

计算机网络笔记

上官凝

2021 年 3 月 5 日

目录

1 计算机网络概述	4
1.1 什么是 internet	4
1.2 网络边缘	4
1.3 网络核心	4
1.4 分组交换中的时延、丢包和吞吐量	5
1.5 协议层次和服务类型	5
1.5.1 服务和服务访问点	5
1.5.2 服务和协议	5
2 应用层	7
2.1 应用层协议原理	7
2.1.1 网络应用架构	7
2.1.2 进程通信	7
2.1.3 因特网提供的运输服务	7
2.2 Web 和 HTTP	8
2.2.1 HTTP 概况	8
2.2.2 非持续链接和持续链接	8
2.2.3 HTTP 报文格式	8
2.2.4 用户-服务器状态: cookies	8
2.3 FTP	8
2.4 Email	8
2.5 DNS	9
2.6 P2P 应用	9
2.7 CDN	9
2.8 TCP socket 编程	9
2.9 UDP socket 编程	9
3 运输层	10
3.1 概述和运输层服务	10
3.2 多路复用与解复用	10
3.3 无连接传输: UDP	12

3.4 可靠数据传输的原理	12
3.4.1 RDT 1.0: 经完全可靠信道的可靠数据传输	13
3.4.2 RDT 2.0: 经具有比特差错信道的可靠数据传输	13
3.4.3 RDT 2.1: 发送方处理出错的 ACK/NAK	15
3.4.4 RDT 2.2: 不使用 NAK 的协议	15
3.4.5 RDT 3.0: 经具有比特差错和分组丢失的信道的可靠信息传输	15
3.4.6 流水线可靠数据传输协议	15
3.4.7 回退 N 步	16
3.4.8 选择重传	16
3.5 面向连接的传输: TCP	17
3.5.1 段结构	18
3.5.2 可靠数据传输	19
3.5.3 流量控制	19
3.5.4 连接管理	19
3.6 拥塞控制原理	19
3.7 TCP 拥塞控制	19
4 网络层-数据平面	20
4.1 导论	20
4.1.1 数据平面	20
4.1.2 控制平面	20
4.2 虚电路和数据报网络	20
4.2.1 虚电路网络	21
4.2.2 数据报网络	21
4.3 路由器组成	21
4.3.1 输入端口处理和基于目的地转发	21
4.3.2 交换结构	22
4.3.3 输出端口	22
4.3.4 何处出现排队	24
4.3.5 分组调度	24
4.4 网际协议: IPv4、寻址、IPv6 及其他	25
4.4.1 数据报格式	25
4.4.2 分片	26
4.4.3 Ipv4 编址	27
4.4.4 NAT: 网络地址转换	32
4.4.5 ICMP	33
4.4.6 Ipv6	33
4.5 通用转发和 SDN	33
4.5.1 匹配	34
4.5.2 行动	34

4.5.3 OpenFLow 有关“匹配 + 行动”的运行实例	34
5 网络层-控制平面	35
5.1 导论	35
5.2 路由选择算法	35
5.2.1 路由 (route) 的概念	35
5.2.2 路由选择算法	36
5.2.3 链路状态路由选择算法	36
5.2.4 距离向量选择算法	37
5.3 因特网中自治系统内部的路由选择	37
5.3.1 RIP	38
5.3.2 OSPF	38
5.4 ISP 之间的路由选择: BGP	39
5.4.1 通告 BGP 路由信息	39
5.4.2 确定最好的路由	39
5.5 SDN 控制平面	40
5.6 ICMP: 因特网控制报文协议	40
6 链路层和局域网	41
6.1 引论和服务	41
6.1.1 链路层提供的服务	41
6.1.2 链路层在何处实现	42
6.2 差错检测和纠正技术	42
6.2.1 奇偶校验	42
6.3 多路访问链路和协议	42
6.3.1 信道划分协议	43
6.3.2 随机接入协议	43
6.3.3 轮流协议	45
6.3.4 DOCSIS: 用于电缆因特网接入的链路层协议	46
6.4 交换局域网	46
6.4.1 链路层寻址和 ARP	46
6.4.2 关于 ARP 缓存	47
6.4.3 Ethernet: 以太网	48

第 1 章 计算机网络概述

第 1.1 节 什么是 internet

一个网，或者图的角度来理解，结点包括主机及其上的应用程序，和路由器、交换机等网络交换设备；边包括通信链路，分为接入网链路和主干链路。还有协议，协议定义了在两个或多个通信实体之间交换的报文格式和次序，以及在报文传输和/或接收或其他事件方面所采取的动作

第 1.2 节 网络边缘

端系统（主机）、C-S 模式、P2P 模式。网络设施的 TCP 和 UDP 服务

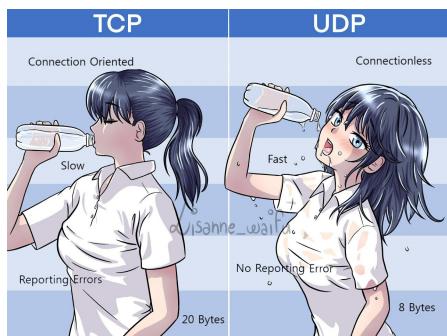


图 1.1: TCP 与 UDP

2.1.4 因特网提供的传输服务

- | | |
|---|--|
| TCP service: <ul style="list-style-type: none"><input type="checkbox"/> 面向连接: 客户进程和服务器进程需要建立连接<input type="checkbox"/> 发送进程和接收进程之间可靠传输<input type="checkbox"/> 流量控制: 发送进程不会“压垮”接收进程<input type="checkbox"/> 拥塞控制: 网络超载时抑制发送进程<input type="checkbox"/> 不提供: 及时性，最低带宽保证 | UDP service: <ul style="list-style-type: none"><input type="checkbox"/> 发送进程和接收进程之间不可靠传输<input type="checkbox"/> 不提供: 连接建立，可靠传输，流量控制，拥塞控制，及时性，最低带宽保证 |
|---|--|

图 1.2: TCP 与 UDP

第 1.3 节 网络核心

电路交换和分组交换

电路交换：为线路预留端到端资源，链路带宽（尤其是瓶颈带宽）决定了通信能力，专用资源不共享，一旦建立起来就可以保证性能，需要建立连接。网络资源（如带宽）被划分成片，可以频分 (FDM)、时分 (TDM)、波分 (WDM)

分组交换：存储-转发（分组每次移动是一跳），在转发之前，结点必须收到整个分组

分组交换可以提升网络的容量：假设线路是 1Mbps，每个用户在活跃的时候用掉 100kbps，有 10% 的时间是活跃的。若使用分组交换，则 ≥ 10 个用户活跃的概率，由二项分布可得为

$$1 - \sum_{n=0}^9 \binom{35}{n} p^n (1-p)^{35-n}$$

第 1.4 节 分组交换中的时延、丢包和吞吐量

分组延迟的来源有四：结点处理延迟、排队延迟、传输延迟、传播延迟。传输延迟是指将分组发送到链路上的时间。

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

第 1.5 节 协议层次和服务类型

1.5.1 服务和服务访问点

1. 服务 (Service): 低层实体向上层实体提供它们之间的通信的能力
 - (a) 服务用户 (service user)
 - (b) 服务提供者 (service provider)
2. 原语 (primitive): 上层使用下层服务的形式，高层使用低层提供的服务，以及低层向高层提供服务都是通过服务访问原语来进行交互的
3. 服务访问点 SAP (Services Access Point): 上层使用下层提供的服务通过层间的接口
 - (a) 例子: 邮箱
 - (b) 地址 (address): 下层的一个实体支撑着上层的多个实体，SAP 有标志不同上层实体的作用
 - (c) 可以有不同的实现，队列
 - (d) 例子: 传输层的 SAP: 端口 (port)

1.5.2 服务和协议

1. 服务与协议的区别
 - (a) 服务 (Service): 低层实体向上层实体提供它们之间的通信的能力，是通过原语 (primitive) 来操作的，垂直
 - (b) 协议 (protocol): 对等层实体 (peer entity) 之间在相互通信的过程中，需要遵循的规则的集合，水平
2. 服务与协议的联系

- (a) 本层协议的实现要靠下层提供的服务来实现
- (b) 本层实体通过协议为上层提供更高级的服务

第 2 章 应用层

第 2.1 节 应用层协议原理

2.1.1 网络应用架构

网络核心中没有应用层功能，网络应用只在端系统上存在，快速网络应用开发和部署。应用层可能的应用架构：客户-服务器模式（C/S），或者对等模式（P2P），或者混合体

客户-服务器模式 服务器：一直运行，并有固定的 IP 和熟知的端口号；客户机：与互联网有间歇性的连接，可能是动态 IP 地址，不直接与其它客户端通信

对等体体系结构 每一个节点既是客户端又是服务器；自扩展性——新 peer 节点带来新的服务能力，当然也带来新的服务请求；参与的主机间歇性连接且可以改变 IP 地址；难以管理

2.1.2 进程通信

不同主机上的进程通过交换报文进行通信。对每对通信进程，我们通常将这两个进程之一标识为客户端，另一个标识为服务器。其各自的定义如下：

在一对进程之间的通信会话场景中，发起通信（即在该回话开始时发起与其他进程的联系）的进程被标识为客户端，在回话开始时等待联系的是服务器

进程编址需要 IP 地址和端口号

2.1.3 因特网提供的运输服务

因特网（更一般的是 TCP/IP 网络）为应用程序提供了两个运输协议，即 UDP 和 TCP。

TCP Service TCP 服务模型包括面向连接服务和可靠数据传输服务。TCP 链接是全双工的，即连接双方的进程可以在此链接上同时进行报文收发，当应用程序结束发送时，需要拆除该链接。TCP 还有拥塞控制机制

第 2.2 节 Web 和 HTTP

2.2.1 HTTP 概况

web 页面是由对象组成的。对象可以是 HTML 文件、JPEG 图像、Java 小程序、声音剪辑文件等。Web 页含有一个基本的 HTML 文件，该基本 HTML 文件又包含若干对象的引用（链接）。通过 URL 对每个对象进行引用。访问协议，用户名，口令字，端口等。

HTTP 是无状态的，即服务器不保存有关客户请求的任何有关信息

2.2.2 非持续链接和持续链接

非持久连接在一个 TCP 连接上最多传输一个对象，持续连接可以发送多个对象。又由于持续连接可以采用流水，效率会更高

关于响应时间模型：往返时间 RTT：一个小的分组从客户端到服务器，在回到客户端的时间（传输时间忽略）。响应时间是 $2RTT + \text{传输时间}$ （握手 + 请求和响应）

2.2.3 HTTP 报文格式

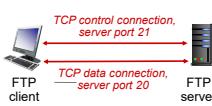
2.2.4 用户-服务器状态：cookies

cookies 是在用户端系统中维护的，由用户的浏览器管理

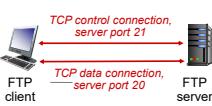
第 2.3 节 FTP

FTP 使用了两个端口

FTP：控制连接与数据连接分开

- FTP 客户端与 FTP 服务器通过端口 21 联系，并使用 TCP 为传输协议
 - 客户端通过控制连接获得身份确认
 - 客户端通过控制连接发送命令浏览远程目录
 - 收到一个文件传输命令时，服务器打开一个到客户端的数据连接
 - 一个文件传输完成后，服务器关闭连接
- 
- 服务器打开第二个 TCP 数据连接用来传输另一个文件
 - 控制连接：带外（“out of band”）传递
 - FTP 服务器维护用户的状态信息：当前路径、用户帐户与控制连接对应
 - 有状态

FTP：控制连接与数据连接分开

- FTP 客户端与 FTP 服务器通过端口 21 联系，并使用 TCP 为传输协议
 - 客户端通过控制连接获得身份确认
 - 客户端通过控制连接发送命令浏览远程目录
 - 收到一个文件传输命令时，服务器打开一个到客户端的数据连接
 - 一个文件传输完成后，服务器关闭连接
- 
- 服务器打开第二个 TCP 数据连接用来传输另一个文件
 - 控制连接：带外（“out of band”）传递
 - FTP 服务器维护用户的状态信息：当前路径、用户帐户与控制连接对应
 - 有状态

第 2.4 节 Email

SMTP 和 HTTP 挺像的，总结如下：

其中拉协议指，在方便的时候，某些人在 web 服务器上装载信息，用户使用 HTTP 从该服务器拉取这些信息。特别是 TCP 连接是由想接收文件的机器发起的。而推协议，指发送邮件服务器把文件推向接收邮件服务器。特别是 TCP 连接是由要发送文件的机器发起的。

SMTP: 总结

- SMTP 使用持久连接
- SMTP 要求报文（首部和主体）为 7 位 ASCII 编码
- SMTP 服务器使用 CRLF、CRLFCRLF 决定报文的尾部

HTTP 比较：

- HTTP：拉（pull）
- SMTP：推（push）
- 二者都是 ASCII 形式的命令 / 响应交互、状态码
- HTTP：每个对象封装在各自的响应报文中
- SMTP：多个对象包含在一个报文中

值得注意的是，SMTP 要求每个报文（包括他们的体）采用 7bit ASCII 码格式。如果包含了非 7bit ASCII 字符（如具有重音的法文字符）或二进制数据（如图形文件），则必须按照 7bit ASCII 进行编码。

第 2.5 节 DNS

第 2.6 节 P2P 应用

第 2.7 节 CDN

第 2.8 节 TCP socket 编程

第 2.9 节 UDP socket 编程

第3章 运输层

第3.1节 概述和运输层服务

在应用程序看来，运输层（PPT 为传输层）为运行在不同主机上的应用进程提供了进程间的逻辑通信。从应用程序的位置来看，通过逻辑通信，运行不同进程的主机好像直接相连一样；实际上，这些主机也许位于地球的两侧通过很多路由器和多种不同类型的链路相连。同应用层一样，运输层也是只有运行在端系统上的。在发送方，将应用层的报文（拆分）并封装为报文段；在接收方做逆处理，从收到的报文段中取出载荷，重组为报文。

网络层服务是主机间的逻辑通信，而运输层是进程间的逻辑通信，它依赖于网络层的服务（继承带宽、延迟的限制）并对网络层的服务进行增强（解决数据丢失、顺序混乱，并加密）。有些服务是可以加强的：不可靠 → 可靠、安全。但有些服务是不可以被加强的：带宽，延迟

类比：两个家庭的通信（Ann 家的 12 个小孩给另 Bill 家的 12 个小孩发信）

1. 主机：家庭
2. 进程：小孩
3. 应用层报文：信封中的信件（可以类比信封为包装的报文段附加的部分）
4. 传输协议：Ann 和 Bill（为家庭小孩提供复用解复用服务）
5. 网络层协议：邮政服务（家庭·家庭的邮包传输服务）

在本书中，我们将 TCP 和 UDP 的分组统称为报文段，而将数据报名称留给网络层分组。

TCP 和 UDP 最基本的责任是，将两个端系统间 IP 的交付服务扩展为运行在端系统上的两个进程之间的交付服务。将主机间交付扩展到进程间交付被称为运输层的多路复用与多路分解（transport-layer multiplexing and demultiplexing）。TCP 力求为每一个通过一条拥塞网络链路的连接平等地共享网络链路带宽。

第3.2节 多路复用与解复用

1. 在发送方主机多路复用：从多个套接字接收来自多个进程的报文，根据套接字对应的 IP 地址和端口号等信息对报文段用头部加以封装（该头部信息用于以后的解复用）
2. 在接收方主机多路解复用：根据报文段的头部信息中的 IP 地址和端口号将接收到的报文段发给正确的套接字（和对应的应用进程）

为了将报文交给正确的套接字

1. 主机中每个套接字应分配一个唯一的标识
2. 报文段中有特殊字段指示要交付的套接字
3. 发送方传输层需在报文段中包含目的套接字标识(多路复用)
4. 接收方传输层需将报文段中的目的套接字标识与本地套接字标识进行匹配, 将报文段交付到正确的套接字(多路分解)

回忆一下 2.7 节, 一个进程(作为网络应用的一部分)有一个或多个套接字, 它们相当于在网络和进程之间传递数据的门户 端口号是 socket 标识的重要组成部分, 是一个 16 位的二进制数,

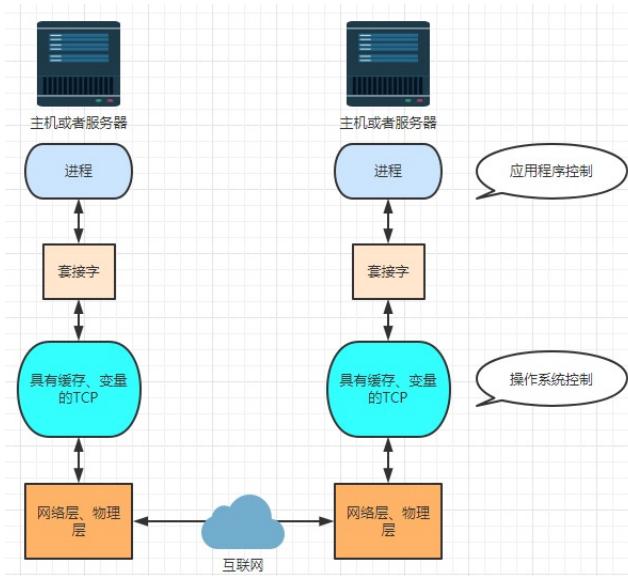
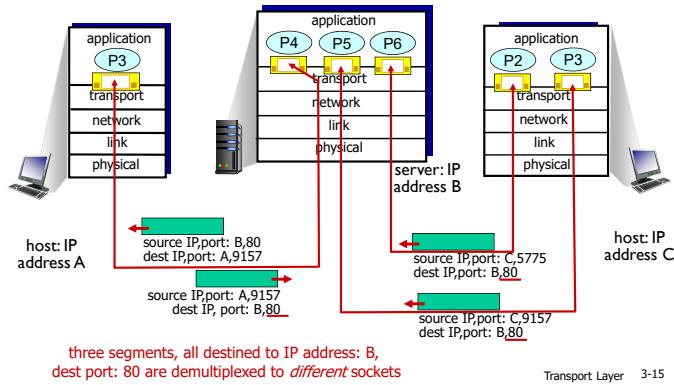


图 3.1: 应用进程、套接字、运输层

其中 0~1023 作为保留端口号给公共域协议使用, 称众所周知的端口号。一般实现公共域协议的服务器会绑定到这个区域内。在主机上的每一个套接字都能够分配到一个端口号, 当接收方传输层接收到一个 UDP 报文时, 检查其中的目标端口号, 并将这个报文交付到具有该端口号的套接字。

值得注意的是, 在多路解复用的过程中, UDP 的 socket 选择标识为报文段中的二元组(目的 IP, 目标端口号), 而 TCP 用的标识是(源 IP, 源 PORT, 目标 IP, 目标 PORT)的四元组。所以对于 UDP, 具备相同目标 IP 地址和目标端口号, 即使是源 IP 地址或/且源端口号不同的 IP 数据报, 也会被传到相同的目标 UDP 套接字上。而对于 TCP, 服务器能够在一个 TCP 端口上同时支持多个 TCP 套接字: 每个套接字由其四元组标识(有不同的源 IP 和源 PORT)。比如 Web 服务器对每个连接客户端有不同的套接字(非持久对每个请求有不同的套接字)。在上图的实际实现中, 有一个初始的 socket, 每接收到一个对应到本 PORT 的连接请求就“fork”出来一个新的 socket 来相应

面向连接的解复用: 例子



Transport Layer 3-15

第 3.3 节 无连接传输: UDP

UDP 即用户数据报协议，其报文结构为源端口号、目的端口号、长度、检验和（奇偶校验）应用数据（报文）。对于 UDP 检验和的确定，其规则如下：UDP 校验和就是二进制反码求和（先求和然后再求反码），但在求和过程中假如首位溢出需要进位，需要回卷，即把前面多出去的 1 加到最后。比如下面这个例子：

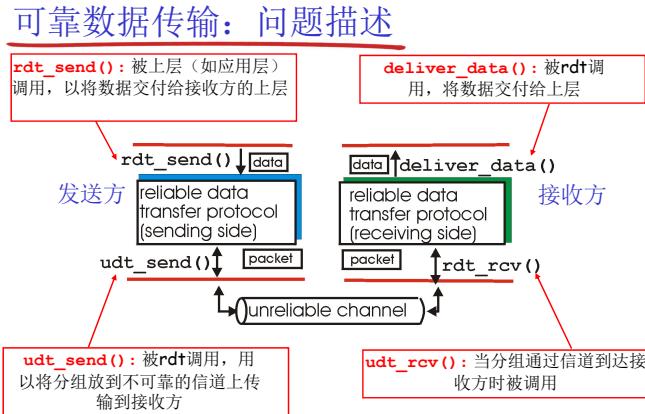
两组数据分别为 1001 和 1111，则求和时，由于首位溢出需要回卷，则为：

$$\begin{array}{r}
 1001 \\
 1111 \\
 \hline
 & 1 \\
 \hline
 1001
 \end{array}$$

取反码得到校验和为 0110。在接收端进行校验时，将校验范围与校验和相加，若为 0xFFFF 则通过校验

第 3.4 节 可靠数据传输的原理

可靠数据传输 (rdt, reliable data transfer) 在应用层、传输层、数据链路层都很重要。可靠数据传输命题大致如下图所示：



Transport Layer 3-26

表 3-1 可靠数据传输机制及其用途的总结

机制	用途和说明
检验和	用于检测在一个传输分组中的比特错误
定时器	用于超时/重传一个分组，可能因为该分组（或其 ACK）在信道中丢失了。由于当一个分组延时但未丢失（过早超时），或当一个分组已被接收方收到但从接收方到发送方的 ACK 丢失时，可能产生超时事件，所以接收方可能会收到一个分组的多个冗余副本
序号	用于从发送方流向接收方的数据分组按顺序编号。所接收分组的序号间的空隙可使接收方检测出丢失的分组。具有相同序号的分组可使接收方检测出一个分组的冗余副本
确认	接收方用于告诉发送方一个分组或一组分组已被正确地接收到。确认报文通常携带着被确认的分组或多个分组的序号。确认可以是逐个的或累积的，这取决于协议
否定确认	接收方用于告诉发送方某个分组未被正确地接收。否定确认报文通常携带着未被正确接收的分组的序号
窗口、流水线	发送方也许被限制仅发送那些序号落在一个指定范围内的分组。通过允许一次发送多个分组但未被确认，发送方的利用率可在停等操作模式的基础上得到增加。我们很快将会看到，窗口长度可根据接收方接收和缓存报文的能力、网络中的拥塞程度或两者情况来进行设置

3.4.1 RDT 1.0：经完全可靠信道的可靠数据传输

这是最简单的情形。注意到下列问题是重要的，发送方和接收方有各自的状态。¹ 在这个简单的协议中，一个单元数据和一个分组没有区别；因为信道完全可靠，接收端不需要反馈信息；由于假定了接收速率和发送速率一样，也不需要限流。

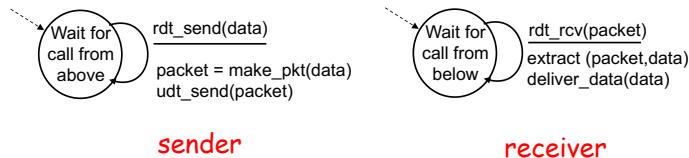
3.4.2 RDT 2.0：经具有比特差错信道的可靠数据传输

假定顺序不被打乱，但是有些比特可能受损（翻转）。处理此类模型的基本思想是“基于肯定确认和否定确认的重传机制的可靠数据传输协议”，称为自动重传请求协议 (Automatic Repeat reQuest, ARQ)。ARQ 使用了以下 4 种机制（书上没写第一个）：

¹ 本书使用的 FSM 规范：引起变迁的事件先是在表示变迁的横线上方，事件发生时所采取的动作显示在横线下方，如果事件/动作为空，则使用符号 \wedge ，以分别明确地表达缺少动作或事件。初始状态用虚线表示。

Rdt1.0: 可靠信道上的可靠传输

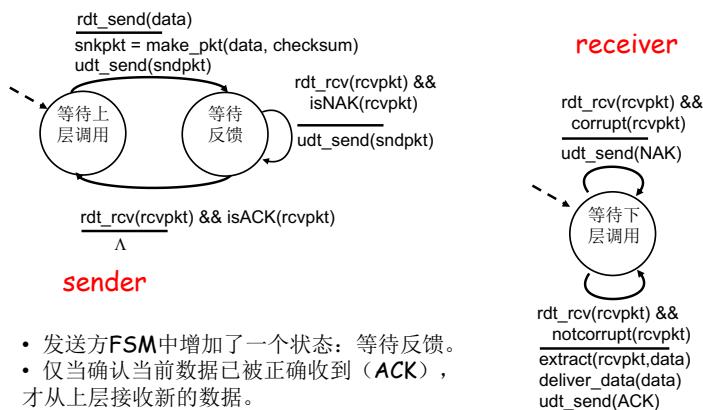
- 下层信道是完全可靠的（理想情况）
 - 没有比特错误
 - 没有分组丢失
- 发送方和接收方使用不同的FSM:
 - 发送方：从上层接收数据，封装成分组送入下层信道
 - 接收方：从下层信道接收分组，取出数据交给上层



Transport Layer 3-31

1. 发送方差错控制编码、缓存
2. 差错检测：使用校验码
3. 接收方反馈：接收方向发送方回送控制报文（“肯定确认”ACK 和“否定确认”NAK）
4. 重传：收到有差错的需要重传

rdt2.0: FSM specification



Transport Layer 3-33

注意下列事实很重要：当发送方处于等待 ACK 或 NAK 的状态时，它不能从上层获得更多的数据。因此 rdt2.0 这样的协议被称为停等 (stop-and-wait) 协议

3.4.3 RDT 2.1：发送方处理出错的 ACK/NAK

rdt2.0 有一个 fatal flaw，即由于信道的不可靠性，无法保证在反馈信号回送给发送方时，反馈信息分组可以无误到达。所以在除去增加纠错比特位使得接收方可以直接恢复原有信息这种办法以外，还可以采取发送方冗余重传的方式。收到损坏的 ACK/NAK 分组时，直接重传即可。但是这种方案可能会在接收方造成分组冗余，于是可以在发送分组上加一个序号（标志位），表示是初传还是重传

3.4.4 RDT 2.2：不使用 NAK 的协议

1. 接收方
 - (a) 对每一个正确接收的分组发送 ACK
 - (b) ACK 中显式携带所确认分组的序号
 - (c) 若收到出错的分组、或不是期待接收的分组，重发对前一个正确接收分组的 ACK
2. 发送方：若 ACK 的序号不是所期待的（表明当前分组未被确认），重发当前分组
3. 为后面的一次发送多个数据单位做一个准备
 - (a) 一次能够发送多个
 - (b) 每一个的应答都有:ACK, NACK; 麻烦
 - (c) 使用对前一个数据单位的 ACK，代替本数据单位的 nak
 - (d) 确认信息减少一半，协议处理简单

3.4.5 RDT 3.0：经具有比特差错和分组丢失的信道的可靠信息传输

使用一个倒计时装置，发送方等待 ACK 一个合理的时间（链路层的 timeout 时间是确定的，传输层 timeout 时间是适应式的），到时还没有收到 ACK 就重传。问题是如果仅仅是延迟了，可能会导致数据冗余。用序列号可以解决，但是接收方在发出 ACK 时必须指明接收的序列号。因为分组序号在 0 和 1 之间交替，因此 rdt3.0 有时被称为比特交替协议

需要注意的是，尽管 rdt3.0 是一个正确的协议，其停等协议的属性导致了它的性能不佳

3.4.6 流水线可靠数据传输协议

参考多周期 CPU 和流水线 CPU 的构造，想想为什么会有流水（并行）和如何抛出精确异常（回退 N 步和选择重传）。为了选择重传，接收方需要设置缓冲区缓存失序的包。

流水线技术对可靠数据传输可以带来如下影响：

1. 必须增加序号范围，因为每一个输送的分组（不计算重传的）必须有一个唯一的序号，而且也许由多个在输送中的未确认报文

2. 协议的发送方和接收方需缓存多个分组。发送方需那些已发送但没有确认的分组（可能重传），接收方需要已经正确接受的分组（可能乱序或者有中间的分组丢失）
3. 所需序号范围和对缓冲区的要求取决于数据传输协议如何处理丢失、损坏以及延迟过大的分组。

传输的窗口包括可以发送但还没有发出去的分组，和已经发出去但还没有 ACK 的分组，也有可能包括已经 ACK 的分组。他只是要求已发出去但没有被确认的包的数量最多为 N，最坏情况下，窗口中的包都是发出去却未收到 ACK 的。

处理异常主要有两种方法：回退 N 步（GBN）和选择重传（SR）

3.4.7 回退 N 步

允许发送方发送多个分组而不需要等待确认。GBN 的基本思想是，将分组按照一个数组进行放置，一个滑动窗口作为分组的可视范围，那些已被发送但还没有确认的，以及（由于收到 ACK 而）做好准备发送的分组的许可序号范围。

从另一个角度来看，许可序号是有限的，当一个 ACK 回到发送方时，就将这个序号传递给下一个没有被分配序号的分组，让他称为“可用，还未发送”状态。这类似于一种流水的折返跑接力赛，假设共有 N 个接力棒（窗口长度），拿到接力棒的选手进入准备状态（窗口内），这些选手每经过一定的时间就出发，到达终点就返回（ACK），返回之后将接力棒交给正好在窗口之后的分组。

分组序号承载在分组首部的一个固定长度的字段中，TCP 有一个 32bit 的序号字段，不过它是按照字节流进行计数的

GBN 的工作原理简单来说，就是

1. 哪里跌倒从哪里站起来：一旦某一个分组传输失败，那后面的都需要重新传输
2. 最高 ACK：接收方仅对正确收到的、序号连续的一系列分组中的最高序号进行确认
3. 失序复读：若收到失序的分组，丢弃（不在接收端缓存），并重发前一次（或者再之前）的 ack 分组（已正确收到、序号连续的一系列分组中的最高序号）
4. 累积确认：若 ACK 包含序号 q，表明“序号至 q 的分组均正确收到”
5. 一次性滑动：如果收到序号 q 的 ACK（即使没有收到之前的），整体滑动发送窗口，使基序号 = q+1
6. 超时重传：发送方只对基序号分组使用一个定时器，发送方重传发送窗口中从基序号开始的所有分组

计时：GBN 在应对超时采用的是维护一个计时器，记录最早的已发送的但还未确认的分组

3.4.8 选择重传

SR 协议通过让发送方仅重传那些它怀疑在接收方出错（丢失或受损）的分组而避免了不必要的重传。SR 协议与 GBN 有一些不同：因为 SR 的每一个分组都是独立的

GBN的发送方

- 收到上层的发送请求:
 - 若发送窗口满: 拒绝请求
 - 若发送窗口不满: 构造分组, 发送
 - 若原来发送窗口为空: 对基序号启动一个定时器
- 收到正确的ACK:
 - 更新基序号 (滑动窗口)
 - 若发送窗口空: 终止定时器
 - 若发送窗口不空: 对基序号启动一个定时器
- 收到出错的ACK:
 - 不做处理
- 定时器超时:
 - 启动定时器, 重发从基序号开始的所有分组

Transport Layer 3-56

1. 计时器: SR 的每一个分组都要有自己的一个计时器, 因为超时发生后只能发送一个分组
2. 窗口移动: 如果收到 ACK 是 send_base (窗口的第一个), 窗口基序号移动到具有最小序号的未确认分组处
3. 发送新的分组: (这两个都是) 发送在窗口内且还没发出去的分组
4. 收到 ACK: 标记这个分组为已接受
5. 重复 ACK: 接收方在接收到窗口头之前的分组时, 还是需要发送 ACK, 因为这个分组有可能时因为它的 ACK 没有成功到达发送方或者发送方超时, 导致发送方重传, 所以还是需要通知发送方
6. 窗口大小: 窗口长度必须小于等于序号空间的一半

第 3.5 节 面向连接的传输: TCP

TCP 即传输控制协议, 是因特网运输层的面向连接的可靠的运输协议。为了提供可靠数据传输, TCP 依赖于前面提到的许多基本原理, 包括差错检测、重传、累积确认、定时器以及用于序号和确认号的首部字段。

总的来讲, TCP 是一种较为“繁琐”的传输协议, 双方需要相互握手, 并采取很多措施, 用时间来换取传输的正确性。TCP 传输有如下特点:

1. 点到点通信: 一个发送者, 一个接收者
2. 全双工: 可以同时双向传输数据
3. 面向连接: 通信前双方先握手 (交换控制报文), 建立数据传输所需的状态 (缓存、变量等)
4. 可靠、有序的字节流: 不保留报文 (应用程序的输出) 边界

5. 流水式发送报文段: 发送窗口由拥塞控制和流量控制机制设置

6. 流量控制: 发送方不会令接收方缓存溢出

TCP 可从缓存中取出并放入报文段中的数据量受限于最大报文段长度 (MSS)，它通常根据最初确定的由本地发送主机发送的最大链路层帧长度 (即所谓的最大传输单元 (MTU)) 来设置。注意 MSS 是指在报文段里**应用层数据**的最大长度，而不是包括首部的 TCP 报文段最大长度。建立连接时，每个主机可声明自己能够接受的 MSS，缺省为 536 字节

3.5.1 段结构

TCP 报文段由首部字段和一个数据字段组成。总的来看，TCP 报文段为如下格式：

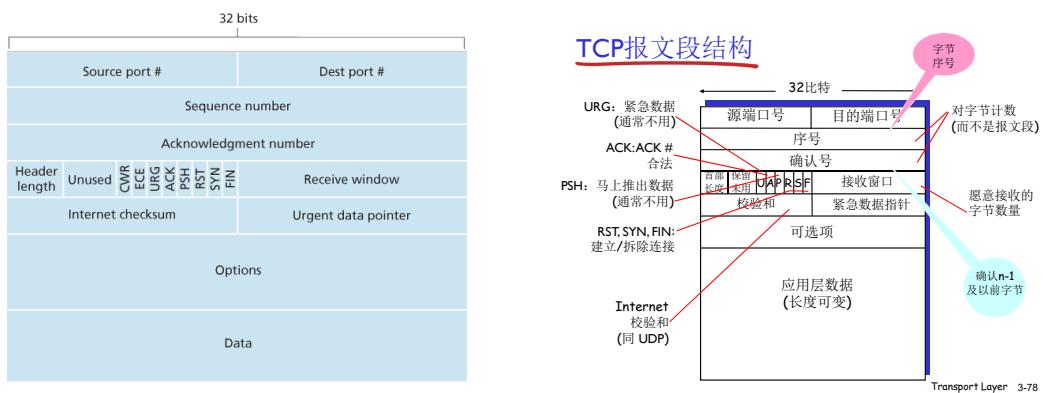


Figure 3.29 TCP segment structure

对部分内容的解释如下：

1. 4bit 的首部长度字段：由于 TCP 选项字段的原因，TCP 首部的长度是可变的。
2. 可选与变长的选项字段：用于发送方和接收方协商最大报文段长度 MSS 时，或在高速网络环境下用作窗口调节因子时使用
3. 还定义了一个时间戳选项

序号和确认号 TCP 把数据看成是一个无结构的、有序的字节流。一个报文段的序号是该报文段首字节的字节流编号。所以说相邻的报文段，其编号是不相邻的，间隔为 MSS。对于确认号，接收方发送的确认号是期望从发送方接到的下一**字节**的序号。TCP 采用的是提供累积确认

3.5.2 可靠数据传输

3.5.3 流量控制

3.5.4 连接管理

第 3.6 节 拥塞控制原理

第 3.7 节 TCP 拥塞控制

第 4 章 网络层-数据平面

网络层是协议栈中最复杂的层次之一，因此我们将仔细考察两种用于构造网络层分组交换的方法，即数据报模式和虚电路模式，并且理解编址在传递分组到目的主机的重要作用。转发涉及在单一的路由器中从一条入链路到一条出链路到传送。路由选择涉及到一个网络的所有路由器，他们经路由选择协议共同交互，以决定目标分组从源到目的地结点所采用的路径。“虚电路和数据报网络”部分在第六版的书上，其余部分由第六版和第七版整理而成。

因特网的网络层有三个主要组件：① IP 协议 ② 路由选择部分（决定了数据报从源到目的地所经过的路径，会计算出用于在网络中转发分组的转发表）③ 报告数据报中的差错和对某些网络层信息请求进行相应的设施

第 4.1 节 导论

网络层的关键功能：转发（通过单个路口的过程）和路由（从源到目的的路由规划路径过程）。

4.1.1 数据平面

4.1.2 控制平面

第 4.2 节 虚电路和数据报网络

与传输层提供的面向连接和无连接服务相对应，网络层提供了连接和无连接服务（注意这里不叫面向连接）。注意网络层和传输层服务之间的巨大差异：

1. 网络层向运输层提供主机到主机的服务，而运输层向应用层提供进程到进程的服务
2. 在至今为止的主要的计算机网络体系结构中（因特网、ATM、帧中继等），网络层提供且仅提供连接和无连接两种服务中的一种，而不同时提供两种服务。仅在网络层提供连接服务的计算机网络称为虚电路网络，仅在网络层提供无连接服务的计算机网络成为数据报网络
3. 在运输层实现面向连接的服务和在网络层实现连接服务是完全不同的，运输层是在位于网络边缘的端系统中实现的，网络层也要在位于网络核心的路由器中实现。

4.2.1 虚电路网络

一条虚电路的组成如下：① 源和目的主机之间的路径（即一系列链路和路由器）；② VC号，沿着该路径的每段链路的一个号码；③ 沿着该路径的每台路由器中的转发表表项。属于一条虚电路的分组将在它的首部携带一个 VC 号。因为一条虚电路在每条链路上可能具有不同的 VC 号，每台中间路由器必须用一个新的 VC 号替代每个传输分组的 VC 号。该新的 VC 号从转发表获得。

换句话说，当一个分组离开端主机或者路由器时，会携带着一个 VC 号，当它途径（已经设定好的路径上的）路由器时，它的 VC 号会发生相应的改变，而改变规则是由转发表决定的。这么大的一个网络，会有更多的路径组合，再加上路由器会有多个出入口，那会产生很多很多的表项更换关系。转发表又是怎么确定应该如何更改 VC 号的呢？有如下原理：无论何时跨越一台路由器创建一条新的虚电路，转发表就增加了一个新表项。类似地，无论何时终止一条虚电路，沿着该路径每个表中的相应项将被删除。

那么，为什么还需要这个转换，而不是在每条链路上简单保持相同的 VC 号呢？第一，逐电路代替该号码减少了在分组首部中 VC 字段的长度。第二（but more importantly），可以大大简化虚电路的建立。**特别是在具有多个 VC 号的路径，其上的每条链路要求……**（第六版电子书 P227 中间）

4.2.2 数据报网络

在数据报网络中，一个端系统要发送分组，就为该分组加上目的端系统的地址，然后将分组推进网络中。在传播过程中，经过的每一台路由器都使用分组中的目标地址来转发该分组。特别是，每台路由器中有一个将**目的地址**映射到链路接口的转发表；当分组到达路由器时，路由器使用该分组的目的地址在转发表中查找适当的输出链路接口。

数据报网络维护的转发表是由目的地址范围（由一个前缀匹配表示，它其实可以是另一个前缀匹配的前缀，在匹配时按照特殊优先的原则不会冲突，即最长前缀匹配规则）和对应的转发到的链路接口组成。

虽然在数据报网络中的路由器不维持连接状态信息，但他们无论如何 (whatever) 在其转发表中维持了转发状态信息。

第 4.3 节 路由器组成

路由器由四个组件构成：路由选择处理器（在控制平面，软件）、输入端口、输出端口、交换结构（都是在数据平面、硬件）。

4.3.1 输入端口处理和基于目的地转发

输入端口的线路端接功能与链路层处理实现了用于各个输入链路的物理层和链路层。在输入端口中执行的查找对于路由器的运行是至关重要的。转发表是由路由选择处理器计算和更新的，但转发表的一份影子副本通常会被放在每个输入端口。**使用在每个输入端口的影子副本，转发决策能在每个输入端口本地做出，无需基于每个分组调用集中式路由选择处理器，因此避免**

了集中式处理的瓶颈。尽管“查找”在输入端口处理中可认为是最为重要的动作，但必须采取许多其他动作：

- ① 必须出现物理层和链路层处理
- ② 必须检查分组头版本号、检验和以及寿命字段，并且重写后两个字段
- ③ 必须更新用于网络管理的计数器（如接收到的 IP 数据报的数目）

所以输入端口步骤如下：线路端接 → 数据链路处理（协议，拆封）→ 查找，转发，排队 → 交换结构

为了保证查找的效率，会用嵌入式片上 DRAM 和更快的 SRAM（用作一种 DRAM 缓存）内存来设计。三态内容可寻址存储器也常被用于查找

4.3.2 交换结构

交换节点位于一台路由器的核心部位。有三种交换技术：

经内存交换 最简单的路由器是传统的计算机，在输入端口与输出端口之间的交换是在 CPU（路由选择处理器）的直接控制下完成的。输入和输出端口的功能就像在传统操作系统中的 I/O 设备一样。许多现代路由器（尤其适合小容量路由器）用内存进行交换，但目的地址由查找和将分组存储（交换）进适当的内存存储位置是由输入线路卡来处理的

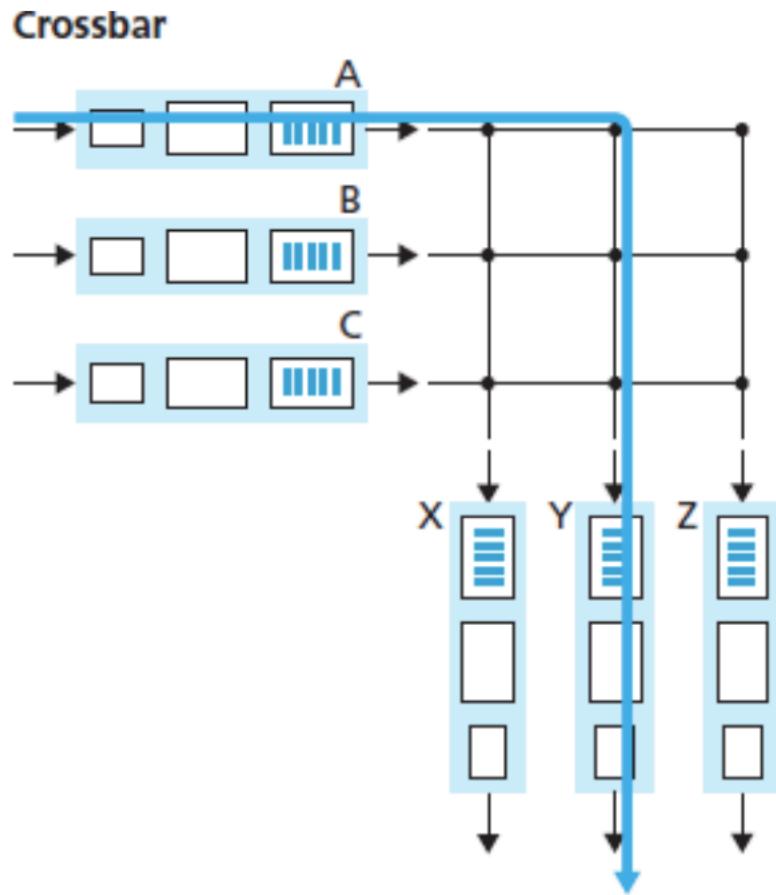
经总线交换 让输入端口为分组预先计划一个交换机内部标签（首部），指示本地输出端口。该分组能由所有输出端口收到，但只有与该便签匹配的端口才能保存该分组。然后标签在输出端口被去除，因为其仅用于交换机内部来跨越总线。因为每个分组必须跨越单一总线，故路由器的交换带宽受总线速率的限制。对于运行在小型局域网和企业网中的路由器来说，通过总线交换通常是足够的。

经互联网络交换 克服单一、共享式带宽限制的一种方法是，使用一个更复杂的互联网络。纵横式交换机就是一种由 $2N$ 条总线组成的互联网络，它连接 N 个输入端口与 N 个输出端口（如下图），交叉结点可以选择打开和闭合。

与前面两种交换方式不同，纵横式网络能够并行转发多个分组。纵横式交换机是**非阻塞的**，即只要没有其他分组被转发到该输入端口，转发到输出端口的分组将不会被到达输出端口的分组阻塞。但如果来自两个不同输入端口的分组，则某个时间经给定总线仅能够发送一个分组

4.3.3 输出端口

交换结构 → 排队（缓存管理）→ 数据链路处理（协议、封装）→ 线路端接



4.3.4 何处出现排队

显然在输入端口和输出端口都会形成排队，这些队列占用了路由器的缓存空间，当缓冲区满时将会产生丢包。在下面的讨论中，设输入链路速率和输出链路速率均为 R_{line} ，且有 N 个输入端口和 N 个输出端口。定义交换结构传送速率 R_{switch} 为从输入端口到输出端口能够移动分组的速率。速率的单位均为（分组/秒）

输入排队 和交换结构传送速率与输入速率有关。考虑纵横式结构，并做如下假定：

1. 所有链路速度相同
2. 一个分组能够以一条输入链路接受一个分组所用的相同的时间量，从任意一个输出端口传送到给定的输出端口（也就是说，去除排队的时间，分组经过输入部分和经过交换结构的时间相同）
3. 分组在输入队列是 FCFS 的，在交换结构中只要输出端口不同就是并行的，否则除其中一个外会被阻塞

会出现如下情况：假设输入端口 A 和 B 头部的两个分组 1 和 2 都要发往接受端口 a，不妨假设交换结构决定发送分组 1。那么这个时候，即使分组 2 后面的分组 3 是发往端口 b（并无竞争），它也需要等待。这种现象称为输入排队交换机中的线路前部（HOL）阻塞。由于 HOL 阻塞，只要输入链路上的分组到达速率达到其容量的 58%，在某些假设前提下，输入队列的长度就将无限制增大。

输出排队 输出端口的排队和输出端口发送分组的速率与交换结构的传送速率有关。假设 $R_{switch} = N \times R_{line}$ ，这时候输入端口不会产生排队，但是输出端口很有可能排队。

当缓冲区满的时候，要么丢弃到达的分组（弃尾），要么删除一个或多个以排队的分组。在某些情况下，在缓存填满之前就丢弃一个分组（或在其首部加上标记）的做法是有利的，这可以向发送方提供一个拥塞信号。这些分组丢弃和标记策略统称为主动队列管理（AQM）

路由器缓存 假定需要路由器缓存来吸收流量负载的波动。多年以来，用于缓存长度的经验方法是 [RFC 3439]，缓存数量 B 等于平均往返时延 RTT 乘以链路的容量 C 。这个结果是对于少量 TCP 流的排队动态性分析得到的。当有大量的 TCP 流（如 N 条）流过一条链路时，缓存所需要的数量是 $B = RTT \times C / \sqrt{N}$

4.3.5 分组调度

现在讨论确定次序的问题，即排队的分组如何经输出链路传输的问题。这块参考 OS 的进程调度，三种策略分别是 FIFO、priority 和 RR

先进先出 如果没有足够的缓存空间来容纳到达的分组，队列的分组丢弃策略则确定该分组是否将被丢弃（丢失）或者从队列中去除其他分组以便为到达的分组腾出空间。

优先权排队 在实践中，网络操作员可以配置一个队列，这样携带网络管理信息的分组（例如，由源或目的 TCP/UDP 端口号所标示）获得超过用户流量的优先权；此外，基于 IP 的实时话音分组可能获得超过非实时流量的优先权。

在非抢占式优先权排队规则下，一旦分组开始传输，就不能打断。

循环和加权公平排队 在循环排队规则下，分组像使用优先权排队那样被分类。但是是采用 Round-Robin 的方式来发送各个类的分组。一个所谓的保持工作排队规则是在有（任何类的）分组等待传输时，不允许链路保持空闲。当寻找给定的类的分组但没有找到时，保持工作的循环规则将立即检查循环序列中的下一个类。

加权公平排队（WFQ）规则中，到达的分组被分类并在合适的等待区域排队，同样采用 Round-Robin 的方式来对各个类进行轮转。WFQ 和循环排队的不同之处在于，每个类在任何时间间隔内可能收到不同数量的服务。具体而言，每个类被分配一个权 w_i 。使用 WFQ 方式，在类 i 有分组要发送的任何时间间隔中，第 i 类将确保接收到的服务部分等于 $w_i / (\sum w_j)$ 部分。

第 4.4 节 网际协议：IPv4、寻址、IPv6 及其他

在这节中，我们将关注点转向今天的因特网网络层的关键方面和著名的网际协议（IP）。今天有两个版本的 IP 正使用：IP 版本 4（IPv4）和 IPv6

4.4.1 数据报格式

IPv4 中的关键字段（部分）如下：

1. 服务类型（TOS）区分不同类型的数据报（如，以下特别要求低时延、高吞吐量或可靠性的数据报）
2. 标识、标志、片偏移与 IP 分片有关。IPv6 不允许在路由器上对分组分片
3. 寿命（TTL）
4. 协议通常仅当到达最终目的地才有用。指示了其数据部分应当交给哪个特定的运输层协议。如：值为 6 是 TCP，值为 17 是 UDP。注意在 IP 数据报中对协议号所起的作用，类似于运输层报文段中端口号字段所起的作用
5. 首部检验和与校验和的反码（被称为因特网检验和）存放在检验和字段中。注意在每台路由器上必须重新计算检验和并再次存放到原处，因为 TTL 字段及可能的选项字段会改变。为什么 TCP/IP 在运输层和网络层都执行差错检测？首先，IP 只对 IP 首部计算了检验和，而 TCP/UDP 检验是对整个 TCP/UDP 报文段进行的；其次，TCP/UDP 和 IP 不一定属于同一个协议栈。原则上，TCP 可以运行在一个不同的协议（如 ATM）上，而 IP 能够携带不一定传递给 TCP/IP 的数据
6. 选项字段允许 IP 首部被扩展。首部选项很少使用，因此对每个数据报首部不包括选项字段中的信息，这样可以节约开销。由于比较麻烦，在 IPv6 中已经去掉了 IP 选项

7. 数据（有效载荷）一般承载的是要交付给目的地的运输层报文段，不过也可以承载其它类型的报文段，比如 ICMP 报文段

注意到一个 IP 数据报有总长为 20 字节的首部（假设无选项¹）。如果数据报承载一个 TCP 报文段，则每个（无分片的）数据报共承载了总长 40 字节的首部以及应用层报文

IP数据报格式

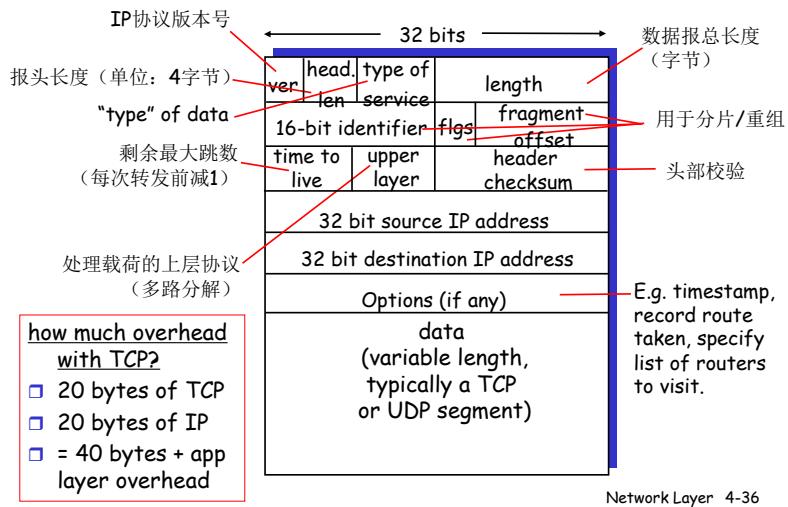


图 4.1: IPv4 数据报格式

4.4.2 分片

为何需要分片？ 众所周知，每个 IP 数据报封装在链路层帧中从一台路由器转移到下一台路由器，故 IP 数据报的大小严格受到链路层帧能承载的最大数据量，即最大传送单元（Maximum Transmission Unit, MTU）的限制。然而实际上，发送方和目的地之间的每段链路可能使用不同的链路协议，且每种协议有着不同的 MTU。

那么，当一个路由器从某条链路收到一个 IP 数据报，通过检查转发表确定出链路并且该链路的 MTU 比该 IP 数据报的有效载荷要小，这时解决问题的办法就是将原数据报拆分成几个较小的数据报，即片（fragment）

分片需要做什么额外的处理？ 片在到达目的地运输层之前需要重新组装。不过为了坚持网络内核保持简单的原则，IPv4 的数据报重新组装工作放到端系统中，而不是放到网络路由器中。

当目的主机从相同源收到一系列数据报时，需要确定这些数据报中的某些是不是一些原来较大数据报的片。如果是的话，必须确定何时收到了最后一片，并且如何（以何种顺序）将这些片拼接到一起以形成初始的数据报。

IPv4 将标识、标志、片偏移字段放在 IP 数据报的首部中。当生成一个数据报时，发送助局在为该数据报设置源和目的地址的同时贴上标识号。同一个较大数据报的不同片之间

¹ 这个在书上这一节“首部长度”部分有说明

1. 具有初始数据报的源地址、目的地址与标识号
2. 标志位:
 - (a) MF(more fragments): 最后一个分片的 MF=0, 其余分片的 MF=1
 - (b) DF(don't fragment): DF=1 表示不允许对数据报分片
3. 偏移字段用来指定该片被放在初始数据报的那个位置

分片的长度

1. 由于分片偏移量只有 13 比特, 除最后一个分片外, 其余分片的数据长度应为 8 字节的整倍数。
2. 假设原始报头的长度为 H , 则分片的数据长度 N 应为满足以下条件的最大整数: $N \leq MTU - H$ (N 为 8 的倍数)

分片有什么问题吗? 首先, 它使路由器和端系统更为复杂。其次, 分片能够被用于生成致命的 DoS 攻击, 由此攻击者发送了一系列古怪的、无法预期的片。如 Jolt2 攻击 (发送偏移量都不是 0 的小片的流), 或者发送交迭的 IP 片 (这些片的偏移量被设置的不能是当地排列起来)

4.4.3 Ipv4 编址

网络接口与 IP 地址 网络接口是主机或路由器与物理链路之间的边界, 而 IP 要求每台主机和路由器接口拥有自己的 IP 地址。因此, *technologically*, 一个 IP 地址与一个接口相关联, 而不是与包括该接口的主机或路由器相关联。

IPv4 地址是一个 32 位的数, 通常用点分十进制表示, 即地址的四个字节分别转换为十进制。除了在 NAT 后面的接口 (4.4.4 节会讨论) 以外, 每个接口都必须有一个全球唯一的 IP 地址。IP 地址的高位由子网确定, 而低位由机器接口确定。

子网 一个子网要具备以下两个条件:

1. 一个子网内的节点 (主机或者路由器) 它们的 IP 地址的高位部分 (子网号) 相同, 这些节点构成的网络的一部分叫做子网
2. 无需路由器介入, 子网内各主机可以在物理上相互直接到达 (在 IP 的角度看只有一跳, 不经过路由器, 不过在链路层来看可以经过几个交换机 blabla)

路由器和交换机有什么区别? 引用自知乎

工作层次不同 交换机主要工作在数据链路层 (第二层); 路由器工作在网络层 (第三层)。

转发依据不同 交换机转发所依据的对象是 MAC 地址 (物理地址); 路由转发所依据的对象是 IP 地址 (网络地址)

主要功能不同 交换机主要用于组建局域网，而路由主要功能是将由交换机组好的局域网相互连接起来，或者接入 Internet。交换机能做的，路由都能做。交换机不能分割广播域，路由可以。路由还可以提供防火墙的功能。路由配置比交换机复杂。

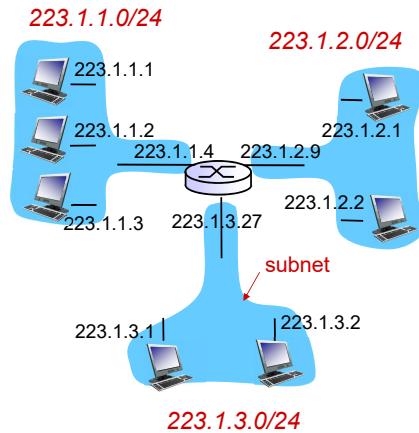
不严格的类比 交换机是看门大爷，路由是邮差。

详细的可以看同问题下的其他回答

子网

方法:

- 要判断一个子网，将每一个接口从主机或者路由器上分开，构成了一个个网络的孤岛
- 每一个孤岛（网络）都是一个都可以被称之为 **subnet**.



子网掩码: 11111111 11111111 11111111 00000000

Subnet mask: /24

网络层: 数据平面 4-41

图 4.2: 子网示意图

如何判断子网（纯子网）

- ① 要判断一个子网，将每一个接口从主机或者路由器上分开，构成了一个个网络的孤岛。
- ② 每一个孤岛（网络）都可以被称为 **subnet**。

注：将接口从主机或者路由器上面分开，也就是将接口与设备分离开来，只观察各接口之间在 IP 层的连接关系（因为 IP 线路本来就是在各个接口之间连接的嘛。这样子，对于一个路由器的不同接口，就可以正确归属到它所属于的子网之中了）

注 2：子网并不局限于连接多台主机到一个路由器接口的以太网段，还可以包括比如两个路由器接口之间直接连着那根线

注 3：前面所说的概念是纯子网（如上面图中的每一个颜色区域），不过其实从这个区域外面来看，这整个的网络也可以叫一个子网（非纯子网）。尽管网内交流需要经过路由器，但是这个局域对于外界是封闭的，到达路由器之后就是网内部的事了

子网掩码 在因特网文献中，子网也称为 IP 网络或直接称为网络。IP 编址为这个子网分配一个地址，如 223.1.1.0/24，其中的/24 记法，有时称为子网掩码，指示了 32 比特中的最左侧 24 比

特定义了子网地址。即，任何其他要连到 223.1.1.0/24 网络的主机都要求其地址具有 223.1.1.xxx 的形式。

单播、广播和多播（组播）地址 具体内容参考这篇文章 单播地址（A、B、C 类）被划分为网络号和主机号两部分：① 网络号：标识一个物理网络 ② 主机号：标识该物理网络上的一个网络接口。同一个物理网络上的网络接口，它们的 IP 地址具有相同的网络号。

广播是对所有人都发送信息，一般来说是局域网范围内的广播。

组播是发送给属于这个组播组的成员。组播组的维护是通过一些特定的协议，如 IGMP。发送到一个 D 类地址，那么属于这个组播组的所有成员都会收到

地址分配 因特网中的每个接口必须具有唯一的 IP 地址，为在因特网范围内保证 IP 地址的全局唯一性：

1. 网络号由 ICANN 统一分配
2. 主机号由网络管理员统一分配
3. 建立私有网络的组织可以自己选择网络号，但同样必须保证每个网络号在私有网络内的唯一性

特殊的地址 全 0 或全 1 的网络号及主机号是特殊地址，从不分配给特定的网络接口：

1. 网络号有效、主机号全为 0 的地址：保留给网络本身。
2. 网络号有效、主机号全为 1 的地址：保留作为定向广播，即在网络号指定的网络中广播（仅用作目的地址）
3. 32 位全 1 的地址：本地广播地址，表示仅在发送节点所在的网络中广播（仅用作目的地址）
4. 32 位全 0 的地址：指示本机（仅用作源地址）
5. 网络号为 0、主机号有效的地址：指代本网中的主机。
6. 形如 127.xx.yy.xx 的地址：保留作为回路测试，发送到这个地址的分组不输出到线路上，而是送回内部的接收端。

内网(专用)IP 地址：

1. 专用地址：地址空间的一部份供专用地址使用
2. 永远不会被当做公用地址来分配，不会与公用地址重复
3. 路由器不对目标地址是专用地址的分组进行转发
4. 专用地址范围
 - (a) Class A: 10.0.0.0-10.255.255.255, MASK 255.0.0.0

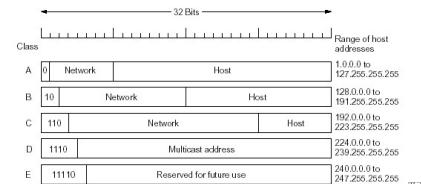
- (b) Class B: 172.16.0.0-172.31.255.255, MASK 255.255.0.0
(c) Class C: 192.168.0.0-192.168.255.255, MASK 255.255.255.0

网络数量与地址数量

- A、B、C类地址对应不同规模的网络
- 一个A类、B类及C类地址可提供的接口地址数：
 - A类地址： $2^{24}-2 = 16777214$
 - B类地址： $2^{16}-2 = 65534$
 - C类地址： $2^8-2 = 254$
- A类、B类、C类地址的个数：
 - A类地址： $2^7-2 = 126$
 - B类地址： $2^{14}-2 = 16382$
 - C类地址： $2^{21}-2 = 2097152$

IP 地址分类

- Class A: 126 networks , 16 million hosts
- Class B: 16382 networks , 64 K hosts
- Class C: 2 million networks , 254 host
- Class D: multicast
- Class E: reserved for future



平面 4-44

无类别域间路由选择 因特网的地址分配策略被称为无类别域间路由选择（Classless Interdomain Routing, CIDR）。如上文所述，我们对 IP 地址进行了分类，如 A 类的第一个字节是网络号。但在 CIDR 中，网络号可以是任意长度的。当使用子网寻址时，32 比特的 IP 地址被划分为两部分，并且也具有点分十进制数形式 $a.b.c.d/x$ ，其中 x 指示了第一部分（网络号）中的比特数。也就是一种按需分配的思路。

形式为 $a.b.c.d/x$ 的地址的 x 位最高比特构成了 IP 地址的网络部分，并且经常被称为该地址的前缀（prefix）（或网络前缀）。在这里可以借助于子网掩码（4.4.3）来在路由器查表的时候对原 IP 地址进行处理。根据 RFC950 定义，子网掩码是一个 32 位的 2 进制数，其对应网络地址的所有位都置为 1，对应于主机地址的所有位置都为 0。子网掩码告知路由器，地址的哪一部分是网络地址，哪一部分是主机地址，使路由器正确判断任意 IP 地址是否是本网段的，从而正确地进行路由。这样做的目的是为了让掩码与 IP 地址做按位与运算时用 0 遮住原主机数，而不改变原网络段数字，而且很容易通过 0 的位数确定子网的主机数（2 的主机位数次方 - 2，因为主机号全为 1 时表示该网络广播地址，全为 0 时表示该网络的网络号，这是两个特殊地址）。

（想想为什么上面的掩码都是 255?）255 转成二进制就是 11111111 啊……

获得 IP 地址 我们先看一个组织是如何为其设备得到一个地址块的，然后再看一个设备（如一台主机）是如何从某个组织的地址块中分配到一个地址的。

获取一块地址 某网络管理员会与他的互联网服务提供商（Internet Service Provider, ISP）联系，该 ISP 可能会从已分给他的更大地址块中提供一些地址。而这也就需要一个全球性的权威机构，具有管理 IP 地址空间并向各 ISP 和其他组织分配地址块的最终责任。这就是因特网名字和编号分配机构（Internet Corporation for Assigned Names and Numbers, ICANN）。它也管理 DNS 根服务器。

获取主机地址：动态主机配置协议 系统管理员通常手工配置路由器中的 IP 地址（常常在远程通过网络管理工具进行配置）。主机地址更多的是使用动态主机配置协议（Dynamic Host Configuration, DHCP）来完成。

DHCP 允许主机自动获取（被分配）一个地址，而这个操作是不需要网络管理员来参与的。不过，DHCP 也允许网络管理员手动配置，来为指定主机分配固定的 IP。除了主机 IP 地址分配外，DHCP 还允许一台主机得知其他信息，例如他的子网掩码、他的第一跳路由器地址（常称为默认网关）与他的本地 DNS 服务器地址。

DHCP 的目标：允许主机在加入网络的时候，动态地从服务器那里获得 IP 地址：

1. 可以更新对主机在用 IP 地址的租用期-租期快到了
2. 重新启动时，允许重新使用以前用过的 IP 地址
3. 支持移动用户加入到该网络(短期在网)

由于 DHCP 具有将主机连接进一个网络的网络相关方面的自动能力，故他又常被称为即插即用协议（plug-and-play protocol）或零配置（zeroconf）协议。

DHCP 的操作流程 DHCP 是一个客户-服务器 (C-S) 协议。对于一台新到达的主机而言，DHCP 是一个四个步骤的过程，这四个步骤是：

1. 主机广播“DHCP discover”报文 [可选]
2. DHCP 服务器用“DHCP offer”提供报文响应 [可选]
3. 主机请求 IP 地址：发送“DHCP request”报文
4. DHCP 服务器发送地址：“DHCP ack”报文

DHCP 服务器发现 一台新到达的主机可以通过 DHCP 发现报文来做到这一点。客户在 UDP 分组中向端口 67 发送该发现报文。在 IP 封装时使用广播目的地址 **255.255.255.255** 和“**本主机**”源 IP 地址 **0.0.0.0**（因为自己的 IP 还没有被分配，所以加了引号）。

DHCP 服务器提供 DHCP 服务器收到一个 DHCP 发现报文时，用 DHCP 提供报文向客户发出响应，该报文向该子网的所有节点广播，仍然使用 IP 广播地址 255.255.255.255（书上说这个的原因是在子网中可能存在几个 DHCP 服务器）。当然在子网中可能存在几个 DHCP 服务器，每台服务器提供的报文包括收到的发现报文的事务 ID、向客户推荐的 IP 地址、网络掩码以及 IP 地址租用期

DHCP 请求 新到达的客户从一个或多个服务器中选择一个，并向选中的服务器用 DHCP 请求报文进行响应，回显配置的参数

DHCP ACK 服务器用 DHCP ACK 报文对 DHCP 请求报文进行响应，证实所要求的参数

4.4.4 NAT: 网络地址转换

NAT 的基本想法由 NAT 使能路由器来完成一个专用网络中所有主机的对外通讯工作。就好比 NAT 使能路由器是一个快递站, 它为每一个要寄出去的包裹都换上“nat 快递”的盒子, 并以自己的名义寄出去; 同样, 将每一个寄给“nat 快递”的包裹, 在更改为真实的收件人之后, 传递给内部的主机。

地址空间 10.0.0.0/8 是一块保留的地址空间, 用于是一个专用网络 (private network) 或具有专用地址的地域 (realm with private address) (具有专用地址的地域是指其地址仅对该网络中的设备有意义的网络)。而 NAT 则是在这个专用网络和专用网络以外的因特网之间的 interface 动机: 本地网络只有一个有效 IP 地址:

1. 不需要从 ISP 分配一块地址, 可用一个 IP 地址用于所有的 (局域网) 设备-省钱
2. 可以在局域网改变设备的地址情况下而无须通知外界
3. 可以改变 ISP(地址变化) 而不需要改变内部的设备地址
4. 局域网内部的设备没有明确的地址, 对外是不可见的-安全

NAT: Network Address Translation

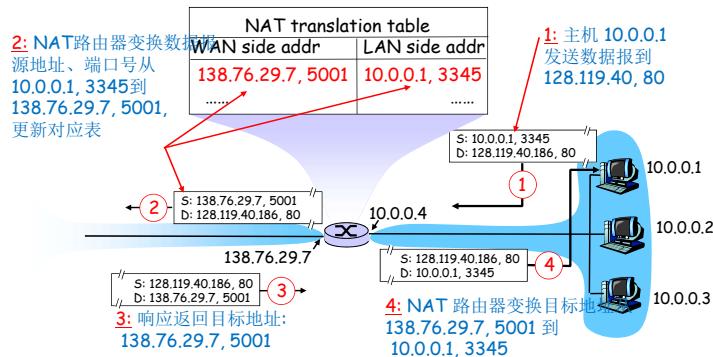


图 4.3: NAT 的执行过程示例

NAT 使能路由器对于外部世界的行为就像一个具有**单一 IP 地址的单一设备**, 而内部网络中不同设备的区分是依赖于端口号来完成的。所以从本质上讲, NAT 使能路由器对外界隐藏了家庭网络的细节。NAT 路由器上会维护一张 NAT 转换表 (NAT transaction table), 并且在表项中包含了端口号及其 IP 地址。

实现: NAT 路由器必须:

1. 外出数据包: 替换源地址和端口号为 NAT IP 地址和新的端口号, 目标 IP 和端口不变。远端的 C/S 将会用 NAP IP 地址, 新端口号作为目标地址
2. 记住每个转换替换对 (在 NAT 转换表中) (源 IP, 端口) vs (NAP IP, 新端口)
3. 进入数据包: 用 (源 IP, 端口) 替换目标 IP 地址和端口号, 采用存储在 NAT 表中的 mapping 表项

4.4.5 ICMP

放到控制平面来说

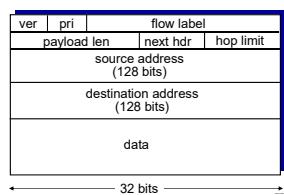
4.4.6 IPv6

IPv6 使用固定的 40 字节头部，且数据报在传输过程中不允许分片。IPv6 中引入的最重要的变化显示在其数据报格式中：

1. 扩大的地址容量。地址长度增加到了 128 比特，且引入了任播地址的概念，可以使数据报交付给一组主机中的任意一个（如，可用于向一组包含给定文档的镜像站点中的最近一个发送 HTTP GET 报文）
2. 简化高效的 40 字节首部。由于许多 IPv4 字段已被舍弃或作为选项
3. 流标签。IPv6 有一个没有被严格定义的流的概念

IPv6 头部 (Cont)

Priority: 标示流中数据报的优先级
Flow Label: 标示数据报在一个“flow.”
 (“flow”的概念没有被严格地定义)
Next header: 标示上层协议



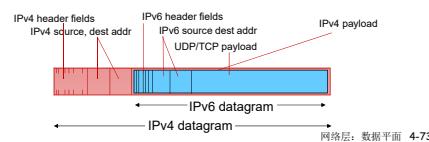
和IPv4的其它变化

- **Checksum:** 被移除掉，降低在每一段中的处理速度
- **Options:** 允许，但是在头部之外，被“Next Header”字段标示
- **ICMPv6:** ICMP的新版本
 - 附加了报文类型, e.g. “Packet Too Big”
 - 多播组管理功能

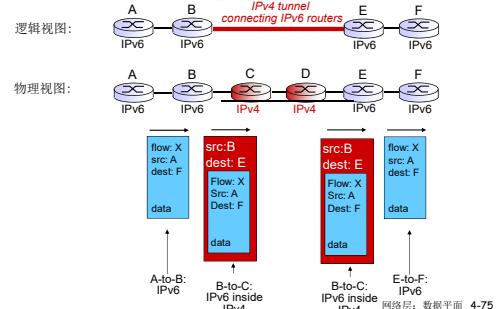
当然，变革都是需要时间的，现在则采用了一种称为“隧道”的技术，即在 IPv4 路由器之间传输的 IPv4 数据报之中携带者 IPv6 数据报，如下图：

从IPv4到IPv6的平移

- 不是所有的路由器都能够同时升级的
 - 没有一个标记日 “flag days”
 - 在IPv4和IPv6路由器混合时，网络如何运转？
- **隧道:** 在IPv4路由器之间传输的IPv4数据报中携带IPv6数据报



隧道(Tunneling)



第 4.5 节 通用转发和 SDN

第二层交换机和第三层路由器等中间盒的剧增，并且每种都有自己特殊的硬件、软件和管理界面，会带来麻烦。基于目的地转发的特征可以总结为两个步骤：查找目的 IP 地址（“匹配”），然后将分组发送到有特定端口的交换结构（“动作”）

4.5.1 匹配

4.5.2 行动

4.5.3 OpenFlow 有关“匹配 + 行动”的运行实例

第 5 章 网络层-控制平面

控制平面作为一种网络范围的逻辑，不仅控制沿着从源主机到目的主机的端到端路径键的路由器如何转发数据报，而且控制网络层组件和服务如何配置和管理。

第 5.1 节 导论

回顾：2 个网络层功能：

1. 转发：将分组从路由器的一个输入端口移到合适的输出端口
2. 路由：确定分组从源到目标的路径

2 种构建网络控制平面功能的方法：

1. 每个路由器控制功能实现（传统）：在每一个路由器中的单独路由器算法元件，在控制平面进行交互
2. 逻辑上集中的控制功能实现（software defined networking）：一个不同的（通常是远程的）控制器与本地控制代理（CAs）交互

第 5.2 节 路由选择算法

5.2.1 路由（route）的概念

① 路由：按照某种指标（传输延迟，所经过的站点数目等）找到一条从源节点到目标节点的较好路径

1. 较好路径：按照某种指标较小的路径
2. 指标：站数，延迟，费用，队列长度等，或者是一些单纯指标的加权平均
3. 采用什么样的指标，表示网络使用者希望网络在什么方面表现突出，什么指标网络使用者比较重视

② 路由器-路由器之间的最优路径 = 主机对之间的最优路径

1. 路由器连接子网，子网到路由器之间的跳数就一跳，必须要走

2. 路由器到下一跳路由器(节点到节点)之间的最优路径找到了
3. 也就找到了从源子网向目标子网所有主机对之间的最优路径
4. 大大降低了路由计算的规模
5. 在路由计算中按照子网到子网的路径计算为目标,而不是主机到主机

5.2.2 路由选择算法

路由选择算法 (routing algorithm): 网络层软件的一部分, 完成路由功能

可以用图来形式化描述路由选择问题。在网络层路由选择的环境中, 图的节点表示路由器, 这是做出分组转发决定的点; 连接这些节点的边表示这些路由器之间的物理链路; 一条边的开销可以反映出对应链路的物理长度, 他的链路速度, 或与链路相关的金钱上的开销。一般而言, 路由选择算法的一种分类方式是根据该算法是集中式还是分散式来划分。

集中式路由选择算法以所有节点之间的连通性和所有链路的开销为输入 (就是整个图!)。计算本身可以在某个场点进行, 或者在每台路由器的路由选择组件中重复进行。然而, 这里的主要 point 是, 集中式算法具有关于连通性和链路开销方面的完整信息 (Dijkstra?)。具有全局状态信息的算法常被称作为链路状态 (Link State, LS) 算法

在分散式路由选择算法中, 路由器以迭代、分布式的算法计算出最低开销路径。每个节点仅有与其直接相连链路的开销即可直接工作。然后, 通过迭代计算过程以及与相邻节点的信息交换, 一个节点逐渐计算出到达某目的节点或一组目的节点的最低开销路径。后面将会介绍一个距离向量 (Distance-Vector, DV) 算法的分散式路由选择算法。

第二种广义分类方式是根据算法是静态的还是动态的来进行分类。在静态路由选择算法中, 路由随时间的变化非常缓慢, 通常是人工进行调整。而动态路由选择算法随着网络流量负载或拓扑发生变化而改变路由选择路径。一个动态算法可以周期性运行或者直接响应拓扑或开销的变化而运行。

第三种分类方式是根据是负载敏感的还是负载迟钝的进行划分负载敏感算法中, 链路开销会动态变化以反映出底层链路的当前拥塞水平。不过当今的因特网路由选择算法 (如 RIP、OSPF 和 BGP) 都是负载迟钝的。

5.2.3 链路状态路由选择算法

LS 路由的基本工作过程

1. 发现相邻节点, 获知对方网络地址
2. 测量到相邻节点的代价 (延迟, 开销)
3. 组装一个 LS 分组, 描述它到相邻节点的代价情况
4. 将分组通过扩散的方法发到所有其它路由器 (以上 4 步让每个路由器获得拓扑和边代价)
5. 通过 Dijkstra 算法找出最短路径 (这才是路由算法)
 - 每个节点独立算出来到其他节点 (路由器 = 网络) 的最短路径

LS 和 DV 算法的比较

消息复杂度 (DV胜出)

- **LS:** 有 n 节点, E 条链路, 发送报文 $O(nE)$ 个
 - 局部的路由信息; 全局传播
- **DV:** 只和邻居交换信息
 - 全局的路由信息, 局部传播

收敛时间 (LS胜出)

- **LS:** $O(n^2)$ 算法
 - 有可能震荡
- **DV:** 收敛较慢
 - 可能存在路由环路
 - count-to-infinity 问题

健壮性: 路由器故障会发生什么 (LS胜出)

LS:

- 节点会通告不正确的链路代价
- 每个节点只计算自己的路由表
- 错误信息影响较小, 局部, 路由较健壮

DV:

- DV 节点可能通告对全网所有节点的不正确路径代价
 - 距离矢量
- 每一个节点的路由表可能被其它节点使用
 - 错误可以扩散到全网

2种路由选择算法都有其优缺点, 而且在互联网上都有应用

- 迭代算法: 第 k 步能够知道本节点到 k 个其他节点的最短路径

OSPF 是一种 LS 协议, 被用于 internet 上; IS-IS 被用于 internet 主干中, Netware

5.2.4 距离向量选择算法

DV 是一种迭代的、异步的和分布式的算法, 而 LS 算法是一种使用全局信息的算法。说他是分布式的, 是因为每个节点都要从一个或多个直接相连的邻居接受某些信息, 执行计算, 然后将其计算结果分发给邻居 (而 LS 是分发的计算源数据分组! 且要从所有连通片内的顶点都要接收!)。说他是迭代的, 是因为此过程一直要持续到邻居之间无更多信息可交换为止。(有趣的是, 此算法是自我终止的, 即没有计算应当停止的信号, 他就停止了。) 说他是异步的, 是因为他不要求所有节点相互之间步伐一致的操作。

DV 的核心算法是 Bellman-Ford 算法。DV 算法的执行和无穷级数问题看这篇。记住 DV 算法的特点是好消息传的快, 而坏消息传的慢。

表 5.1: LS 算法和 DV 算法的比较

链路状态 LS	距离矢量 DV
链路状态信息在全网传播	距离矢量仅向邻居发送
每个节点仅传播可靠的信息:	节点传播的信息可能不正确:
亲自测量的本地链路代价	有些距离矢量是“道听途说”的
节点计算的路由不传播,	节点计算的路由要传播,
错误不扩散	会造成错误扩散

第 5.3 节 因特网中自治系统内部的路由选择

由于规模以及 ISP 希望按照自己的意愿运行路由器等原因, 路由器组织引入自治系统 (Autonomous System, AS) 来解决。自治系统是由处于同一个管理域下的网络和路由器组成的集合。

每个 AS 被赋予一个 AS 编号，由 ICANN 分配；同一个 AS 中的路由器运行相同的选路协议（称 Intra-AS 选路协议）；不同 AS 中的路由器可以运行不同的 Intra-AS 选路协议。AS 与外界的接口被称作网关路由器，即在一个 AS 内、直接连接到其它 AS 的路由器。网关路由器之间运行 Inter-AS 选路协议；所有 AS 必须运行相同的 Inter-AS 选路协议。（注意这两个用词是不一样的）

5.3.1 RIP

路由信息协议（Routing Information Protocol, RIP）是一种 DV 算法，其距离矢量定义为跳数，每条链路的 cost 为 1，最大跳数为 15（16 就是不可达）。每个节点每隔 30 秒（或者在有请求的时候）和邻居交换 DV，通告（advertisement, AD）。每个通告包括至多 25 个目标子网，代价值和描述 etc。RIP 报文封装在 UDP 报文中发送，使用 UDP 端口 520（**RIP 是一个应用层协议！**即一个应用层进程为了完成网络层的功能，借用了传输层的协议）

5.3.2 OSPF

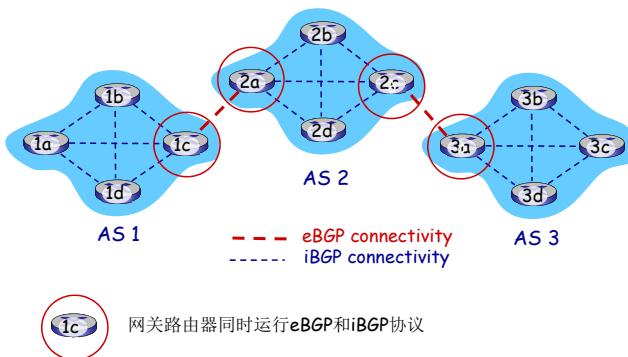
开放最短路径优先（Open Shortest Path First, OSPF）是一种 LS 协议，他使用洪泛链路状态信息和 Dijkstra 算法。OSPF 的“高级”特性（在 RIP 中没有的）有：

1. 安全：所有的 OSPF 报文都是经过认证的（防止恶意的攻击）
2. 允许有多个代价相同的路径存在（在 RIP 协议中只有一个）
3. 对于每一个链路，对于不同的 TOS 有多重代价矩阵（不同的代价评价标准）
 - (a) 例如：卫星链路代价对于尽力而为的服务代价设置比较低，对实时服务代价设置的比较高
 - (b) 支持按照不同的代价计算最优路径，如：按照时间和延迟分别计算最优路径
4. 对单播和多播的集成立支持：Multicast OSPF (MOSPF) 使用相同的拓扑数据库，就像在 OSPF 中一样
5. 在大型网络中支持层次性 OSPF

OSPF 分组被封装在 IP 包中传输，协议号为 89。路由器周期性地、或在链路状态改变时发送 OSPF 链路通告。

层次性的 OSPF 对于一个比较大的网络，OSPF 支持如下分区操作：将整个网络分为很多个 area（本地区）和一个 backbone（主干区），每一个 area 相对独立，只在本地之内进行信息的洪泛（链路状态通告）和 LS 网络构建。所以，每个节点只拥有本地内部的拓扑和代价信息，而关于其他区域，只知道去它的方向（最短路径）（通过边界路由器）。主干区域总是包含本 AS 中所有的区域边界路由器，并且还可能包含了一些非边界路由器。

做一个不一定恰当的比喻，一个 AS 是一个国家，每一个本地区域是一个城市，城市内部的路况对本城市以内的人公开，而对外面城市保密。每一个城市有一些城门，他们各自有通往主干区的一些道路。主干区负责对所有城际物流进行调配。



第 5.4 节 ISP 之间的路由选择: BGP

边界网关协议(Broder Gateway Protocol, BGP)是一种自治系统间路由选择协议(inter-autonomous system routing protocol)是一种“事实上的”标准，他是将互联网各个 AS 粘在一起的胶水。在 BGP 中，分组并不是路由到一个特定的目的地址，相反是路由到 CIDR(无类别域间路由)化的前缀，其中每个前缀表示一个子网或一个子网的集合。在 BGP 的世界中，一个目的地址可以采用如 138.16.68/22 的形式，对于这个例子包括 1024 个 IP 地址。因此，一台路由器的转发表将具有形式为 (x, I) 的表项，其中 x 是一个前缀(掩码)， I 是该路由器的接口之一的接口号。

BGP 提供给每个 AS 以下方法：① eBGP：从相邻的 Ases 那里获得子网可达信息 ② iBGP：将获得的子网可达信息传遍到 AS 内部的所有路由器 ③ 根据子网可达信息和策略来决定到达子网的“好”路径。

5.4.1 通告 BGP 路由信息

自治系统彼此并未实际发送报文，相反是路由器在发送报文。在 BGP 中，每对路由器通过使用 179 端口的半永久 TCP 连接交换路由选择信息。每条直接连接以及通过该链接发送的 BGP 报文，称为 BGP 连接(BGP connection)。注意到 iBGP 连接并不是与物理链路对应

5.4.2 确定最好的路由

当路由器通过 BGP 连接通告前缀时，他在前缀中包括一些 BGP 属性(BGP attribute)。用 BGP 术语来说，前缀及其属性称为路由(route)。两个较为重要的属性是 AS-PATH 和 NEXT-HOP。前者包含了通告已经通过的 AS 的列表，可以用于检查和防止通告环路、多路径选择。在向其他 AS 转发时，需要将自己的 AS 号加在路径上。后者是 AS-PATH 起始的路由器接口的 IP 地址，作用是如果从当前 AS 到下一跳 AS 有多个链路，NEXT-HOP 可以告诉对方是通过哪个转发的。

在上面的图中假设从路由器 1d 到路由器 3d 附加一跳物理链路。在这种情况下，从 AS1 到 3d 的右边端口有两条路径。那么，对于路由“AS2 AS3 x”，其属性 NEXT-HOP 是路由器 2a 左边接口的 IP 地址。对于路由“AS3 x”，其属性 NEXT 是路由器 3d 最左边接口的 IP 地址

热土豆路由选择

第 5.5 节 SDN 控制平面

第 5.6 节 ICMP：因特网控制报文协议

因特网控制报文协议 (Internet Control Message Protocol, ICMP)，被主机和路由器用来彼此沟通网络层的信息。它最典型的用途是差错报告。ICMP 通常被认为是 IP 的一部分（即在网络层），但从体系结构上讲它位于 IP 之上，这是因为 ICMP 报文是作为 IP 有效载荷承载的，他的封装解封装都和 TCP、UDP 一样。

ICMP 报文有一个类型字段和一个编码字段，并且包含引起该 ICMP 报文首次生成的 IP 数据报的首部和前 8 个字节

第 6 章 链路层和局域网

在链路层的讨论中，我们将看到两种截然不同类型的链路层信道。第一种是广播信道，第二种是点对点通信线路，这在诸如长距离链路连接的两台路由器之间，或用户办公室计算机与他们所连接的临近以太网交换机之间等场合经常能够发现。

第 6.1 节 引论和服务

运行链路层协议（即第二层）的任何设备都称为节点（node），包括主机、路由器、交换机、网桥和 Wi-Fi 接入点等。沿着通信路径，连接各相邻节点通信信道的是链路（link），包括有线链路、无线链路、局域网。第二层协议数据单元叫帧（frame），封装数据报。数据链路层负责从一个节点通过链路将（帧中的）数据报发送到物理相邻节点。

网络层选定从源节点到目标节点的路径（指定出行方案），而具体的实施就交给了链路层。路径由一系列路由器和链路组成，路径上的链路可能不同。路径由一系列路由器和链路组成，路径上的链路可能不同。

6.1.1 链路层提供的服务

链路层所提供的服务细节能够随着链路层协议的不同而变化。链路层协议能够提供的可能服务包括：

1. 成帧（framing）。帧的结构由链路层协议规定。
2. 链路接入。媒体访问控制（Medium Access Control, MAC）协议规定了帧在链路层上传输的规则。
3. 可靠交付。（注意其实链路层是可以有可靠交付的选项的）与运输层可靠交付服务类似，链路层的可靠交付服务通常是通过确认和重传取得的。链路层可靠交付通常用于易于产生高差错率的链路，例如无线链路，其目的是本地（也就是在差错发生的链路上）纠正一个差错，而不是通过运输层或应用层协议迫使进行端到端的数据重传
4. 差错检验和纠正。通常让发送节点在帧中包含差错检验比特，让接收节点进行检查。相比于运输层和网络层有限形式的差错检测，链路层的通常更复杂，并且用硬件实现，而且能够更准确地确定帧中的差错出现的位置。
5. 半双工和全双工。半双工：链路可以双向传输，但一次只有一个方向。半双工通信时，提供收发转换。

6.1.2 链路层在何处实现

链路层的主体部分在网络适配器（network adapter）中实现的，网络适配器有时候也称为网络接口卡（Network Interface Card, NIC）。位于网络适配器核心的是链路层控制器，该控制器通常是一个实现了许多链路层服务（成帧、链路接入、差错检测等）的专用芯片

第 6.2 节 差错检测和纠正技术

链路层通常提供的两种服务是比特级差错检测和纠正，即对从一个节点发送到另一个物理上的临近节点的链路层帧中的比特损伤进行检测和纠正。为了保护比特免收差错，使用差错检测和纠正比特（Error-Detection and Correction, EDC）来增强数据 D。现在来研究在数据传输中检测差错的三种技术：奇偶校验（用来描述差错检测和纠正背后隐含的基本思想）、检验和方法（通常更多的用于运输层）和循环冗余检测（通常更多的用于在适配器中的链路层）

由于在组成原理笔记中已经涉及到此部分内容，在计网笔记中仅记录一小部分。

6.2.1 奇偶校验

奇偶校验的一种进阶办法是二维奇偶校验（two-dimensional parity）方案，这里 D 中的 d 个比特被划分为 i 行 j 列。对每行和每列计算奇偶值。产生的 $i+j+1$ 个奇偶比特构成里差错检验比特。这样接收方可以利用列和行对索引来实际识别发生差错的比特并纠正他。

接收方检测和纠正差错的能力被称为前向纠错（Forward Error Correction, FEC）。在网络环境中，FEC 可以单独应用，或与链路层自动重传请求（Automatic Repeat-reQuest, ARQ）技术一起使用。FEC 技术的价值在于可以减少所需的发送方重发的次数

第 6.3 节 多路访问链路和协议

点对点链路由链路一端的单个发送方和链路另一端的单个接收方组成。第二种是广播链路，它能够让多个发送和接收节点都连接到相同的、单一的、共享的广播信道上。在本节，先研究一个对链路层很重要的问题：如何协调多个发送和接收节点对一个共享广播信道的访问，这就是多路访问问题。节点通过多路访问协议来规范他们在共享的广播信道上的传输行为。由于多个节点可能在同时传输帧，那么这时候所有节点同时收到很多帧；也就是说，传输的帧在所有的接收方处碰撞（collide）。通常，当碰撞发生时，没有一个接受节点能够有效的获得任何传输的帧；在某种意义上，碰撞帧的信号纠缠在一起，因此，涉及此次碰撞的所有帧都丢失了。

我们将任何多路访问协议划分为 3 种类型之一：信道划分协议（channel partitioning protocol），随机接入协议（random access protocol）和轮流协议（taking-turns protocol）。在理想情况下，对于速率为 R bps 的广播信道，多路访问协议应该具有以下所期望的特性：

1. 当仅有一个节点发送数据时，该节点具有 R bps 的吞吐量
2. 当有 M 个节点发送数据时，每个节点吞吐量为 R/M bps。这不必要求 M 个节点中的每一个节点总是有 R/M 的瞬间速率，而是每个节点在一些适当定义的时间间隔内应该有 R/M 的平均传输速率

3. 协议是分散的；这就是说不会因某主节点故障而使整个系统崩溃
4. 协议是简单的，使实现不昂贵

6.3.1 信道划分协议

比如可以有时分多路复用（TDM）和频分多路复用（FDM）可以用。TDM 将时间划分为时间帧（time frame），并进一步划分每个时间帧为 N 个时隙（slot）。（应当注意的是，不应当把 TDM 时间帧与在发送和接收适配器之间交换的链路层数据单元相混淆。为了减少混乱，我们将后者称为分组）第三种信道划分协议是码分多址（Code Division Multiple Access, CDMA）。CDMA 为每个节点分配一种不同的编码

打一个比方，TDM 就像是不同的人在不同的时刻讲话；而 FDM 是不同的组在不同的小房间里通信；而 CDMA 则是不同的人使用不同的语言讲话

具体的工作原理是：

1. 将每个比特时间进一步划分为 m 个微时隙（称 chip）
2. 每个节点被分配一个唯一的 m 比特码序列（称 chip code）
3. 发送方编码：发送“1”=发送 chip code；发送“0”=发送 chip code 的反码
4. 信号干扰：多个节点发送的信号在信道中线性相加
5. 接收方解码：用发送方的 chip code 与信道中收到的混合信号计算内积，恢复出原数据
6. 前提条件：任意两个 chip code 必须是相互正交的
7. CDMA 允许所有节点同时使用整个信道！

6.3.2 随机接入协议

在随机接入协议中，一个传输节点总是以信道的全部速率进行发送。他并没有节点之间的预先协调，而是在有碰撞时，涉及碰撞的每个节点反复的重发他的帧（分组），直到该帧无碰撞为止。但是当一个节点经历一次碰撞时，他不必立刻重发该帧。相反，他在重发之前等待一个随机时延。涉及碰撞的每个节点独立地随机选择时延。

时隙 ALOHA 在对时隙 ALOHA 的描述中，我们做以下假设：

- 所有帧由 L 比特组成
- 时间被划分为长度为 L/R 秒的时隙，每个时隙可以发送一帧
- 节点只在时隙开始的时候传帧
- 节点在时钟上是同步的
- 如果两个或多个节点在一个时隙传输，所有的站点都能检测到冲突

而时隙 ALOHA 的运行过程如下：① 当节点获取新的帧，在下一个时隙传输 ② 传输时没有检测到冲突，成功（节点能够在下一时隙发送新帧）③ 检测时如果检测到冲突，失败（节点在每一个随后的时隙以概率 p 重传帧直到成功）。时隙 ALOHA 的优点很明显，比如① 节点可以以信道带宽全速连续传输 ② 高度分布：仅需要节点之间在时隙上的同步 ③ 简单。

当只有一个活跃节点时，时隙 ALOHA 工作出色，可是当由多个活跃节点时，将会出现两个要考虑的效率问题。首先，当有多个活跃节点时，一部分时隙将碰撞（而浪费掉）。第二个考虑是，时隙的另一部分是“空闲”的，因为所有活跃节点由于概率传输策略会节制传输。刚好有一个节点传输的时隙称为一个成功时隙（successful slot）。

时隙多路访问协议的**效率**定义为：当有大量的活跃节点且每个节点总有大量的帧要发送时，长期运行中成功时隙的份额。当有 N 个活跃节点时，由概率论知识（二项分布）时隙 ALOHA 的效率是 $Np(1 - p)^{N-1}$ 。当 N 趋于 ∞ 时极限效率为 $1/e \approx 0.37$

ALOHA 在纯 ALOHA 是一个非时隙、完全分散的协议，他不要求节点在时钟上同步，当一帧首次到达，节点立刻将该帧完整的传送进广播信道，如果发生了碰撞，就立即以概率 p 传输该帧，否则等待一个帧传输时间。优点是真的简单，无须在时间节点上进行同步。但是相应的，冲突的概率增加，极限效率减为 $1/2e \approx 1.75$

载波侦听多路访问 在时隙和纯 ALOHA 中，一个节点传输当决定独立于连接到这个广播信道上的其他节点的活动。特别是，一个节点不关心在他开始传输时是否有其他节点碰巧在传输。所以我们增加两个重要的规则：

1. 说话之前先听。这一点称为载波侦听（carrier sensing），即一个节点在传输之前先听信道，直到监测到一段长时间没有传输，然后开始传输
2. 如果与他人同时开始说话，停止说话。这一点被称为碰撞检测（collision detection），即当一个传输节点在传输时一直同时在侦听此信道。如果他监测到另一个节点正在传输干扰帧，就停止传输，在重复“侦听-空闲时传输”循环之前等待一段随机时间

这两个规则包含在载波侦听多路访问（Carrier Sense Multiple Access, CSMA）和具有碰撞检测的 CSMA（CSMA with Collision Detection, CSMA/CD）协议族中。这里，我们将考虑一些 CSMA 和 CSMA/CD 最重要的和最基本的特性。

如果所有节点都被载波侦听了，为什么还会发生碰撞 由传播延迟造成：两个节点可能侦听不到正在进行的传输。这里要学会读下面这个时空图，颜色区域的意思是分组从一个几点向双向传输，在时间轴上的占用段。而两种颜色的交叠区域即为冲突时间。显然广播信道的端到端信道传播时延（channel propagation delay）（即信号从一个节点传播到另一个节点所花费的时间）在决定性能方面起着关键的作用。该传播时延越长，载波侦听节点不能侦听到网络中另一个节点已经开始传输的机会就越大。

具有碰撞检测的载波侦听多路访问 注意上面那个图中，在“碰撞检测/放弃时间”之后， B 和 D 都停止了传输动作。显然，在多路访问协议中加入碰撞检测，通过不传输一个无用的、（由来自另一个节点的帧干扰）损坏的帧，将有助于改善协议的性能。终止传输之后，适配器等待一个

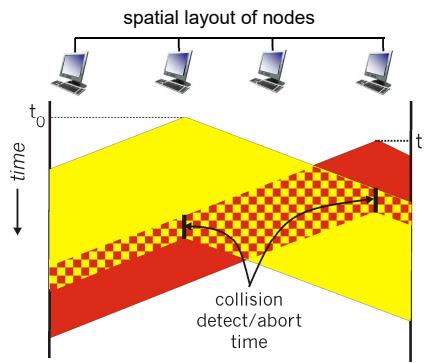


图 6.1: 其实是 CSMA/CD 的图

随机时间量。等待一个随机（而不是固定）的时间量的需求是明确的——如果两个节点同时传输帧，然后这两个节点等待相同固定的时间量，他们将持续碰撞下去。

二进制指数后退算法解决了间隔时间的选择的问题。特别的，当传输一个给定帧时，在该帧经历了一连串的 n 此碰撞后，节点随机的从 $\{0, 1, 2, \dots, 2^n - 1\}$ 中选择一个个 K 值。对于以太网，一个节点等待的实际时间量是 $K \cdot 512$ 比特时间， n 能够取的最大值在 10 以内

CSMA/CD 效率 CSMA/CD 效率定义为：当有大量的活跃节点，且每个节点由大量的帧要发送时，帧在信道中无碰撞的传输的那部分时间在长期运行时间中所占的份额。令 d_{prop} 表示信号能量在任意两个适配器之间传播所需的最大时间。令 d_{trans} 表示传输一个最大长度的以太网帧的时间。则可以列出下面的近似式：

$$\text{efficiency} = \frac{1}{1 + 5d_{prop}/d_{trans}}$$

6.3.3 轮流协议

多路访问协议的两个理想特性是：① 当只有一个节点活跃时，该活跃节点具有满吞吐量；② 当有 M 个节点活跃时，每个活跃节点的吞吐量接近平均分配。ALOHA 和 CSMA 具有第一个特性但不具备第二个特性。所以引出了下面要讨论的轮流协议们。主要讨论两种比较重要的：轮询协议和令牌传递协议。

轮询协议 要求一个节点被指定为主节点，他轮询每个节点，并告诉他能够传输的帧的最数量。主节点通过观察在信道上是否缺乏信号，来决定一个节点何时完成了帧的发送。

轮询协议引入了轮询时延（轮询开销），且从节点会受到更多限制（借用 OS 进程调度的话讲，一个没有在传输信号的节点会阻塞（影响）其他节点）。更严重的问题是，如果主节点 down 了，整个系统都会崩溃掉

令牌传递协议 一个称为令牌（token）的小的特殊帧在节点之间以某种固定的次序进行交换（像是一个互斥锁）。当一个节点收到令牌时，仅当他有一些帧要发送时，他才持有这个令牌，并发送最大数目的帧数；否则，他立即讲这个令牌向下一个节点转发。但是，令牌本身会消耗一定

的带宽，而且，一个节点的故障可能会是整个信道崩溃。或者一个节点偶然忘记了释放令牌，则必须调用某些恢复步骤使令牌返回到循环中。如果令牌丢失，则需要通过复杂机制重新生成令牌。

6.3.4 DOCSIS：用于电缆因特网接入的链路层协议

PPT 上面没有，不想看了 orz

第 6.4 节 交换局域网

先来看一下三种范围的网络概念：

1. 局域网 LAN (Local Area Network)：将小范围内的计算机及外设连接起来的网络，范围在几公里以内
2. 城域网 MAN (Metropolitan Area Network)：通常覆盖一个城市的范围（几十公里），如有线电视网、宽带无线网等。城域网要能支持数据、音频和视频在内的综合业务，服务质量好，支持用户数量多
3. 广域网 WAN (Wide Area Network)：通常覆盖一个国家或一个洲（一百公里以上），规模和容量可任意扩大

6.4.1 链路层寻址和 ARP

每一块网络适配器(网卡)固定分配一个地址，称为 MAC 地址，也称物理地址、硬件地址、链路层地址等。MAC 地址长 6 个字节，一般用由“:”或“-”分隔的 6 个十六进制数表示。MAC 地址由 IEEE 负责分配，每块适配器的地址是全球唯一的：网卡生产商向 IEEE 购买一块 MAC 地址空间（前 3 字节）生产商确保生产的每一块网卡有不同的 MAC 地址。MAC 地址固化在网卡的 ROM 中，不过现在用软件改变网卡的 MAC 地址也是可能的。

事实上，并不是主机或路由器具有链路层地址，而是他们的适配器（即网络接口）具有链路层地址，就像 IP 地址一样。重要的是注意到链路层交换机并不具有与他们的接口（这些接口是与主机和路由器相连的）相关联的链路层地址。适配器的 MAC 地址具有扁平结构（这与层次结构相反），而且无论适配器到哪里用都不会变化。与之形成对照的是，IP 地址具有层次结构（即一个网络部分和一个主机部分），而且当主机移动时，主机的 IP 地址需要改变。

适配器可以接收一个并非向他寻址的帧。当适配器接收到一个帧时，将检查该帧中的目的 MAC 地址是否与他自己的 MAC 地址匹配。如果匹配，该适配器提取出封装的数据报，并将该数据报沿协议栈向上传递。不匹配就丢弃。目的 MAC 地址共有三种类型：

1. 单播地址：适配器的 MAC 地址，地址最高比特为 0
2. 多播地址：标识一个多播组的逻辑地址，地址最高比特为 1
3. 广播地址：**ff:ff:ff:ff:ff:ff**

地址解析协议 在网络层地址（如因特网的 IP 地址）和链路层地址（即 MAC 地址）之间进行转换，就是地址解析协议（Address Resolution Protocol, ARP）的任务。

每台主机或路由器在其内存中具有一个 ARP 表，这张表包含 IP 地址到 MAC 地址的映射关系。该 ARP 表也包含一个寿命（TTL）值。注意这张表不必为该子网上的每台主机和路由器都包含一个表项；某些可能从未进入到该表中，某些可能已经过期。从一个表项放置到某 ARP 表开始，过期时间通常是 20 分钟。

假如 cache 命中失败，采用老的策略，并在收到 ARP 响应后将地址绑定缓存到 ARP 表中 当一个主机要发送一个数据报时，该数据报要 IP 寻址到本子网上另一台主机或路由器。发送主机需要获得给定 IP 地址的目的主机的 MAC 地址。可是如果发送方的 ARP 表中当前没有该目标主机的表项时（是不是有一点像 page fault 呢），发送方便用 ARP 协议来解析这个地址。

首先，发送方构造一个称为 ARP 分组（ARP packet）的特殊分组。然后他广播包含 B 的 IP 地址的 ARP 查询包（DEST MAC ADDRESS = FF-FF-FF-FF-FF-FF）。当接收方收到这个查询包时，（由于广播地址）每个适配器都把该帧中都 ARP 分组向上传递给 ARP 模块。这些 ARP 模块都检查他的 IP 地址是否与 ARP 分组中的目的 IP 地址相匹配。与之匹配的一个给查询的主机发送回一个带有所希望映射的相应 ARP 分组。

6.4.2 关于 ARP 缓存

1. 每个节点在内存中维护一个地址映射（绑定）表，称 ARP 表。
2. 每次发送数据报前先查询 ARP 缓存，找不到需要的地址映射再发送 ARP 请求进行查询，并在收到 ARP 响应后将地址绑定缓存到 ARP 表中。
3. ARP 缓存中的信息动态更新，并在超时（一般为 15~20 分钟）后删除。
4. 从 ARP 请求中获取地址绑定信息：节点可以收到全部的 ARP 请求报文，从而可以将发送节点的地址绑定缓存到自己的 ARP 表中。
5. ARP 是即插即用的（节点自己创建 ARP 的表项无需网络管理员的干预）节点在启动时自动广播自己的地址绑定：
 - (a) 节点 A 在启动时主动广播一个 ARP 请求，在目标字段内填入自己的 IP 地址。
 - (b) 收到 ARP 请求的节点将 A 的地址绑定保存在自己的 ARP 表中。
 - (c) 若 A 收到 ARP 响应，报告 IP 地址重复错误。

发送数据报到子网以外 注意每台主机只有一个 IP 地址和一个适配器。但是，一台路由器对它每个接口都有一个 IP 地址。对路由器的每一个接口，（在路由器中）也有一个 ARP 模块和一个适配器。当然，网络中的每个适配器都有自己的 MAC 地址。

当一台主机发送一个数据报到子网以外时，它填写目标地址的 IP 地址，以及子网中那个路由器接口的 MAC 地址（通过 ARP 获得）。而此路由器在发送的帧中填入数据报要被转发的下一个正确接口。如在第四章中讨论的，这是通过查询路由器中的转发表来完成的。转发表告诉这台路由器该数据报要走的路径。不过，下一跳的 MAC 地址也是通过 ARP 方法获得的

6.4.3 Ethernet: 以太网

以太网是第一个广泛应用的局域网技术，也是现在占主导地位的有线局域网技术。其优点是比其他的局域网技术简单、成本低

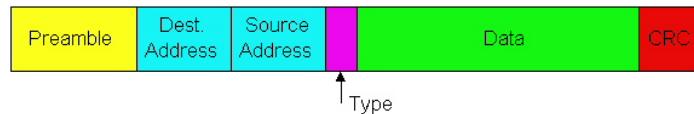
物理拓扑

1. 总线型拓扑：以同轴电缆作为共享传输媒体（总线），所有节点通过特殊接口连接到这条总线上
2. 基于集线器（hub）的星型拓扑：一个物理层设备，把从一个端口进入的物理信号（光，电）放大后立即从其它端口输出
3. 今天的以太网采用基于交换机的星型拓扑：主机通过双绞线或光纤连接到交换机。交换机在端口之间存储-转发帧。交换式以太网是无冲突的！

注意：交换机能增加总带宽：该交换机通过处理帧的 MAC 地址，从一个输入端向一个或多个输出端转发分组，增加了网络总带宽。如一个单个的以太网段只能提供 100 Mpbs 的带宽，若输入与输出主机两两不同的话，则以太网交换机可提供 $100 \times n/2$ Mpbs 的带宽（n 为交换机输入和输出端口的数目）。而此时这些主机之间同时可以进行双工通信，而不会互相干扰

碰撞域 & 广播域 碰撞域描述了一组共享网络访问媒体的网络设备覆盖的区域（就是会信道冲突的区域）。广播域是指广播分组能直接到达的区域（就是一个子网内）

以太帧结构



- **Preamble** (前导码)：
 - 7个10101010字节，后跟一个10101011字节，用于在发送方和接收方之间建立时钟同步
- **Type**: 指出Data所属的高层协议（如IP、ARP等），每个协议有一个编号
- **Data**: 46~1500字节，不足46字节填充至46字节
- **CRC**: 对dest addr.、src addr.、type和data四个字段计算得到的CRC码

以太网帧结构