址，然后再查询，所以下面两行是解析 IP 地址的过程。其他同上.（如果是在本地查，第一行就写成 nslookup *qt=ns edu.cn）

【DNS 报文】：请求和回答结构相同，12 字节头部+问题+回答+权威+附加

【注册域名】：向 DNS 数据库插入记录（域名，权威服务器名，NS）、（权威服务器名，IP，A）

***创建一个新的网站时，要找到.com 的顶级域名称服务器，在这里面交钱注册，注册信息有两条，分别是 NS 类型和 A 类型，让这个顶级域名称服务器存这两条注册信息：**

(networkutopia.com, dns1.networkutopia.com, NS)（该网站的域名与可以解析这个网站的权威 DNS 的域名）

(dns1.networkutopia.com, 212.212.212.1, A)（该网站的域名与该网站的 IP）

***************以上均是 C/S 架构，且除 DNS 外均是 TCP 连接**************************
***************P2P 是另一种对等架构，了解即可*************************************
P2P（文件共享，下载器，skype）、混合（即时通讯）
【P2P 特点】:
1.对等方并不为服务提供商所拥有，而是受用户控制的桌面计算机和膝上计算机；
2.没有服务器；
3.任意两个点都可以直接通过 **TCP** 连接；
4.对等点可以间歇性连接并且可以改变 IP 地址
***在客户-服务器文件分发中，该服务器必须向每个对等方发送该文件的一个副本，服务器承受了极大的负担，并且消耗了大量的服务器带宽；而在 P2P 文件分发中，每个对等方能够像其他任何对等方重新分发它已经收到的该文件的任何部分，从而在分发过程中协助该服务器。
【分发时间】所有 N 个对等方得到该文件的副本所需要的时间。（分发时间取决于每个对等方如何向其他对等方分发该文件的各个部分）
***P2P 具有自扩展性：对等方除了是比特的消费者还是它们的重新分发者
P2P 文件分发协议 BitTorret：对等方即是接收者又是分发者
文件分为块（chunk），对等方下载不同的块，同时为其他对等方上载所拥有的块，直至文件下载完毕。
算法原则：下载稀缺资源优先，上传给向我传送速度最快的节点，随机发送给其他节点
具体过程见纸质 PPT

（从应用层角度向下看套接字）
套接字：进程与计算机网络之间（运输层和应用层之间）的编程接口（应用层调用运输层服务）
***套接字有推（客户端通过套接字将消息推到链路）/拉（从链路将消息通过套接字拉到客户端）
***同主机的通信：套接字由 OS 控制；异主机的通信：套接字由协议控制（所以如果想捕服务器和客户端的包，必须分别在两台机器上运行二者，不然同台机器直接 OS 就控制了，根本不经过网卡，捕不到包）
*****【区别】TCP 套接字编程&UDP 套接字编程*****
1.TCP:
1)通信之前服务器端必须先运行，服务器端会先创建欢迎套接字并等待用户访问（欢迎套接字是单一进程串行，始终存在）
2）客户端建立套接字时要写入 IP 地址和端口号。客户端创建套接字的时候，就创建了与服务器之间的 TCP 连接。客户端之后传数据的时候就只需要发数据本身即可，不需要再每次都写上 IP 地址和端口号了。
3）当客户请求到达的时候，服务器会为每条请求新建一个套接字，然后服务器回传完消息这个套接字就关闭了。所以客户端可以并发同时相应多个客户端的请求。(并行，用完即关)
4）四元组标识（源 IP，源 port，目的 IP，目的 port）
2.UDP:
1)没有握手过程，所以服务器端无需先运行起来
2）发送方每次发送一个分组都要包含数据+目的 IP+目的端口号。

3）只要给相同服务器发消息，即只要目的 IP+目的端口号相同，那么就用同一个服务器套接字接收（串行，始终存在）,而不是每个进程都要给其新建一个套接字。
4）二元组标识（目的 IP，目的 port）

套接字编程：
UDP 套接字由目的主机 IP、目的进程端口号唯一标识
UDP 客户端：指定目的主机名（或 IP 地址）和端口号→创建套接字→发送报文
UDP 服务器：指定自己的端口号→绑定套接字→等待请求
TCP 套接字由源 IP、源端口、目的 IP、目的端口唯一标识
TCP 客户端：指定目的主机名（或 IP 地址）和端口号→创建套接字→**发送连接请求**（进行三次握手）→发送报文
TCP 服务器：指定自己的端口号→创建欢迎套接字，等待→收到连接请求，（创建新线程）**创建新的套接字为此客户专用**

运输层：

**运输层位于应用层和网络层之间，其为运行在不同主机上的应用进程提供直接的通信服务。运输层调用网络层主机间通信服务，向上提供进程的逻辑通信服务
【运输层协议】运输层协议为运行在不同主机上的应用进程之间提供了**逻辑通信功能**（应用进程只需要使用运输层提供的逻辑通信功能彼此发送报文，而无需考虑承载这些报文的物理基础设施的细节）
**运输层协议在端系统实现（因为路由器这种网络核心没有运输层），封装端口号（进程地址），向下传递报文段（segment）
【区别】
1.网络层提供了**主机**之间的**逻辑通信（源 IP,目的 IP）(源与目的节点之间链路的每两个节点之间传输时这两个值始终不变)**；网络层的分组叫**数据报**
2.运输层为运行在**不同**主机上的**进程**之间提供了**逻辑通信(源 IP,源端口号，目的 IP,目的端口号)（源与目的节点之间链路的每两个节点之间传输时端口号会改变）**；运输层的分组叫**报文段**
***注意：网络路由器仅作用于该数据包的网络层字段！路由器不检查封装在该数据报的运输层报文段的字段！即在端系统中，运输层协议将来自应用进程的报文移动到网络边缘（网络边缘即网络层），但对有关这些报文在网络核心如何移动并不作任何规定。中间路由器既不处理也不识别运输层加在应用层报文的任何信息。**
***但!运输层协议能够提供的服务受制于底层网络层协议的服务模型。因为网络层协议无法为主机之间发送的运输层报文段提供时延&带宽保证，所以运输层也无法提供时延&带宽保证!，但是即使网络层 IP 协议不可靠，运输层仍能够为应用程序提供可靠的数据传输服务、拥塞控制和流量控制(均是 TCP 提供的)**
***一个**进程**由一个或多个套接字，所以发送方的进程数据必须通过套接字从进程传到运输层，接收方运输层也是将**数据交给套接字，套接字再将数据给进程**。
【两进程通信过程】发送端将从应用进程（通过套接字）接收到的报文划分成小块，加上头部，形成运输层**报文段**，向下传递给网络层，网络层将其封装成网络层分组（数据报）并向目的地发送；接收端网络层从数据报中提取运输层报文段，并将该报文段向上交给运输层，

运输层处理接收到的报文段，使该报文段中的数据为接受应用进程使用。
【多路复用】： 在源主机从不同套接字中收集数据块，并未每个数据块封装上首部信息从而生成报文段，然后将报文段传递到网络层
***运输层多路复用的要求：1.套接字有唯一标识符；2.每个报文段有特殊字段（源端口号字段+目的端口号字段）来指示该报文段所要交付到的套接字
【多路分解】： 将运输层报文段中的数据交付到正确的套接字 分配一个端口号，当报文段到达主机时，运输层检查报文段中的目的端口号，并将其定向到相应的套接字，然后报文段中的数据通过套接字进入其所连接的进程。
***分解服务的实现过程：在主机上的每个套接字 nengou
【端口号】：16bit，范围 0~65535，0~1023 为周知端口号。开发应用程序时必须为其分配一个>1024 的端口号
****所有进程都有自己的套接字及相应的端口号**

【UDP】

*****【对比】*****
1.多路分解过程中的套接字确定：
1) UDP 套接字由二元组（目的 IP 地址，目的端口号）标识（所以如果两个 UDP 报文段有相同的目的 IP 和目的端口号，那么这两个报文段就将通过相同的目的套接字被定向到相同的目的进程），使用全部 2 个值来将报文段定向分解到相应的套接字。
2) TCP 套接字由四元组（源 IP 地址，源端口号，目的 IP 地址，目的端口号）标识（所以每个 TCP 连接，使用全部 4 个值来将报文段定向分解到相应的套接字（任意一个值不同就会被定向到不同的套接字，每一个 TCP 连接都会为其创建不同的套接字，以为其特定使用）
2.有新连接请求的时候套接字的创建情况：
1） UDP 若新连接请求的二元组和之前的相同，则不会再创建新的 UDP 套接字，所以多个进程串行通过一个 UDP 套接字（始终有，一个，串行）。如果二元组不同则会创建新的套接字，但只要二元组同进程就只能串行通过其相应的套接字，而不是每个进程都有一个专有的套接字。
2） TCP 始终有一个欢迎套接字（始终有，一个，串行）来等待其他进程的连接，一旦有一个进程请求连接，则会根据其四元组为其创建一个新的套接字（即用即创，用完即灭，多个并行），服务器主机可以支持很多并行的 TCP 套接字，每个套接字与一个进程相联系，并尤其四元组来表示每个套接字。对每一队请求/响应创建一个新的套接字并在数据传输结束吼关闭。
3. 所提供的服务：
1） UDP：提供运输层协议的最低服务，即复用/分解功能+少量的差错检测（不纠错!）
2） TCP：
4. 对应用层的控制：
1) UDP 只要应用进程将数据传给 UDP,UDP 就会将此数据打包进 UDP 报文段并立即将其传递给网络层
2) TCP 有拥塞控制机制，所以当源和目的主机间的一条或多条链路拥塞时会遏制运输层 TCP 发送方。并且 TCP 会重传报文段直到目的主机收到报文并确认，所以 TCP 不考虑可靠交付所需要的时间。
5. 连接的建立：
1） UDP 在数据传输之前无握手，不会引入建立连接的时延
2） TCP 在数据传输之前要三次握手，会引入建立连接时延，但这能保证可靠性

6. 连接状态：

1) UDP 不维护连接状态，也不跟踪参数

2）TCP 要在端系统中维护连接状态。此连接状态包括接收和发送缓存、拥塞控制参数以及序号与确认号的参数。

7. 分组=首部+数据

1）UDP 首部 8 字节，报文段结构=源端口号+目的端口号+长度+检验和+应用数据报文（32bit=8+24）

2）TCP 首部 20 字节，报文段结构=

应用	应用层协议	下面的运输协议
电子邮件	SMTP	TCP
远程终端访问	Telnet	TCP
Web	HTTP	TCP
文件传输	FTP	TCP
远程文件服务器	NFS	通常 UDP
流式多媒体	通常专用	UDP 或 TCP
因特网电话	通常专用	UDP 或 TCP
网络管理	SNMP	通常 UDP
名字转换	DNS	通常 UDP

图 3.6　流行的因特网应用及其下面的运输协议

【UDP 用户数据报协议】无连接服务（因为使用 UDP 在发送报文段之前，发送方和接收方的运输层实体之间没有握手）：不可靠，不可调节流量，不处理分组丢失、超时、乱序

【UDP 的服务内容】：多路分解和复用，少量的差错检验

【UDP 的优势】：无需建立连接（因此不会引入建立连接的时延，此为 DNS 运行在 UDP 之上的主要原因），无连接状态（所以 UDP 可以支持更多的活跃用户），分组头部字段开销小，关于发送什么数据以及何时发送的应用层控制更为精细（即应用层更可控，实时应用适合 UDP）

【UDP 报文段】：源端口、目的端口、报文总长度（字节）、校验和、应用层数据

【UDP 差错检验】：16 位累和，溢出则回卷，最后取反码，作为校验码**（UDP 无纠错能力，**其只能丢弃受损报文段或者将受损的报文段交给应用程序并给出警告**）**

【UDP 传输数据过程】

UDP 从应用进程得到数据，附加上用于多路复用/分解服务的源和目的端口号字段，以及两个其他的小字段，然后将形成的报文段交给网络层。网络层将运输层报文段封装到一个 IP 数据报中，然后尽力而为地将此报文段交付给接收主机。如果该报文段到达接收主机，UDP 使用目的端口号将报文段中的数据交付给正确的应用进程。

【DNS 应用 UDP 过程】

当一台主机中的 DNS 应用程序想要进行一次查询时，它构造了一个 DNS 查询报文并将其交给 UDP.无需执行任何与运行在目的端系统中的 UDP 实体之间的握手，主机端的 UDP 为此报文添加首部字段，然后将形成的报文段交给网络层。网络层将此 UDP 报文段风装进一个 IP 数据报中，然后将其发送给一个名字服务器。在查询主机中的 DNS 应用程序则等待对该查询的相应。如果它没有受到响应，则要么试图向另一个名字服务器发送该查询，要么通知调用的应用程序她不能获得响应。

****

【可靠数据传输】：建立在不可靠信道之上，向上层提供服务，适用于一般的计算机网络

rdt：可靠数据传输协议
udt：不可靠数据传输协议
接口：上层调用发送 rdt_send()，下层调用接收 rdt_rcv()
依赖：调用下层发送 udt_send()，调用上层接收 deliver_data()
***以下为 rdt 协议，紫色标注为每次新增的内容!
【Rdt1.0】：假设下层信道完全可靠。收到分组后只进行打包或解包，然后直接递交。
【Rdt2.0/自动重传请求协议 ARQ（停等协议）】：假设下层信道可能出现比特差错（接收方一定会收到分组）。
***该报文协议使用了控制报文肯定确认 ACK 和否定确认 NAK，接收方可以让发送方知道哪些内容被正确接收，哪些有误需要重传。基于 ACK&NAK 的重传机制的可靠数据传输协议称为自动重传请求协议 ARQ。
***ARQ 协议中还需要三种协议功能来处理比特差错:
【**差错检测**】：检验和字段发送方发送分组前需打包校验码。接收方收到后需要进行校验。
【**接收方反馈**】：接收方向发送方反馈校验结果 ACK/NAK(0=NAK,1=ACK,所以反馈分组只需要 1bit 长度)。发送方发送一个分组后等待反馈。
【**重传**】：发送方收到 ACK 后继续发送下一个分组，否则重新发送当前分组。
*****【注意】***发送方发送完数据，一定要等待接收方的反馈信息，只有接收到 ACK 之后才会继续从上层获得数据来传输，如果接收到 NAK 也不会从上层再取数据而是会重传刚刚的分组。→【停等协议】：发送方发送一个分组后停止发送，等待应答。**

【Rdt2.1】：ACK/NAK 应答可能受损。**使用接收方到发送方的肯定确认和否定确认（即接收方应答信息加检验和），发送报文加序号，。**
【序号】：发送方发送分组前标注 1 比特序号。接收方收到后，若校验出错，则回复 NAK（再给我传一个）；若校验无错，检查是否是为希望接收到的分组，若是则接收，否则不接收，无论是否接收都要回复 ACK（收到了，别再传了）。发送方发送完毕后等待，收到反馈后先校验，若校验出错（即应答有比特出错，发送方读不出来到底时 ACK 还是 NAK）或者收到 NAK，都对当前分组进行重传，并继续等待，直到收到 ACK 再改变序号发送下一个。
*******接收方只需要检查序号及可以确定接收到的分组是否需要重传。
*******应答报文 ACK/NCK 不需要指明它们要确认的分组序号，因为发送方知道所接受的响应报文一定是相应其最近发送的数据分组而生成的。
*******因为是停等协议，所以只需要 1bit 的序号（只有 0/1）（发送方收到 0 号包的 ACK 才能发 1 号包，收到 1 号包的 ACK 才能发 2 号包（而 2 号包模 2 序号实际为 0，所以发的 0 号包），0/1 之间不断转换即足够了。
【Rdt2.2】：去除 NAK：**接收方应答信息加序号**。
接收方收到分组后，在处理完成后，一律回复 ACK 和当前最新正确接收的分组序号（想要得到下一个）。发送方若收到损坏应答，重传当前分组；否则根据应答序号，若为当前分组序号，则发送下一个分组，否则重传当前分组。
【Rdt3.0/比特交替协议（停等协议）】：假设下层信道可能出比特差错可能丢失分组（之前是会出错但不丢），**发送方每次发送一个分组（包括第一次分组和重传分组）时，启动定时器。**
【倒计数定时器】：发送方在**发送一个分组后启动倒数定时器并等待**，若超时未收到回复信息，则重传当前分组，并重启定时器。倒数时间内若收到损坏应答，或者收到非当前序号（上一个序号）的 ACK，**不进行立即重传，而是什么都不做，等待计时器超时**（以减少信道中的冗余数据分组）。只有在倒数时间内收到当前序号的 ACK 时，才停止计时器并发送下

一个分组。

【发送方/信道利用率】发送方式实际忙于发送比特到信道的那部分时间/发送时间（发现停等协议的发送方利用率很低，所以采用流水线技术，允许发送方发送多个分组而无需等待确认）
【流水线协议】：提升信道利用率
【增加序号范围】：多个已发送但未确认的分组需要有唯一的序号标识
【发送方&接收端缓冲】：缓冲从最早的已发送但未确认的分组至当前已发送的所有分组，且不能超过 N 个（窗口长度），以备重传。
【窗口滑动】：仅当最早的已发送但未确认的分组得到确认后，进行窗口滑动，并发送窗口内新增的未发送的分组。
【激活发送方的事件】：上层调用 rdt_send()
【解决流水线的差错恢复 2 种基本方法】回退 N 步 GBN，选择重传 SR
【回退 N 步 GBN】:
发送方:
窗口长度限制：k 比特序号最多缓冲 2^k 个
单一计时器：仅当窗口滑动时（收到 ACKn≥base）启动计时器；若应答损坏或收到之前分组的应答，则什么都不做。
超时回退 N 步重传：若计时器超时，则将窗口内所有已发送未确认的分组进行重传，并重启计时器。
接收方:
不接受乱序到达：不进行缓存，丢弃所有乱序到达的分组。
累积确认：对正确收到的分组进行累积确认，ACKn 代表序号 n 之前的所有分组都进行了确认。

【选择重传 SR】:
发送方:
窗口长度限制：k 比特序号最多缓冲 2^{k-1} 个
N 个计时器：发送方维护分组的接收标记，并给每个分组都维护一个计时器，每当计时器超时，进行当前分组的重传。
接收方:
缓存失序分组：缓存失序到达的分组直至有序才批量交付上层。
立即确认：收到每个分组（包括已经有序递交成功的分组，避免由于应答丢失导致的发送方无限等待）立即发送 ACK 确认。

TCP 传输控制协议，面向连接服务：可靠性（有序实时）、流量控制（接收方过载）、拥塞控制（路由器拥塞）
缓存数据流：应用层向 TCP 的数据缓存（建立连接时建立）中输送/拿取数据
最大报文段长度 MSS：每个报文段中可封装的最长应用层报文长度。通常为 1460 字节，以太网链路层承载的最大传送单元 MTU 减去 TCP/IP 头部 40 字节。
TCP 报文段：源端口、目的端口、序号 SEQ、确认号 ACK、TCP 首部长度（通常为 20 字节）、可选选项字段、标志字段、接收窗口（用于流量控制）、校验和、紧急数据指针、加长选项字段（通常为空）、数据。

标志字段：SYN（建立连接请求），ACK（表示确认连接或确认号字段 ACK 有效），FIN（连接拆除标志），URG（包含紧急数据），PSH（直接递交数据），RST（接收方找不到套接字）。

TCP 连接管理：
三次握手：客户端发送请求报文段（SYN 报文段，随机选择初始序号 SEQ，SYN=1）
→服务器分配缓存，向客户响应连接（SYNACK 报文段，ACK=收到的 SEQ+1，随即选择初始序号 SEQ，SYN=1）
→客户端分配缓存，开始发送数据（ACK=收到的 SEQ+1，SYN=0）
关闭连接：客户端发送关闭请求（FIN=1）→服务器发回确认→服务器发送关闭请求（FIN=1）
→客户端发回确认（等待若干时间后关闭）
客户状态机：已发送 SYN→已进行三次握手→已发送 FIN→已收到 ACK→已收到 FIN→关闭
服务器状态机：监听请求→收到 SYN 并返回 SYNACK→收到返回报文并开始传输→收到 FIN 并返回 ACK→已发送 FIN→收到 ACK，关闭连接

可靠数据传输：流水线协议 GBN 和 SR 的结合
序号 SEQ：发送方填写的该报文段中数据的首字节编号，起始序号随机
确认号 ACK：接收方**累积确认**，填写希望接收到的下一个数据字节的编号
缓存乱序到达：接收方缓存乱序到达的分组，但不立即给予确认
单一定时器：发送方只对最早发送但未被确认的报文段计时，但只重传一个报文段。
往返时间估计：EstimatedRTT=（1-α）EstimatedRTT（初始值为 1）+αSampleRTT（α=0.125）
超时（重传）时间估计：DevRTT=（1-β）DevRTT+β|SampleRTT-EstimateRTT|（β=0.25）
TimeoutInterval=EstimatedRTT+4DevRTT（发生超时则直接加倍）
快速重传：接收方等待 N 个按序到达的报文段一起累积确认。但当收到乱序到达报文段，或缺失报文段被填充完毕时立即发送 ACK。发送方除了超时重传外，当收到 3 个冗余 ACK 时立即重传该报文段。

流量控制：避免接收方数据流缓存溢出
接收窗口 rwnd：接收方将剩余缓存大小告知发送方（TCP 头部字段），发送方控制数据的发送，保证已发送但未确认的数据量不大于接收窗口。当缓存空间满（接收窗口=0），发送方仍继续发送 1 字节数据报文段，直到缓存清理。

拥塞控制：减少由于 IP 网络繁忙导致的繁忙和丢包，避免加剧拥塞
网络层辅助拥塞控制：路由器提供反馈信息（一比特分组首部，直接反馈/经接收方反馈）。
ATM ABR：发送方在数据中夹杂资源管理信元（RM），RM 信元到达接收方后返回发送方。
EFCI（显式转发拥塞指示）比特：位于数据信元，交换机将其置 1 表示发生拥塞。
NI（轻度拥塞）比特、CI（轻度拥塞）比特：位于 RM 信元。交换机可改变 NI、CI 值；若接收方收到的多数近来的数据信元 EFCI=1，则将 CI 置 1。
显式速度 ER：位于 RM 信元，交换机可降低其值，其值为路径上所有交换机最小速率。
缺陷：增加交换机的计算负荷

端到端拥塞控制：端系统（运输层）通过观察网络行为来推断网络状态（例如冗余 ACK）
TCP 的拥塞控制：既避免拥塞，又能充分利用网络资源
拥塞窗口 cwnd：发送方保证已发送但未确认的数据量不大于拥塞窗口和接收窗口的最小值。
TCP 带宽探测：加性增，乘性减

慢启动阶段：初始 cwnd 为较小值（1 个 MSS），在慢启动阶段呈指数级增长（收到每个 ACK 时+1MSS，因此 cwnd 每隔 RTT 翻倍）。
拥塞避免模式：若发生超时，则将 cwnd 置为初始值 1MSS，重新开始慢启动，并且当 cwnd 增长到超时值一半时进入拥塞避免模式：进入拥塞避免模式后，cwnd 呈线性增长（收到每个 ACK 时+1/cwnd 个 MSS，因此 cwnd 每隔 RTT+1）。
快速恢复（TCP Reno）：若收到 3 个冗余 ACK，进行快速重传，并选择快速恢复：cwnd 置为当前值的一半，并进入拥塞避免模式（线性增长）。

网络层：

向上提供主机间逻辑通信服务（因特网：尽力而为）
部署在路由器，封装 IP 地址（源和目的），向下传递数据报（datagram）
路由器：网络层分组交换机，基于网络层报文决定分组出口
转发：网路层负责将分组移动到适当的输出链路（动作）
路由选择：网络层决定分组通过何种路径送至接收方（算法）
转发表：（分组首部值，输出链路）路由算法的产物，存在于每台路由器，决定分组出口

虚电路网络：网络层的连接服务，分组传输路径固定，且有序（ATM，帧中继体系）
与运输层区别：主机到主机；无法同时提供无连接服务；也在路由器实现（参与建立过程）
VC 号：每段链路（方向）对应一个 VC 号，并由分组首部携带，路由器转发时根据转发表更改之。
连接建立：发送信令报文进行虚呼叫，路由器在转发表中新建表项（拆除时删除），指定 VC 号，并预留链路资源。

数据报网络：网络层的无连接服务，分组传输路径不固定，且乱序（因特网）
目的地址转发：路由器根据分组目的地址（IP）检索转发表，决定输出链路接口
最长前缀匹配：转发表只存储每个链路出口对应的地址前缀，分组选择最长的匹配项

对比的方面	虚电路服务	数据报服务
思路	可靠通信应当由网络来保证	可靠通信应当由用户主机来保证
连接的建立	必须有	不需要
终点地址	仅在连接建立阶段使用，每个分组使用短的虚电路号	每个分组都有终点的完整地址
分组的转发	属于同一条虚电路的分组均按照同一路由进行转发	每个分组独立选择路由进行转发
当节点出故障时	所有通过出故障的节点的虚电路均不能工作	出故障的节点可能会丢失分组，一些路由可能会发生变化
分组的顺序	总是按发送顺序到达终点	到达终点时不一定按发送顺序
端到端的差错处理和流量控制	可以由网络负责，也可以由用户主机负责	由用户主机负责

路由器工作原理：分组转发平面（硬件）/路由选择控制平面（软件）
输入端口：查找（存储转发表副本）并**排队等待**进入交换结构转发
交换结构：经内存交换（路由选择处理器控制），经总线交换（根据分组首部标签），经互联网络交换（交换结构控制器控制接口闭合）
输出端口：分组**排队**等待输出
线路前部阻塞：（在输入端口进入交换结构时，输出端口不同可并行传送）可并行传送的分组须在队列中等待不可并行传送的分组进行交换。

IP 网际协议：
IPv4 数据报：版本（v4），首部长度（20 字节），服务类型（上层服务类型），数据报长度（首部 20 字节+数据长度），标识号（IP 分片第几片），标志（是否最后一片 0），偏移（该片在完整 IP 数据报的位置），寿命（TTL，经一路由器后-1），上层协议（TCP/UDP），首部校验和（16 比特累和，由于 TTL 值不断变化，需每台路由器进行校验并更改校验和），源 IP，目的 IP，扩展选项（一般为空），数据（有效载荷，网络层报文段）
最大传输单元 MTU：链路层能承载的最大数据报长度（各不相同），以太网为 1500 字节
IP 数据包分片（fragment）：以适应不同的链路层协议的 MTU，在发送方分片，接收方组装

IPv4 编址：网络号（+子网号）+主机号
接口：主机、路由器与物理链路之间的边界，每个接口对应一个 IP 地址（同一设备可有不同 IP）
子网：路由器接口连接的隔离的网络岛（划分子网：减少路由表规模）
子网掩码：表示网络地址所占位数（/x 或 11111111000000 表示）
判断两台主机是否位于同一子网：网络号+子网号是否一致（分别和子网掩码做与运算）

因特网：无类别域间路由选择 CIDR
地址分配（before CIDR）：
A：1.0.0.0~127.255.255.255（8 位网络号，第 1 位为 0）
B：128.0.0.0~191.255.255.255（16 位网络号，前 2 位为 10）
C：192.0.0.0~223.255.255.255（24 位网络号，前 3 位为 110）
D、E：多播地址，前 3 位为 111
私有域 IP：
A：10.0.0.0~10.255.255.255（10.XXX.XXX.XXX）
B：172.16.0.0~172.31.255.255（172.16~172.32）
C：192.168.0.0~192.168.255.255（192.168.XXX.XXX）
禁用 IP：
（每个子网）主机号全为 0（网段 IP）、主机号全为 1（广播 IP）
255.255.255.255：全球广播
0.0.0.0：默认 IP

获取自己的 IP 地址：动态主机配置协议 DHCP（即插即用）
DHCP 服务器：维护可用 IP 地址池，主机加入或离开网络时分配或回收 IP 地址
流程（DHCP 报文：UDP，发送端口 68，接收端口 67）：
DHCP 发现：广播，源 IP（0.0.0.0），目的 IP（255.255.255.255），事务 ID
DHCP 提供：广播，源 IP（自身 IP），目的 IP（255.255.255.255），事务 ID，分配的 IP，子网

掩码等，IP 租用时间
DHCP 请求：（由于可能存在多个 DHCP 服务器，客户从多个提供中选择一个 IP，并告知所有 DHCP 服务器）广播，源 IP（0.0.0.0），事务 ID，选择的 IP，提供该 IP 的 DHCP 服务器 IP，IP 租用时间
DHCPACK：广播，响应 DHCP 请求
DHCP 提供的信息：子网掩码，默认网关（路由器地址），本地 DNS 服务器地址

网络地址转换 NAT：内网私有 IP→公网 IP
NAT 转换表：路由器维护内网 IP+端口号与公网 IP+端口号的映射（每个公网 IP 可提供 65535 个映射）。路由器收到报文后检索 NAT 转换表对其目的 IP 和端口号进行更改并转发。
NAT 穿越：当一台主机位于一个 NAT 后面，就不能充当服务器接受 TCP 连接请求。需要一台已经与该主机建立 TCP 连接的设备充当中介。
UPnP：允许外部主机使用 TCP 和 UDP 向 NAT 后面的主机发起通信会话

地址解析协议 ARP：IP 地址→物理地址（RARP：物理地址→IP 地址）
ARP 缓存：（IP，MAC，TTL）主机向某目的 IP 发送数据报时，先查询 ARP 缓存有无接收方的 MAC 地址。
ARP 请求：若 ARP 缓存中无此字段，将自己的 IP 地址和 MAC 地址写入广播包（填写目的 IP，目的 MAC 全为 F），向网络内设备进行广播。网络内主机收到广播后都将此映射写入自己的 ARP 缓存，并递交至网络层检查目的 IP 是否是自己，若为自己，向发送方返回自己的 MAC 地址。
若与接收方不在同一子网内：MAC 地址填写路由器 MAC，源 IP 与目的 IP 不变

主机发送分组流程：
1）将自身 IP 和目的 IP 分别与本机所处子网掩码进行与运算，若结果相同，则说明目的主机就在本网络中：执行 ARP 协议，目的 MAC 地址填写目的主机 MAC。
2）否则：执行 ARP 协议，目的 MAC 地址填写当前网关（路由器）MAC，发至路由器。

路由器分组转发处理流程：
特定主机路由：管理员人工为特定目的的主机指定输出端口
默认路由：路由表不提供明确指示时提供的默认输出端口
1）从首部提取目的 IP，与路由表中记录的各出口网络的子网掩码进行与运算，若得出的网络地址与路由表中记录的该出口网络的网络地址相同，则说明目的主机就在此网络，直接交付该分组。
2）若该目的 IP 符合特定主机路由，则将其转发至指定下一条路由器。
3）查找路由表，根据指示转发至下一跳路由器
4）若在路由表中检索不到，则通过默认路由进行转发
5）若无默认路由，报告出错

因特网控制报文协议 ICMP：差错报告（ping，源抑制，traceroute 的 TTL 过期）
ICMP 报文：由 IP 数据报承载，类型字段+编码字段

IPv6：地址容量扩大，固定 40 字节数据报首部（无扩展选项字段）
不分片：不允许路由器进行分片和组装，若分组大小超出链路层载荷，路由器丢掉分组并返

回一个 ICMP（ICMPv6）差错报文
去除首部校验：将校验任务交给运输层和链路层，减少路由器耗时
IPv6 数据报：版本（v6），流量类型，流标签（标记需特殊处理的数据报），有效载荷，下一个首部（上层协议 TCP/UDP），跳限制（TTL），源 IP，目的 IP，数据
IPv4 到 IPv6 的迁移：隧道法，经过 IPv4 设备时，将 IPv6 数据报装入 IPv4 数据报。

路由选择算法：生成路由转发表（虚电路网络和数据报网络）
问题描述：给定一组路由器，和他们之间的链路，求从源路由器到目的路由器的最优路径
全局式/分散式路由选择算法：要求给定全局网络信息还是节点附近链路信息
静态/动态路由选择算法：是否根据网络状态的变化随时做出反应

链路状态算法 LS（全局式路由选择算法）
原理：如果结点 C 位于结点 A 到结点 B 的最短路上，则这条路也一定包含 C 到 A 的最短路
汇集树：给定源点到所有目的结点的最短路径形成以源点为根的树（可能有多个）
Dijkstra 算法：单源点到所有结点的最短路径：基于点的迭代
变量：到各结点最短路长 d，到各结点最短路径后序结点 p，到各结点最短路是否确定标记
初始化：初始化源点到邻居结点最短路长为距离，其他点为无穷大。
迭代选点：每次迭代，找出未确定的点中最短路长最小的点 w，将其标志改为确定。
迭代松弛：遍历该点的每个未确定最短路的邻居 v，d[v]=min(d[v],d[w]+v 到 w 的距离)，若 d[v]更新，p[v]同时更新。
出口：到所有结点最短路都已确定。
振荡：当边权值（路径长度）为链路负载时，计算结果会受上次和其他节点影响而频繁变化
解决方法：使每个节点在同一周期的不同时刻执行算法
获取链路状态信息的方法：探测相邻链路状态和费用，并洪泛之

洪泛：将收到的每一个包，向除了来源外所有链路广播。用于散布路由信息
洪泛控制：计数器（TTL）法；记录分组路径；产生时记录序号（类似时间），路由器记录（源路由器，最大序号），遇到序号不比最大序号大的分组，则说明该分组已重复到达或过期，丢弃之，否则转发。
选择性洪泛：将包发送到与正确方向接近的链路

距离向量算法 DV（分散式路由选择算法）：异步、迭代、自我终止、分布式
Bellman-ford 方程：x 到 y 的最短路长为 x 的所有邻居 v 到 y 的最短路与 x 到 v 距离之和的最小值
变量：当前节点到各点的最短路长（距离向量），邻居点到任意点的最短路长（从邻居获取），最短路经过的下一节点（转发表）
初始化：距离向量等于直接距离（链路费用），向所有邻居发送自己的距离向量
更新触发：每当相连链路的直接距离发生改变，或从邻居收到距离向量更新
更新动作：对距离向量中的每个目的节点，执行 bellman-ford 方程。若距离向量产生任何改变，向所有邻居发送自己的距离向量。
路由选择环路：若链路费用增加，将会产生无限更新的现象
解决方法：
最大量度值：当链路费用达到最大度量值时，视为此链路已断开（不可达）
水平分割：从一个方向发来的路由信息而引发的更新，不能发回该方向

路由中毒：网络故障时，告诉邻居到某点距离为无穷（下毒）
毒性反转：若 z 通过 y 到达 x，则 z 通知 y：z 到 x 距离为无穷，避免 y 通过 z 找 x
保持时间：当链路费用超过最大度量值时，在路由器中保持一段时间再清除
触发更新：出现网段故障时立即发送更新包，而不必等下次发送路由更新包

层次路由选择：
自治系统 AS：一组处在相同管理控制（路由算法）下的路由器，通过网关路由器对外连接
热土豆路由：进行自治系统间路由时，AS 尽快将分组交至网关路由器（找最短路）

自治系统内部路由选择协议：面向性能（最速）
路由选择信息协议 RIP（UDP 520）：DV
路径长度定义为条数（最大度量值 15 跳）
邻居间每隔 30 秒交换一次跳数向量表（RIP 响应报文），超过 180 秒未收到报文则认为该邻居不可达（下毒，触发更新）
每台路由器维护距离向量表（最少跳数）、转发表（最少跳数路径后继节点）

开放最短路优先 OSPF：洪泛链路状态信息+Dijkstra
路由器周期性、触发性地向 AS 所有路由器洪泛其链路状态信息
支持层次结构：将 AS 划分为多个区域（其中一个为主干区域）
区域内路由器：区域内分组转发
区域边界路由器：区域间连接转发，连接主干、非主干区域，维护内外两个向量表、转发表
主干路由器：为 AS 内其他区域提供路由选择
自治系统边界路由器：与其他 AS 交换信息

自治系统间路由选择协议：面向策略（可达）
边界网关协议 BGP
从相邻 AS 获得可达信息 Ebgp：发送可达子网前缀
向 AS 内部广播 Ibgp：路由器得知一个前缀后，在转发表中增加一项
决定分组转发路径：政治经济文化因素（手动设定）优先→路径最短→热土豆→其他策略
多宿桩网络：客户 AS 可以控制 BGP 通告以拒绝向 ISP 桩 AS 提供流量承载

链路层：

部署在交换机、网络适配器（网卡），封装 MAC 地址，负责点对点递交链路层帧（frame）
提供服务：
组帧：将上层数据报封装并拆分成帧（字符记数法：帧头用一个域记录字符个数；带字符填充的首尾字符定界：使用特定字符 flag 标记帧首尾，并使用 esc 填充；带位填充的收尾标记定界：使用特殊位串标记帧首尾）
链路接入：媒体访问控制协议 MAC，点对点链路传输规则
差错校验：
奇偶校验/二维奇偶校验
循环冗余检测 CRC：

生成多项式：事先协定，长度 r+1 比特（最高位为 1）
数据→CRC 码：数据码左移 r 位，补 0，加模 2 除（按位异或）生成多项式所得 r 位余数
校验：若 CRC 码除生成多项式为 0，则正确

多路访问协议：广播链路（非点对点链路）的共享信道多路访问问题
信道划分协议：电路交换：预留资源保证带宽，1/n 传输速率，存在静默期，用户数固定
时分多路复用 TDM：将时间划分为时间帧，分配时间帧内的不同时隙
频分多路复用 FDM：将信道划分为不同频段
码分多址 CDMA：分配不同的编码格式，使能够同时传输

随机接入协议：享受满带宽，不分配资源，竞争信道（轻负载时延迟小，重负载时效率低）
ALOHA：组帧完成后立即传送。如遇碰撞，则每隔一个帧传输时间以概率 p 重传。
时隙 ALOHA：将时间划分为时隙，每个时隙等于传送一帧的时间，每个帧只在时隙开始时可以传送。如遇到碰撞，在时隙结束前检测之，并在后续时隙以概率 p 重传。（当遇到碰撞，该时隙被浪费）
载波侦听多路访问 CSMA：发送数据前先监听信道状态，若信道空闲则发送。若发送过程中产生冲突，则等待随机时间后重新开始发送过程。
1-坚持型 CSMA：若信道忙，则持续监听，直至信道空闲
非坚持型 CSMA：若信道忙，则等待随机时间后再进行监听
p-坚持型 CSMA：信道分槽，对于每个时间槽，若信道空闲，则以概率 p 发送数据，概率 1-p 等待下一个时间槽。若忙，等待下一个时间槽。
带碰撞检测的载波侦听多路访问 CSMA/CD：若发送数据过程中检测到冲突，立即停止传送，以避免浪费时间。等待随机时间（应当随碰撞次数增加）后，重新传送。
确定随机等待时间：二进制指数后退法：经历第 n 次碰撞后随机从 $0\sim2^n-1$ 中取随机数，作为等待时间量（乘传送 512 比特所需时间）。若经历 10 次碰撞，放弃传送。

轮流协议：（轻负载时延迟大，重负载时效率高）
轮询协议：主节点以循环方式遍历每个节点，询问是否需要发送数据
令牌传递协议：令牌在节点中轮流传递，持有令牌时才可发送数据。

有限征用协议：将结点进行分组，单一时间片内只允许某个组内竞争
适应树搜索：所有站点作为树叶（最底层）组成二叉树，初始为所有站点都可发送。当产生冲突时，下一个时间片只允许当前可发送范围的左子树发送。当发送成功或没有数据发送时，选择当前可发送范围右侧子树。（深度优先搜索）

以太网：无连接，不可靠
帧结构：前同步码，源 MAC，目的 MAC，类型（IP，ARP），数据，CRC 码

链路层交换机：链路层分组交换机，提供点对点连接，基于链路层帧决定分组出口
消除链路碰撞：交换机设有帧缓存，同一时间不会在网段传输多于一个帧
交换机表：（目的 MAC，出口，表项创建时间）当收到待转发的帧时，存储来源 MAC、来源接口、当前时间。若一段时间内该 MAC 没有再传来帧，将记录删除。（自学习，即插即用）
转发流程：查找交换机表，若查到则根据其转发（若指示的出口与来源接口相同则丢弃）。若交换机中没有目的 MAC，向除来源外所有出口广播该帧。

访问 WEB 网页 www.google.com 完整流程（精简）

1.获取本机 IP 地址和当前网络信息：DHCP 协议

Bob 的主机生成 DHCP 发现报文，封装至目的端口 67、源端口 68 的 UDP 报文段，再封装至目的 IP（255.255.255.255）、源 IP（0.0.0.0）的 IP 数据报中，最后封装至源 MAC 为自己 MAC、目的 MAC 为（ff：ff：ff：ff：ff：ff）的以太网帧中广播。本网段 DHCP 服务器收到报文后，分配 IP 地址和租借时间，和当前网段子网掩码、默认网关 IP、本地 DNS 服务器 IP 一并写入 DHCP 提供报文返回给 Bob。Bob 收到后，广播发送 DHCP 请求报文，DHCP 服务器收到后响应 DHCP ACK 报文。Bob 收到后，记录自己的 IP、默认网关 IP、本地 DNS 服务器 IP。

2.获取网关路由器的 MAC 地址：ARP 协议

Bob 封装源 IP 为自己 IP、目的 IP 网关路由器 IP 的 ARP 查询报文，再封装至源 MAC 地址为自己 MAC、目的 MAC 为（ff：ff：ff：ff：ff：ff）的以太网帧中广播。网关路由器收到后递交至网络层，发现 IP 地址与自己相同，返回 ARP 应答报文，其中包含自己的 MAC 地址。Bob 收到后，将网关路由器 IP 和 MAC 地址存入 ARP 缓存。

3.获取 www.google.com 的 IP 地址：DNS 查询

Bob 分别将自己的 IP 和本地 DNS 服务器 IP 与子网掩码做按位与运算，发现结果不相同，说明其不在当前网络。Bob 封装 DNS 查询报文，目的 IP 为本地 DNS 服务器 IP，目的 MAC 为网关路由器 MAC。网关路由器收到后，根据目的 IP 地址和按 BGP 配置好的转发表进行最长前缀匹配并转发之，经过数个路由器，直到递交到本地 DNS 服务器。本地 DNS 服务器检索数据库，找到 www.google.com 对应的 IP 地址，将 DNS 应答报文返回给 Bob。Bob 从报文中获得 IP 地址。

4.与服务器建立 TCP 连接

Bob 向刚获取的目的 IP、80 端口发送 TCP SYN 报文，封装网关路由器 MAC 并发送。经过层层转发后，目的主机 80 端口的 TCP 套接字收到报文，为之新建一个专用 TCP 套接字，并返回 TCP SYNACK 报文，Bob 收到后进入 TCP 连接状态。

5.HTTP 请求

Bob 的浏览器将 URL 封装至 HTTP GET 报文，通过 TCP 套接字发送至目的主机。目的主机将 Web 对象放入 HTTP 响应报文中返回。Bob 收到后提取网页的 HTML，显示到屏幕上。

本地 DNS 查不到：去根名称服务器查

（Web 缓存代理服务器，cookie，关闭 tcp）