

计算机网络

Computer Network

0

课程简介

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

主要内容

- 课程信息
- 期末成绩
- 课程内容
- 学习方法



内容纲要

1

课程信息

2

考核方式

3

课程内容

4

学习方法

5

计算机网络概述



课程信息

课程名称（中）	计算机网络
课程名称（英）	Computer Network
课程获奖	2016年福建省教学改革项目 2019年厦门大学一流本科课程建设计划立项
主讲教师	黄 炜 副教授 (whuang@xmu.edu.cn)
教学助理	1班： 老 师 (@xmu.edu.cn) 2班： 老 师 (@xmu.edu.cn)
上课时间地点 (理论课)	1班：周一8:00~9:40，单周周三8:00~9:40，海韵104； 2班：周一10:10~11:50，单周周三10:10~11:50，海韵104；
上课时间地点 (实验课)	1班：双周周三8:00~9:40，海韵实验楼202。 2班：双周周三10:10~11:50，海韵实验楼202。

课程活动（翻转课堂）

- 学生课下学习理论
- 理论课
 - 答疑、习题讲解
 - 项目进展汇报
- 实验课
- 习题课
- 期末测试



内容纲要

1

课程信息

2

考核方式

3

课程内容

4

学习方法

5

计算机网络概述



期末成绩

- 卷面成绩 (50%)

- 期末统一笔试

- 平时成绩 (50%)

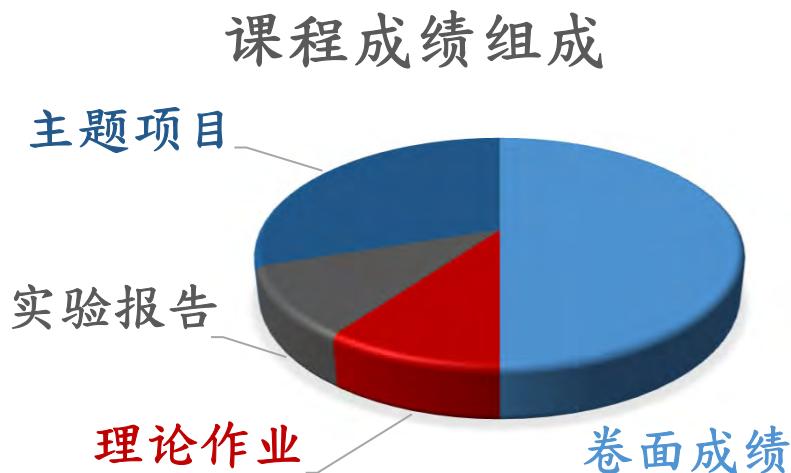
- 纸质作业 (约10%)

- 实验报告 (约10%)

- 课程项目 (约30%)

- 考勤成绩 (倒扣分)

- 按校规，课程或实验三分之一以上缺勤不得参加期末考试



课程项目

- 完成形式

- 以4人为一组，推举1人为组长，不许跨班组队
 - 每周课轮流上台报告和点评

- 内容

- 代码阅读
 - 程序设计与实现

内容纲要

1

课程信息

2

考核方式

3

课程内容

4

学习方法

5

计算机网络概述



高内聚、低耦合

- 软件工程的高内聚、低耦合
 - 内聚是就其中任何一个模块的内部特征而言的。
 - 耦合是就多个模块组成的系统中各模块关联关系而言的。
- 计算机网络的高内聚、低耦合：网络协议分层
 - 计算机网络探讨不同环境下从硬件到软件的双机互联，及过程中遇到的问题
 - 概念的组织形式：抽象出共同的接口，构建出一个模型
 - ISO/OSI七层协议模型、TCP/IP四层协议模型、五层协议模型



计算机网络分层

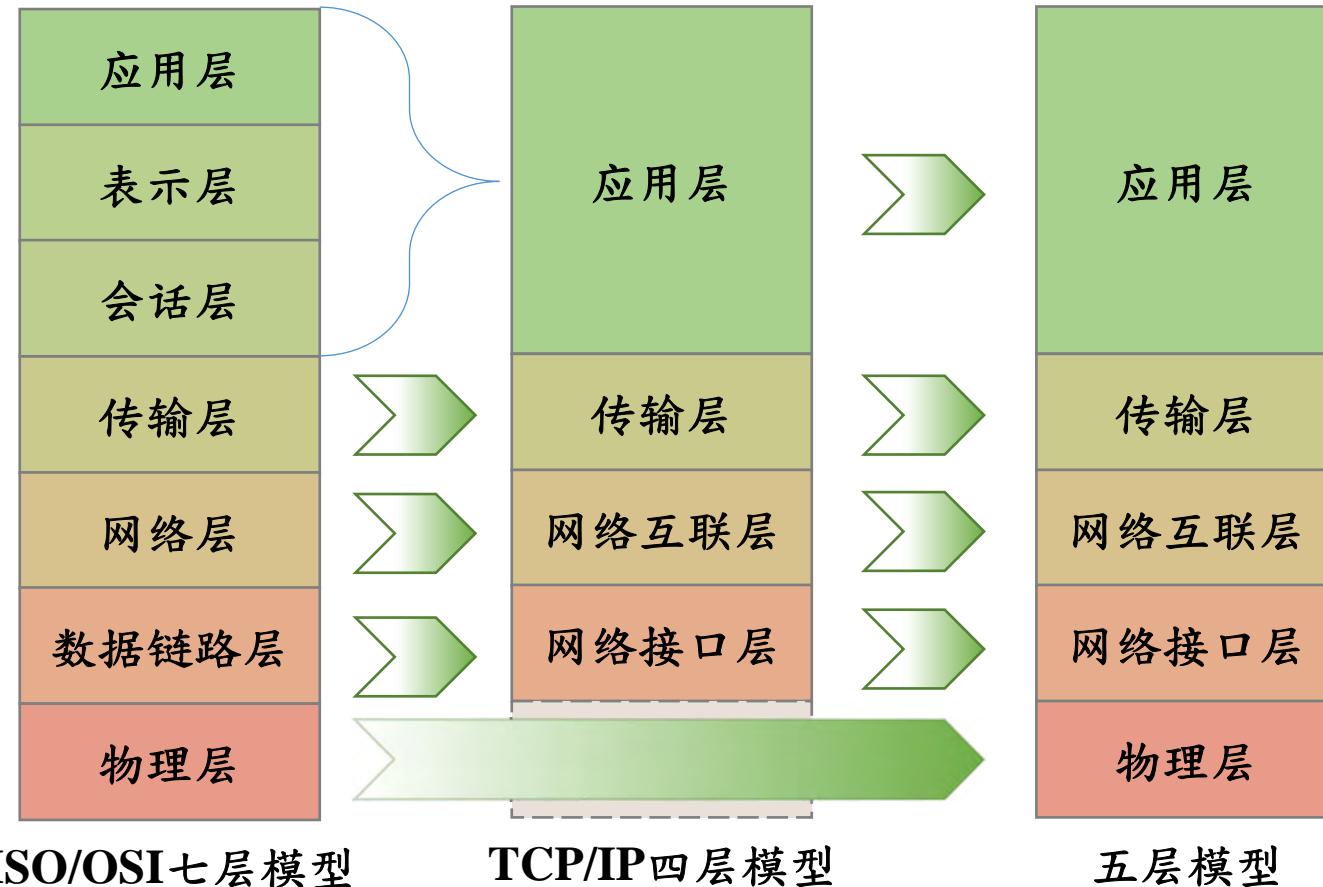
- 不同组织有自己的协议栈

- TCP/IP
- ISO/OSI
- Microsoft
- VOIP
- VPN/Security
- IBM
- Apple



计算机网络学网络通信中的各层

• 计算机网络ISO/OSI七层架构



计算机网络学网络通信中的各层

• 比喻图



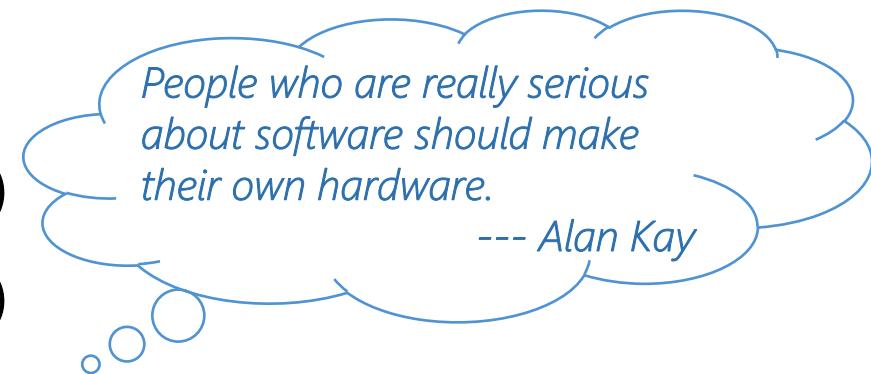
物理层：硬件

- 主要作用

- 机械、电子、定时接口通信信道上的原始比特流传输。

- 主要内容

- 传输介质（光、电、无线电）
 - 近程通信（标准、电气特性）
 - 远程通信（调制、复用）
 - 差错控制（发现错误、纠正）
 - 局域组网（材质、设备、策略）



数字链路层

- 主要作用

- 物理寻址，并将原始比特流转变为逻辑传输线路。
- 多方共享（竞争）使用底层设备、链路流量和差错控制

- 主要内容

- 局域组网（编址、拓扑、竞争机制）
- 扩展设备（网桥、交换机）
- 远程连接（技术、速率、虚拟专用网）
- 广域组网（下一跳、路由）

网络层

- 主要作用

- 性能各异的主机间互联，寻径和错误控制
 - 控制子网的运行，如逻辑编址、分组传输、路由选择。

- 主要内容

- 互连协议（编址、转发、重组、差错报告机制）
 - 支撑协议（获取地址、地址转换、地址解析）



传输层

- 主要作用
 - 提供可靠或轻快的网络进程之间的通信
- 主要内容
 - 数据报文（编址、进程间通信）
 - 传输控制（停等、流量控制、拥塞控制、建立和撤除）



会话层

- 主要作用

- 不同机器上的用户之间建立及管理会话。
- 会话建立、撤销
- 传输同步
- 面向连接的交互活动管理
 - 口令认证
 - 数据传输规范



表示层

- 主要作用
 - 信息的语法语义以及它们的关联
 - 举例
 - 加密解密
 - 转换翻译
 - 压缩解压缩

应用层

- 主要作用
 - 提供通用应用程序，完成用户和软件转换信息的交互
- 主要内容
 - 域名系统
 - 电子邮件
 - 文件访问
 - 万维网络

其它需求

- 信息安全
 - 可靠性、真实性、保密性、完整性、不可抵赖性
- 服务质量：QoS
 - 带宽、延时、抖动（算钱的时候有用）
- 新一代技术
- 网络与社会

内容纲要

1

课程信息

2

考核方式

3

课程内容

4

学习方法

5

计算机网络概述



学习方法

- 记住术语
 - 简称、全称
- 理解原理
 - 为什么、怎么来；搜集相关资料：网络、图书馆
- 亲自实验
 - 注重锻炼编程能力，不要死记硬背
- 积累经验

Practice, Practice, Practice
No Magic



Preface by the Author

- Instructors should impress on students the importance of concepts and principles: specific technologies may become obsolete in a few years, but **the principles will remain.**
- Because programming and experimentation are crucial to helping students learn about networks, **hands-on experience** is an essential part of any networking course.



内容纲要

1

课程信息

2

考核方式

3

课程内容

4

学习方法

5

计算机网络概述



因特网概述

- 因特网

- 起源于美国，世界上最大的国际性计算机互联网
- 网络由若干结点和连接这些结点的链路组成。
 - 网络的node是结点；而数据结构中树的node是节点
 - 连接在因特网上的计算机都称为主机（host）

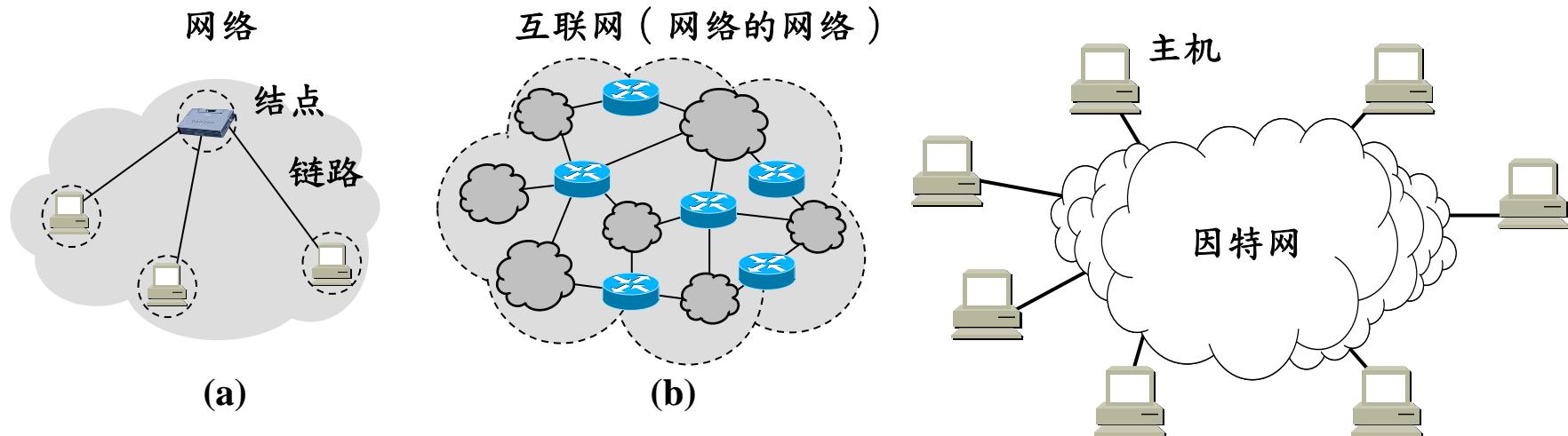
- 互联网：网络的网络（network of networks）

- 第一个网络指的是主机通过共享设备和介质连接为局域网
- 第二个网络指的是通过路由器将局域网连成广域网



网络与因特网

- 网络把许多计算机连接在一起。
- 因特网则把许多网络连接在一起。



因特网

- 因特网发展的三个阶段

1

单个网络到
互联网

1983 年 TCP/IP 协议成 ARPANET 的
标准协议。

2

三级结构的
因特网

建成主干网、地区
网和校园（企业）
网三级结构网。

3

多层次 ISP
结构的因特网

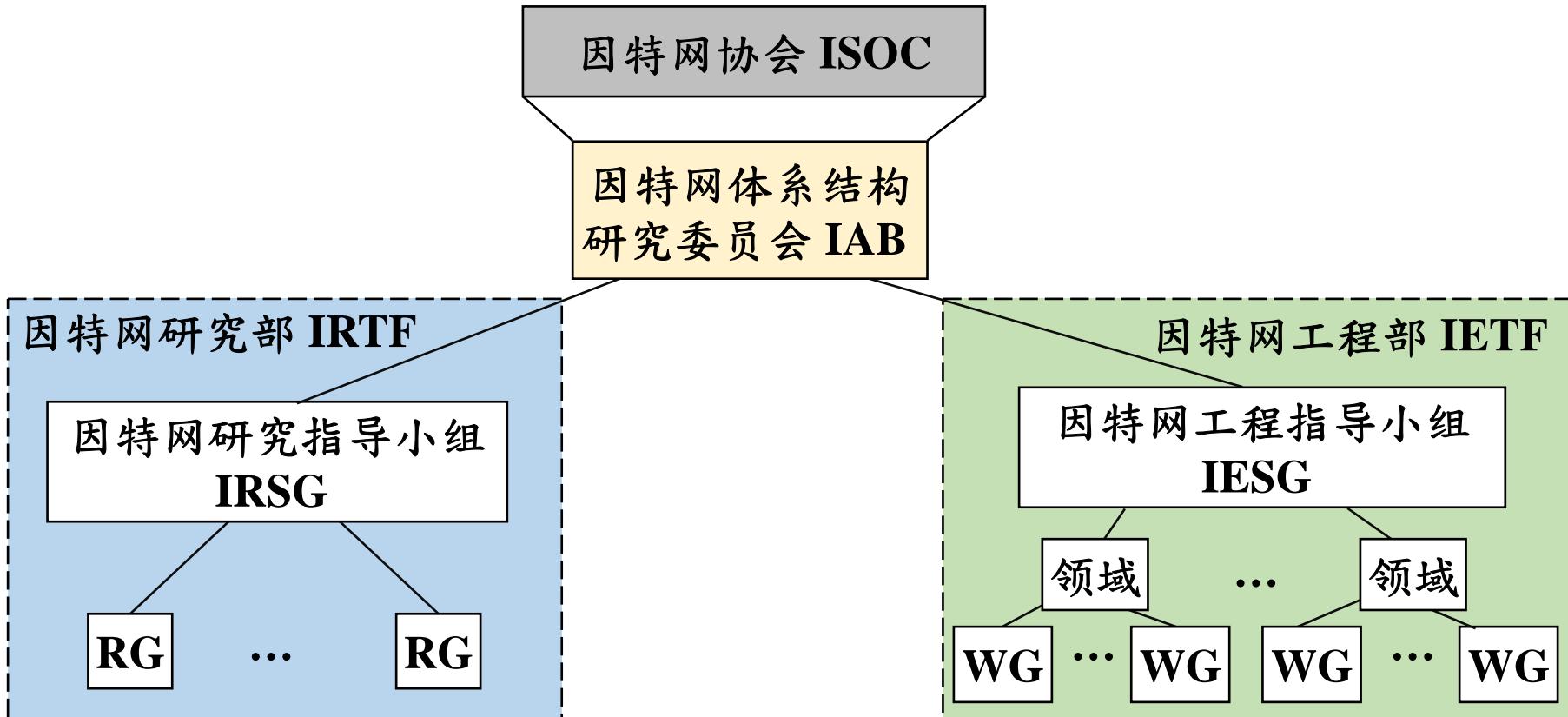
出现了因特网服务
提供者（ISP）。

- 因特网的迅猛发展始于 20 世纪 90 年代。
- 由欧洲原子核研究组织 CERN 开发的万维网 WWW
(World Wide Web)广泛使用在因特网上，方便了广大非网
络专业人员使用网络，驱动因特网指数级增长。



关于因特网的标准化工作

- 因特网协会架构



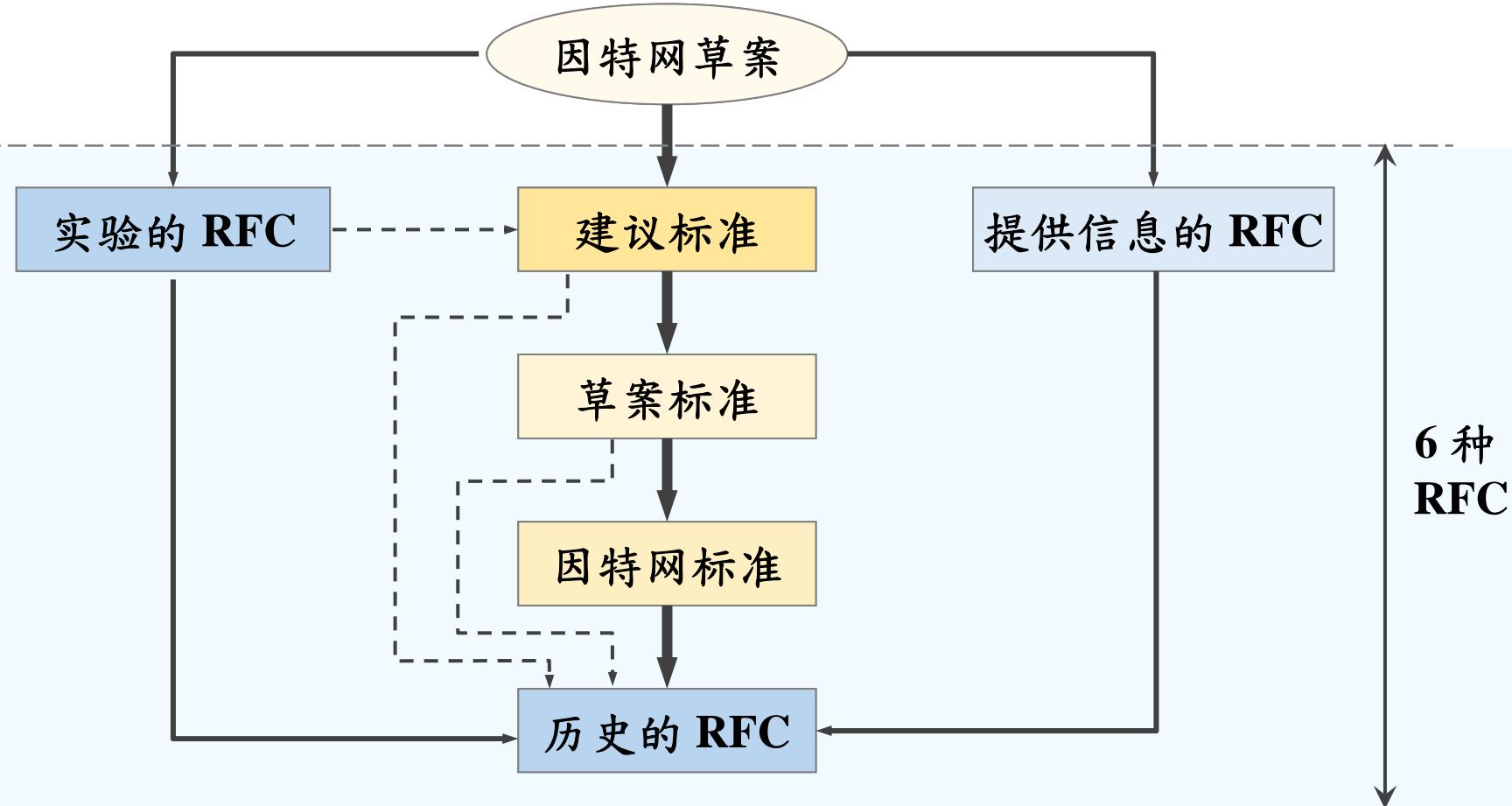
关于因特网的标准化工作

- 制订因特网的正式标准要经过以下的四个阶段
 - 因特网草案(**Internet Draft**)
 - 在这个阶段还不是 RFC 文档。
 - 建议标准(**Proposed Standard**)
 - 从这个阶段开始就成为 RFC 文档。
 - 草案标准(**Draft Standard**)
 - 因特网标准(**Internet Standard**)



关于因特网的标准化工作

- 制订因特网的正式标准要经过四个阶段



计算机网络在我国的发展

- 1980年：铁道部开始进行计算机联网实验。
- 1989年11月：我国首个公用分组交换网建成运行。
- 1994年4月20日：我国用64 kb/s专线正式连入因特网。
- 中国教育和科研计算机网CERNET
 - China Education and Research NETwork，简称为中国教育网，是由国家建设，教育部负责管理，清华大学等高等学校承担建设和管理运行的全国性学术计算机互联网络。
- 中国互联网络信息中心 CNNIC
 - Network Information Center of China，每年两次公布的我国因特网的发展情况。



计算机网络

Computer Network

0

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

计算机网络

Computer Network

1

传输媒介

理论课程

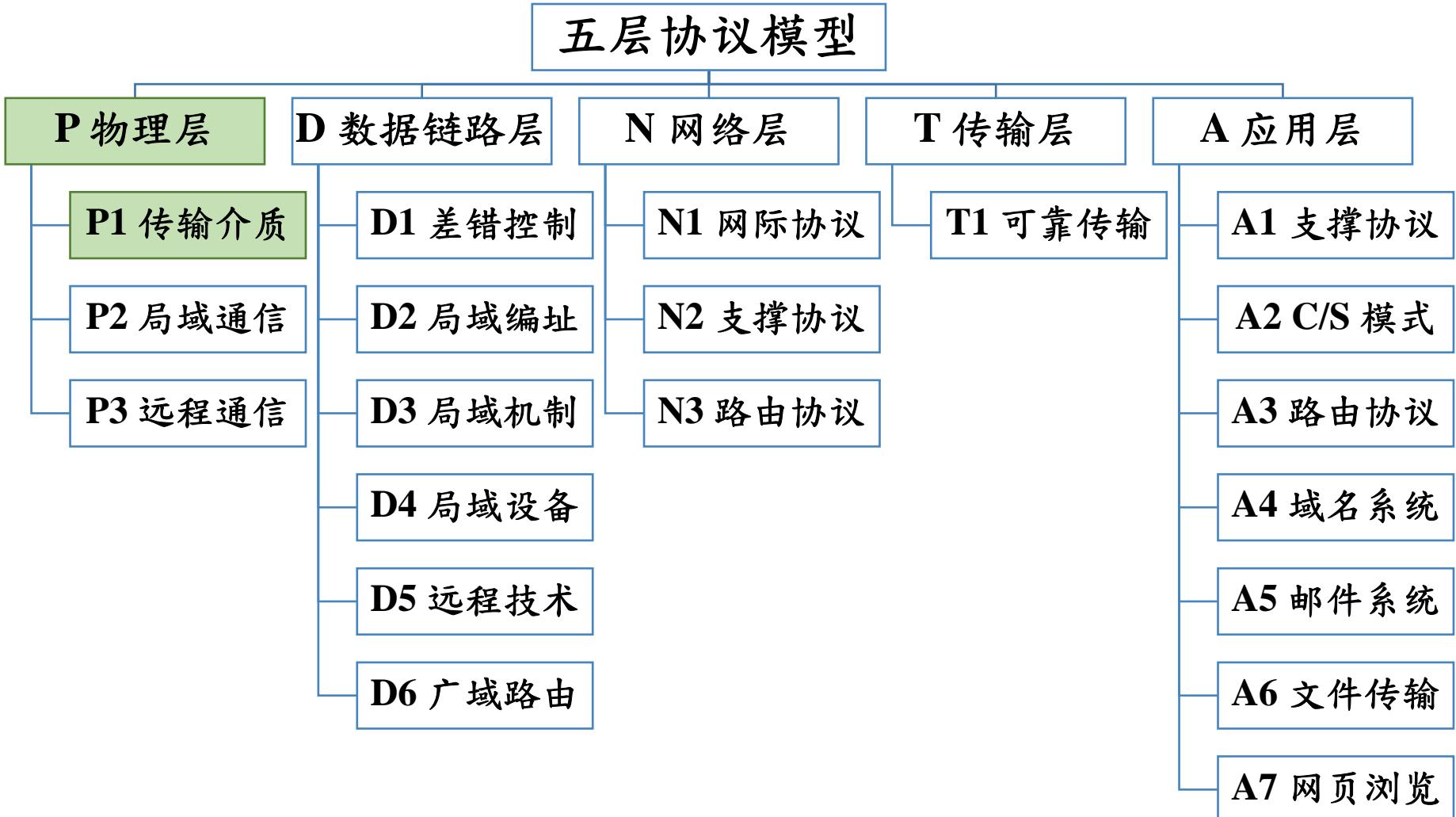


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 通信基本模型
- 引导型传输媒体
 - 屏蔽双绞线，非屏蔽双绞线，同轴电缆
 - 光纤：单模和多模
- 非引导型传输媒体
 - 红外线，激光，无线电波（镭射）、卫星
- 介质间的权衡



对应课本章节

- PART II Data Communication Basics
 - Chapter 5 Overview Of Data Communications
 - Chapter 7 Transmission Media



内容纲要

1

通信的基本概念

2

传输介质的分类

3

传输介质介绍

4

介质的选用标准

5

小结



通信模型

- 通信的基本模型



- 数据通信

- 信息（Information）是任意形式
- 传输系统采用物理系统
- 多个信息源可以共用底层的传输介质



信息的基本单位

- 网络通信的基本单位是：位（ bit，b ）
 - 有2种事件出现的概率各是50%，其信息量是1 bit。
 - 有 2^N 种事件出现的概率各是 2^{-N} ，其信息量是N bit。
 - 一共是N种情况，和事件的具体意义无关。
- 内存存储的基本单位是：字节（ byte，B ）
 - $1 \text{ byte} = 2^3 = 8 \text{ bits}$
 - 因为 2^1 、 2^2 位都无法表示26个字母，故而寻求 2^3 。
 - KB（ 2^{10}B ）、MB（ 2^{20}B ）、GB（ 2^{30}B ）

信号

- 信号：信息的载体，利用物理量携带信息
- 信道：信号的传输媒介。
- 信道：调制信道和编码信道。
- 信道噪声：噪声和干扰的总称。

数字和模拟信号

- 数字信号 (digital signal)

- 指自变量是离散的、因变量也是离散的信号

- 模拟信号 (analog signal)

- 指信息参数在给定范围内表现为连续的信号。

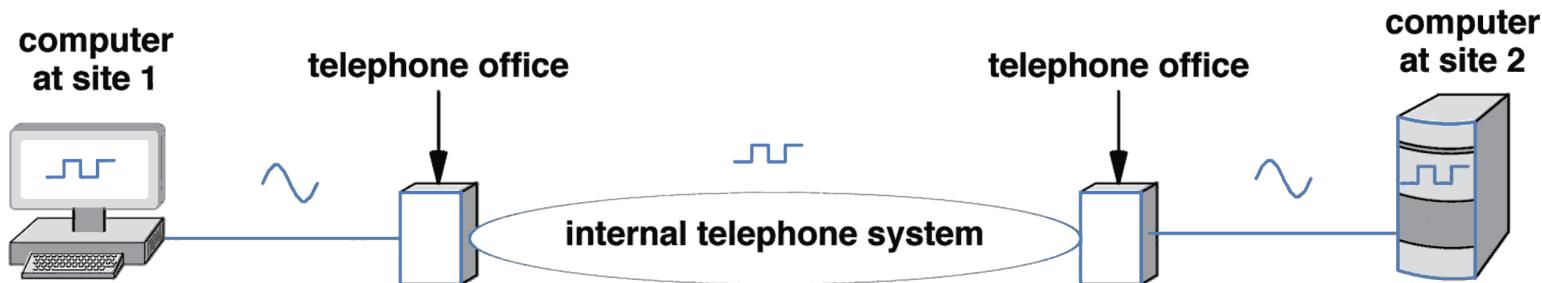
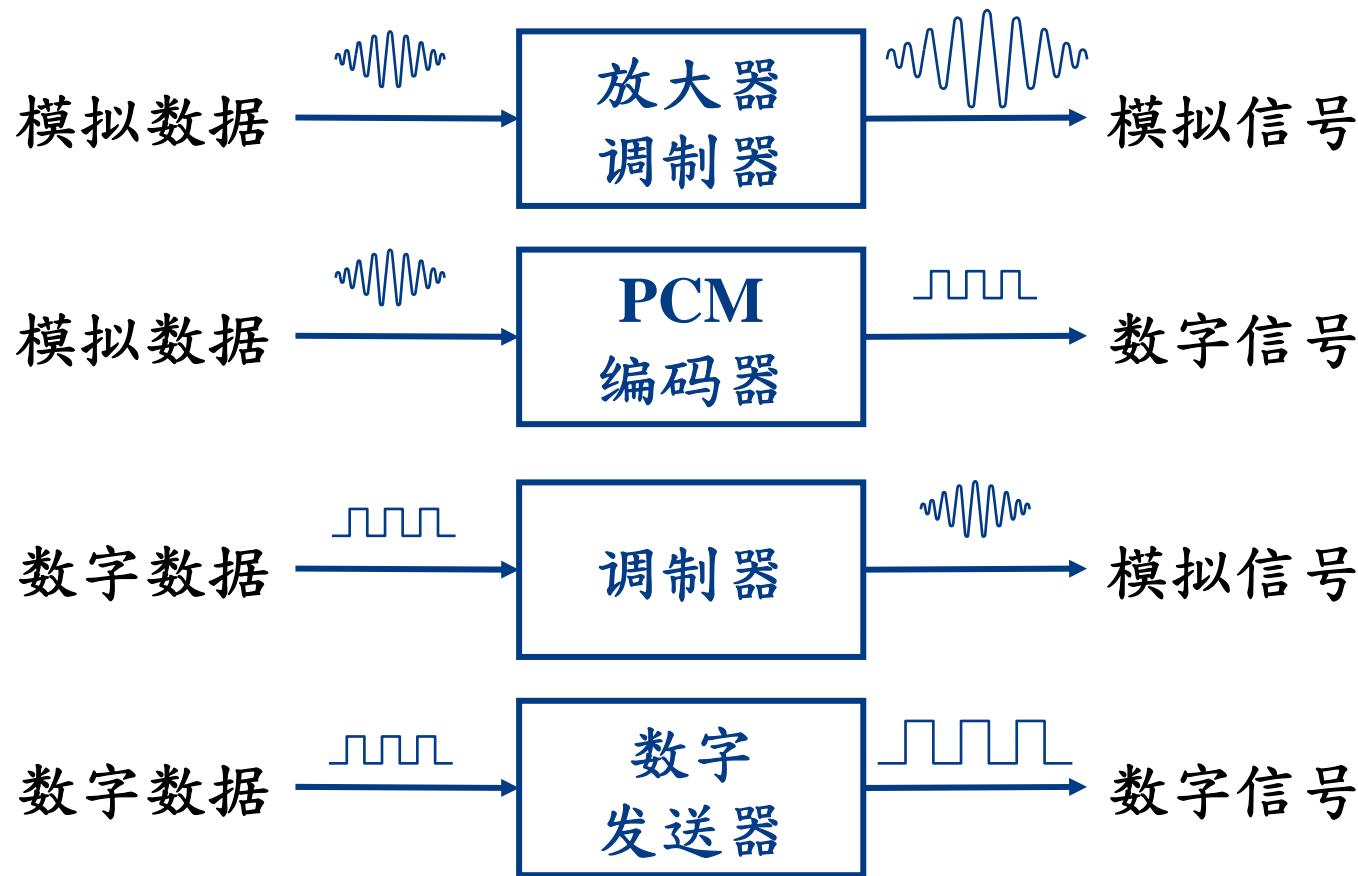


Figure 10.10 Illustration of digital and analog signals (denoted by a square wave and a sine wave) that occur when a dialup modem is used to send data from one computer to another.

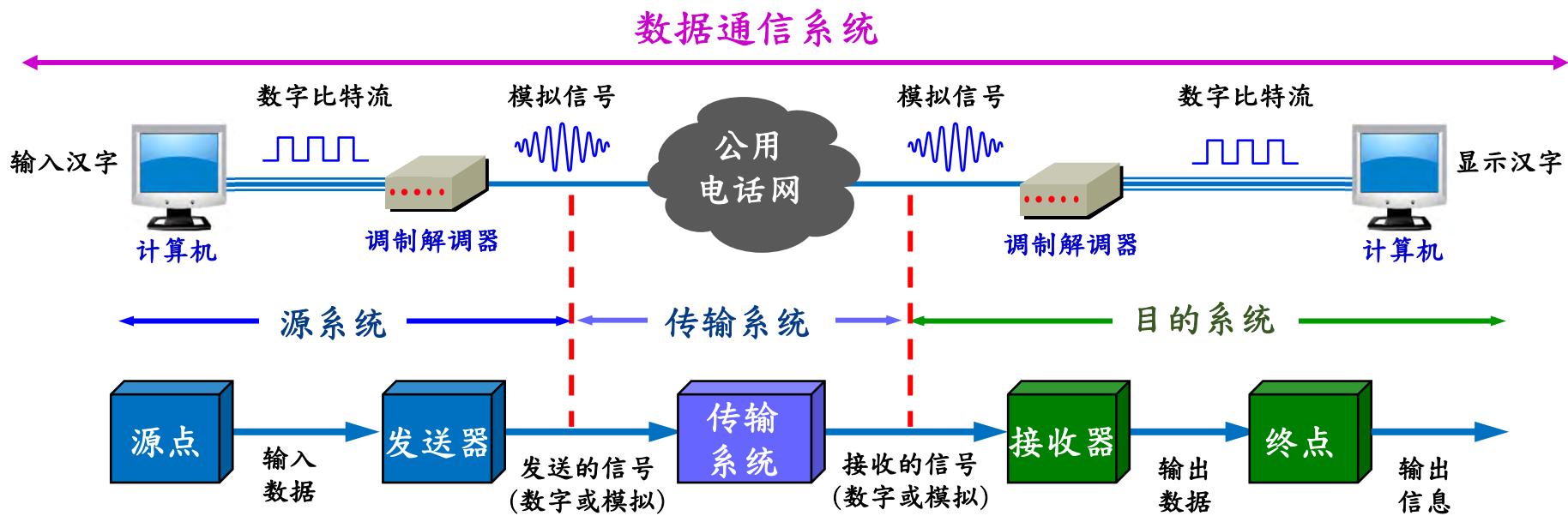
模拟/数字的数据/信号转换

- A/D和D/A转换



A/D和D/A通信系统模型

- 通信系统模型



内容纲要

1

通信的基本概念

2

传输介质的分类

3

传输介质介绍

4

介质的选用标准

5

小结

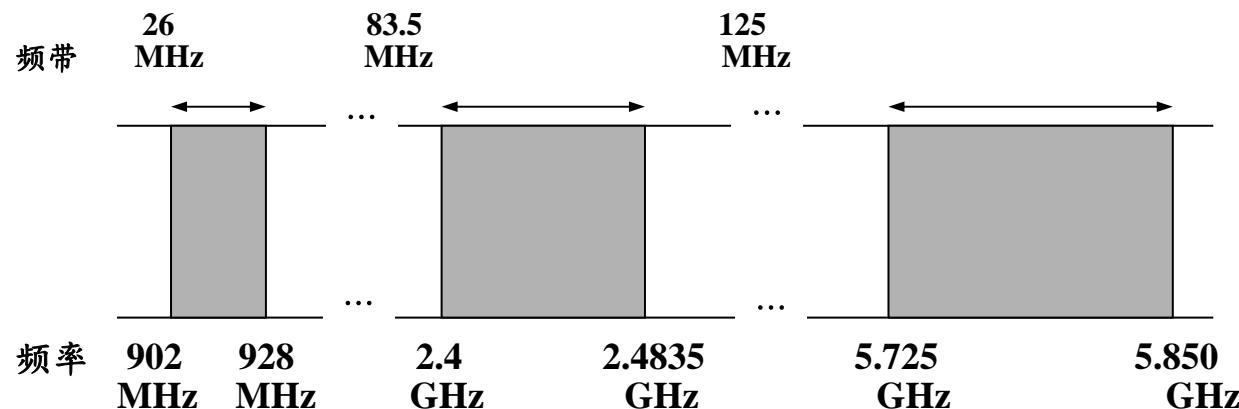


导引型传输媒体



非导引型传输媒体

- 无线传输所使用的频段很广。
 - 举例：红外线、激光、卫星
- 短波通信主要靠电离层的反射，但通信质量较差。
- 微波在空间主要是直线传播。
 - 地面微波接力通信
 - 卫星通信



按能量形式划分

- 电的
 - 双绞线，同轴电缆
- 光的
 - 光纤，红外线，激光
- 电磁波（无线电波）
 - 地面无线电，卫星



内容纲要

1

通信的基本概念

2

传输介质的分类

3

传输介质介绍

4

介质的选用标准

5

小结



双绞线（ Twisted Pair ）

- 平行导线的问题

- 随机电磁噪声（ noise ）是普遍存在的
- 电磁辐射碰到金属时会产生微弱信号干扰通信信号
- 金属可以吸收辐射，起到屏蔽（ shield ）作用

- 三种导线可以减小干扰

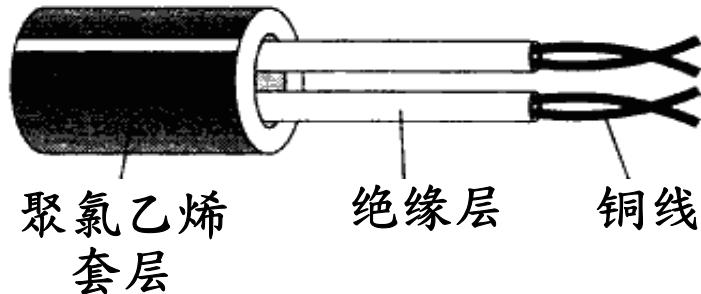
- 非屏蔽双绞线（ Unshielded Twisted Pair ）：柔韧性
- 同轴电缆（ Coaxial Cable ）：屏蔽能力
- 屏蔽双绞线（ Shielded Twisted Pair ）：折中



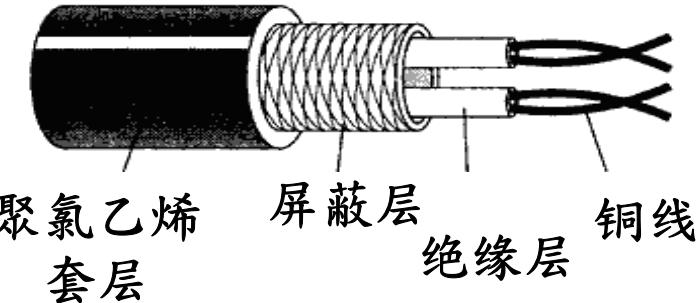
不同类型的电缆

- 三种不同类型的电缆示意图

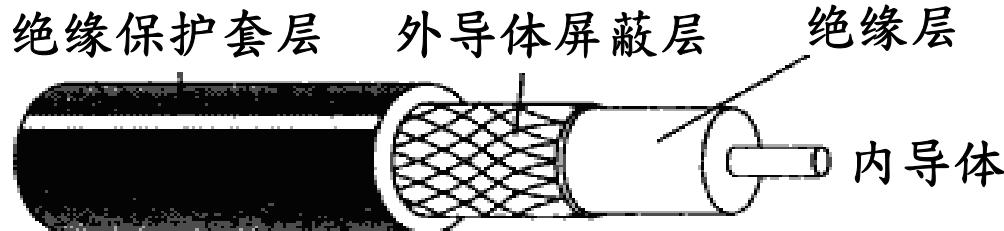
无屏蔽双绞线 UTP



屏蔽双绞线 STP



同轴电缆



双绞线

- 既能传输模拟信号又能传输数字信号；
- 通信距离一般为几到几十公里
 - 距离太长，信号会衰减，需要用中继器进行整形和放大。

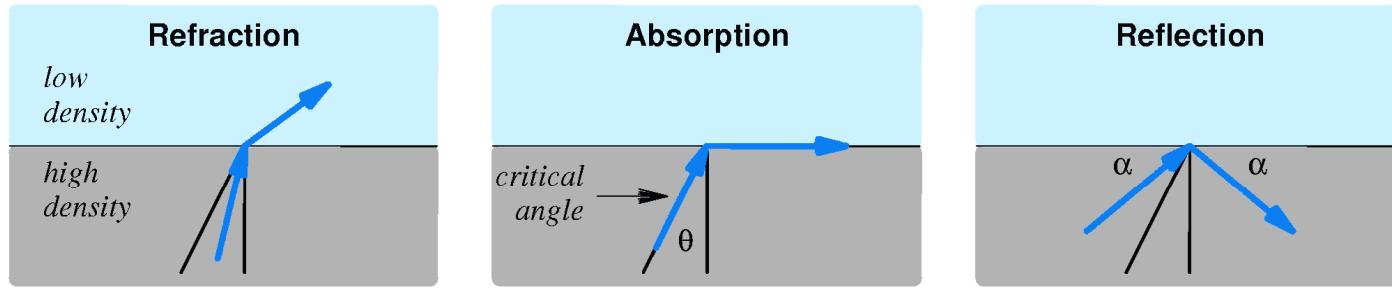
UTP类别	作用	带宽	系统
CAT-1	以往用在传统电话网络（ POTS ）、 ISDN 及门钟的线路。		电信 系统
CAT-2	以往常用在 4 Mbit/s 的令牌环网络。		
CAT-3	曾经常用在 10Mbps 以太网络。	16MHz	计算 机网 络
CAT-4	常用在 16 Mbit/s 的令牌环网络。	20MHz	
CAT-5	常用在快速以太网（ 100 Mbit/s ）中。	100MHz	
CAT-5e	常用在快速以太网及吉比特以太网（ 1000Mbit/s ）。	125MHz	
CAT-6	比 CAT-5 与 CAT-5e 高出一倍半。	250MHz	

同轴电缆

- 按阻抗分为两类：**50Ω同轴电缆和75Ω同轴电缆**。
 - **50Ω**：基带同轴电缆（baseband coaxial cable）
 - 以10Mb/s传输基带信号的距离可达1km.
 - 用于以太网的标准：10Base2, 10Base5
 - **75Ω**：宽带同轴电缆（broadband coaxial cable）
 - 频率可高达500MHz以上，传输距离可达100km.
 - 用于传输有线电视的模拟信号
 - 分为多个信道
 - 使用电缆调制技术, 电视和数据可在一条电缆上混合传输

光纤

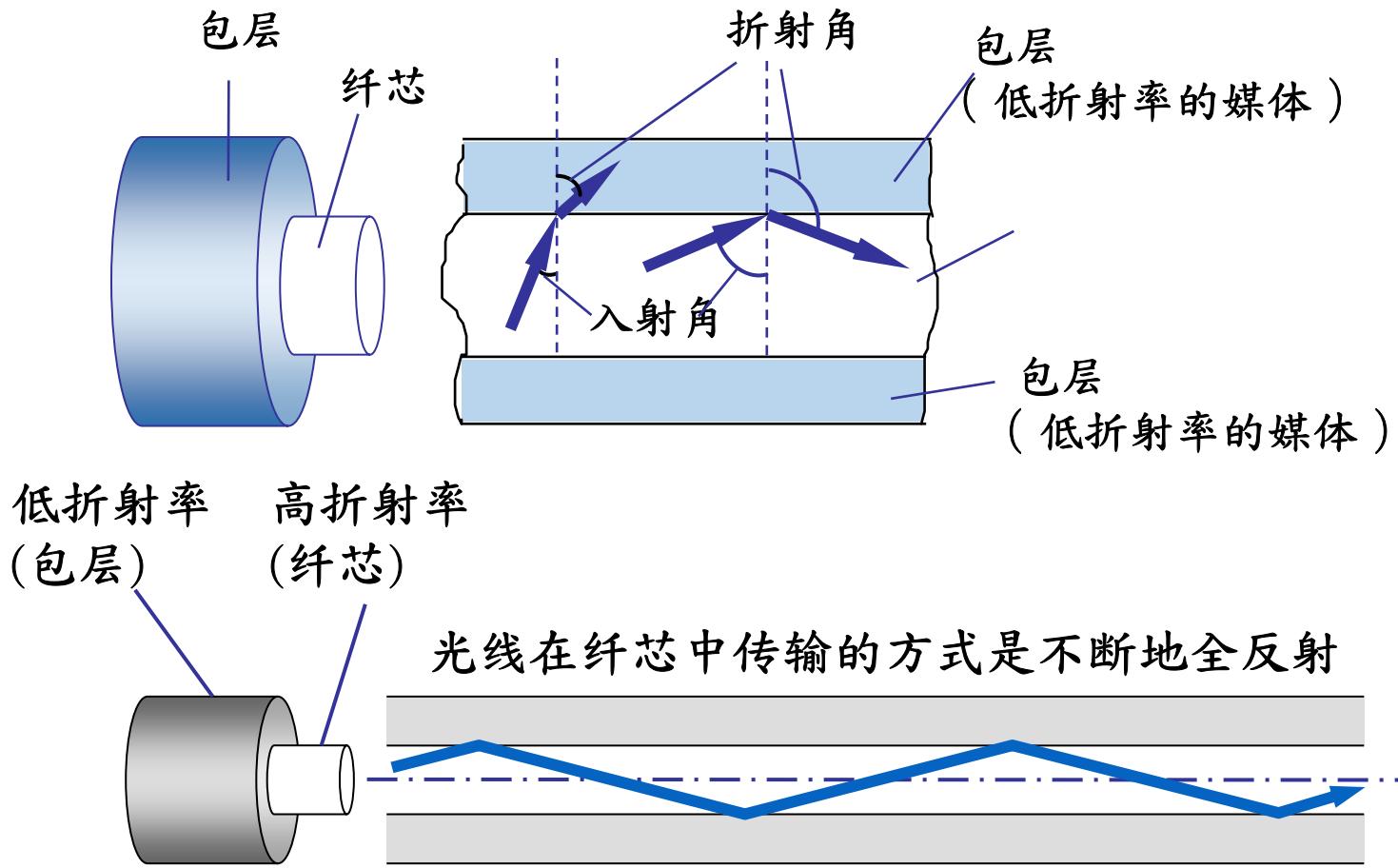
- 光在密度边界的折射、吸收和反射
 - 光的反射会吸收一小部分能量，出现色散（dispersed）



- 发送
 - 发射器：发光二极管（LED）或激光器将光纤的脉冲发送
 - 接收器：使用光敏晶体管来检测脉冲

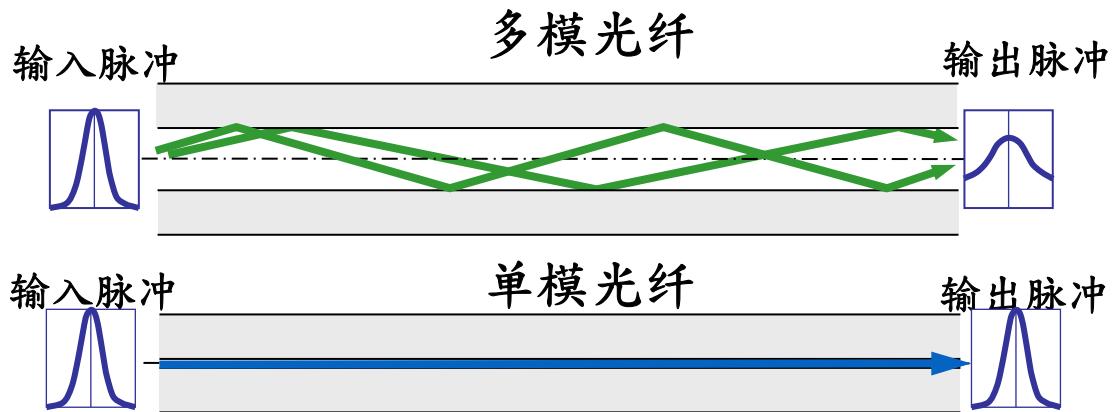
光纤的工作原理

- 光纤很细（ μm 级），因此需要包层



多模光纤与单模光纤

- 多模突变光纤：便宜，纤芯密度不变，覆层间突变
- 多模渐变光纤：纤维密度越接近边缘越大，减少反射
- 单模光纤：贵，直径小、长距离、高比特率



项目	单模光纤	多模光纤
距离	长	短
数据传输率	高	低
光源	激光	发光二极管
信号衰减	小	大
端接	较难	较易
造价	高	低

光缆

• 光缆

- 光纤非常细，直径不到0.2mm，容易损坏
- 一根光缆可包括有一根乃至数百根光纤
- 加上加强芯和填充物提供机械强度
- 必要时还可以放入远供电线
- 最后加上包带层和保护套提高抗拉强度

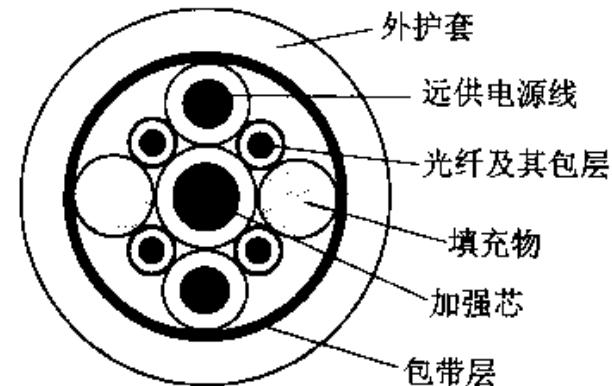


图 3-13 四芯光缆剖面的示意图

光纤与铜导线的比较

- 光纤

- 免受电气噪声干扰，信号损耗小
- 高带宽

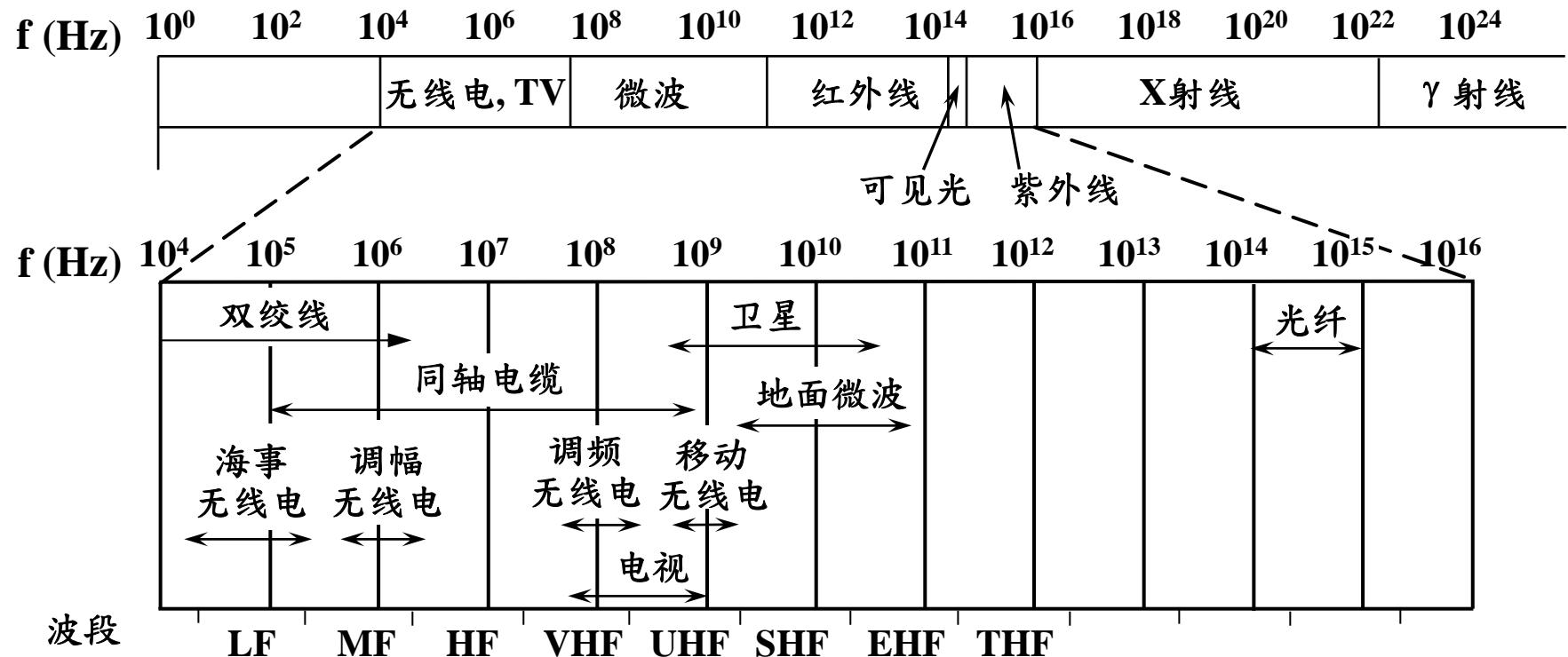
- 铜导线

- 整体费用低
- 不需要专门人员与设备
- 不易折断



物理层的传输媒体

- 电信领域使用的电磁波的频谱



红外（ InfraRed ）通信技术

- 构成：红外线发射、接收装置
 - 不需要天线，适合于室内环境
- 与可见光特性相似，但在可见范围外
 - 扩散快
 - 光滑坚硬的表面反射，不透明物体（包括水蒸气）阻挡
- 速率
 - 低速0.115Mbps；中速1.150Mbps；高速4.000Mbps



点对点激光（Laser）通信

- 数据传输率高，正确率高，信号衰减小，成本高
- 适用于城市楼宇间传输

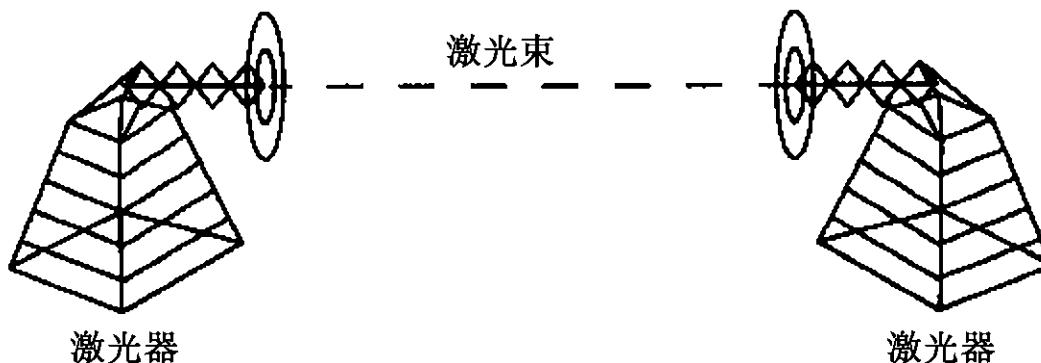
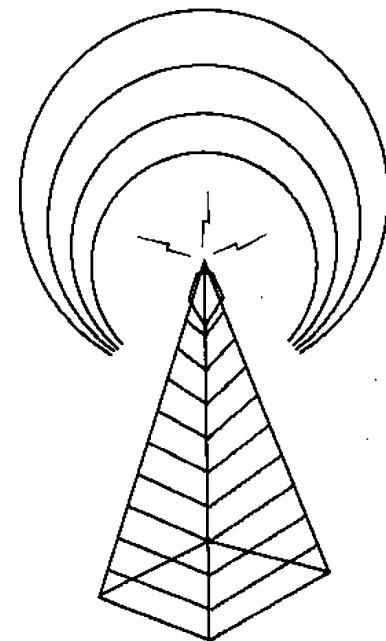


图4-4 光的聚集定向传输

无线电波 (Radio)

- 构成：无线电发射装置，接收装置
 - 计算机连接天线以发送接收射频 (radio frequency)
- 特点：
 - 广泛应用于广播电视系统
 - 传输部分不需要物理介质



无线电发送器

图4-3 信号的全向辐射

卫星 (Satellites)

- 构成：

- 无线电发射装置，接收装置，人造卫星转发装置

- 轨道类型

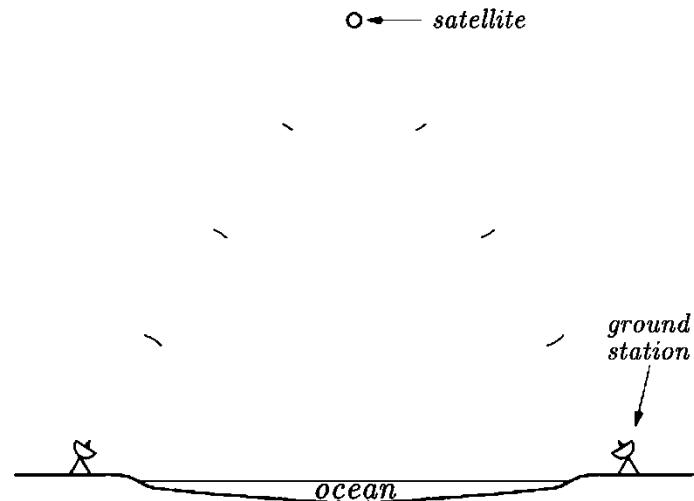
- 低地球轨道：低时延，不断移动

- 时延1~4ms，通常设计为集群

- 中地球轨道：椭圆形，南北极通信

- 地球静止轨道：固定方位，距离远

- 轨道在地月距离十分之一，时延0.2s



微波（Microwave）

- 虽然微波只是无线电波的更高频率版本，但它们的行为方式不同
- 针对在一个单一的方向
- 微波传输可以携带更多的信息。
- 微波不能穿透金属结构。



内容纲要

1

通信的基本概念

2

传输介质的分类

3

传输介质介绍

4

介质的选用标准

5

小结



介质之间的权衡

- 成本：材料、安装、运营、维护
- 数据速率：bps
- 时延：信号传输的时间
- 对信号的影响：衰减和失真
- 环境：对干扰和电气噪声的敏感性
- 安全：对窃听的敏感性

内容纲要

1

通信的基本概念

2

传输介质的分类

3

传输介质介绍

4

介质的选用标准

5

小结



小结

- 数据必须编码成能通过传输介质传输的格式。
 - 这些格式必须随着传输介质而变化，因为每种介质都有其自身的物理特性。
 - 数据编码的技术有许多种，但它们都使用电磁波来进行编码和数据传输。
- 电磁波是能量的物理形式，可通过电磁波谱来描述。
- 随着频率的增加，对数据编码的能力也增加。
- 高频比低频有更多的状态改变，状态改变可用于编码。



计算机网络

Computer Network

1

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

计算机网络

Computer Network

2

局域异步通信

理论课程

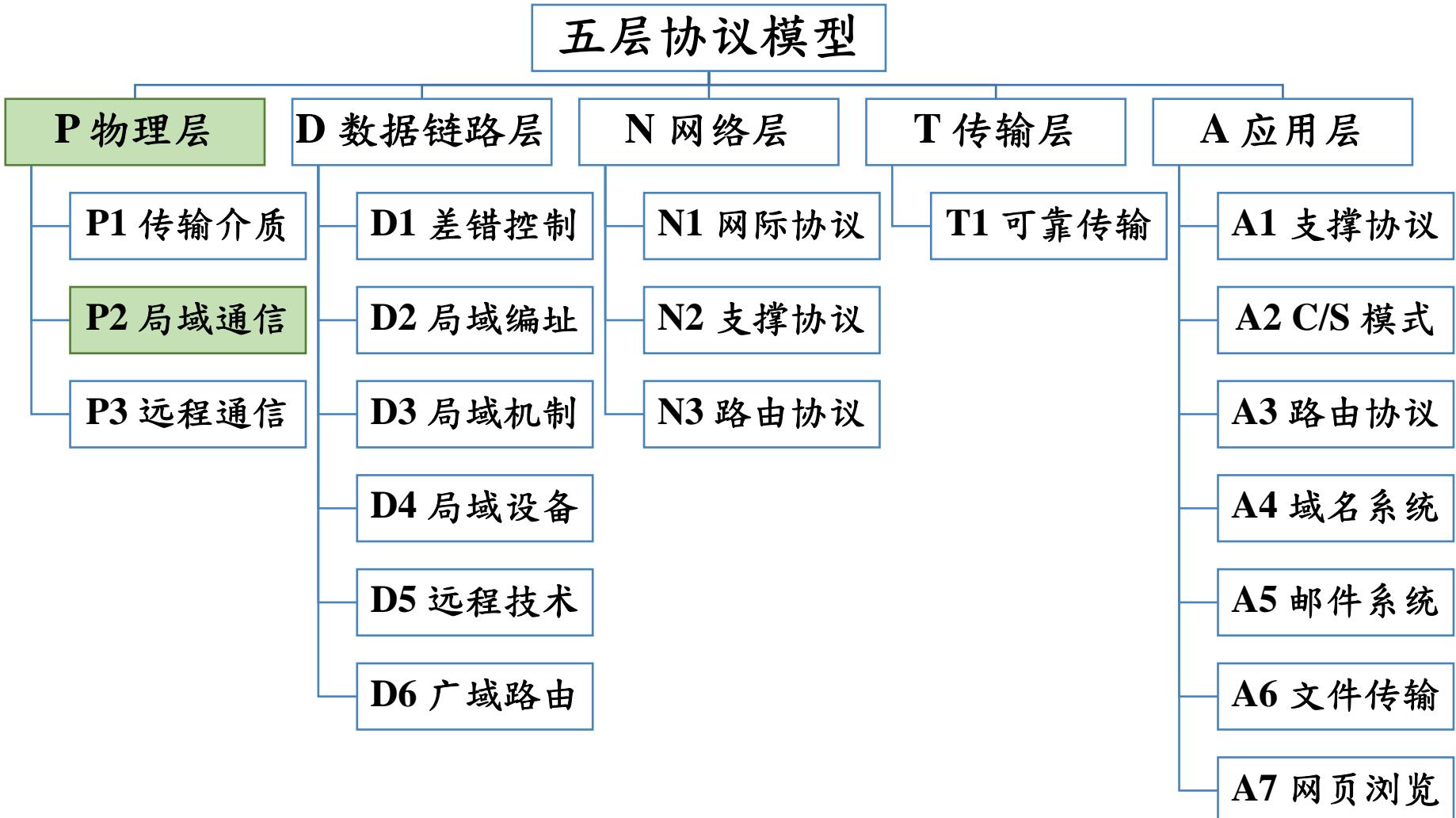


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 传输模式的类别
 - 串行，平行；同步，异步，等时；单工、半双工、全双工
- 多比特下的端序：大端序，小端序
- 异步通信标准：RS-232
 - 电气特性，帧、帧格式
 - 参数：波特率，波特，标准化
- 两个重要通信理论：奈奎斯特定理和香农定理



对应课本章节

- PART II Data Communication Basics
 - Chapter 6 Information Sources And Signals
 - Chapter 9 Transmission Modes



内容纲要

1

传输模式

2

异步通信标准（RS-232）

3

带宽

4

奈氏定理和香农定理



传输模式 (transmission mode)

- 传输模式指的是数据通过底层介质传送的方式
- 串行 (Serial)

名称	传输发生时间	数据项之间的间隔
异步 Asynchronous	在任意时间 发生	两个数据项之间可以有任意的时延
同步 Synchronous	连续 发生	需事先约定有效数据的位置，两个数据项之间没有间隔 (gap)
等时 Isochronous	在常规间隔 发生	两个数据项之间有固定的 (fixed) 间隔 (gap , intervals)



同步通信和异步通信

- 同步通信

- 发送方在数据前附加特定的同步字符，以便建立同步
- 双方在内部同步时钟的控制下逐位收发，传输单位：位
- 数据基本单元保持顺序传输，空数据需填充对应的空单元

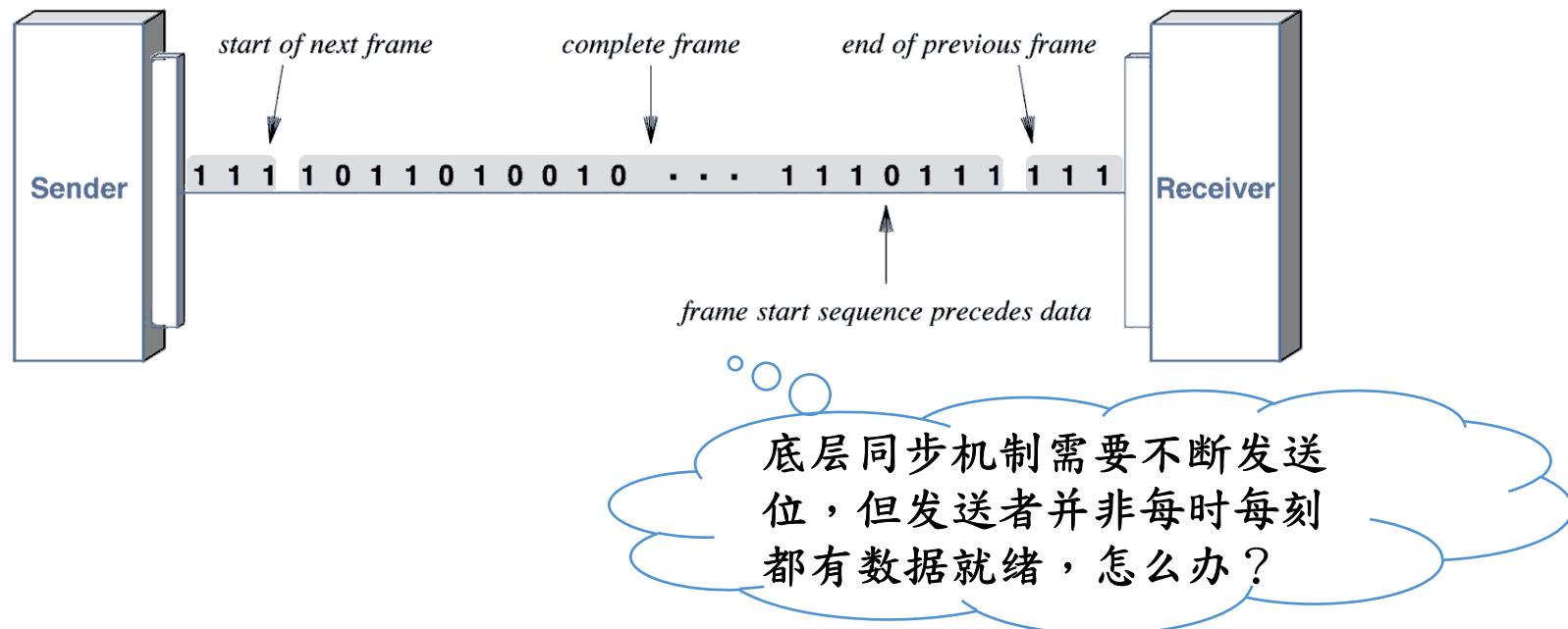
- 异步通信

- 发送方在字符开始和结束处加上标志
- 接收方随时做好接收准备（需要在线）
- 字符间隔任意，不需事先同步，但字符内各位间隔固定



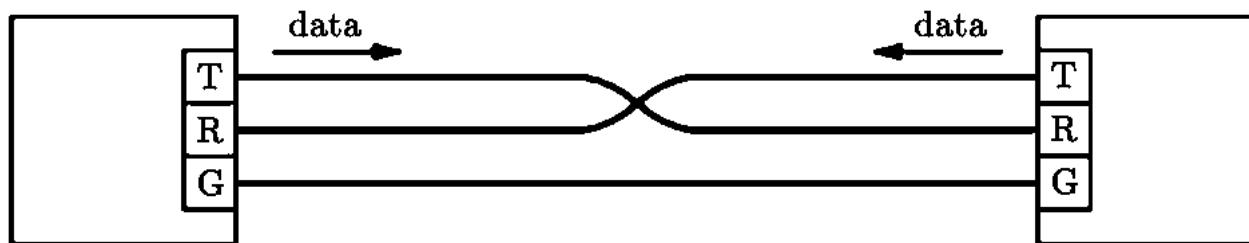
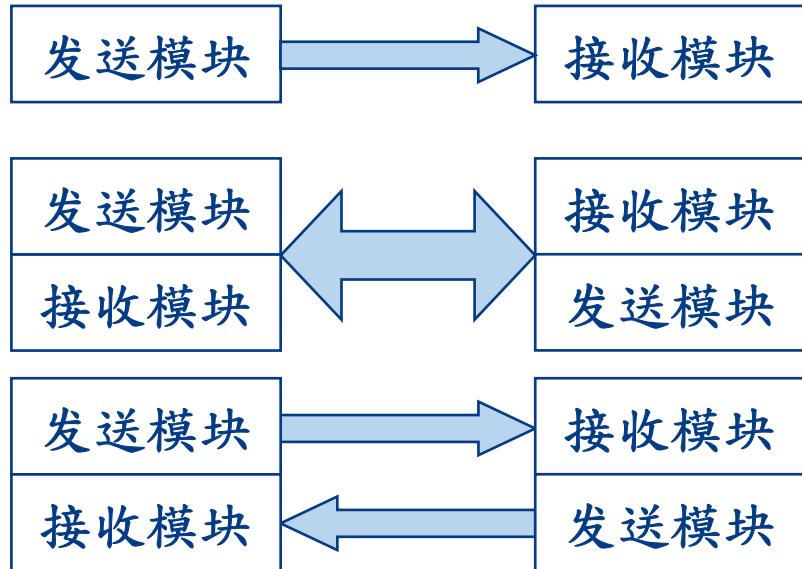
分块（Blocks）和帧（Frames）

- 分块（Blocks）：将数据划分为组
- 帧（Frame）：有格式的分组
 - 发送方无数据时发送空闲序列，有用序列以开始位为起点



异步通信的模式

- 单工：单方向通信
- 半双工：双向，同一时刻单向
- 全双工：同一时刻可双向
- 全双工 RS-232 通讯最小布线



串行通信的端序

- ## • 端 (End)

— 内存存储以低地址为端。

— 网络通信以零时刻为端。

- ## • 大端与小端

1代表1000，为
较大数（高位）

4代表4，
变小数(低位)。

– 大端（ Big-endian ）：大数在端。

— 小端（ Little-endian ）：小数在端。

- 字节与位的端序可能不一样。



廈門大學
XIAMEN UNIVERSITY



信息学院 黄煌
国家示范性软件学院 博士,副教授
School of Informatics Dr. Wei Huang

2020-01-18

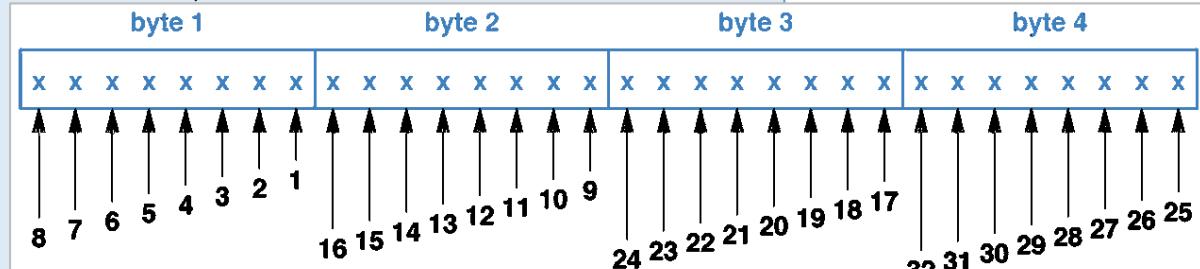
10

字节和位序 (Byte and Bit Order)

- 字节大端、位小端序

```
#include <stdio.h>
```

```
typedef unsigned char BYTE;
int main() {
    BYTE pdw[] = { 0x12, 0x34, 0x56, 0x78 };
    for (size_t i = 0; i < sizeof(pdw); ++i) {
        BYTE b = pdw[i];
        for (size_t j = 0; j < 8; ++j) {
            printf("%X", b & 1);
            b = b >> 1;
        }
        printf(" ");
    }
    return 0;
}
```



输出 : 01001000 00101100 01101010 00011110



内容纲要

1

传输模式

2

异步通信标准 (RS-232)

3

带宽

4

奈氏定理和香农定理



通信标准

- 通信系统的规范标准化
 - 目的：确保不同的供应商构建的通信硬件将实现互操作
 - 标准规范了
 - 发送器每个位在电线保持电压的时间
 - 硬件改变电压的最大速率（ maximum rate ）
- RS-232标准：电子工业协会（ EIA ）
 - EIA标准RS-232-C，通常缩写为RS-232。
 - 一个用于计算机、调制解调器或ASCII终端（ serial ）异步（ asynchronous ）通信标准。



RS-232 电气特性

- RS-232 电气特性

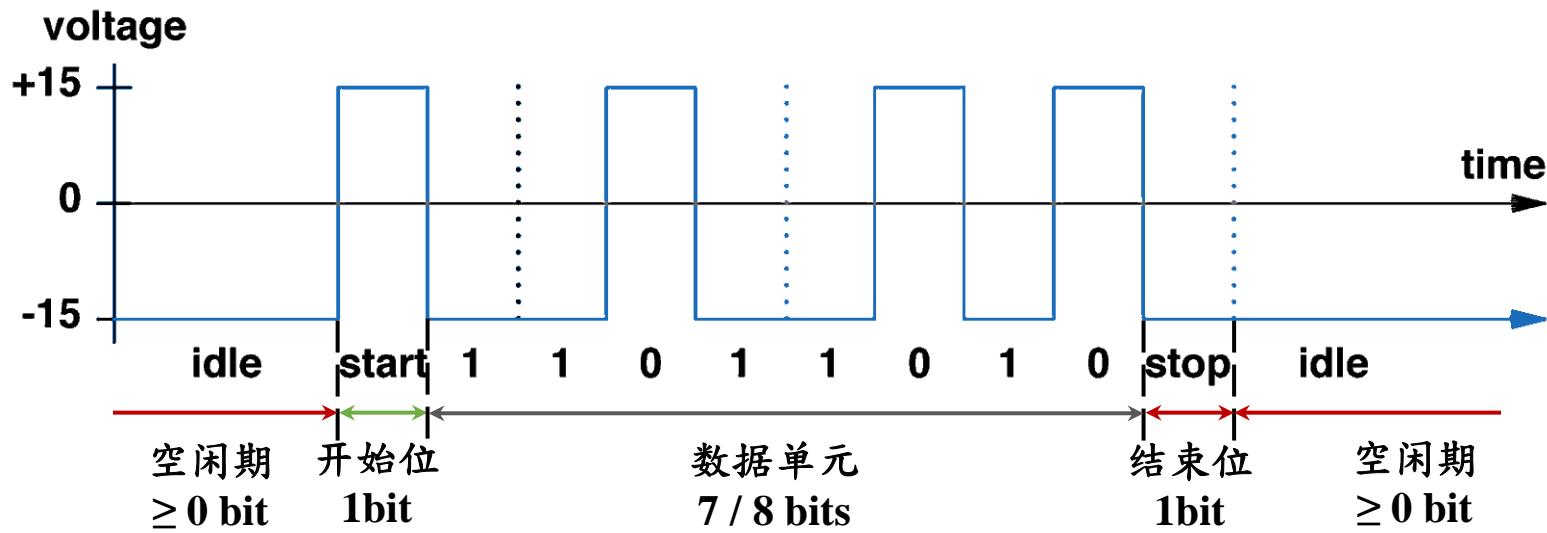
- 连线长度：小于 50ft
- 电压范围：-15V ~ +15V
- 线路编码：负电压表示 1，正电压表示 0
 - 来自电传机（ Teleprinter ）
 - 例如：0: [3v, 15v]; 1: [-15v, -3v]



RS-232的帧格式

- RS-232的帧格式

- 发送每个位使用的时间相同，接收方严格按照时间片顺序完成数据接收，直到结束位为止
- 仅有0和1（无“空”状态）

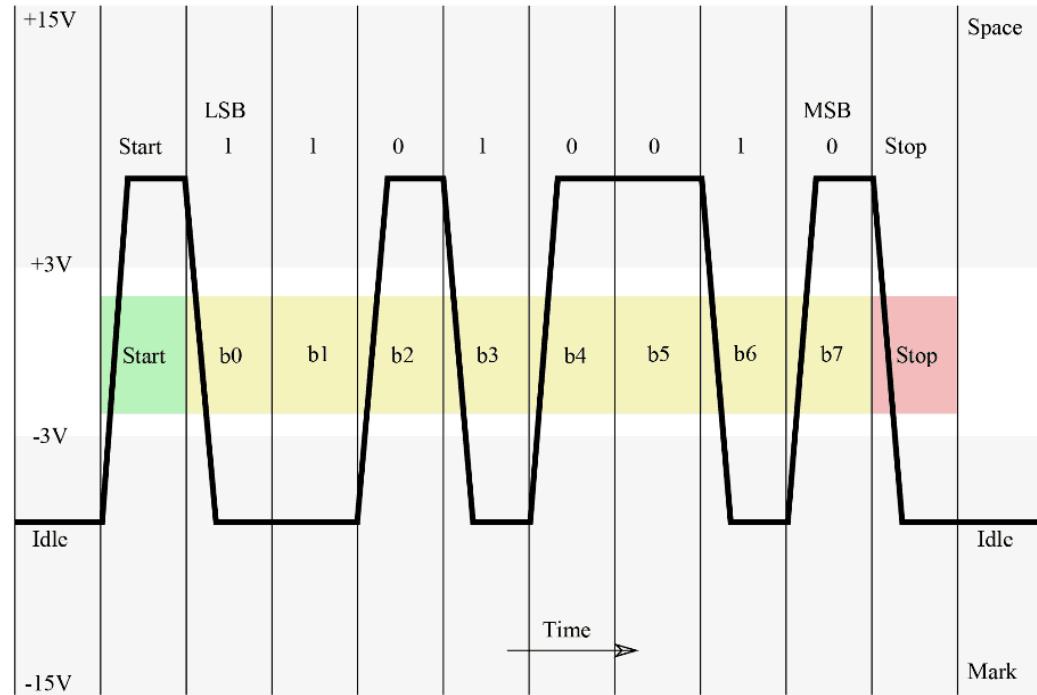


RS-232的帧格式

- RS-232帧格式

- 数据按字节大端序位小端序发送和接收（图：K，0x4B）
- 7位作为一个发送单元

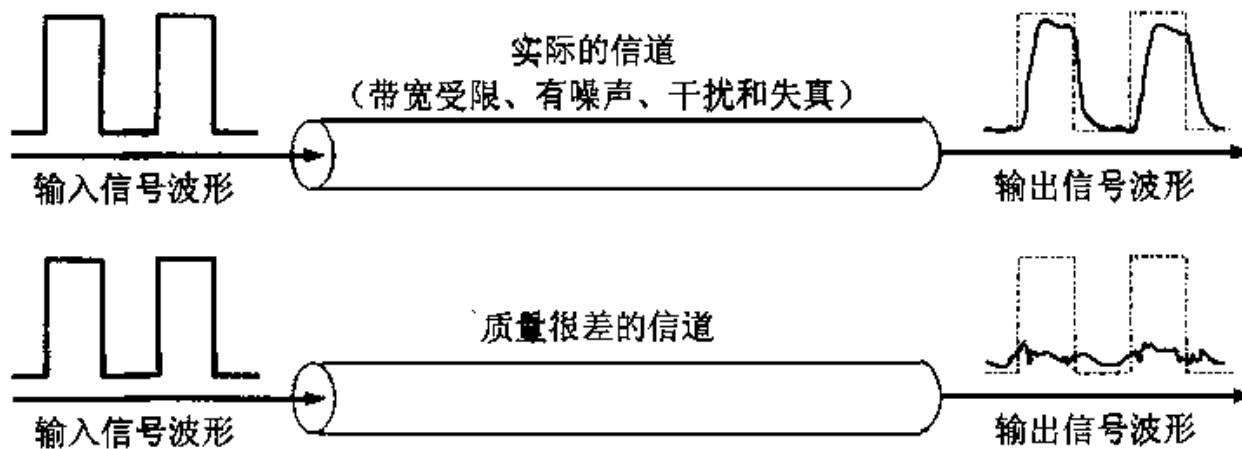
- 有的版本为8位
- 电传机都是可见字符，最高位为**0**，能省则省



真实硬件的局限

- RS-232实际信道

- 该标准并未规定接收器应在位的开始立即测量电压
- 该标准建议分配给位时间的中间过程中抽取样本。



内容纲要

1

传输模式

2

异步通信标准（RS-232）

3

带宽

4

奈氏定理和香农定理



带宽 (Bandwidth)

- 信号带宽的定义

- 传统定义：某个信号具有的频带宽度
 - 新定义：该信号的各种不同频率成分所占据的频率范围
 - 特定信号通常由许多不同的频率成分组成。

- 数字信道的带宽

- 在信道上能够传送的数字信号的速率，即数据率或比特率。
 - 单位：b/s（或bps，bit per second）。
 - 根据香农理论，因为带宽有时也称为吞吐量。



波特率

- 带宽的单位：波特（baud，也称波特率）

- 传输介质中信号变化的速率，每秒传输的符号数。

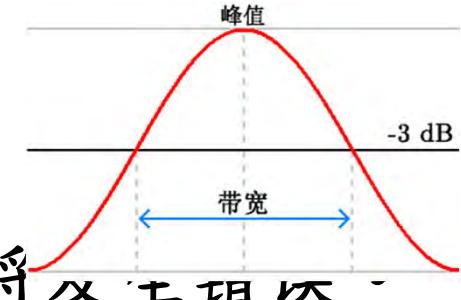
- 真正的硬件不能瞬间改变电压，带宽有限。

- 仅在二进制环境下为bps。

- 发送和接收的硬件配置的波特率不同，将~~发生错误~~

- 原因：接收方计时器的每个位等待时间长度不适当。

- 原理：检测错误时，接收机多次测量每个比特的电压，并比较测量结果。如果电压都不一致，或停止位没有在预期的时间发生时，接收器会报告错误，称为帧错误。



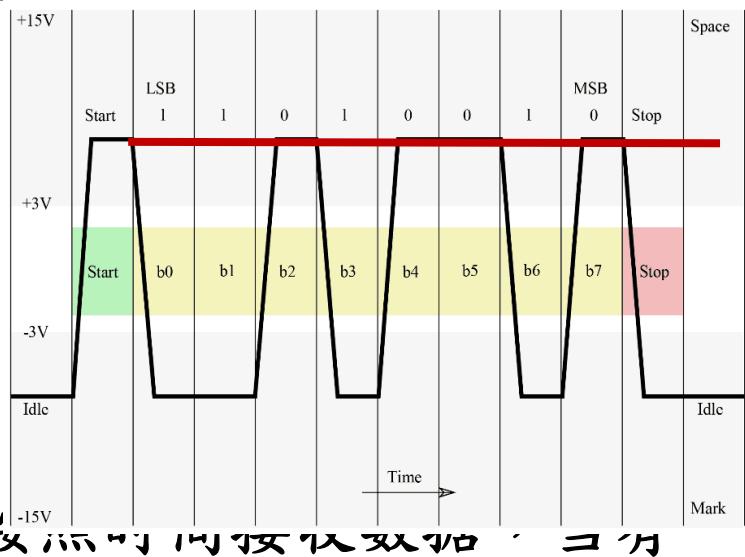
传输中断

- 发送方

- 发送完开始位后，持续发送0，直到结束位的位置仍是0。

- 接收方

- 接收到有效数据开始位后，严格按时间片到，仍然是“0”，接收方认为数据错误，产生中断信号



RJ-45接口

- 10Mbps以太网的RJ-45接口

- 数据通信设备（DCE）、数据终端设备（DTE）

DCE		DTE
接收正 (R+)	Lead #1	发送正 (T+)
接收负 (R-)	Lead #2	发送负 (T-)
发送正 (T+)	Lead #3	接收正 (R+)
未用	Lead #4	未用
未用	Lead #5	未用
发送负 (T-)	Lead #6	接收负 (R-)
未用	Lead #7	未用
未用	Lead #8	未用

– 在UTP】

图3-10 DCE与DTE的连接

全部4对线



内容纲要

1

传输模式

2

异步通信标准（RS-232）

3

带宽

4

奈氏定理和香农定理



奈奎斯特定理 (Nyquist's theorem)

- 奈氏定理：硬件带宽与理论最大数据发送速率间关系
 - 如果传输系统使用 K 个可能的电压值，带宽为 B (半个周期)，则最大数据速率 $D = 2B \log_2 K$ (单位 : bps)
 - 例如：RS-232 使用 2 个电压值，最大数据速率为 $2B$
 - 信息论，特别通讯与信号处理学科中的一个重要基本结论。
 - 如果信号是带限的，并且采样频率大于信号带宽的 2 倍，那么，原来的连续信号可以从采样样本中完全重建出来。
 - 比如声音信号，由人类发出的声音信号中，频率超过 5 kHz 的成分通常非常小，因此以 10 kHz 的频率来采样这样的音频信号就足够了。



香农定理（ Shannon's theorem ）

- 香农定理扩展了奈奎斯特定理
 - 引入噪声的传输系统可以达到的最大数据速率为

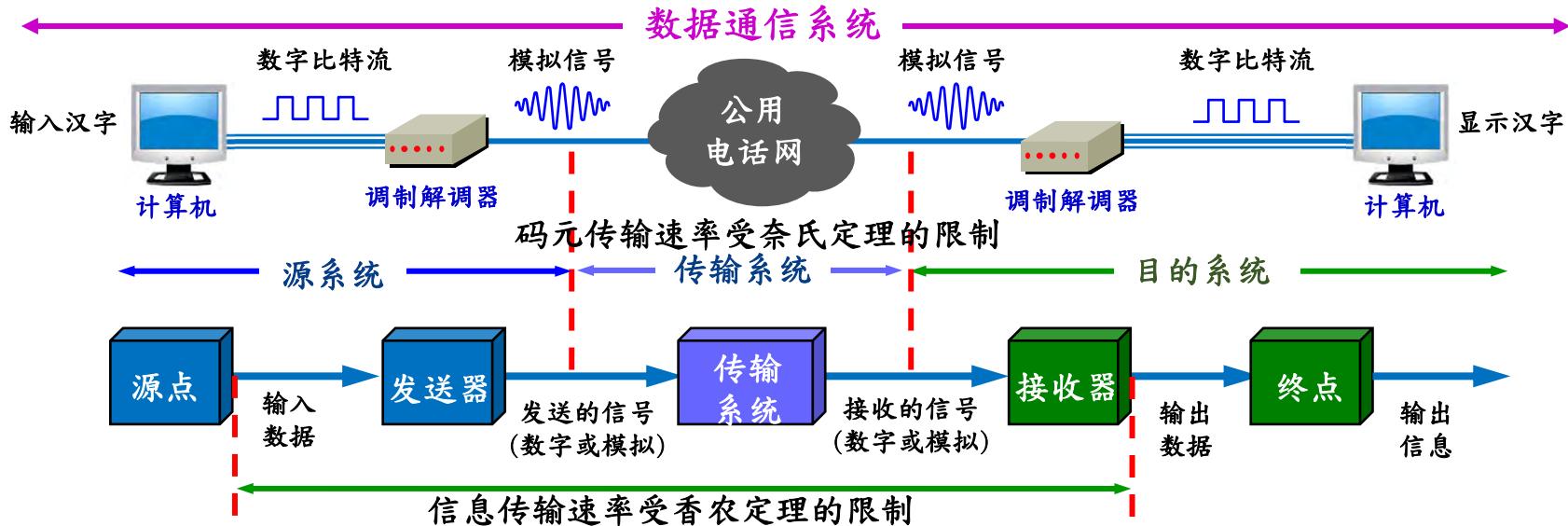
$$C = B \log_2 \left(1 + \frac{S}{N} \right)$$

— 其中， C 为信道容量的有效上限（单位为：bps，位每秒）， B 为硬件带宽， P 是平均信号功率， N 是平均噪声功率。

- $\frac{S}{N}$ 是信噪比 (signal-to-noise ratio) , 其中 $10 \lg \frac{S}{N}$ 的单位为分贝 (dB) 。

两大定理的比较

- 奈奎斯特定理鼓励工程师们探讨如何对信号进行位编码，因为一个聪明的编码允许单位时间传递更多位。
- 香农定理则告诉工程师再多的巧妙编码也难以克服限制实际通信系统中每秒可传输位数目的物理定律。



计算机网络

Computer Network

2

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

3

远距离通信

理论课程

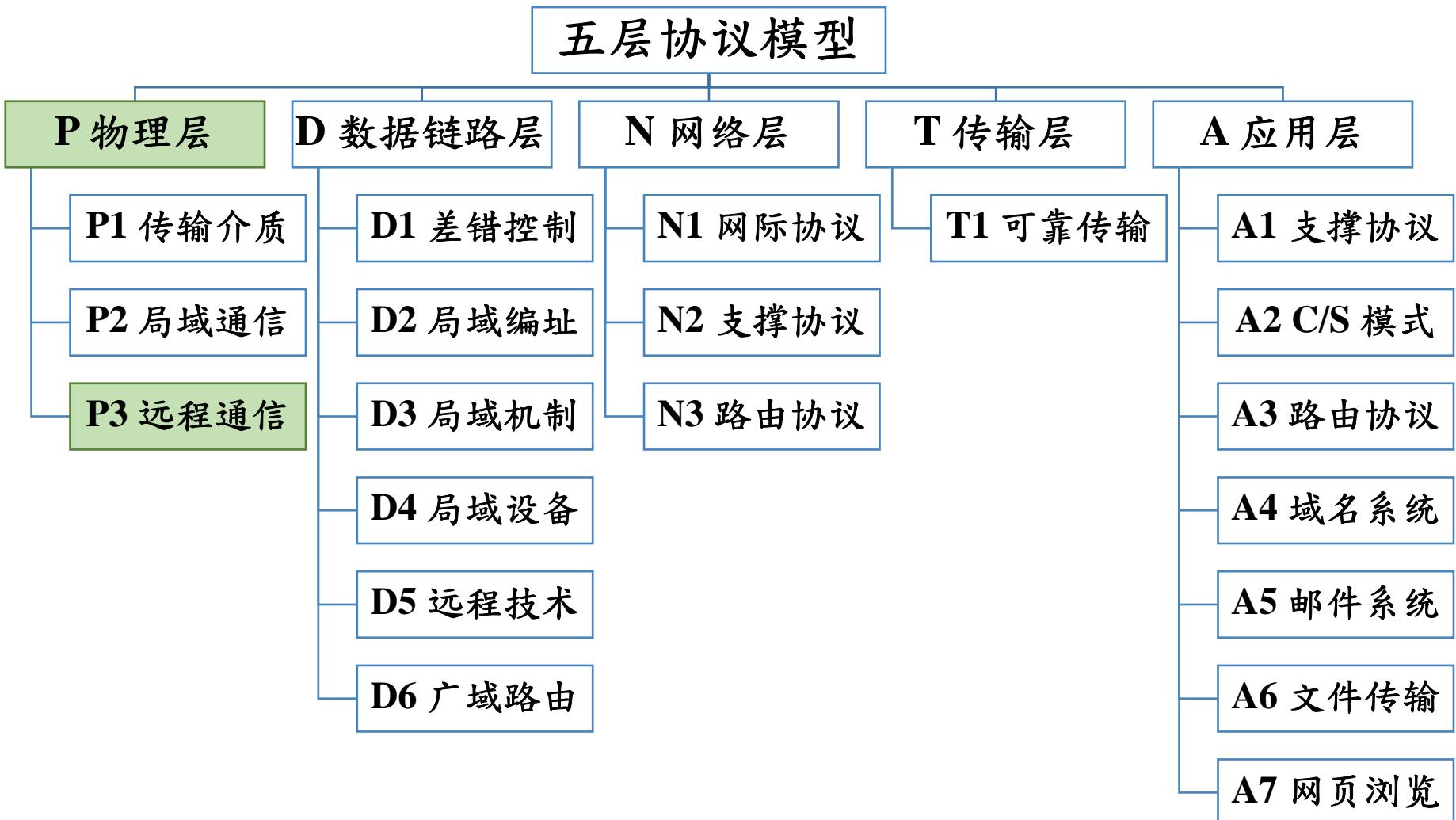


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

五层协议模型



主要内容

- 载波
- 调制和解调
 - 调频、调幅、调相
- 复用和解复用
 - 频分、波分、时分（同步时分、统计时分）、码分
- 基带和宽带



对应课本章节

- PART II Data Communication Basics
 - Chapter 10 Modulation And Modems
 - Chapter 11 Multiplexing And Demultiplexing (Channelization)



内容纲要

1 载波

2 调制和解调制

3 复用和解复用



电流承载信息的局限性

- 电流在铜线上的传播的距离是有限的。
 - 导线电阻将电能转换为热能，叫信号损耗（ signal loss ）
- 结果：误码（ Error Code ）

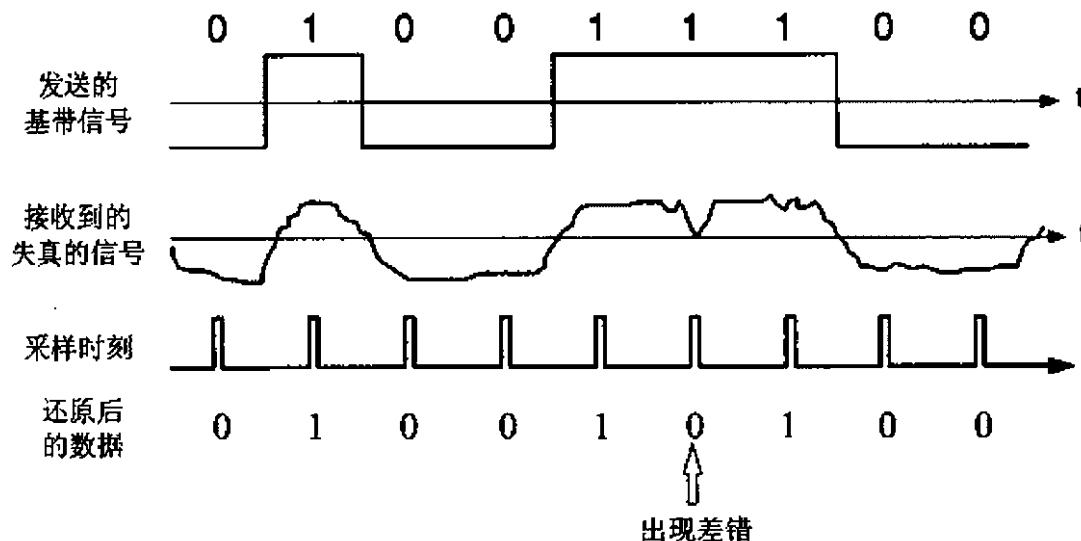


图 3-15 基带信号经电话线路传输后产生误码



载波（ Carrier ）

- 远距离通信系统发送连续震荡的信号称为载波
 - 通常是正弦波（ sine wave ）
 - 连续（ continuous ）振荡（ oscillating ）信号传播得更远
- 载波是特定频率的无线电波，是一种可在频率、幅度或相位方面被调制以传输其它信号的电磁波。
- 即便没有信号被发送，载波仍持续震荡。



内容纲要

1 载波

2 调制和解调制

3 复用和解复用



载波调制

- 调制（ modulation ）

- 使载波的某些特性按信息的波形或信号而变化的处理方法。
 - 特性：频率、幅度、相位.....
 - 表示信息的信号称为调制信号
 - 调制后的信号称为频带信号、射频信号或带通信号

- 解调制（ demodulation ）

- 根据频带信号恢复出调制信号
 - 载波信号将被丢弃



调制技术

- 调制技术

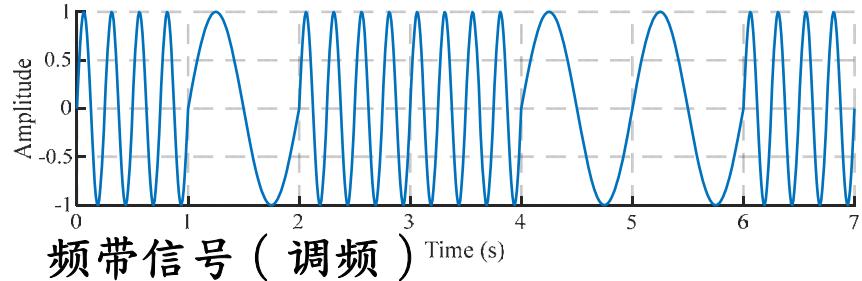
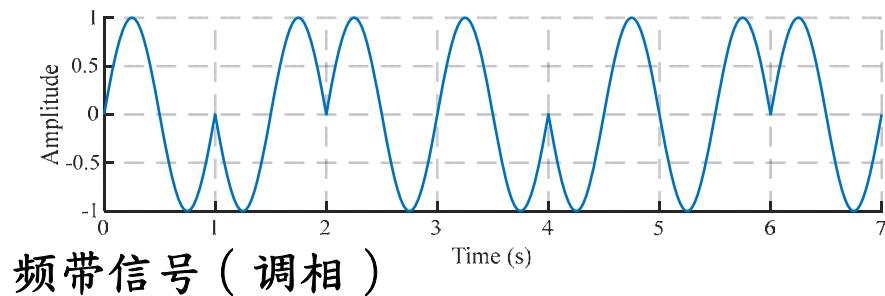
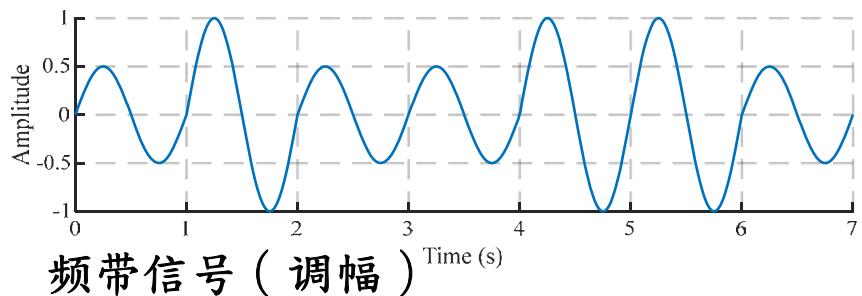
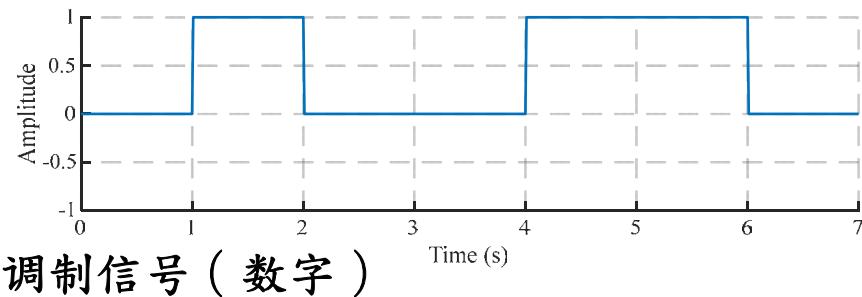
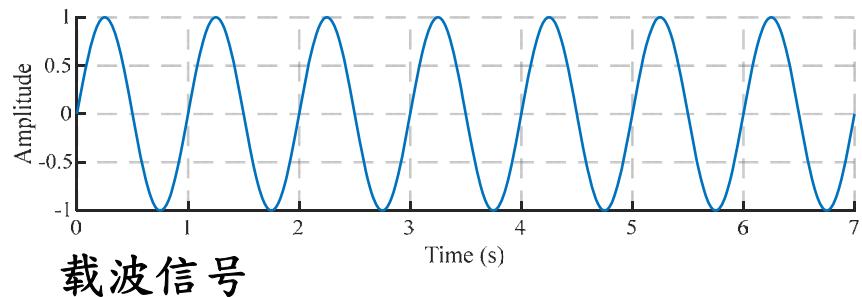
- 调幅 (amplitude modulation , AM)
- 调频 (frequency modulation , FM)
- 调相 (phase shift modulation , PM)

$$F(t) = \boxed{A} \sin(2\pi \boxed{f}t + \boxed{\alpha})$$

振幅 频率 相位

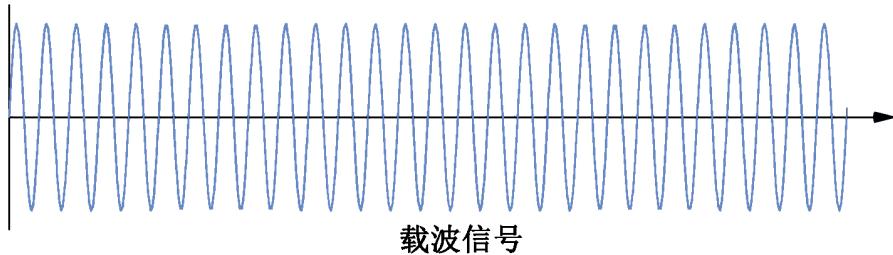
调制：调制信号为数字信号

- 载波信号为模拟信号，调制信号为数字信号

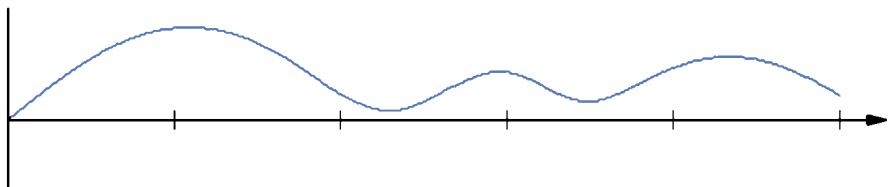


调制：调制信号为模拟信号

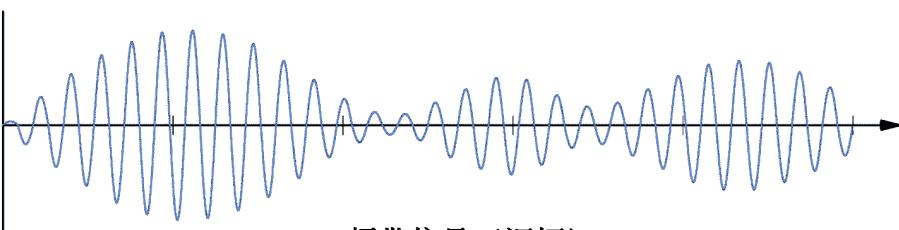
- 载波信号为模拟信号，调制信号为模拟信号



载波信号



调制信号（模拟信号）



频带信号（调幅）

调制器和解调器硬件

- 调制解调器（ Modem ）

- 调制器（ modulator ）是执行调制功能的器件。
- 解调器（ demodulator ）是执行解调功能的器件。
- 制造商经常将调制器与解调器结合成一个单独的双工设备。
- 可以是内置的嵌入式硬件，也可以是外置的独立硬件

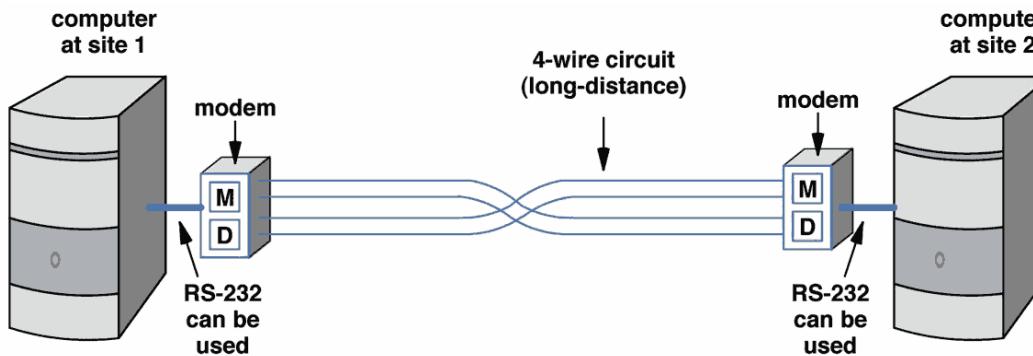


Figure 10.9 Illustration of two modems that use a 4-wire connection.

调制器和解调器硬件

- 调制解调器可以是内置的嵌入式硬件，也可以是外置的独立硬件

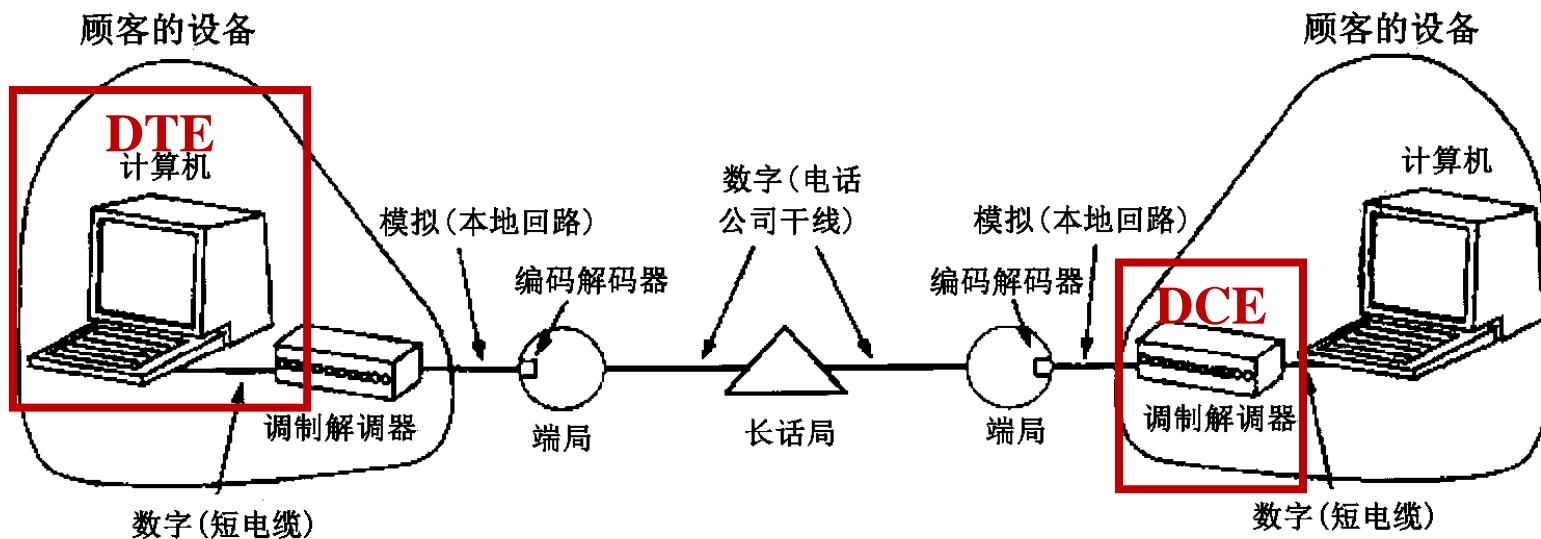


图 2-17 计算机与计算机之间的呼叫必须使用模拟及数字两种传输；
其转换工作由调制解调器 (modem) 及编码解码器 (codec) 完成。

租赁模拟数据电路

- 租赁模拟数据电路
 - 私人企业无法远距离安装电路，允许租用电话公司电路
 - 企业可以安装自身必要的电线
 - 在端点必须安装调制解调器
- 优点：可以在任何时候发送数据
- 缺点：有限的连通性和成本



光、射频、拨号调制解调器

• 通用性

- 除了专用线，调制解调器也可用于与其他的媒体，包括射频传输、光纤和常规的电话连接。

— 原理一致

- 发送端，调制解调器调制载体
 - 接收端，数据从被调制过的载体提取得到。

- 包含电路的拨号调制解调器，模拟电话

- 调制解调器可以模拟拿起听筒，拨号，或者挂了电话。

内容纲要

1 载波

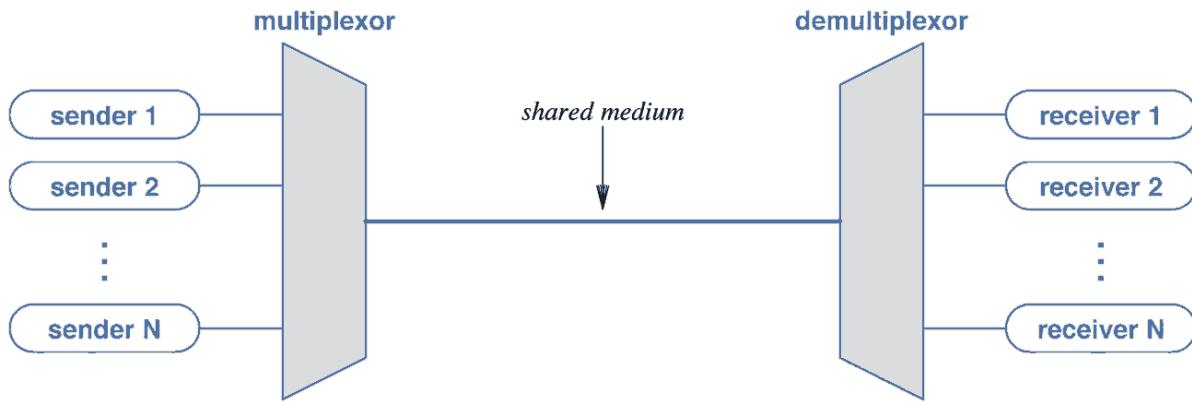
2 调制和解调制

3 复用和解复用



复用的概念

- 复用 (Multiplexing)
 - 指多个信源的信息流组合在一条共享介质上传输
- 解复用 (Demultiplexing)
 - 指将信息流组合分隔回分开的信息流
- 复用器 (Multiplexor) 和解用器 (Demultiplexor)



复用的基本形式

- 复用的基本形式
 - 频分多路复用 (Frequency Division Multiplexing , FDM)
 - 波分多路复用 (Wavelength Division Multiplexing , WDM)
 - 时分多路复用 (Time Division Multiplexing , TDM)
 - 码分多路复用 (Code Division Multiplexing , CDM)
- 关系
 - TDM和FDM广泛使用
 - WDM 是FDM 在光纤的特殊形式

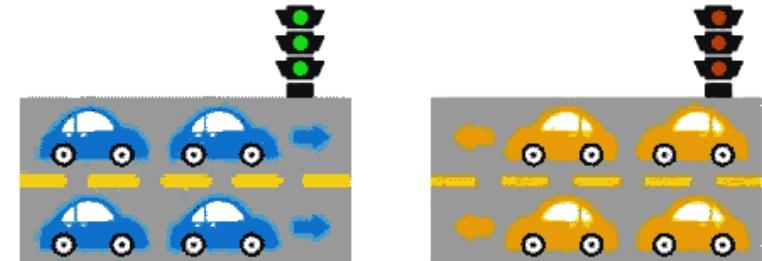


三大运营商的相关技术

- 中国移动
 - GSM，TD-SCDMA，TD-LTE
- 中国联通
 - GSM，WCDMA，FDD-LTE
 - TD-LTE
- 中国电信
 - CDMA2000
 - TD-LTE，FDD-LTE



TD上下行数据分开传输
像单车道运行，通过“信号灯”控制通道为上传或下载



FDD制式4G网络能实现实时的上传与下载，而TD制式4G网络需要通过信号灯切换改变上传与下载通道，对于用户而言，FDD的网络体验相对来说会更胜一筹。

频分多路复用 (FDM)

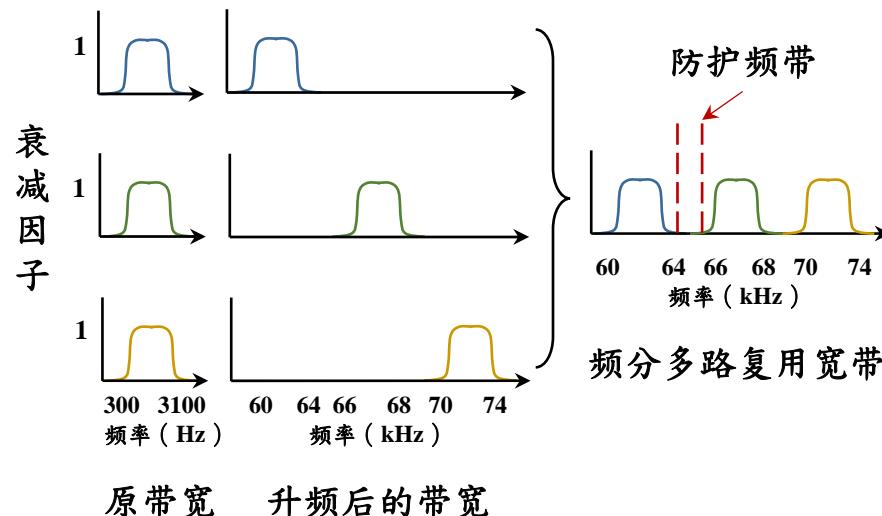
- 频分多路复用 (Frequency-Division Multiplexing)

- 载波带宽被划分为多种不同频带的子信道，每个子信道可以并行传送一路信号的一种多路复用技术。

- 多个载波可以在同一时间通过同一导线不相互干扰。

- 高吞吐量 (throughput)

- 一条电缆中同时传送多路的数字信号，提高了线路利用率。



基带和宽带

- 基带（Baseband）

- 原始电信号所固有的频带
 - 基带信号：将数字信号1或0直接用两种不同电压来表示。

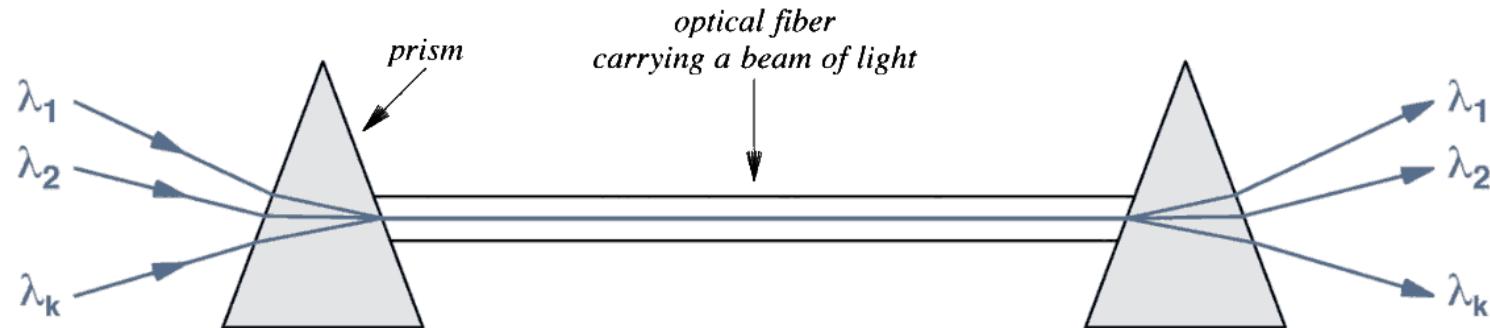
- 宽带（Broadband）

- 同时传输多个信号、传输介质带宽大的通信系统。
 - 宽带信号：将基带信号调制后形成的频分复用模拟信号。
 - 基带信号调制后，其频谱被移到较高的频率处，每一路基带信号的频谱被移到不同频段，合在一起不会互相干扰。



波分多路复用（WDM）

- 波分多路复用（ Wave Division Multiplexing ）。
 - 光波频率很高，习惯用波长表示光波。光的FDM即WDM。
 - 在一根光纤上发送多个光波。
 - 在接收端，光学棱镜用来分离频率。



扩频 (Spread Spectrum)

- 定义

- 将信号的频谱打散到较其原始带宽更宽的一种通信技术

- 收发机制

- 发射器在一组载波频率上发送相同的信号。
 - 接收器检查所有载波频率并使用其中有效的频谱。

- 优点

- 如果一个或多个载波被干扰破坏，调制解调器可以从其他频率中提取数据。



时分多路复用（TDM）

- 时分多路复用（Time Division Multiplexing）
 - 将时间划分为等长的时分复用帧（TDM 帧）。
 - 每个用户在每一帧中占用固定序号的时隙。
 - 所有用户在不同的时间占用同样的频带宽度。
- TDM 信号也称为等时信号。



时分多路复用 (TDM)

- TDM的两大类型

- 同步时分多路复用
(Synchronous TDM)
- 统计时分多路复用
(Statistical TDM)

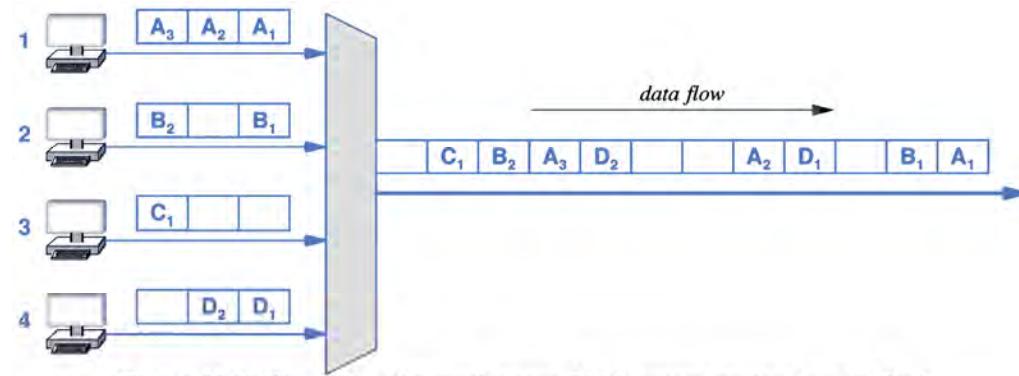


Figure 11.12 Illustration of a synchronous TDM system leaving slots unfilled when a source does not have a data item ready in time.

- 使用时分复用系统传送计算机数据时，由于计算机数据的突发性质，用户对分配到的子信道的利用率一般是不高的。

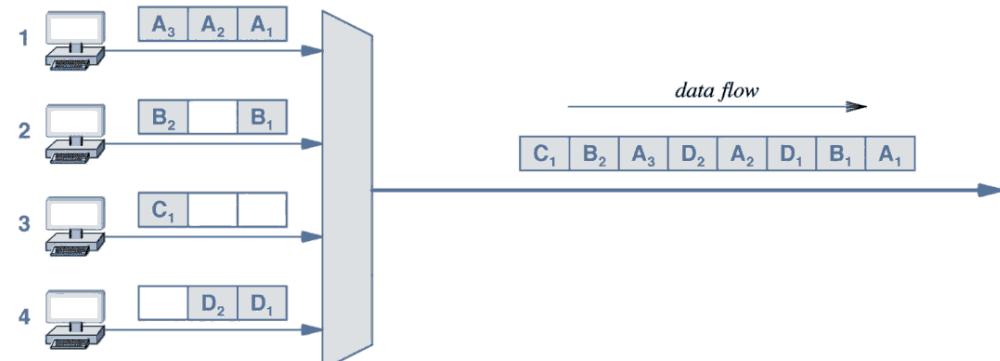


Figure 11.13 Illustration that shows how statistical multiplexing avoids unfilled slots and takes less time to send data.

码分多路复用（CDM）

- 码分多路复用（Code Division Multiplexing）
 - 利用各路信号码型结构正交性而实现多路复用的通信方式
 - 各用户使用特殊挑选的不同码型，因此彼此不会造成干扰。
 - 系统发送的信号有很强的抗干扰能力，其频谱类似于白噪声，不易被敌人发现。



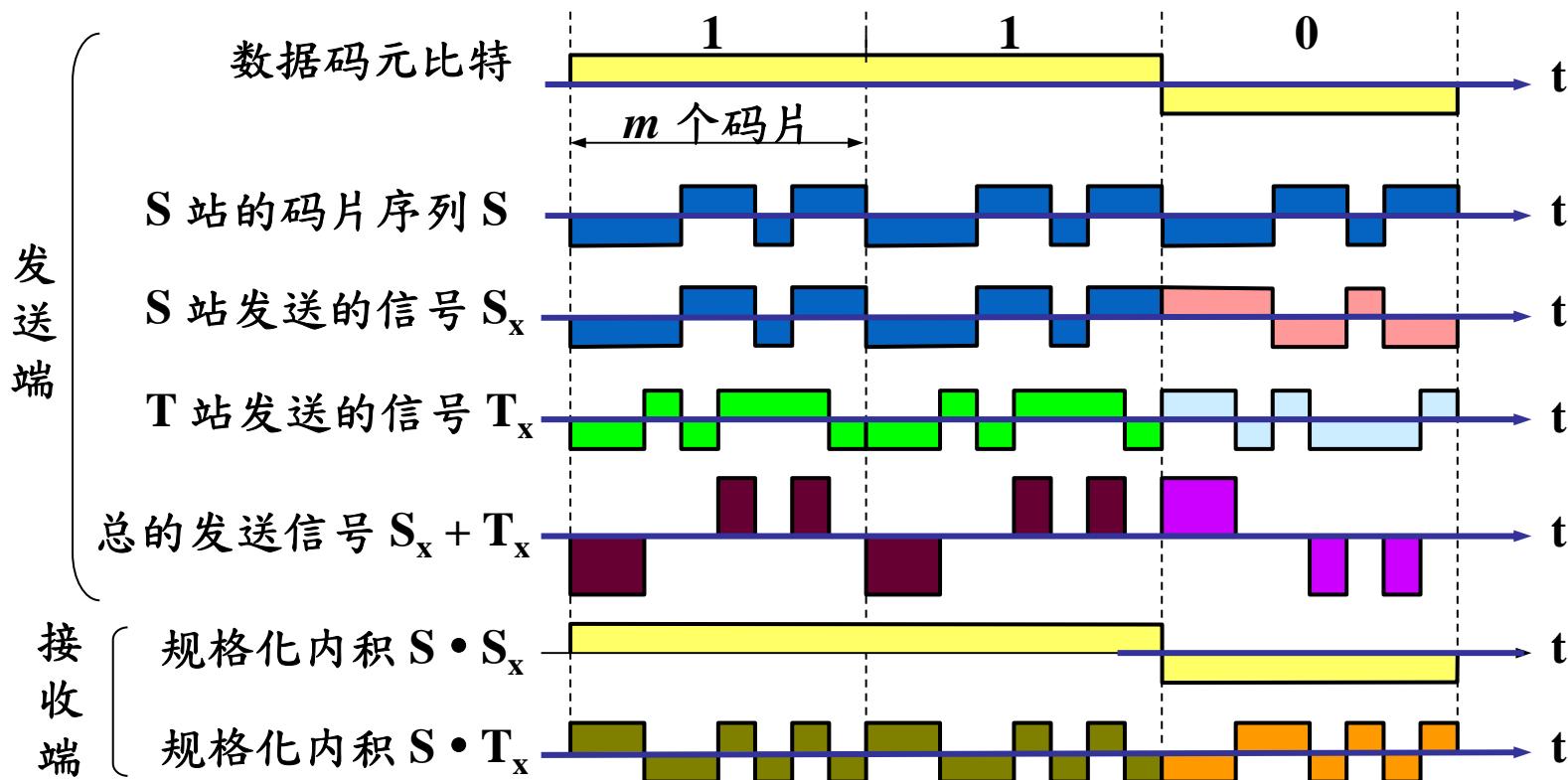
码分多路复用 (CDM)

- 每一位的时间划分为 m 个短的间隔，称为码片(chip)。
- 每个站被指派一个唯一的 m bit 码片序列。
 - 如发送比特 1，则发送自己的 m bit 码片序列。
 - 如发送比特 0，则发送该码片序列的二进制反码。
 - 例如，S 站的 8 bit 码片序列是 00011011。
 - 发送比特 1 时，就发送序列 00011011；发送比特 0 时，就发送序列 11100100。
- S 站的码片序列：(-1 -1 -1 +1 +1 -1 +1 +1)



码分多路复用 (CDM)

- 每个站分配的码片序列各不相同，还须互相正交 (orthogonal)。在实用的系统中是使用伪随机码序列。



3

远距离通信

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

4

可靠信道编码

理论课程

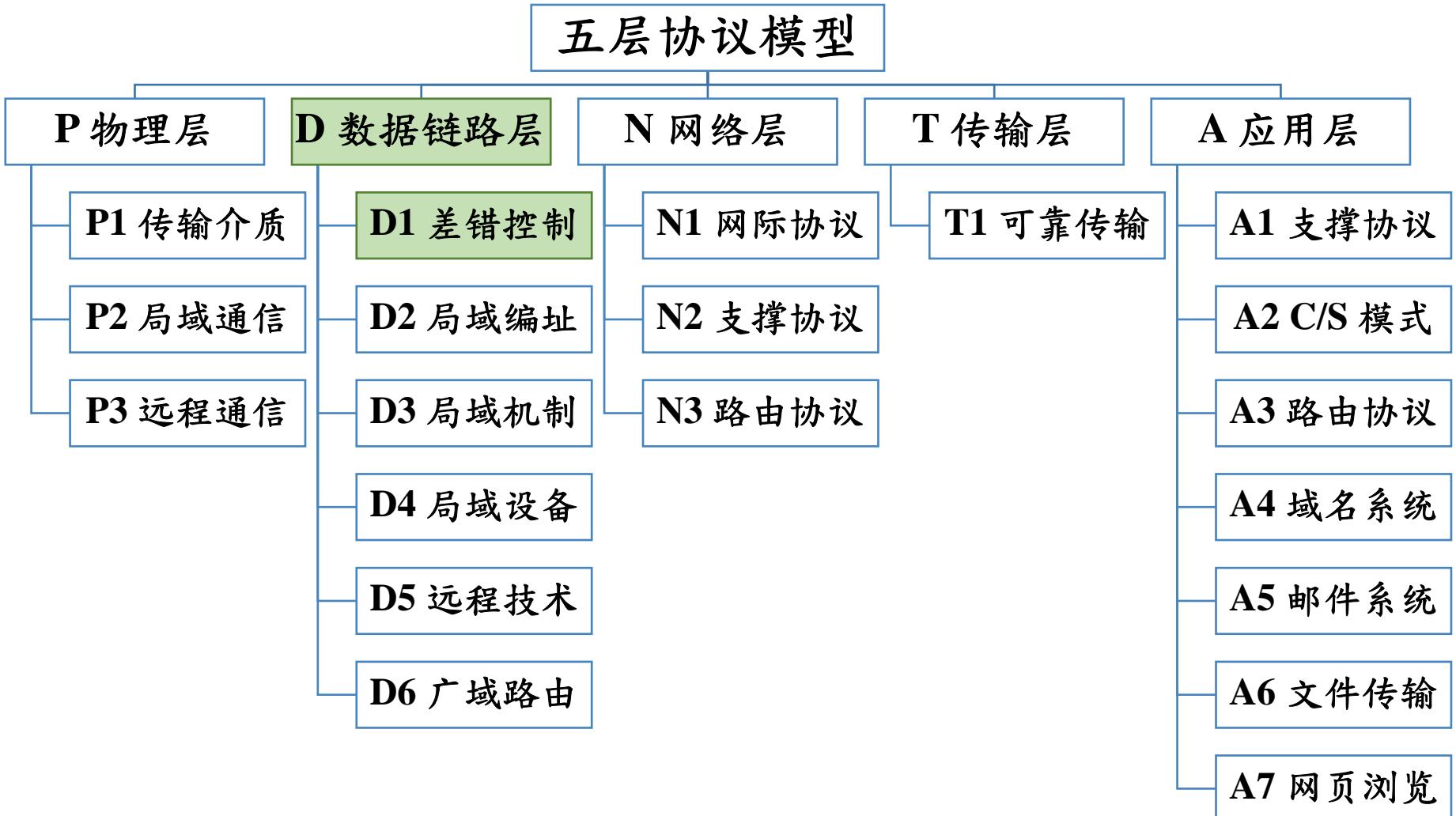


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 传输差错的原因
- 差错控制编码
 - 奇偶校验码、Internet校验和、CRC
 - 计算方法、作用与局限

对应课本章节

- PART II Data Communication Basics
 - Chapter 8 Reliability And Channel Coding

艾滋器官移植事故 台大和成大医院各罚15万

• Reactive vs. Negative



台湾《联合晚报》指出，一位邱姓男子23日不慎在家中坠楼、送到新竹南门医院被判定脑死，24日家属同意捐赠器官，成大医院也接获台大医院通知，24日傍晚台大器官移植小组进开刀房摘除器官，摘下的心肺肾等器官，火速移植给台大和成大医院5名患者，但25日，台大医院发现检验结果书面报告为阳性，以致误将艾滋感染者器官移植给病患。

报道指出，在台湾，现行艾滋病非属重大伤病加上保护感染者，因此不会特别注记在健保卡上，艾滋感染者若因艾滋病到医院就医时，需另外出示医疗卡。因此当意外坠楼的邱姓男子被送到医院急救、判定脑死时，从健保卡无从得知他是否有艾滋病。而邱姓男子家属没有联想到他是同性恋，属于艾滋病高危险群，错失第一道防线。

另外，目前医院没有向台湾卫生主管机关查询列管艾滋病患的程序，而是由医院自行检验。艾滋感染者是在新竹南门医院捐赠器官，血液检体送到台大医院检验，台大协调师出差到新竹，和台大医院检验部电话联系，不知是检验师口误，还是协调师听错，误将reactive(阳性)当成non-reactive(非阳性)，发生严重错失。检验书面报告正确，只是移植团队在摘取器官进行移植时，没有依规再次检视，又错失最后防线。

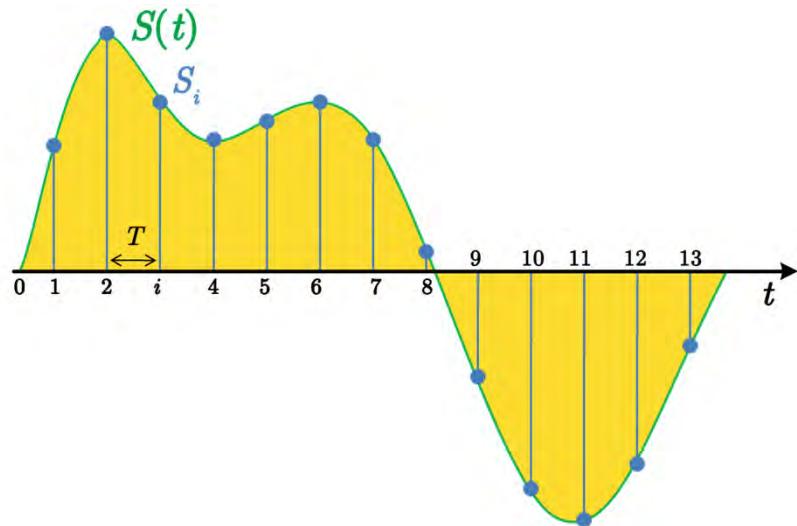
从现实社会到虚拟世界

- 现实世界：连续的

- 时间上的、数值上的，如：挥手，如：声波、光波
- 如果不能找到规律，则需要大量的存储能力

- 计算机世界：离散的

- 时间上：采样；数值上：编码
- 数字化
 - 1号时间、2号时间、.....
 - 1号音量、2号音量、.....



信息熵 (Entropy)

- 六个字符与六位数字的信息量

$$\begin{aligned}
 H(X) &= - \sum_{i=1}^n p(x_i) \log_b p(x_i) \\
 &= - \sum_{i=1}^{256^6} \frac{1}{256^6} \log_2 \frac{1}{256^6} \\
 &= -256^6 \frac{1}{256^6} \log_2 \frac{1}{256^6} \\
 &= 6 \log_2 256 \\
 &= 48 \text{bit} \\
 &= 6 \text{byte}
 \end{aligned}$$

$$\begin{aligned}
 H(X) &= - \sum_{i=1}^n p(x_i) \log_b p(x_i) \\
 &= - \sum_{i=1}^{10^6} \frac{1}{10^6} \log_2 \frac{1}{10^6} \\
 &= -10^6 \frac{1}{10^6} \log_2 \frac{1}{10^6} \\
 &= 6 \log_2 10 \\
 &\approx 19.93 \text{bit} \\
 &\approx 2.49 \text{byte}
 \end{aligned}$$



内容纲要

1

传输差错

2

简单的校验方法

3

循环冗余检验码



传输差错（Transmission Errors）

- 所有通信系统都容易犯错
 - 原因：物理特性；未达到工业标准
 - 小错比大错更难发现
- 这里的差错是自然产生的，不是恶意拼凑的错误
- 分类
 - 干扰（Interference）：元器件电子辐射；宇宙背景辐射
 - 失真（Distortion）：长距离传输会受到干扰
 - 衰减（Attenuation）：通过介质的信号会变弱



传输差错

- 原因

- 闪电、电涌和其他电磁干扰会给电子元件或用于通信的电线带来不必要的电流。

- 现象

- 编码的二进制数据信号，0可能变成1或1变成0。

- 分类：单个比特、突发差错、模糊差错

- 单个比特差错是最常发生的
 - 突发差错是最少发生的



传输差错

- 香农定理：增加信噪比（ signal-to-noise ratio ）
 - 屏蔽线等可以降低噪声，但无法完全消除
 - 然而错误可以被检测（ detect ）到
- 错误检测（ Error detection ）增加开销（ overhead ）
 - 某些情况下错误的比特可以自动纠正
- 错误处理是一个权衡（ tradeoff ）

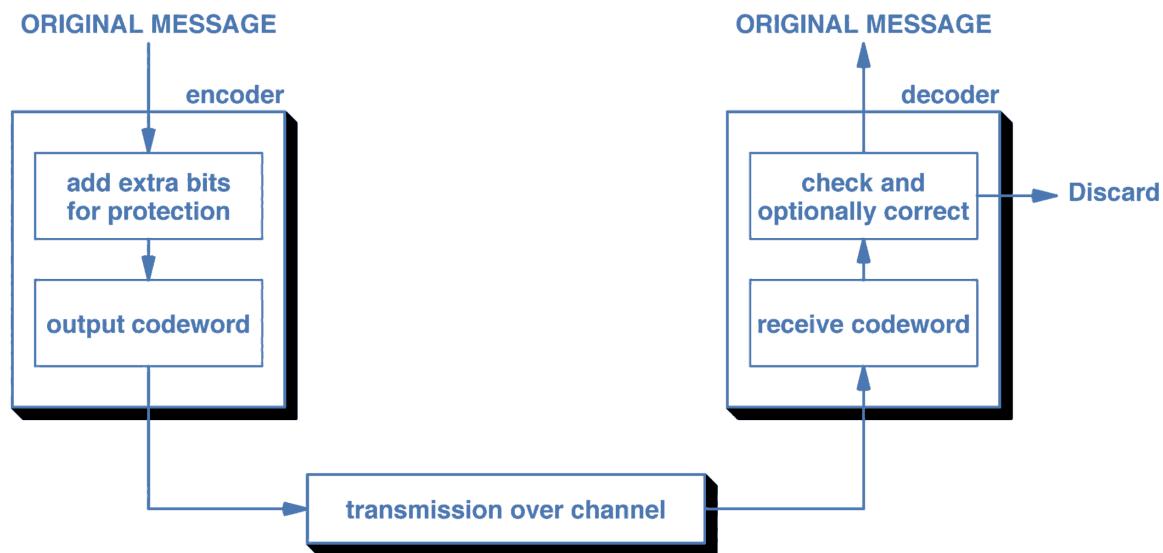


克服信道错误的两种方法

- 冗余可用于纠错
 - 压缩：去除无不确定性的部分（冗余）
 - 校验：利用冗余信息检测或纠正错误
- 信道编码是可用于克服传输错误的数学方法
 - 前向错误纠正（Forward Error Correction，FEC）
 - 自动重传请求（Automatic Repeat reQuest，ARQ）

克服信道错误的两种方法

- 前向错误纠正



- 自动重传请求

- 每当发送消息到对方，对方会立即响应（ACK）。如果没有收到，则假设消息丢失，应重传副本。
- 接收方如果发现信息错误，则丢弃

前向错误纠正技术的两种编码

- 分块错误编码（ Block Error Codes ）
 - 将数据分块(block) , 将冗余(redundancy)加到每块信息中
 - 无记忆(memoryless)：编码依赖于给定块，和前面的块无关
- 卷积错误编码（ Convolutional Error Codes ）
 - 数据被视为位序列，连续计算编码
 - 计算的码取决于当前输入和以前的一些位流
 - 卷积码是有记忆码（ codes with memory ）



内容纲要

1

传输差错

2

简单的校验方法

3

循环冗余检验码



校验位和奇偶校验

- 奇偶校验（ parity check ）是一种机制
 - 发送方根据消息序列计算一个额外的位附加在原有消息序列后，称为奇偶位（ parity bit ）
 - 接收方收到所有位后，校验并丢弃奇偶位。
- 两种机制：奇（ odd ）和偶（ even ）
 - 奇校验中消息和校验位共有奇数个1；偶校验共有偶数个1
 - 发送和接收方应有同样的模式



单个奇偶校验位

- 多说无益，看代码吧

```
unsigned char val=0x5B;
unsigned char pcOdd=1;
printf("Bits: ");
for (size_t i = 0; i < 8; i++) {
    printf("%d ", val >> (7-i) & 1);
}
for (size_t i = 0; i < 8; i++) {
    pcOdd ^= val & 1;
    val >>= 1;
}
printf("\nParity bit: Even: %d; Odd: %d.\n", !pcOdd, pcOdd);
```

Bits: 0 1 0 1 1 0 1 1
Parity bit: Even: 1; Odd: 0.

校验位和奇偶校验

- 奇偶校验码是可检测错误的信道编码中的一种弱形式
 - 可以检测错误，但不能纠正错误
- 只能检测部分错误
 - 奇数个出错：认为是错的；偶数个出错：误认为是对的
 - 未出错：对的
- 行列码

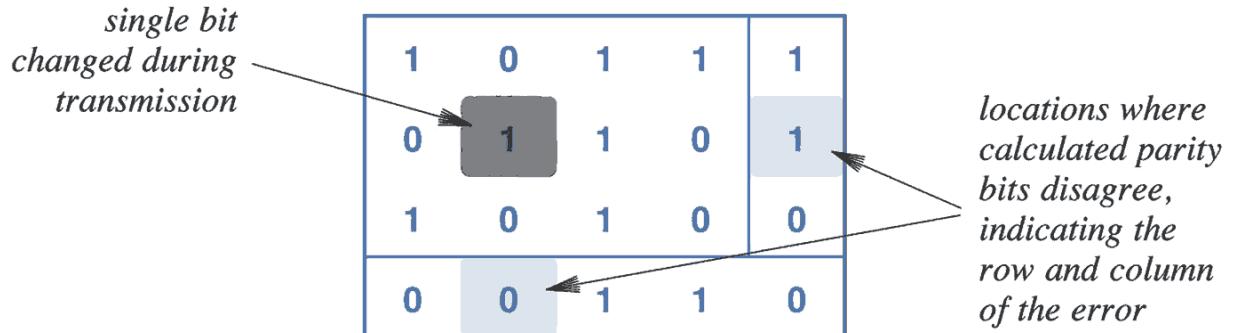


Figure 8.8 Illustration of how a single-bit error can be corrected using a row and column encoding.

块错误码和 (n, k) 符号

- 数据字（ Datawords ）：所有消息的集合
- 码字（ Codewords ）：所有正确编码的集合
- (n, k) 码
 - 数据字有 k 位，校验码 r 位附加在数据字后形成码字
 - 度量的好坏是码率（ Code rate ）： $R=k/n$
- 成功的关键在于选择的是有效码字的可能组合的子集
 - 有效的子集被称为一个码本（ codebook ）
 - 以奇偶校验码为例，一半是码本



汉明距离：代码强度的量度

- 汉明距离 (Hamming distance)

- 两个字符串汉明距离是它们不同位的数量

Dataword	Codeword
0 0	0 0 1
0 1	0 1 0
1 0	1 0 0
1 1	1 1 1

$d(001,010) = 2$	$d(010,100) = 2$
$d(001,100) = 2$	$d(010,111) = 2$
$d(001,111) = 2$	$d(100,111) = 2$

- 码本的 (a) (b) distance)

- 没有理想的信道编码：修改足够多位可转换至合法的码字
 - 码本中任何码字转换成另一个码字所需要改动位数的下限



Internet 校验和

- Internet校验和（ Internet checksum ）

- 数据字不需要固定大小
- 不足16位补0
- 计算校验和
- 发送方在消息后添加校验和

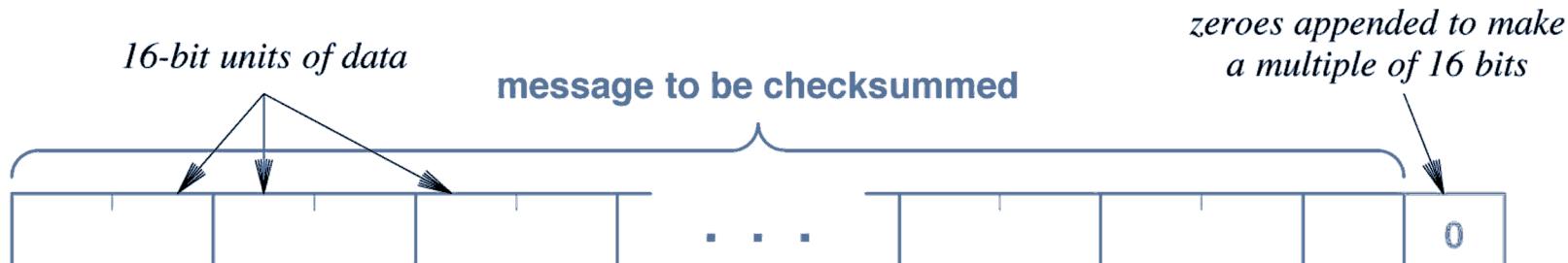
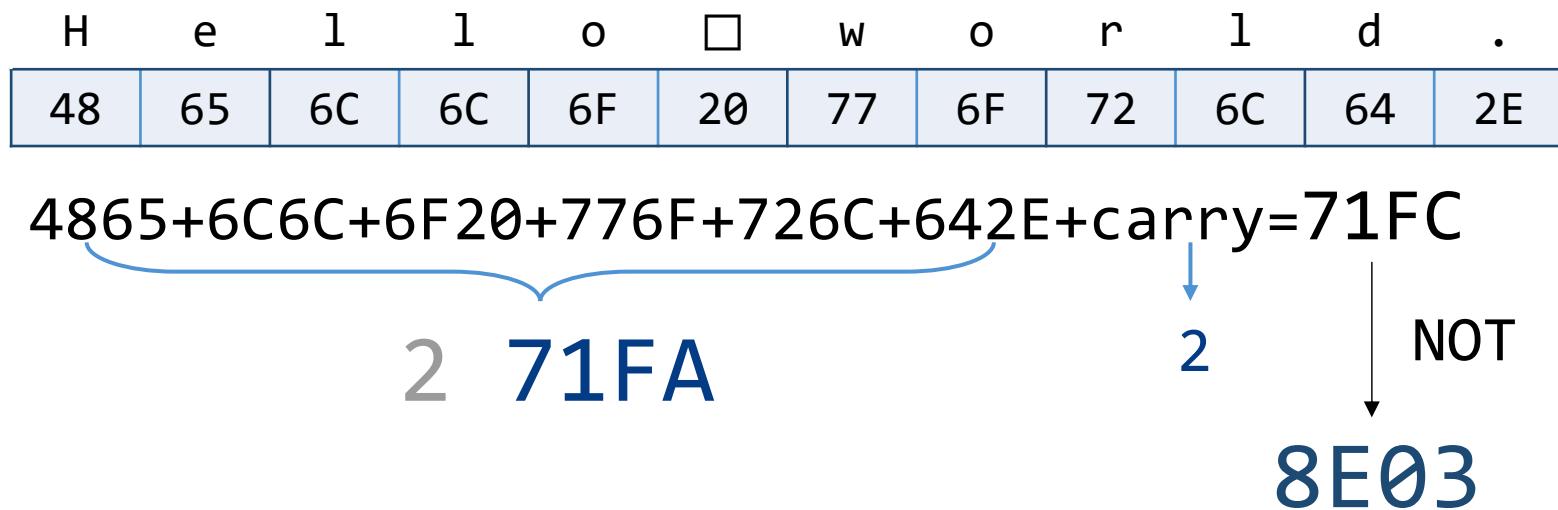


Figure 8.9 The Internet checksum divides data into 16-bit units, appending zeroes if the data is not an exact multiple of 16 bits.

Internet 校验和

- 为了计算校验和，发送者把消息每16位作为一个整数计算总和，如果结果大于16位，则重复上述过程。
- 校验和尺寸小，额外传输开销小



计算Internet Checksum

- 以下代码源于 RFC 1071 文档。

```
/* Compute Internet Checksum for "count" bytes beginning at location "addr". */
register long sum = 0;
while (count > 1)  {
    /* This is the inner loop */
    sum += *(unsigned short *) addr++;  count -= 2;
}

/* Add left-over byte, if any */
if (count > 0)
    sum += *(unsigned char *) addr;

/* Fold 32-bit sum to 16 bits */
while (sum>>16)
    sum = (sum & 0xffff) + (sum >> 16);
checksum = ~sum;
```

内容纲要

1

传输差错

2

简单的校验方法

3

循环冗余检验码



循环冗余检验码（CRC）

- 循环冗余检验码（Cyclic Redundancy Codes）
 - 一种根据网络数据包或文件等数据产生简短固定位数校验码的散列（Hash）函数，主要用来检测或校验数据传输或者保存后可能出现的错误。
- 重要特征
 - 任意消息字长
 - 出色的错误检测
 - 快速硬件实现（位移寄存器，异或门）

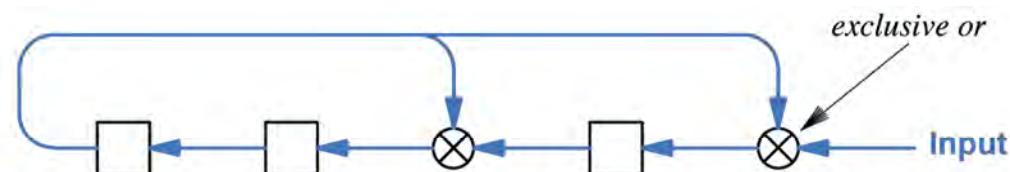


Figure 8.13 A hardware unit to compute a 3-bit CRC for $x^3 + x^1 + 1$.

循环冗余检验码 (CRC)

- 可以视为不带借位的二进制数除法

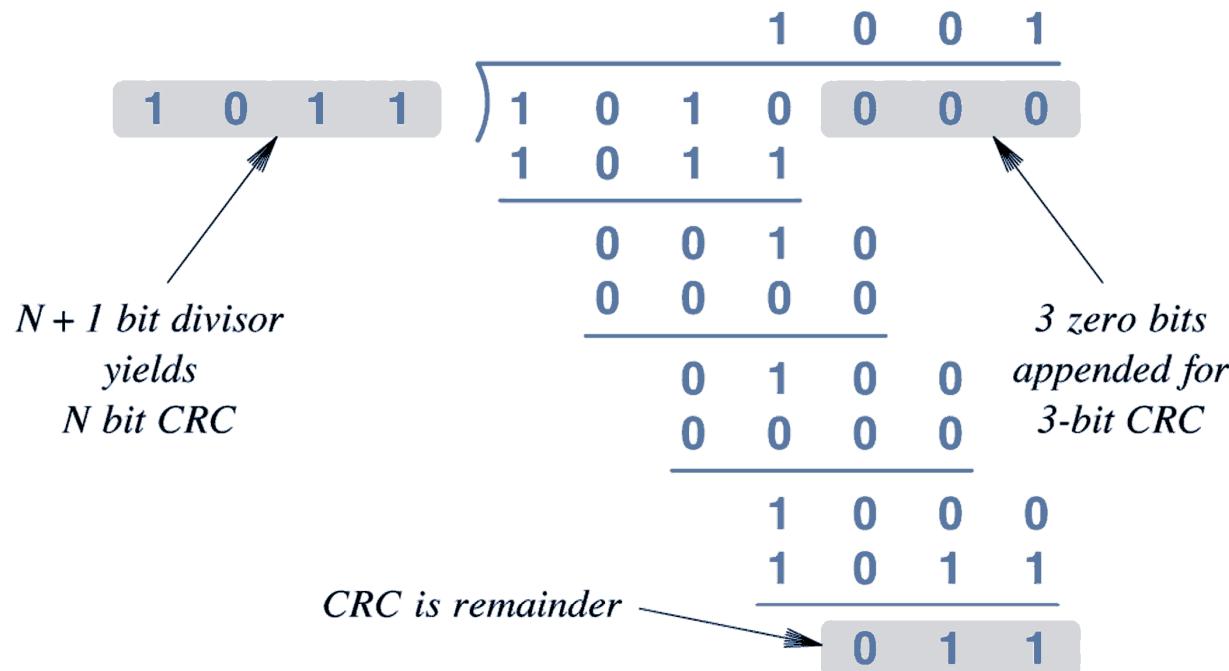


Figure 8.12 Illustration of a CRC computation viewed as the remainder of a binary division with no carries.

循环冗余检验码（CRC）

- 每个位的二进制数作为一个多项式的系数
- 图8.12除数1011可视为 $1 \times x^3 + 0 \times x^2 + 1 \times x^1 + 1 \times x^0 = x^3 + x + 1$
 - 被除数1010000可以视为 $x^6 + x^4$
- 除数多项式称为生成多项式（generator polynomial）
 - 一个理想的多项式是不可约的（只能被自己和1整除）
 - 有一个以上的非零系数多项式可检测所有的单比特错误



生成多项式 $G(x)$ 的国际标准

- 生成多项式 $G(x)$ 的国际标准

- **CRC-8** $x^8 + x^2 + x + 1$

- **CRC-10** $x^{10} + x^9 + x^5 + x^4 + x^2 + 1$

- **CRC-12** $x^{12} + x^{11} + x^3 + x^2 + x + 1$

- **CRC-16** $x^{16} + x^{15} + x^2 + 1$

- **CRC-CCITT** $x^{16} + x^{12} + x^5 + 1$

$x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

突发错误（ Burst Errors ）

- 两种常见错误使得CRC很有用
 - 硬件故障有时会导致损坏一组特定位，如：垂直错误。
 - 在单个位置附近一小部分位的误差特别有用，即突发错误。



帧格式和错误检测机制

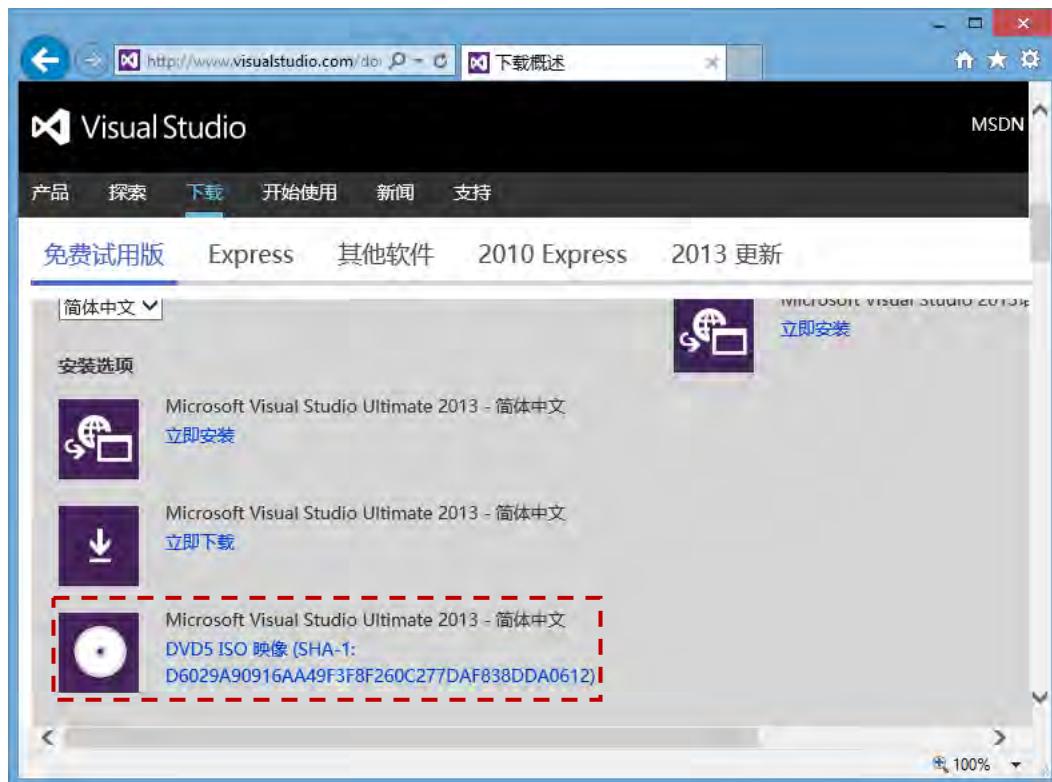
- 网络通常将错误检测信息和每一帧联系起来。
 - 发送方计算信息的校验和或CRC，并随着帧数据发送附加信息。
 - 接收方用相同的方法计算校验值。
 - 一般接收方并不是重复计算校验值再与发送方发来的校验值比较，而是将校验值一起计算。
 - 复杂的发送方，简单的接收方。（好处？）



Figure 7.11 A modification of the frame format from Figure 7.3 that includes a 16-bit CRC.

更多的校验码

- 没有完美的错误检测方案，因为传输错误可以影响更多的信息以及数据。
- 信息安全用的校验和
 - MD5、SHA 系列



计算机网络

Computer Network

4

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

5

分组，帧和编址

理论课程

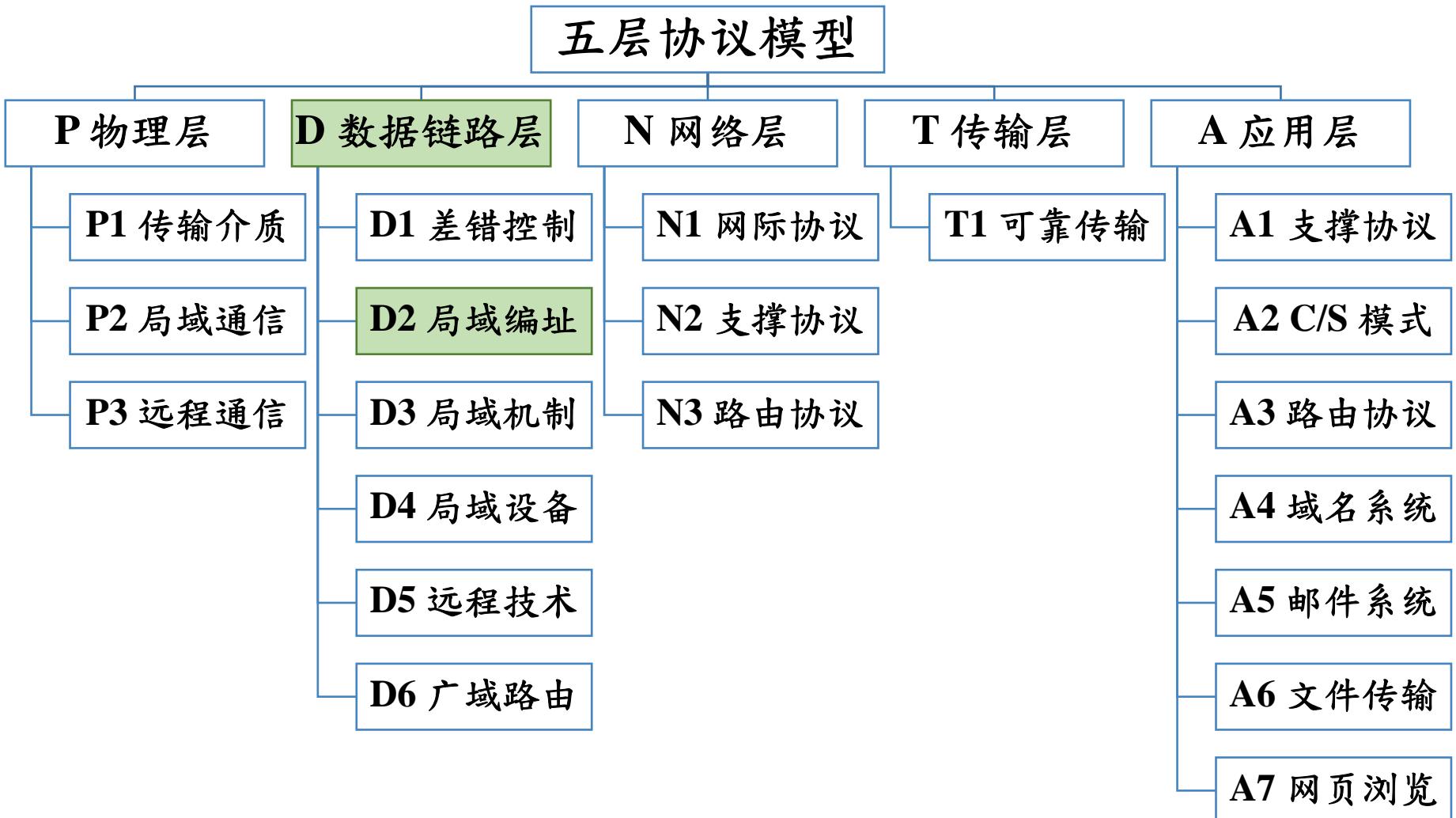


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 电路交换和分组交换
- 解复用和编址
 - MAC地址的组成；单播、多播、广播
- 帧与成帧
 - 帧的构成；以太网的帧格式
- 网卡
 - 作用；混合模式

对应课本章节

- PART III Packet Switching And Network Technologies
 - Chapter 13 Local Area Networks: Packets, Frames, And Topologies

内容纲要

1

分组和交换

2

分组和帧

3

编址：分组解复用

4

以太网的帧格式

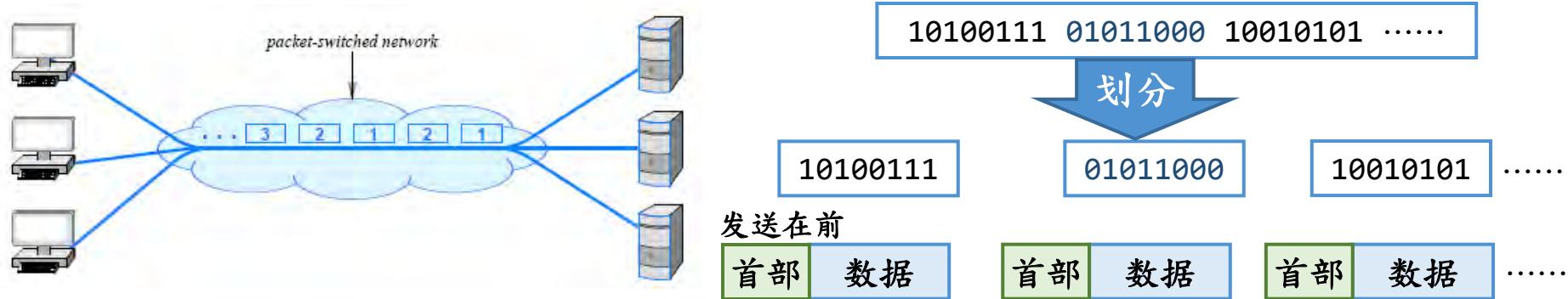
5

网络监听实验



分组 (Packet)

- 分组
 - 网络系统将数据分为小的分块（分组），独立发送。
- 分组交换网中，数据以分组的形式轮流发送



分组交换 (Packet Switching)

- 分组交换

- 以分组为单位进行传输和交换的存储-转发交换方式。
- 采用统计复用，多个来源争夺共享介质使用

- 特点

- 异步
- 无需建立
- 性能各异

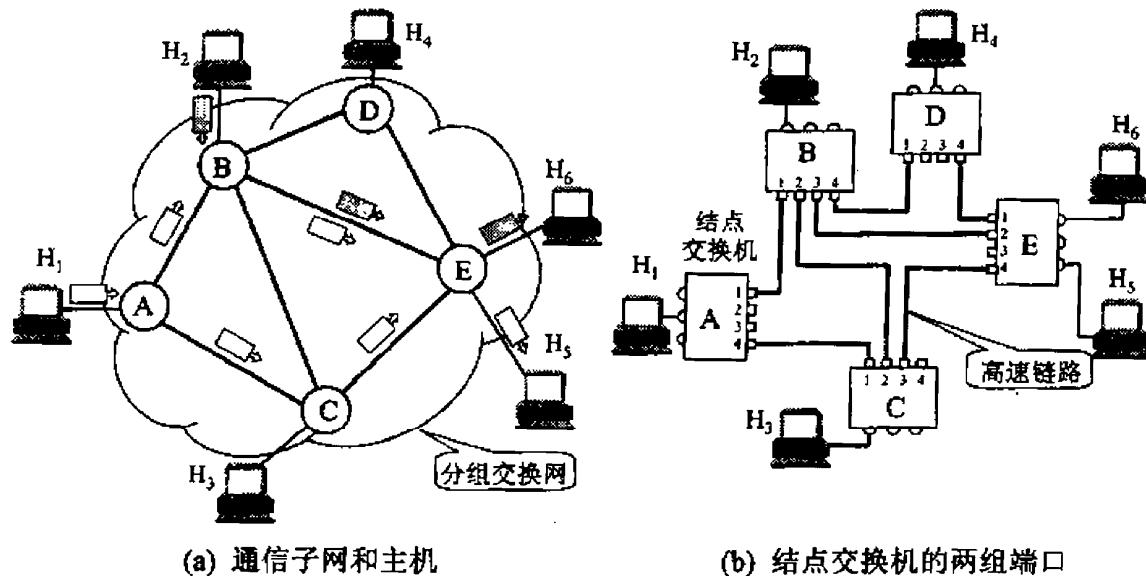


图 1-6 分组交换网的示意图

电路交换 (Circuit switching)

- 电路交换：以电路连接为目的的交换方式
 - 通信之前在通信双方之间建立一条被双方独占的物理通道。
 - 多个电路在共享介质上复用，形成虚拟通路
 - 类似于电话技术：建立线路、线路交互、终止使用

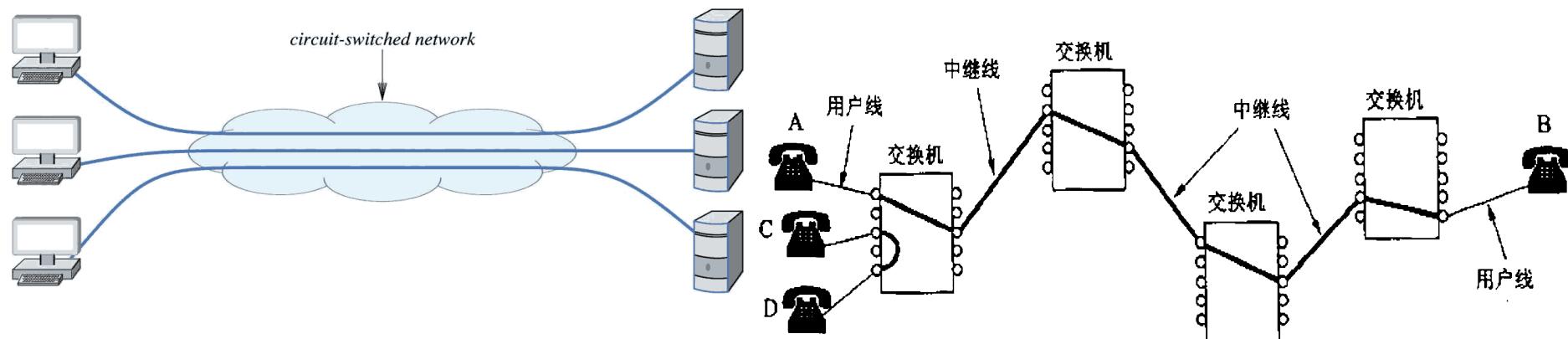


Figure 13.1 A circuit-switched network that provides a direct connection between each pair of communicating entities.

图 1-4 电路交换的示意图

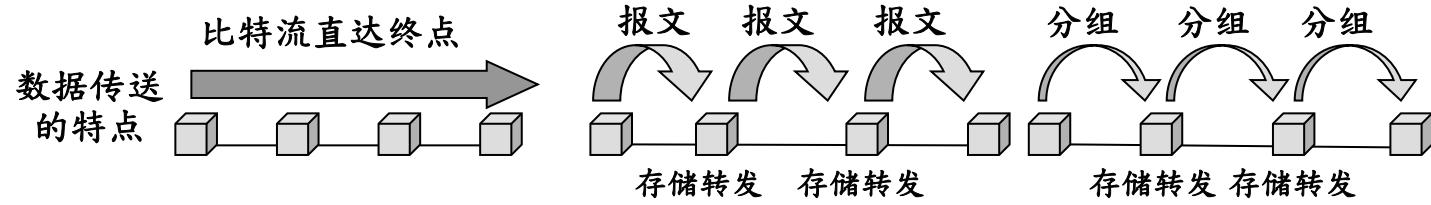
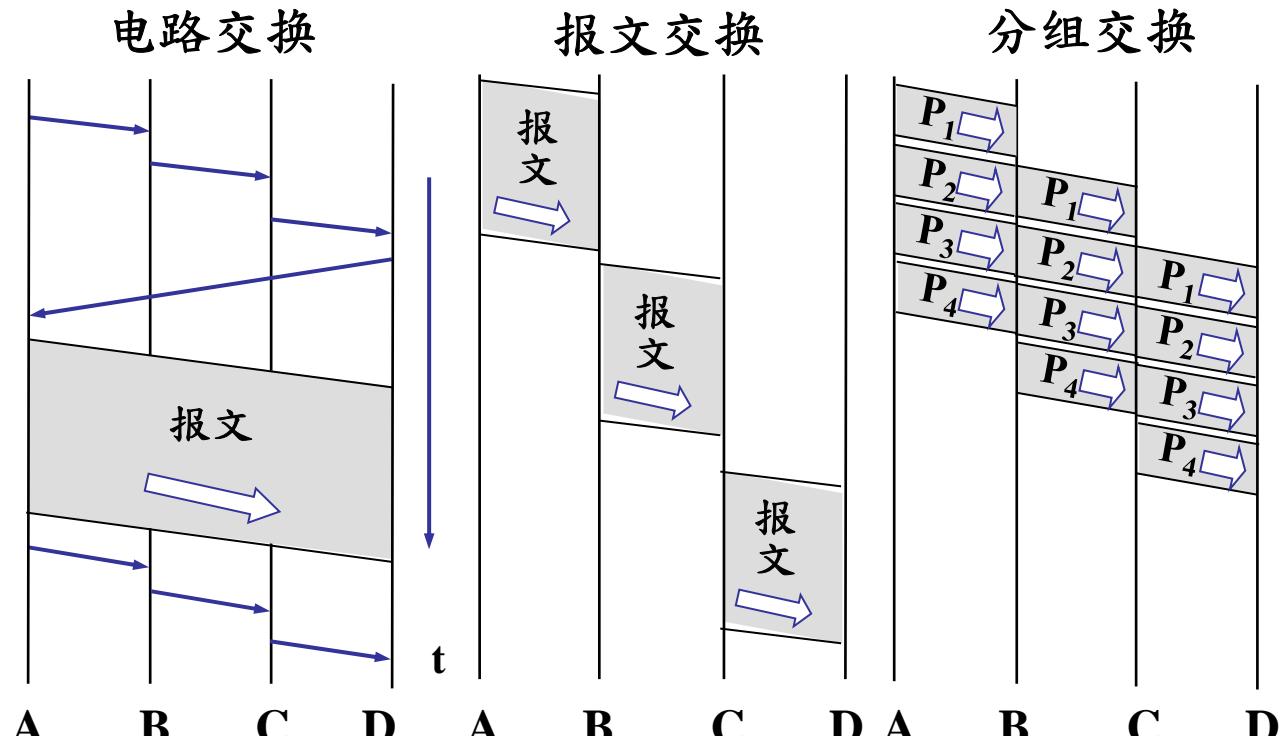
电路交换与分组交换

- 线路交换

- 独立信道
- 实时性高

- 分组交换

- 信道利用率高



内容纲要

1

分组和交换

2

分组和帧

3

编址：分组解复用

4

以太网的帧格式

5

网络监听实验



分组和硬件帧

- 帧 (Frame)

- 帧是用于具体网络类型的分组
 - 在分组交换网络中，每个帧对应一个分组。

- 帧结构

- 帧结构是指添加到位序列或字节序列中的结构
 - 该结构允许发送方和接收方对消息的确切格式达成一致。

- 成帧 (Framing)

- 是指同步通信系统中使接收器知道消息开始和结束的机制。

帧的一般结构

- 头部



- 包含元数据，如用于处理帧的地址信息
- 消息是不透明的，网络只检查帧头（ frame header ）

- 载荷（ Payload ）

- 包含发送的数据，通常比帧头大得多
- 有效载荷包含只对发送者和接收者有意义的字节序列

- 一些技术在帧前发送短的前奏，在帧后发送短的尾曲
- 对帧进行定界：头部开始符、传输终止

利用字节填充定界

- 如何区分正常数据和定界？

- 回顾C语言转义字符\r\n，右斜杠：\r\n

- 字节填充（data stuffing）：插入额外的比特或字节

- 字符填充

- 位填充

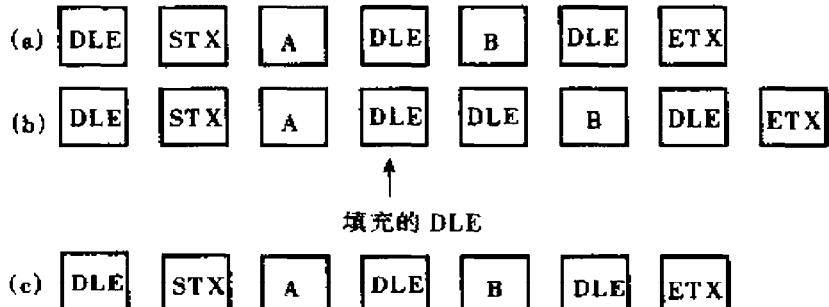


图 3-4 (a) 网络层发出的数据；(b) 经数据链路层填充后的数据；
(c) 数据传送给接收方的网络层。

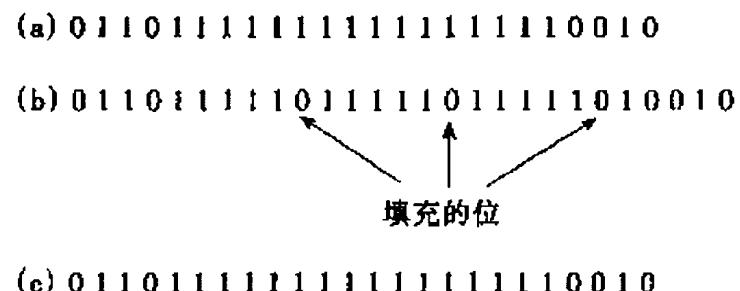


图 3-5 位填充
(a) 原始数据；(b) 线上数据；(c) 删除填充位后接收方存储器内的数据。

内容纲要

1

分组和交换

2

分组和帧

3

编址：分组解复用

4

以太网的帧格式

5

网络监听实验



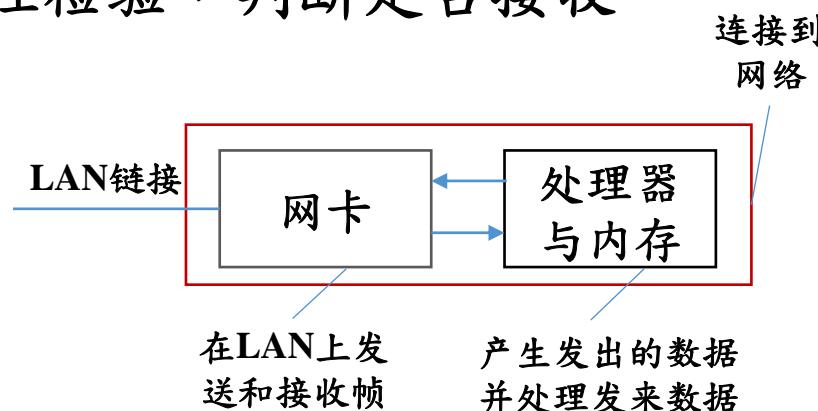
网卡 (Network Interface Card)

- 网卡处理计算机进行网络通讯的设备
 - 由一侧带有插头的电路板组成。
 - 功能像I/O设备，使用中断机制来通知CPU。
 - 作用：处理地址识别、CRC计算、帧识别、发送和接收帧



• 分层处理

- 网卡：检测帧是否存在，有效性检验，判断是否接收
- CPU：判断是否传给上层处理
- 目的：减少CPU的负荷



网卡

- 局域网速率很快，CPU无法以网络速度处理位。
 - 网络速度常为固定，不依赖于连接的计算机的CPU速度。
 - 网卡理解的网络上使用的电信号、该数据被发送或接收的速率，以及网络的帧格式的细节。
 - 网卡完全处理共享介质的收发细节，无需借助CPU
- LAN接口硬件使用物理地址防止收到LAN所有数据包
 - 在局域网中，使用ID号作为地址，解复用不同的接受方。
 - 一旦获得完整帧，将其目的地址与该站的物理地址比较。



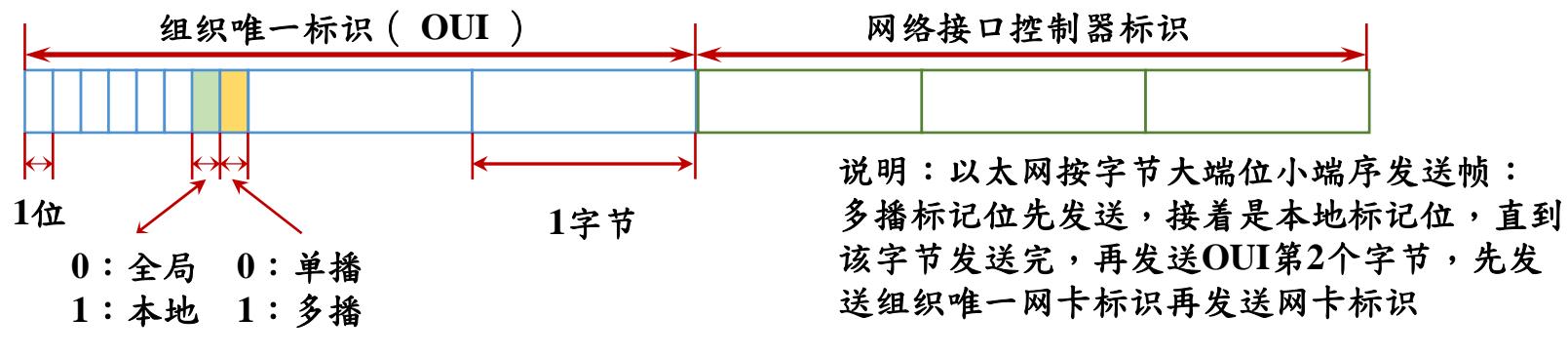
分组标识和解复用

- IEEE 地址严格地讲是站点的标识符（ Identifier ）
 - 局域网内要对物理地址提供检索的功能
- 硬件地址（ hardware address ）别称
 - 物理地址（ physical address ）
 - MAC 地址（ media access address ）
- 需要完成解复用，局域网（ LAN ）的帧须包含
 - 发送者地址，称为源地址（ source address ）
 - 接收者的地址，称为目的地址（ destination address ）



IEEE编址（Addressing）方案

- IEEE保证每张网卡分配到唯一的（unique）地址
- IEEE编址方案中，每个Ethernet地址有48位
 - IEEE向厂家分配组织唯一标识（OUI），即高24位。
 - 第1个字节最低位为0表示单播，1表示多播和广播；
 - 第一个字节最低第2位为0表示全局网址，1表示局域网址。
 - 厂家自行分配网卡标识符，即低24位。



网卡物理地址的配置方式

• 静态地址（Static address）

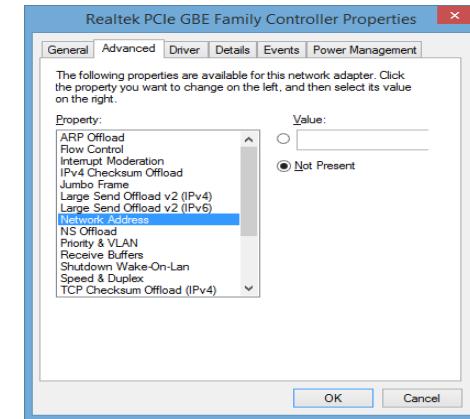
- 出厂前一次性配置，全球唯一，永久使用
- 地址较长，查询慢

• 可配置地址（Configurable address）

- 用户动态分配，局域网内唯一，永久有效也可随需求改变
- 地址短，查询快

• 动态地址（Dynamic address）

- 系统启动时动态分配，局域网内唯一，可能出现冲突
- 地址短，查询快，动态不利于地址映射表的维护



网卡通讯方式：单播，多播和广播

- 单播（ unicast ）帧：一对一
- 多播（ multicast ）帧：一对多，特定地址
 - 当一台计算机需要向多台计算机广播信息时，若使用单播，必须向每台计算机分别发送数据，造成网络负载成倍增长。
 - 网上所有设备的网卡分别拷贝数据帧，并交给CPU处理。需要接收数据帧的设备由CPU向上层传递，不需要数据帧的设备由 CPU 丢弃该帧。这样增加了无关设备的CPU负载。
- 广播（ broadcast ）帧：一对全体，48位全为1

网卡处理分组的算法

- 从分组提取目标地址D
- 如果D符合“我的地址”，则接受并处理分组；
 - 复制帧，中断CPU；将复制后的帧交给CPU
- 否则，如果D符合广播地址，则接受并处理分组；
- 否则，如果D符合多播地址，则接受并处理分组；
 - 多播地址是配置在网卡的，指向一个多播组
- 否则，丢弃该分组，继续等待下一个帧



混杂模式（ Promiscuous Mode ）

- 定义

- 指网卡接收所有经过它的数据流，而不过滤目的地址。

- 用途

- 混杂模式通常用于监听网络上的数据流

- 设置

- 通常安装网络监听软件之后，网卡处于混杂模式

- WinPCAP (Wireshark 需要该软件)
 - libPCAP

内容纲要

1

分组和交换

2

分组和帧

3

编址：分组解复用

4

以太网的帧格式

5

网络监听实验



以太网的帧格式

• 以太网的帧格式

前同步码	SFD	目的地址	源地址	类型	数据	CRC
7字节	1字节	6字节	6字节	2字节	46~1500 字节	4字节

– 前同步码（物理层）：56比特交替出现的0和1

- 提醒接收系统有帧到来，以及使到来的帧与计时器同步。
- 帧首定界符（SFD）：1字节的10101011，帧的开始。
- 目的地址：6字节，目的物理地址。
- 源地址：6字节，源物理地址。

以太网的帧格式

- 以太网的帧格式

前同步码	SFD	目的地址	源地址	类型	数据	CRC
------	-----	------	-----	----	----	-----

7字节 1字节 6字节 6字节 2字节 46~1500 字节 4字节

- 类型：2字节，定义了封装在帧中的数据类型。
- 数据：包含从上层来的数据，必须在**46 到 1500** 字节之间。
 - 如果上层协议产生的数据长度小于46字节，则应将其填补到46字节。
 - 若数据长度超过1500 字节，上层就必须将其进行分片
- 循环冗余检验（CRC）：4字节，CRC-32，用于差错检测。

以太网的帧类型域（ type field ）

- 显式帧类型：帧类型域提供了复用和解复用
- 当帧到达目的地时接收器检查类型字段，使用该值来确定哪个软件模块应该处理帧

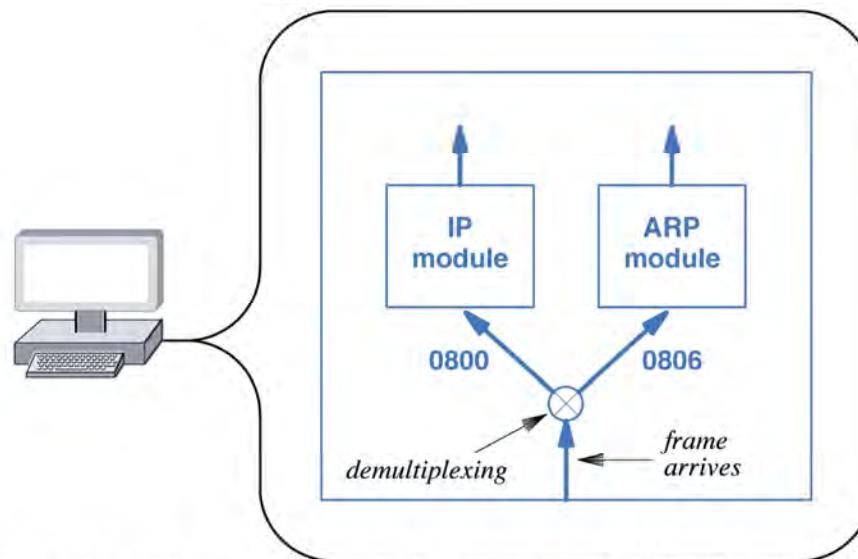


Figure 15.2 Illustration of using the frame type field for demultiplexing.

*令牌环的帧格式

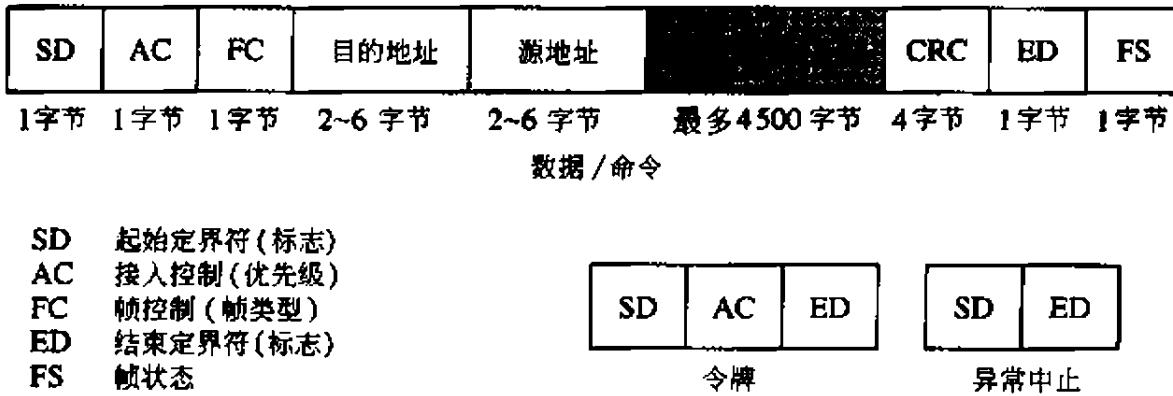


图 3.15 令牌环的帧格式

- 目的地地址(DA)。这个可变长度(2 至 6 字节)字段是下一站的物理地址。
- 源地址(SA)。这个可变长度(2 至 6 字节)字段是前一站的物理地址。
- 数据。这个字段是数据。数据可多到 4500 字节。
- CRC。这个字节为 4 字节长,包含 CRC-32 检错序列(见附录 D)。
- 结束定界符(ED)。这个 1 字节字段指出发送器的数据结束,同时还包含更多的控制信息。
- 帧状态(FS)。这个 FS 字段由接收器设置,指出帧已被读取,或由监督站设置,指出该帧已在环上转了一圈。

令牌帧,令牌帧包括三个字段:SD、AC 和 ED。

异常中止帧,异常中止帧只有两个字段:SD 和 ED。当出现一些问题时,监督站使用异常中止帧来中止令牌传递机制。



IEEE的Ethernet版本 802.3

- 传统Ethernet与802.3 Ethernet区别于类型域的解释
- 新版本兼容旧版本

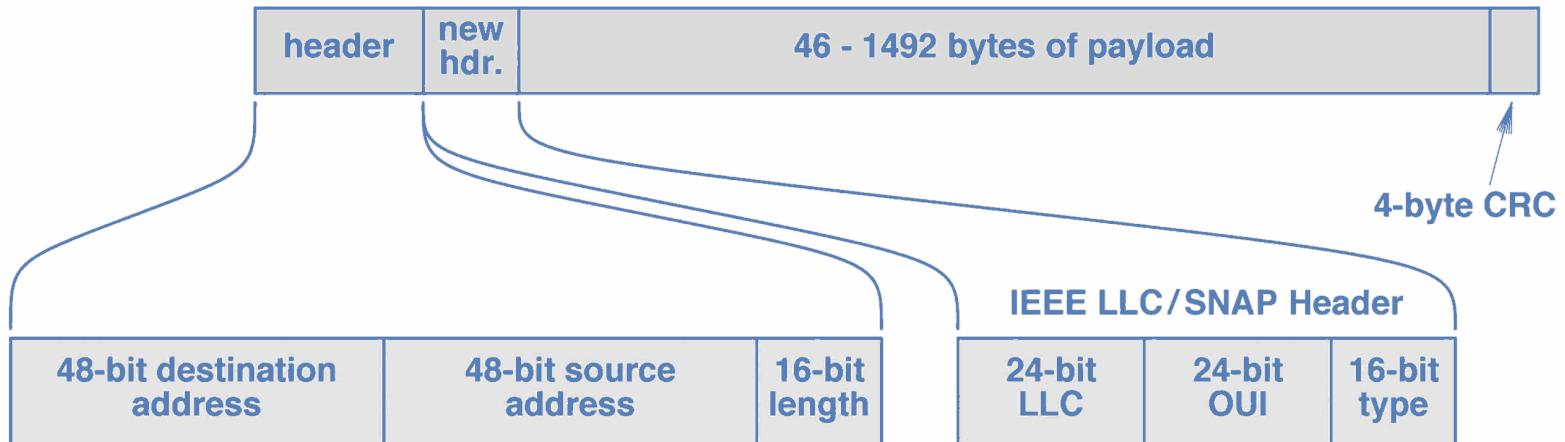


Figure 15.3 The IEEE 802.3 frame format with an LLC/SNAP header.

内容纲要

1

分组和交换

2

分组和帧

3

编址：分组解复用

4

以太网的帧格式

5

网络监听实验



小实验

- 用Wireshark监听本机的收发包，观察MAC地址
 - 在宿舍里（设宿舍内不用路由），PING不同室友的电脑（或QQ传文件），观察包的MAC地址
 - PING公网上的不同主机，观察包的MAC地址
 - 以上实验也可以在机房完成
 - Wireshark程序在FTP上，用法请上网搜教程。
- 得到了什么结论？



监听结果节选

Packet #1

Ethernet Type 2

Destination: FF:FF:FF:FF:FF:FF *Ethernet Broadcast [0-5]*
Source: 00:0C:29:37:5A:1B *VMware:37:5A:1B [6-11]*
Protocol Type: 0x0800

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Protocol: 17 *UDP* [23]
Source IP Address: 0.0.0.0 [26-29]
Dest. IP Address: 255.255.255.255 *IP Broadcast [30-33]*

UDP - User Datagram Protocol

Source Port: 68 *bootpc [34-35]*
Destination Port: 67 *bootps [36-37]*

BootP - Bootstrap Protocol

IP Address Known By Client: 0.0.0.0 *IP Address Not Known By Client [54-57]*
Client Hardware Addr: 00:0C:29:37:5A:1B *VMware:37:5A:1B [70-75]*

DHCP - Dynamic Host Configuration Protocol Requested IP Address

Address: 192.168.7.132 [296-299]

Host Name Address

String: WIN-KG9CLM76UIA [302-316]

计算机网络

Computer Network

5

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

6

网络拓扑，机制 和无线技术

理论课程

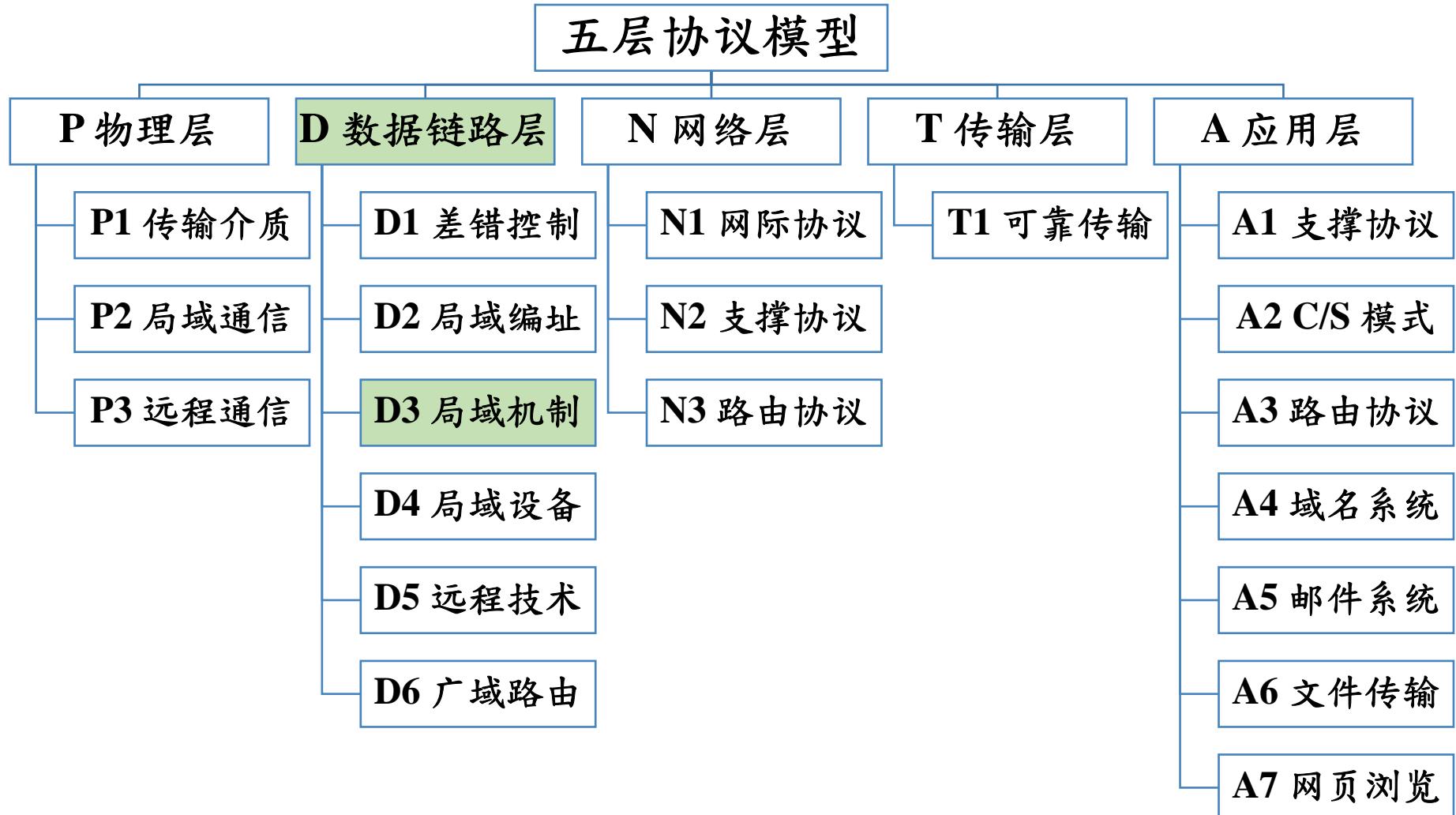


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

主要内容



第5课

分组、交换、网卡、编址、帧格式

第6课

网络拓扑、网络机制、无线网络

第7课

网络布线、接口硬件



主要内容

- 拓扑
 - 星型、环型、总线型、网状；其优点缺点比较
- Ethernet的机制：IEEE 802.3标准
 - Manchester 编码；CSMA/CD机制
- 其它网络
 - LocalTalk；Token Ring；FDDI；ATM
- 无线联网技术：1G~4G；Wi-Fi；Bluetooth；GPS

对应课本章节

- PART III Packet Switching And Network Technologies
 - Chapter 13 Local Area Networks: Packets, Frames, And Topologies
 - 13.8 LAN Topologies
 - Chapter 14 The IEEE MAC Sub-Layer

内容纲要

1

网络拓扑

2

以太网的编码和协议

3

以太网的工作机制

4

其它网络

5

无线网络技术



拓扑 (Topology)

- 拓扑学 (Topology)

- 研究几何图形或空间在连续改变形状后还能保持不变的一些性质的学科。

- 拓扑

- 只考虑物体间的位置关系
 - 不考虑它们的形状和大小
 - 不表示方位。



远距离和近距离通信

- 远距离通信特点
 - 安全、稳定：能迅速从故障中恢复
 - 通信量大
- 近距离通信特点
 - 成本低
 - 便于使用
 - 与物理位置近邻通讯多
 - 与经常通讯的主机通讯多



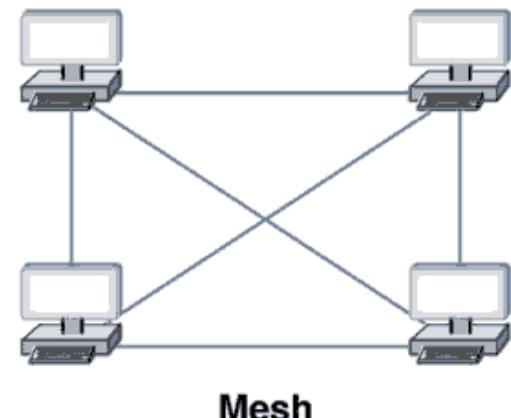
网状拓扑（ Mesh Topology ）

- 网状网络

- 特征：点对点（ Point-to-point ）连接
- 优点：独立安装、独占访问；安全稳定
- 缺点：线多，成本高

- 应用

- 常用于远距离通信



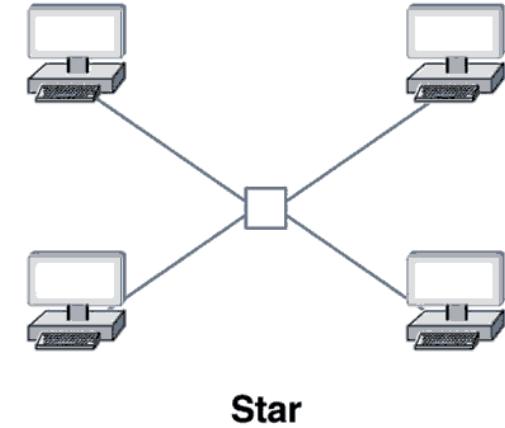
星型拓扑（Star Topology）

- 星型结构

- 一个中心节点
 - 一组计算机通过中心节点实现信息传输
 - 各节点和中心节点之间通过点到点的链路进行连接

- 集线器（Hub）

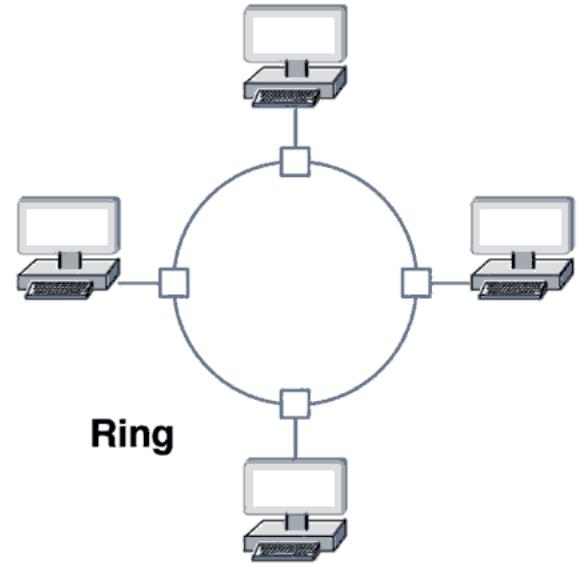
- 集线器到所有计算机的距离并不相等
 - 作为中心点很脆弱，经常放置于网管办公室锁起来



环形拓扑 (Ring Topology)

- 环型结构

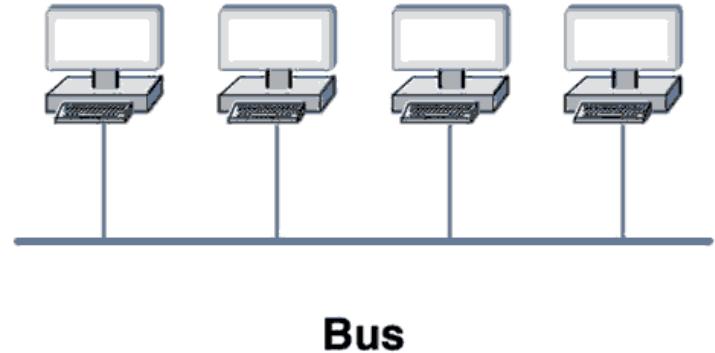
- 各节点通过通信介质连成封闭的环
- 两个邻接的节点之间通过点到点连接
- 非邻接节点之间通信要经过路径上的其它节点
- 不需要是圆形



总线拓扑 (Bus Topology)

- 总线型结构

- 所有的站点共享一条数据通道
- 总线拓扑网络通常由一个单一的长电缆连接到计算机上。
- 任何连接到总线上的计算机都可以通过电缆发送信号，所有的计算机都接收信号。
- 连接到总线网络的计算机必须协调，以确保在任何时候只有一台计算机发送信号，否则将造成混乱的结果。



拓扑悖论

- 10BaseT布线形成以集线器为中心的星型拓扑结构。
- 双绞线以太网像总线结构一样工作，因为所有计算机共享一个通信介质。
 - 在物理上，双绞线以太网采用星型拓扑结构。
 - 在逻辑上，双绞线以太网像总线工作，称为星形总线。

三种拓扑的优缺点

拓扑	优点	缺点
星型	容易在网络中增加新的站点 一个节点断掉不会影响其它节点 数据的安全性和优先级容易控制 易实现网络监控	中心节点的故障会引起整个网络瘫痪
环型	结构简单，容易安装 容易监控通/断情况 节省资源	容量有限 网络建成后，难以增加新的站点 一个节点产生异常情况，会影响其它节点通信
总线	安装简单方便 需要铺设的电缆最短，成本低 某个站点的故障一般不会影响整个网络	介质的故障会导致网络瘫痪 线网安全性低 监控比较困难 增加新站点也不如星型网容易

内容纲要

1

网络拓扑

2

以太网的编码和协议

3

以太网的工作机制

4

其它网络

5

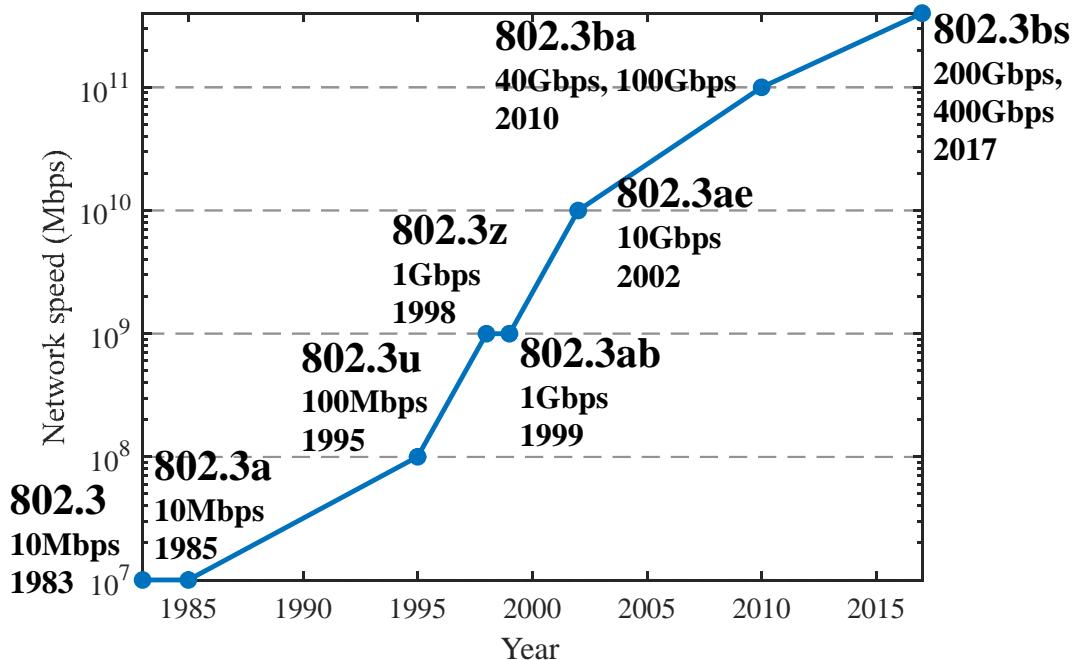
无线网络技术



以太网 (Ethernet)

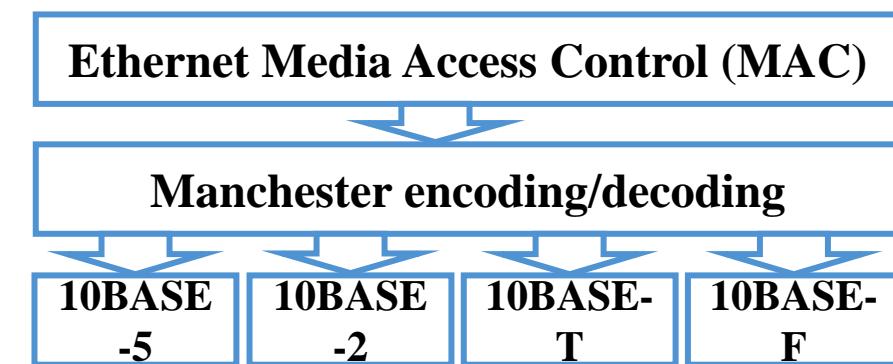
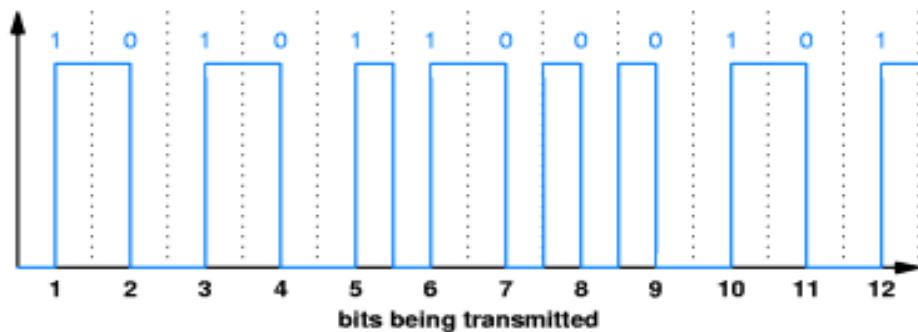
- 以太网发展历史

- 以太网是由Xerox公司在70年代早期发明的。
- 以太网的标准：现在由IEEE制订并维护



曼彻斯特编码（Manchester Encoding）

- 以太网标准规定帧使用曼彻斯特编码发送。
 - 硬件在边缘触发：使用上升和下降边缘编码数据
 - 下降表示0，上升表示1
 - 接收器必须知道每个时隙何时开始和何时结束。
 - 使用前同步：前同步码由64交替的1和0组成，在帧之前发送。



传统以太网

• 以太网版本

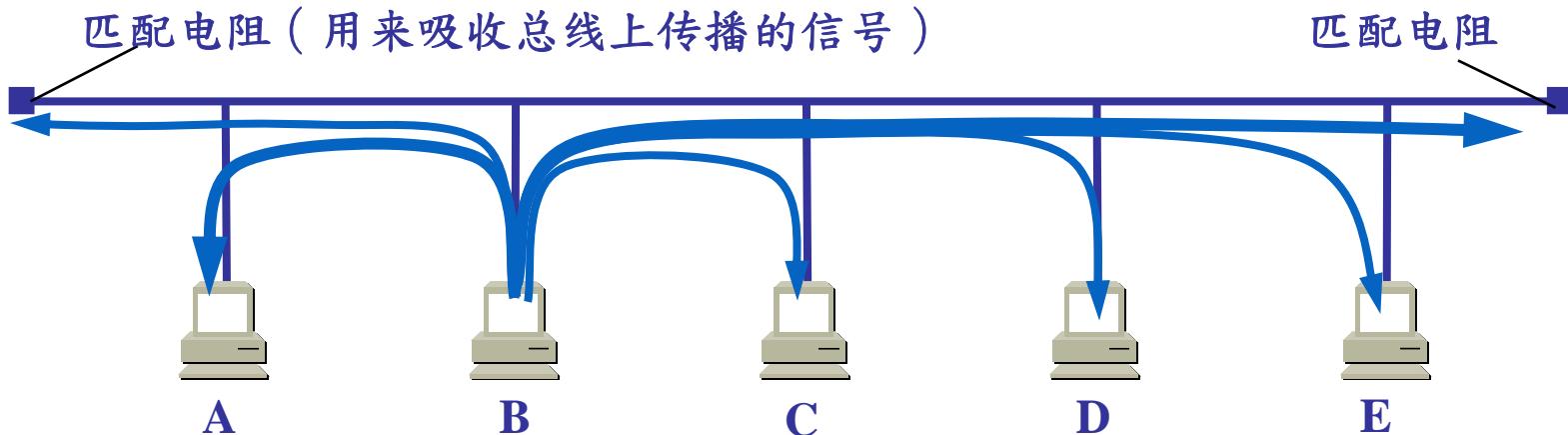
名称	数据速率	线缆类型
双绞线以太网（传统以太网）	10 Mbps	5类
快速以太网	100 Mbps	超5类（5E）
千兆以太网	1 Gbps	6类，光纤
万兆以太网	10 Gbps	仅光纤

• 双工模式

- 传统以太网仅工作在半双工模式
- 万兆以太网仅工作在全双工模式

在传统以太网上共享

- 传统以太网可以视为总线型网络
 - 发送方发送的信号从发送者向共享电缆的两端传播。
 - 总线上每个工作计算机都能检测到 B 发送的数据信号。
 - 当一台计算机发送信号，其它计算机必须等待。

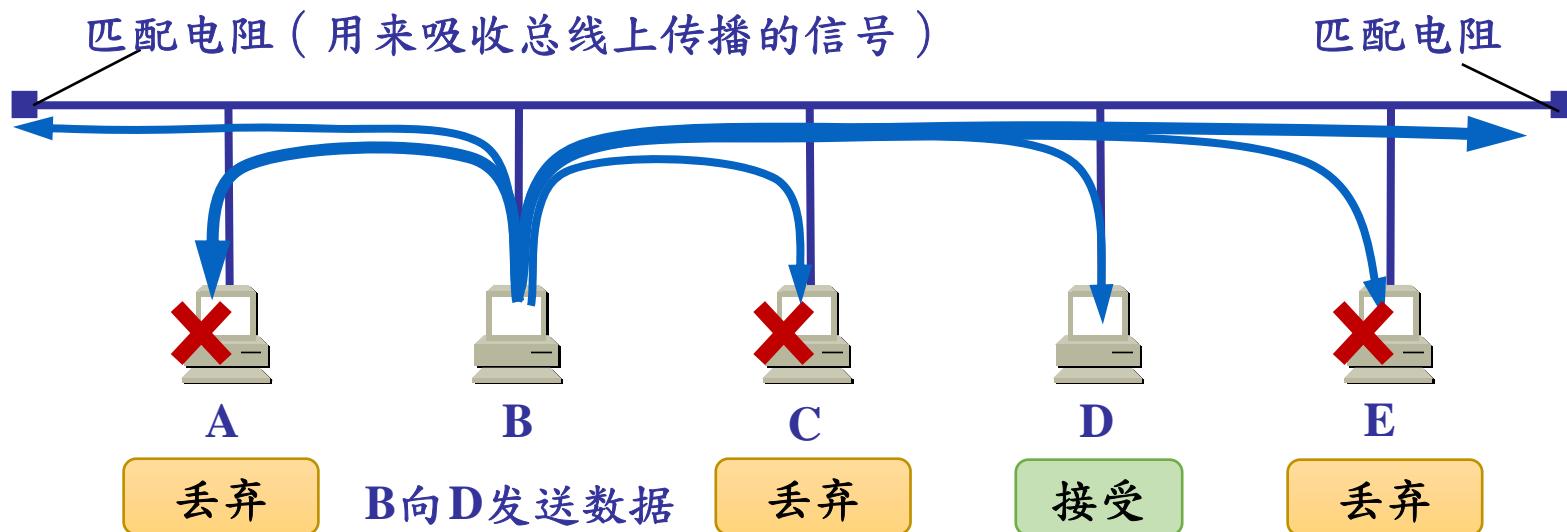


在传统以太网上共享

- 传统以太网可以视为总线型网络

- 具有广播特性的总线上实现了一对一的通信。

- 只有 D 的地址与帧目的地址一致，因此只有 D 才接收这个数据帧。
 - 其它计算机（A, C和E）都检测到目的地址不一致，因此丢弃该帧。



不可靠的交付

- 为了通信的简便，以太网采取了两种重要的措施
 - 无连接：即不必先建立连接即可直接发送数据。
 - 发送的帧不编号，不要确认。
 - 理由：局域网信道质量很好，因信道质量产生差错的概率是很小的。
- 以太网提供的服务是不可靠的尽力而为的交付。
 - 目的站收到有差错的帧时就丢弃。差错纠正由高层来决定。
 - 如果高层发现丢失了一些数据而进行重传，但以太网并不知道这是一个重传的帧，而是当作一个新的数据帧来发送。

争用型的介质访问控制协议

- 带冲突检测的载波监听多路访问技术（ Carrier Sense Multiple Access with Collision Detection ）
 - 载波监听（ Carrier Sense ）
 - 每个站点发送前，先检测总线是否有其它站点在发送数据。
 - 多址接入（ Multiple Access ）
 - 许多计算机以多点接入方式连接在总线上。
 - 冲突检测（ Collision Detection ）
 - 两个站点发送的信号到达电缆同一点，彼此干扰，即冲突。



内容纲要

1

网络拓扑

2

以太网的编码和协议

3

以太网的工作机制

4

其它网络

5

无线网络技术



半双工以太网的工作机制

- 载波监听（先听后发）

- 以太网要求每个站点监听电缆，检测是否已有一个传输正在处理中。它阻止了最明显的冲突问题。

- 当帧发送时，发送方发来对位进行编码的电信号
 - 当没有计算机在发送帧时，以太网不包含电信号



半双工以太网的工作机制

- 冲突检测（边发边听，冲突停发）
 - 当两个站点同时载波侦听到电缆空闲，仍可能发生冲突。
 - 为解决冲突，每个站点在发送过程中监视电缆，如果电缆信号与本站发送的信号不符即认定为冲突，立即终止发送。
 - 两个站点发送的信号到达电缆同一点，彼此干扰，即冲突。
 - 当几个站同时在总线上发送数据时，总线上的信号电压摆动值将会增大（互相叠加）。
 - 计算机边发送数据边检测信道上的信号电压大小。当站检测到的信号电压摆动值超过一定的门限值时，认为总线上至少有两个站同时发送数据，表明产生了碰撞。



半双工以太网的工作机制

- 二进制指数退避（随机延迟后重发）

- 冲突发生后，需要从冲突中恢复
- 标准规定基本退避时间 d ，每个检测到冲突的站点随机选一个介于 $(0,2]$ 的整数 r ，随机延迟 rd 时间
- 如果连续遇到第 n 次冲突，则选择介于 $(0,2^{n-1}]$ 随机数 r ，延迟 rd 时间

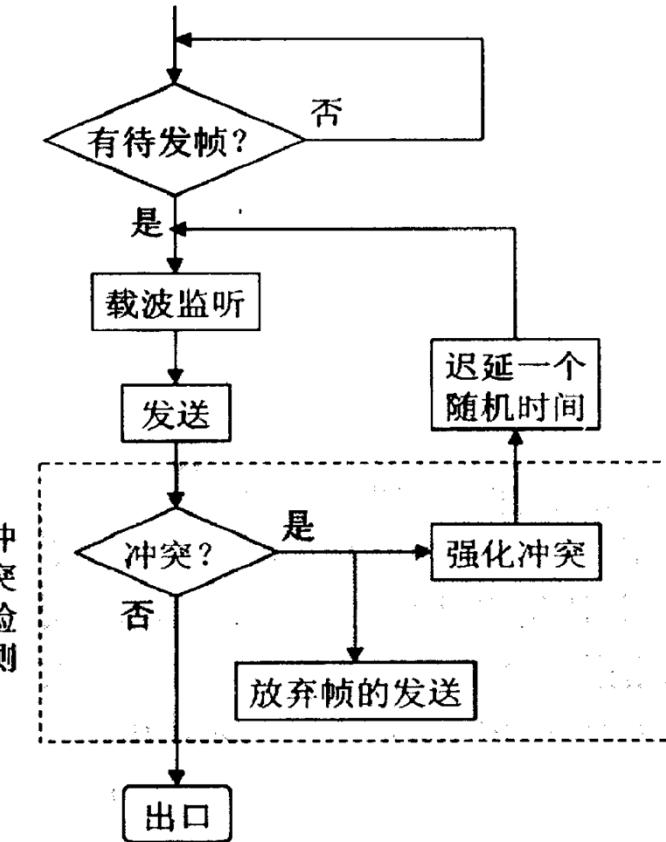
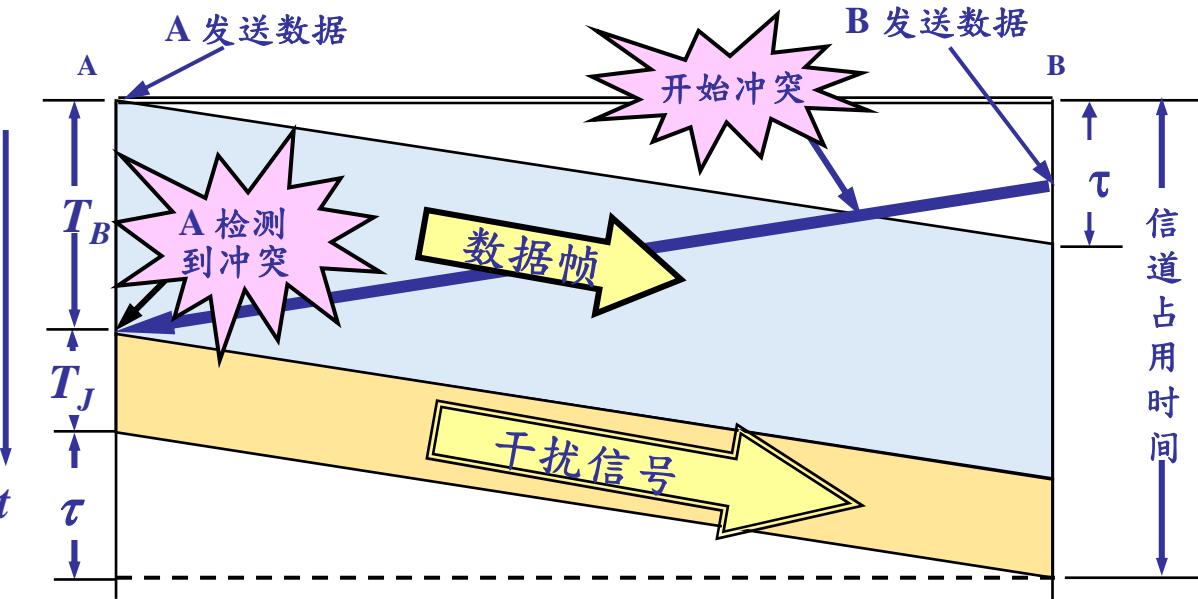


图 5-12 CSMA/CD 的流程图



强化碰撞

- 当发送数据的站一旦发现发生了碰撞时
 - 立即停止发送数据；
 - 再继续发送若干位的人为干扰信号（jamming signal），以便让所有用户都知道现在已经发生了碰撞。



冲突的原因和结果

- 原因：当某站监听到总线空闲时，总线并非真正空闲
 - A向B发出的信息，要经过一定的时间后才能传送到B。
 - B若在A发出的信息到达前发送帧，必然发生碰撞。
 - 因为这时B的载波监听检测不到A所发送的信息
- 结果：是两个帧都变得无用。
 - 站点在发送数据之后的一小段时间内，仍可能遭遇碰撞。
 - 该不确定性使传统以太网的平均通信量远小于最高数据率。
 - 只有30%~40%的效率。



冲突的发现

- 在发生碰撞时，总线上传输的信号产生了严重的失真，无法从中恢复出有用的信息来。
 - 正在发送数据的站一旦发现总线上出现了碰撞，立即停止发送，以免浪费网络资源，等待一段随机时间后再次发送。
- 最先发送帧的站，在发送后至多经过时间 2τ （两倍的端到端往返时延）即可知道发送的帧是否遭受了碰撞。
 - 以太网的端到端往返时延 2τ 称为争用期，或碰撞窗口。
- 经过争用期还没检测到碰撞，才能肯定这次不会碰撞。



二进制指数退避：具体算法

- 意味着以太网能在冲突后迅速恢复，降低竞争性

发生碰撞的站在停止发送数据后，要推迟（退避）一个随机时间才能再发送数据。

基本退避时间取为争用期 2τ 。

从整数集合 $[0, 1, \dots, (2^k - 1)]$ 中随机地取出一个数，记为 r 。重传所需的时延就是 r 倍的基本退避时间。

参数 k 按下面的公式计算：

$$k = \min[\text{重传次数}, 10]$$

当重传达 16 次仍不能成功时即丢弃该帧，并向高层报告。



传统以太网最短有效帧长度

- 传统以太网取 $51.2 \mu\text{s}$ 为争用期的长度。
 - 10 Mbps 以太网，在争用期内可发送 512 bit，即 64 字节。
- 以太网规定了最短有效帧长为 64 字节
 - 若前 64 字节没有发生冲突，则后续的数据不会发生冲突。
 - 检测到冲突立即中止发送，已发送的数据一定小于 64 字节。
 - 凡长度小于 64 字节的帧都是由于冲突而异常中止的无效帧。



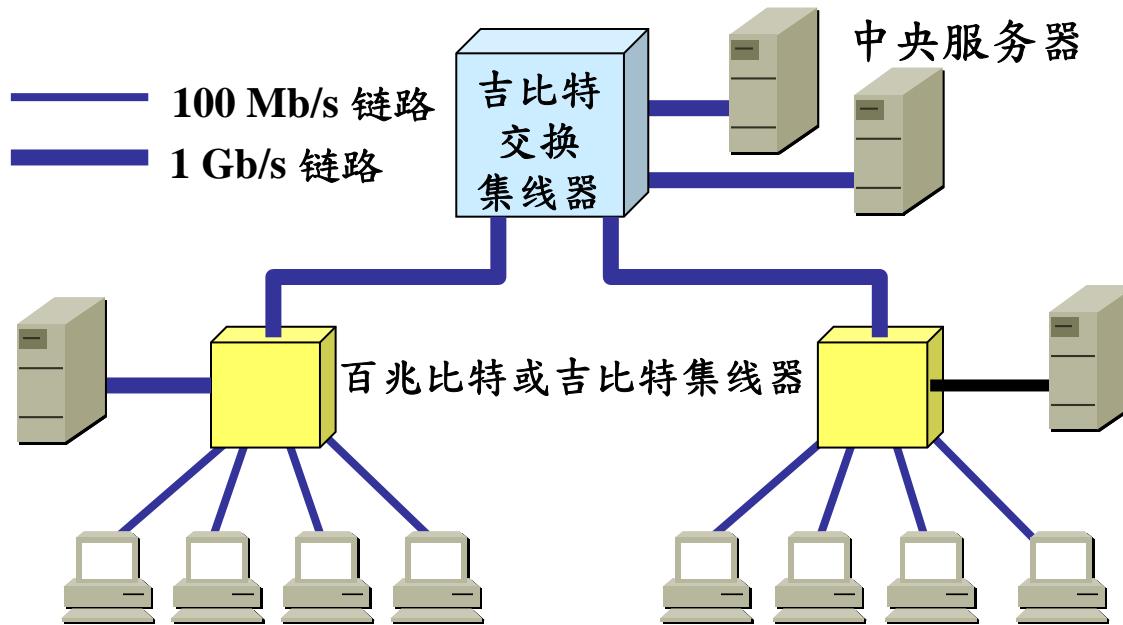
快速以太网

- 半双工模式下的快速以太网在10BaseT基础上
 - 保持最短帧长不变，但将网段最大电缆长度减小到 100 m。
 - 帧间时间间隔从原来的 $9.6 \mu\text{s}$ 改为 $0.96 \mu\text{s}$ 。
- 全双工模式的快速以太网
 - 每个网段只连接2个设备，因而不会发生碰撞
 - 可以是网卡或交换机端口，不能是集线器端口
 - 全双工模式不受碰撞范围限制，大大延伸了网络范围。



千兆以太网

- 802.3 协议的帧格式兼容1Gbps全双工和半双工
 - 半双工下使用 CSMA/CD 协议，全双工不需要CSMA/CD 。
- 10Gbps以太网只使用光纤，只有全双工方式。



使用高速以太网进行宽带接入

- 以太网从 10 Mb/s 到 100 Gb/s 的演进证明了以太网是：
 - 可扩展的、灵活的（多种传输媒体、全/半双工、共享/交换）、易于安装、稳健性好。
 - 以太网所覆盖的地理范围也扩展到了城域网和广域网，因此现在人们正在尝试使用以太网进行宽带接入。
- 以太网接入的重要特点是它可提供双向的宽带通信，并且可根据用户对带宽的需求灵活地进行带宽升级。
- 采用以太网接入可实现端到端的传输，中间不需要帧格式的转换，提高了传输效率和降低了传输成本。



内容纲要

1

网络拓扑

2

以太网的编码和协议

3

以太网的工作机制

4

其它网络

5

无线网络技术



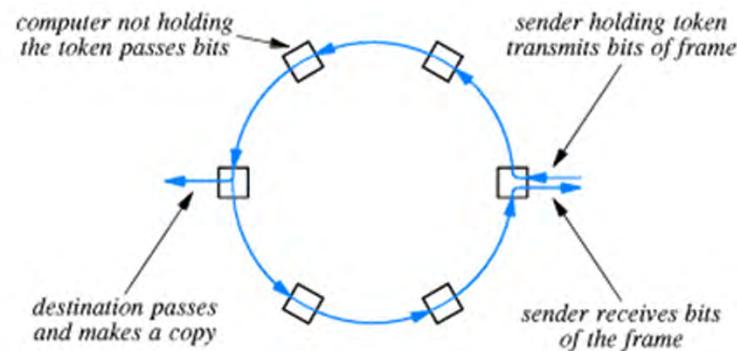
LocalTalk

- Apple公司的总线网络
- 应用了CSMA/CA技术处理介质访问冲突
- 成本低，安装简单，适合苹果机的局域网
- 带宽受限，距离受限，不适合其它品牌设备互联
 - 带宽：230.4kbps
- 在2009年Mac OS X v10.6已不支持



IBM令牌环（ Token Ring ）

- 使用令牌传递网络被称为令牌环网或令牌环。
 - 一台计算机需要发送数据时，必须等待权限才可访问网络。
 - 协调使用一个特殊的，保留的消息称为令牌（ token ），它是不同于正常的数据帧的位模式。
- 令牌环在LAN中过时，用于光缆骨干网高可靠性环境。
- 令牌集中管理，避免了冲突。
- 单点出现故障可造成环网瘫痪。



光纤分布数据互连（ FDDI ）

- **FDDI (Fiber Distributed Data Interconnect)**

- 令牌环技术，光纤互连， 100 Mbps 速率。
- 成本高，越来越少用；两个局域网可能不同标准。

- **FDDI自恢复**

- 使用反向旋转的环，可检测一个故障，并自动恢复。
- 当发生故障时，打破了环，相邻站自动重新配置，使用第二个环，避免失败。

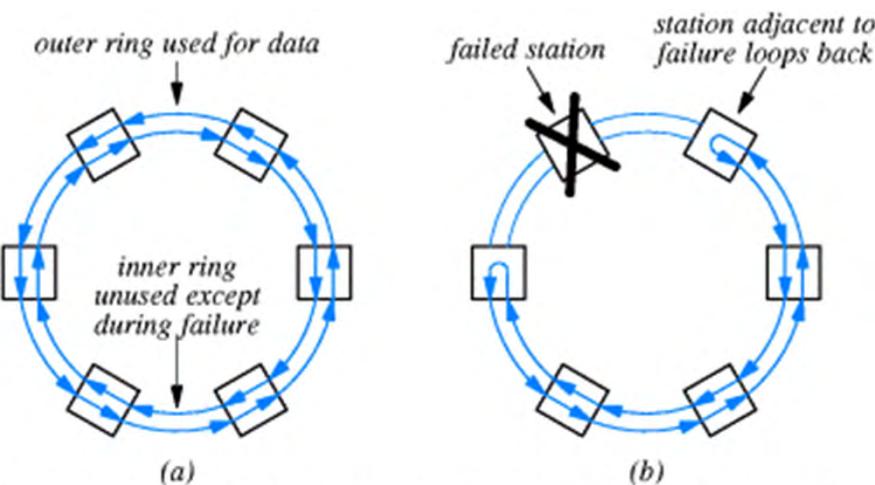
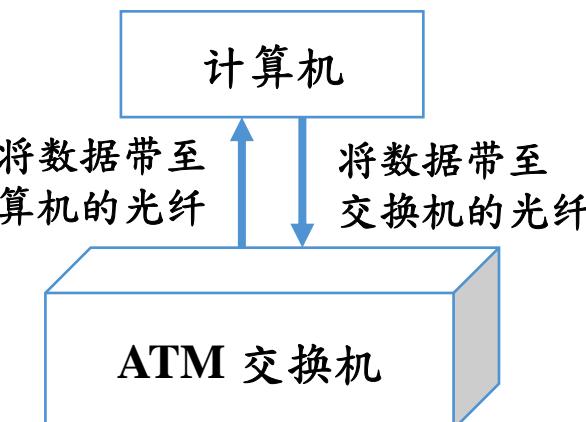


Figure 8.10 (a) An FDDI network with arrows showing the directions that data flows, and (b) the same network after a station has failed. Normally, data travels in one direction. After a station fails, adjacent stations use the reverse path to form a closed ring.

异步传输模式（ATM）

- 异步传输模式（Asynchronous Transfer Mode，ATM）

- 建立在电路交换和分组交换的基础上的一种交换技术。
- ATM网络通过ATM交换机进行互连，是星型拓扑结构。
- 面向连接，没有冲突，抖动小，性能较好。
- ATM交换机工作在155Mbps的或更快的速度。
- ATM交换机通常采用光纤，成本较高。



内容纲要

1

网络拓扑

2

以太网的编码和协议

3

以太网的工作机制

4

其它网络

5

无线网络技术



无线网络分类

- 无线网络
 - 旧时王谢堂前燕，飞入寻常百姓家

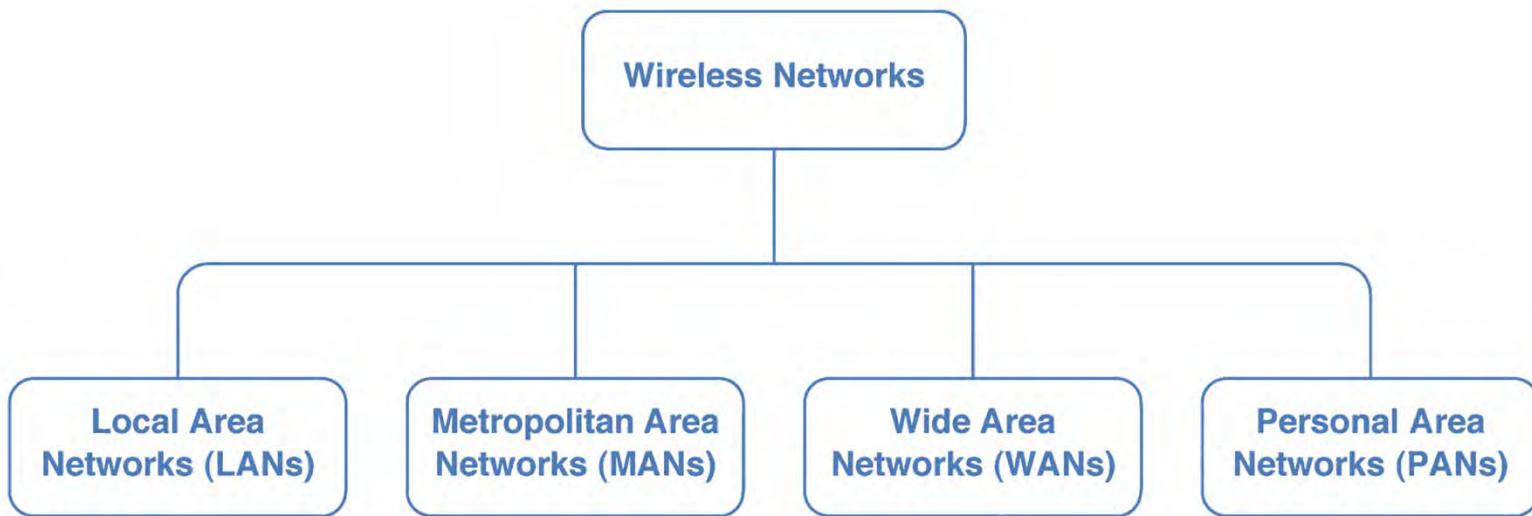


Figure 16.1 A taxonomy of wireless networking technologies.

Copyright © 2009 Pearson Prentice Hall, Inc.

无线LAN与Wi-Fi：802.11abgn

- **Wi-Fi**: Wireless Fidelity (无线保真度，非官方)
 - 802.11 bg : 室内50平方米，室外140平方米。802.11n两倍。

IEEE Standard	Frequency Band	Data Rate	Modulation Technique	Multiplexing Technique
original 802.11	2.4 GHz	1 or 2 Mbps	FSK	DSSS
	2.4 GHz	1 or 2 Mbps	FSK	FHSS
	InfraRed	1 or 2 Mbps	PPM	±none±
802.11a	5.725 GHz	6 to 54 Mbps	PSK or QAM	OFDM
802.11b	2.4 GHz	5.5 and 11 Mbps	PSK	DSSS
802.11g	2.4 GHz	22 and 54 Mbps	various	OFDM

Figure 16.4 Key wireless standards certified by the Wi-Fi Alliance.

Copyright © 2009 Pearson Prentice Hall, Inc.

无线LAN组成

- 无线网的3个构件
 - 接入点（基站）
 - 互联机构（如：交换机、路由器）
 - 无线主机（无线节点或无线站点）
- 两种接入类型
 - 点对点模式（Ad hoc）
 - 基础结构型（infrastructure）



802.11帧格式与协调

- 有线和无线的区别：盲区、重叠区域

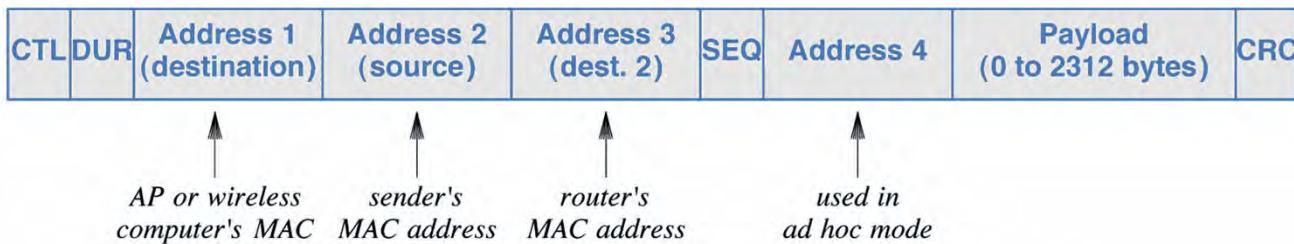


Figure 16.9 The frame format used with an 802.11 wireless LAN.

Copyright © 2009 Pearson Prentice Hall, Inc.

- 接入点 —— AP

- 接入点负责
- 无线计算机负责（成本低）

无线网络技术

- PAN技术
 - 蓝牙（1Mbps，2.4GHz）
- WAN
 - 蜂窝系统、基站集群
- 更新换代

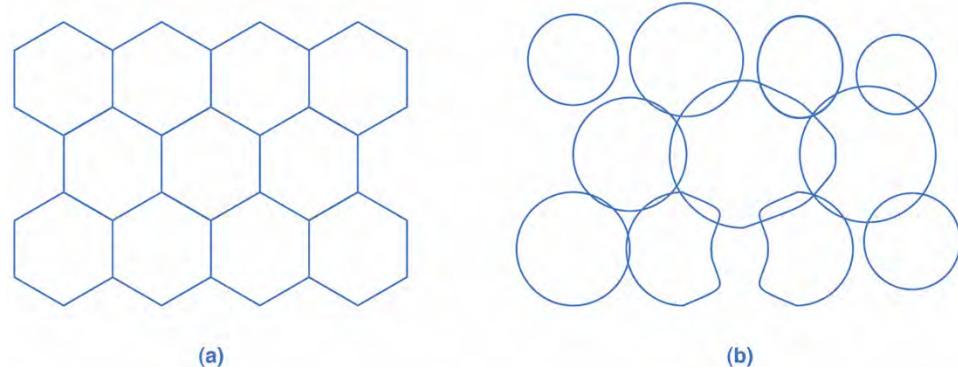


Figure 16.15 Illustration of (a) an idealized cellular coverage, and (b) a realistic version with overlaps and gaps.

Copyright © 2009 Pearson Prentice Hall, Inc.

1G	2G	2.5G	3G	3.5G	4G
模拟信号 语音	数字信号 语音	部分3G 业务	高速数据服务	实时多媒体业务的支持、 含WiFi和卫星	

无线网络技术

- GPS卫星(Global Positioning System，全球定位系统)

- 精确度：2~20m（军用更高）
- 24颗卫星绕行地球轨道
- 6个轨道平面
- 网络时间同步
- 美军的产品
- 中国北斗

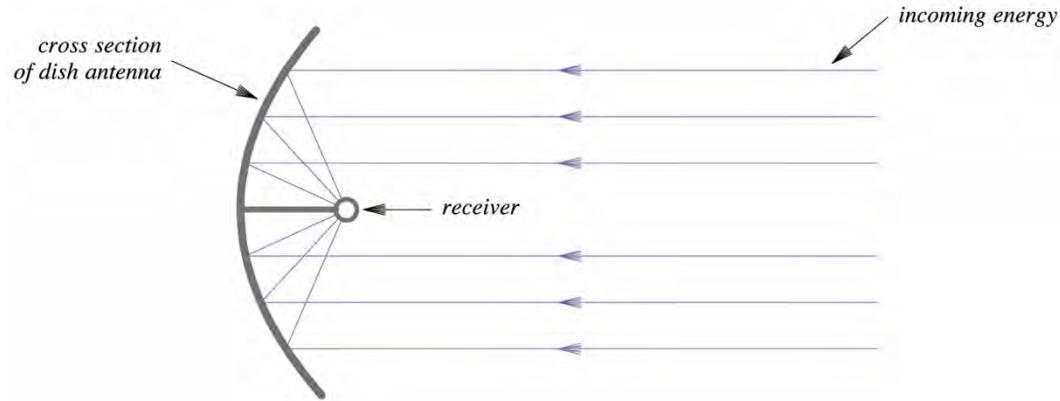


Figure 16.20 Illustration of reflection by a parabolic dish antenna.
Copyright © 2009 Pearson Prentice Hall, Inc.

802.11无线LAN与CSMA/CA

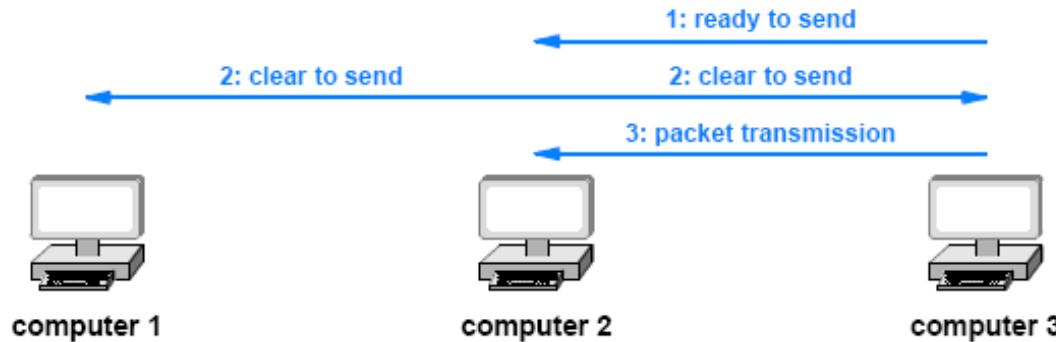
- IEEE802.11标准定义了在11Mbps使用2.4 GHz频段的频率使用无线局域网。
- 蓝牙（ Bluetooth ）标准规定了用于短距离的无线局域网技术
- 无线局域网硬件使用天线通过空气，让其他电脑接收到广播的射频（ RF ）信号。

CSMA/CA

- CSMA/CD 在无线LAN中无法工作
 - 接受者可能无法接受信号，不能够检测到载波



- 无线局域网使用带碰撞避免的载波侦听多址接入
(CSMA with Collision Avoidance , CSMA / CA)



无线局域网技术特点

- 基本技术特点

- 通过一台PC机器加上一块无线网络接口卡构成将多个无线的接入站聚合到有线的网络上
- 采用了新的协议CSMA/CA进行冲突避免
 - 只有当客户端收到网络上返回的ACK信号后才确认送出的数据已经正确到达目的
- 两个节点通过中间节点进行数据通信，要由中间节点通过控制命令进行统一调度控制，增加了额外的网络负担



计算机网络

Computer Network

6

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

7

局域网布线与扩展

理论课程

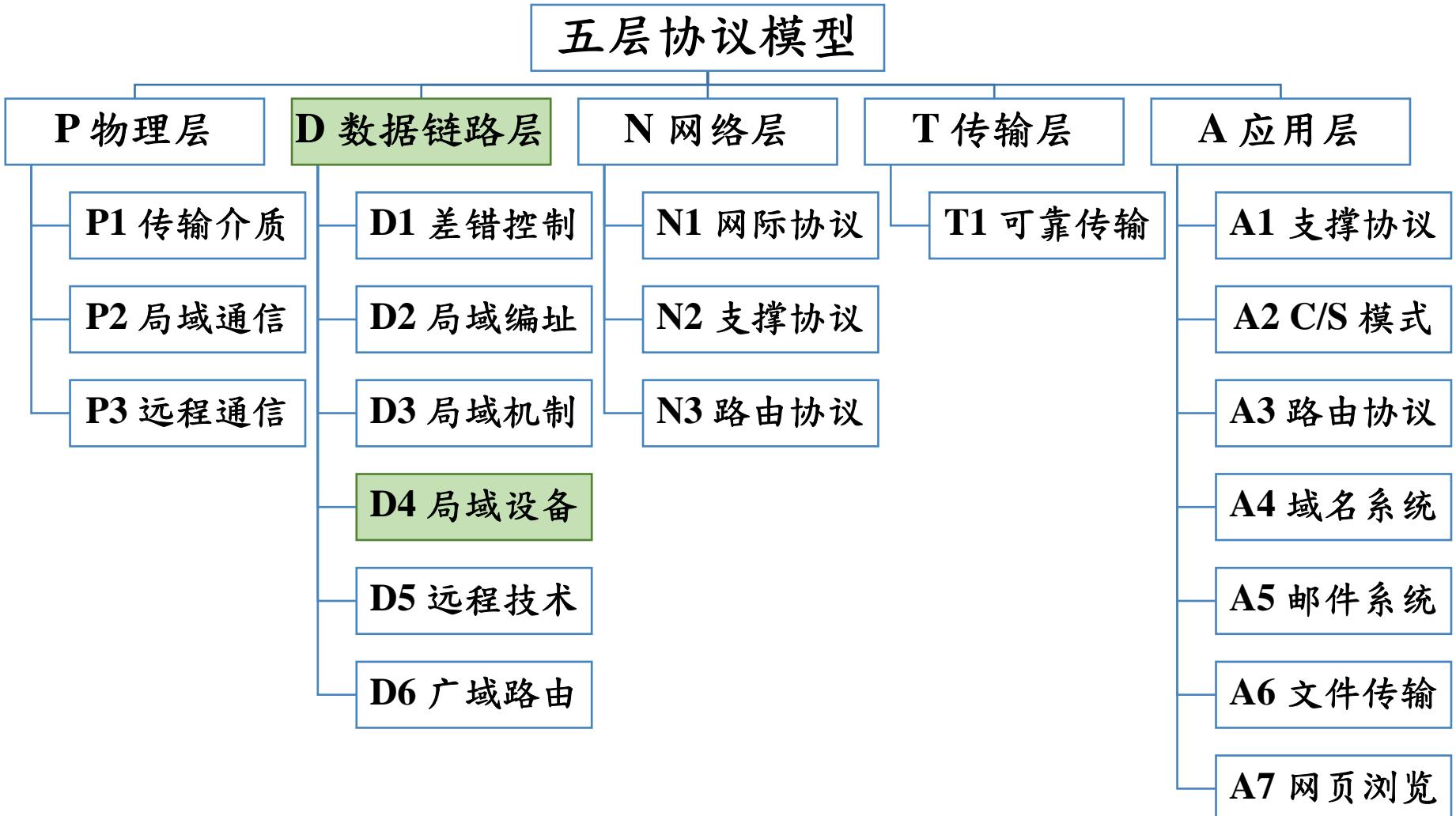


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 布线：粗缆和细缆，介质和速率
 - 各种标准：10/100/1000 Base 5/2/T/F
- 扩展局域网
 - 物理层：中继器、集线器（不隔离冲突域）
 - 优点、局限性、五四三二一原则
 - 数据链路层：网桥、交换机（只隔离冲突域）
 - 网桥转发表、网桥环、交换机内部结构
 - 网络层：路由器（隔离冲突域和广播域）

对应课本章节

- PART III Packet Switching And Network Technologies
 - Chapter 15 Wired LAN Technology (Ethernet And 802.3)
 - Chapter 17 LAN Extensions: Fiber Modems, Repeaters, Bridges, and Switches



内容纲要

1

网络布线

2

中继器

3

集线器

4

网桥

5

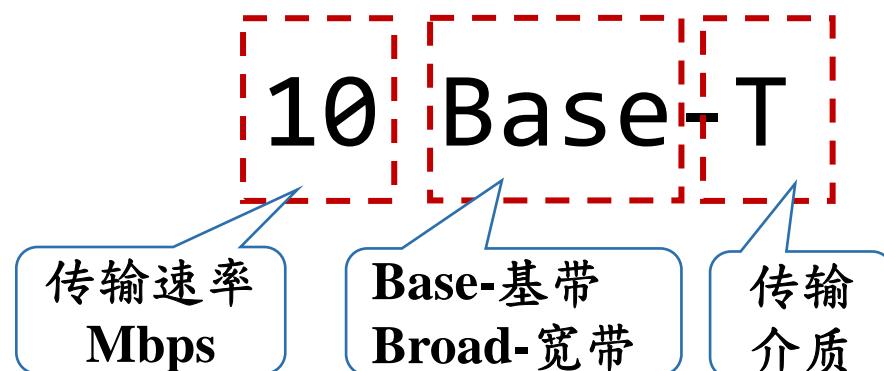
交换机



网卡和网络连接的类型

- 网卡和网络之间使用的连接类型取决于网络技术。
- 以太网的4种不同的物理层

名称	介质	最大长度/段	工作站数目/段	特点
10BASE5	粗同轴电缆	500m	100	适合于主干
10BASE2	细同轴电缆	200m	30	低廉的网络
10BASE-T	双绞线	100m	1024	易于安装和维护
10BASE-F	光纤	2000m	1024	远程工作站连接



Ethernet的三种重要布线

- Ethernet的三种重要布线

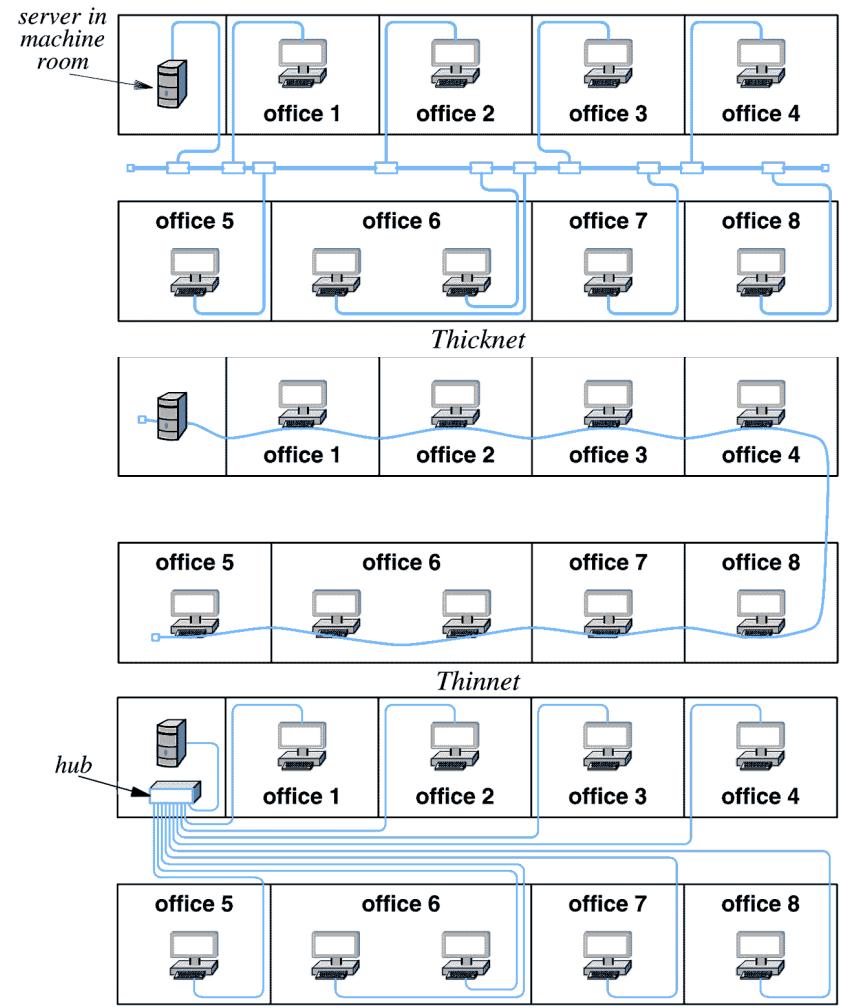
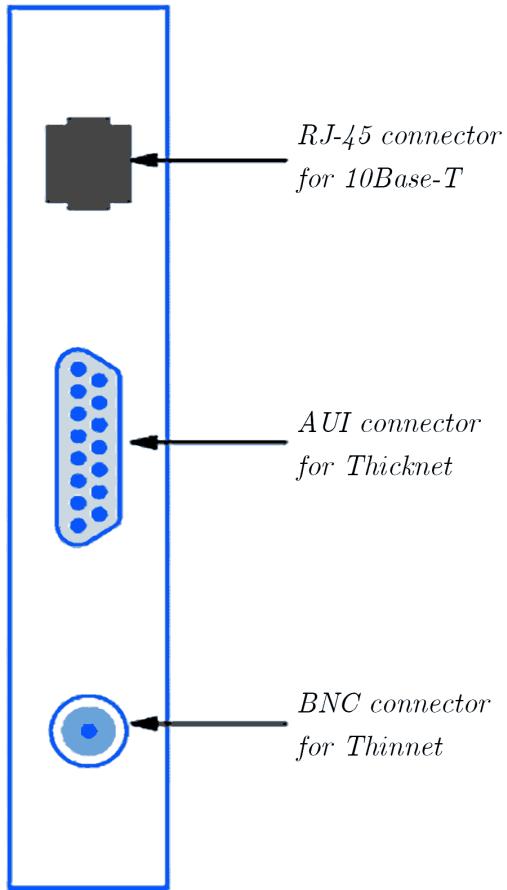


Figure 15.7 Illustration of various LAN wiring schemes that have been used in an office building.

原始粗缆以太网布线（10Base5）

- **10BASE5**：原始粗缆以太网布线（ Thick Wiring ）
- 总线拓扑：共享介质为一个粗的同轴电缆。
 - 连接到网络的每个计算机都需要收发器（ transceiver ）
 - 连接网卡和收发器的电缆称为连接单元接口（ Attachment Unit Interface ， AUI ）。
 - 多路复用器允许多台计算机连接到一个单一的收发器。
- 截至2003年，IEEE 802.3对新

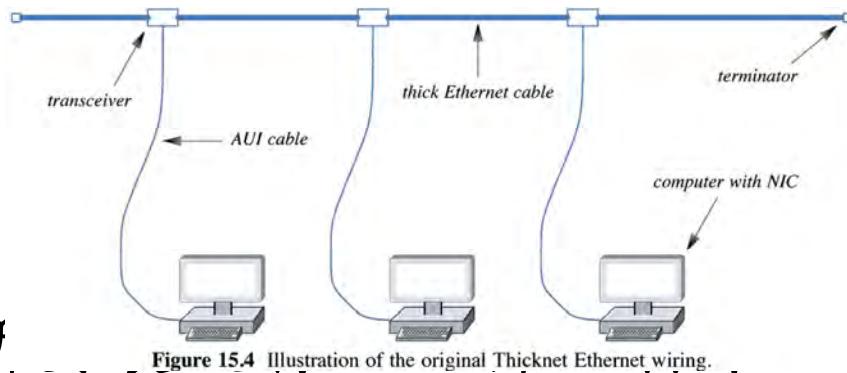
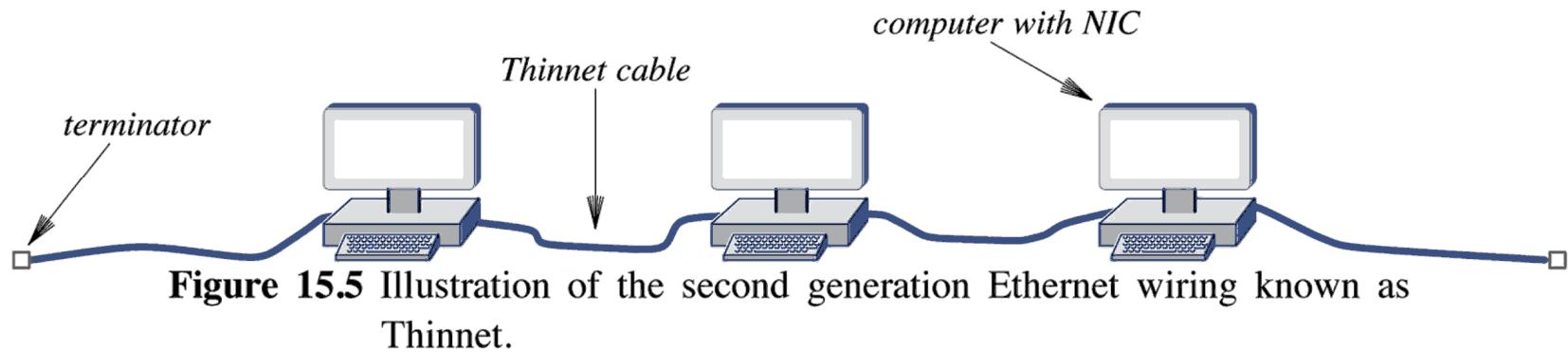


Figure 15.4 Illustration of the original Thicknet Ethernet wiring.

细缆以太网布线（10Base2）

- 10Base2：细缆以太网布线（Thin Wiring）
- 总线拓扑
 - 直接附加到使用BNC连接器的计算机背面，不用AUI。
 - 一般成本较低，无需外部收发器。
- 截至2010年，IEEE 802.3对新安装设备已放弃该标准。



双绞线以太网（10Base-T）

- 10Base-T：双绞线以太网（Twisted Pair Ethernet）

- 星型拓扑

- 使用Ethernet集线器（Hub）

- 计算机和集线器之间使用带RJ-45连接器的双绞线布线。

- T568B颜色顺序

- 白橙、橙、白绿、
蓝、白蓝、绿、
白棕、棕

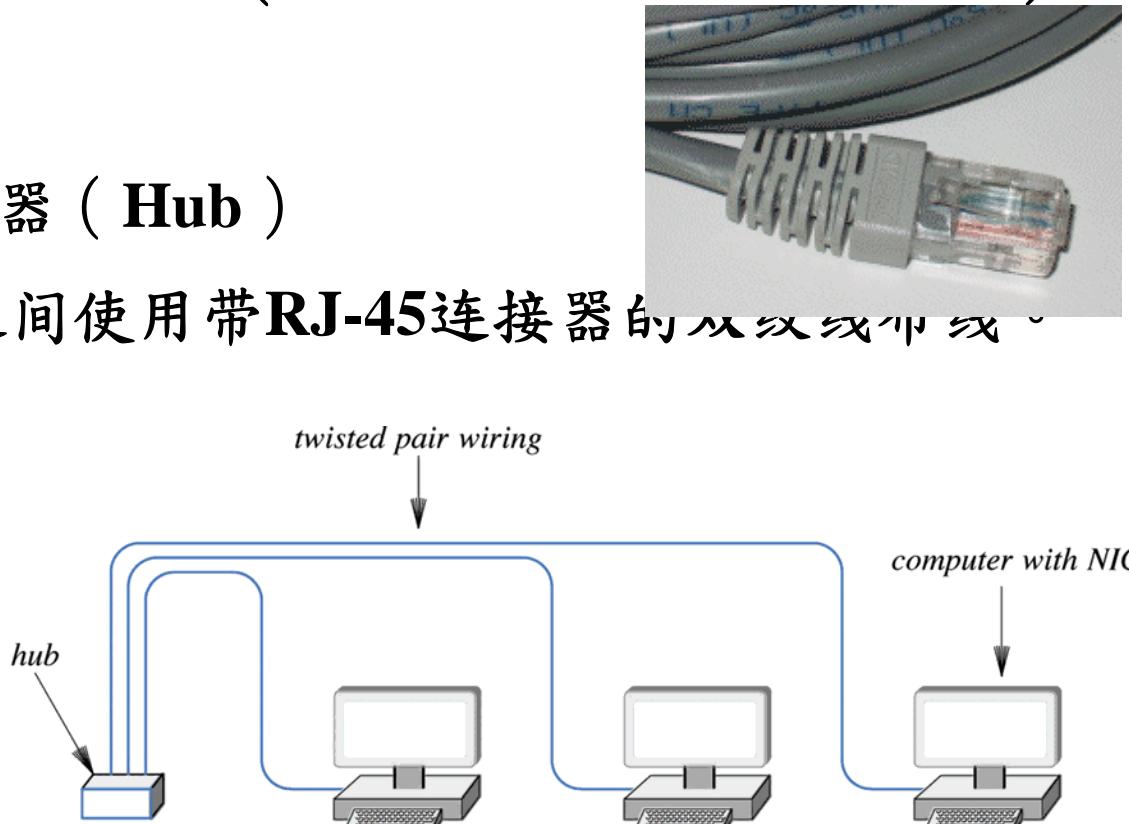


Figure 15.6 Illustration of the third generation Ethernet using twisted pair wiring.

光纤以太网（10Base-F）

- 10Base-F：光纤以太网（Ethernet Over Fiber）
 - 物理：光纤



网卡和布线方案

- 许多网络接口支持多种布线方案，同一时刻只用一种
 - 为了使人们改变布线方案而不改变接口的硬件
- 网络技术独立于布线方案
 - 任何网络技术可以使用多个布线方案，且逻辑拓扑可能不同的物理拓扑。
 - 原始LocalTalk的布线方案使用收发器。
 - 集线器布线通常与LocalTalk一起使用。
 - 集线器布线通常与IBM令牌环一起使用。



光纤调制解调器

- 光纤调制解调器和光纤提供计算机和远程LAN的连接。
 - 光纤的延迟低且带宽高，可跨越几公里的距离正确操作。
- 光纤调制解调器工作在物理层。

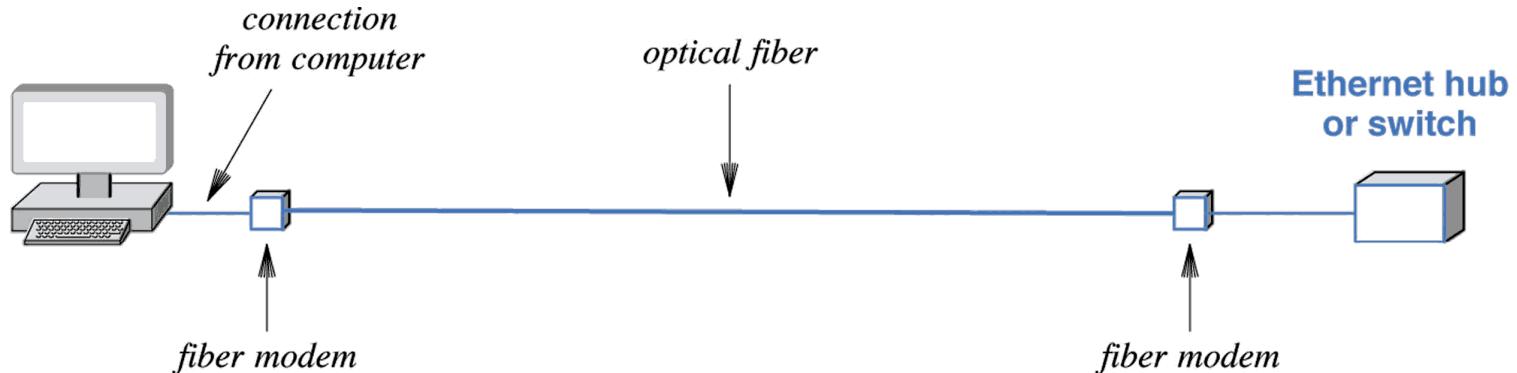


Figure 17.1 Illustration of fiber modems used to provide a connection between a computer and a remote Ethernet.

内容纲要

1

网络布线

2

中继器

3

集线器

4

网桥

5

交换机



中继器

- 中继器

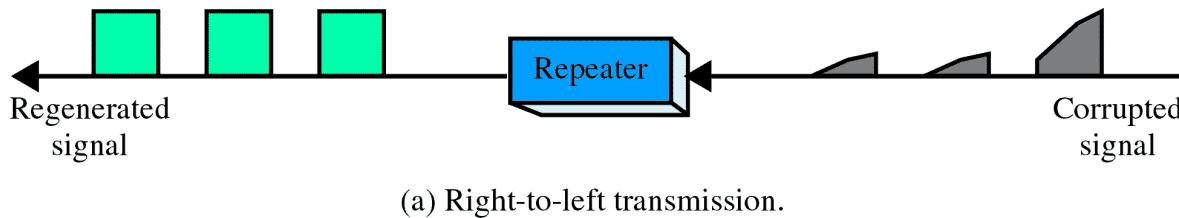


- 中继器（Repeater）是工作在物理层上的连接设备。
 - 中继器不理解帧格式，也没有物理地址。
 - 中继器不区分对应于有效帧和其他电信号的信号。
 - 适用于完全相同的两类网络的互连
 - 扩展局域网上的计算机不知道中继器是否将它们分开。

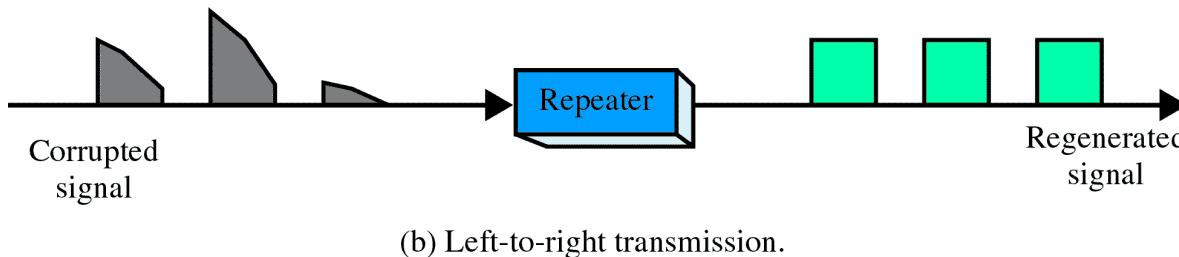
中继器 (Repeaters)

- 主要功能

- 通过对数据信号的重新发送或者转发，来扩大网络传输的距离。中继器连接两个称为段（ segments ）的以太网电缆。



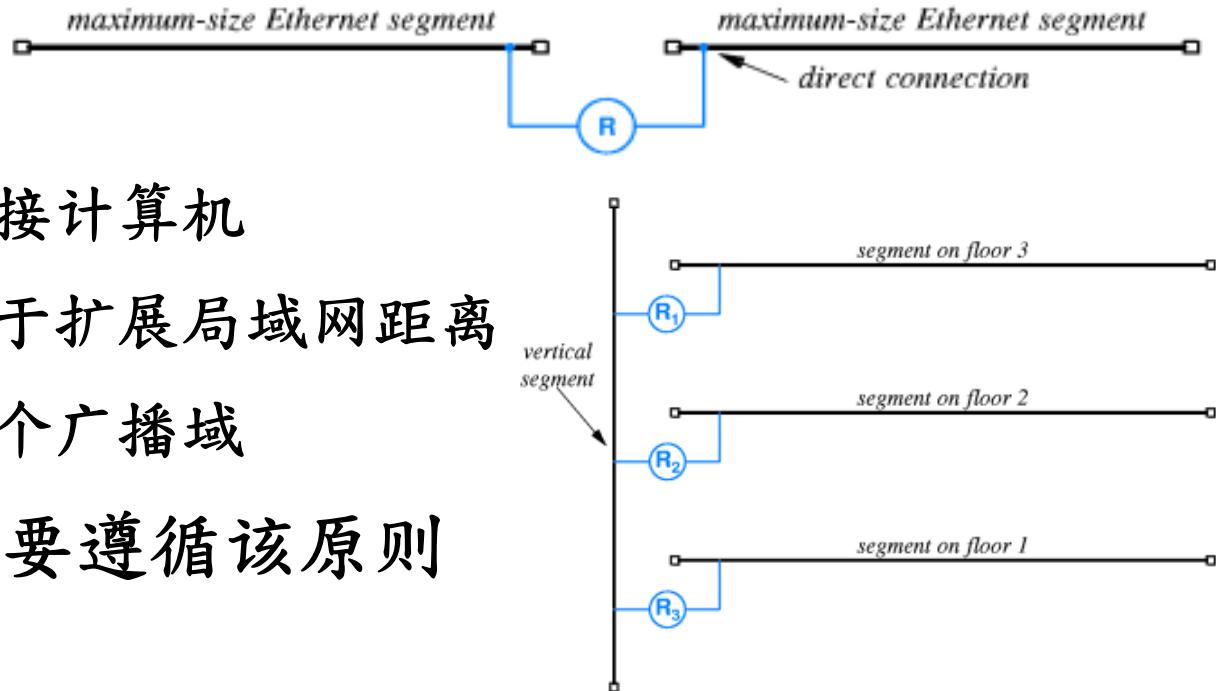
(a) Right-to-left transmission.



(b) Left-to-right transmission.

网络设计的“五四三二一”原则

- 网线传输距离是有限的，如果节点间的距离太远，需要使用中继器来放大信号继续传输。最多只能有：
 - 5个网段
 - 4个中继器
 - 3个网段可以连接计算机
 - 2个网段只能用于扩展局域网距离
 - 它们共同处于1个广播域
- 用交换机组网也要遵循该原则



内容纲要

1

网络布线

2

中继器

3

集线器

4

网桥

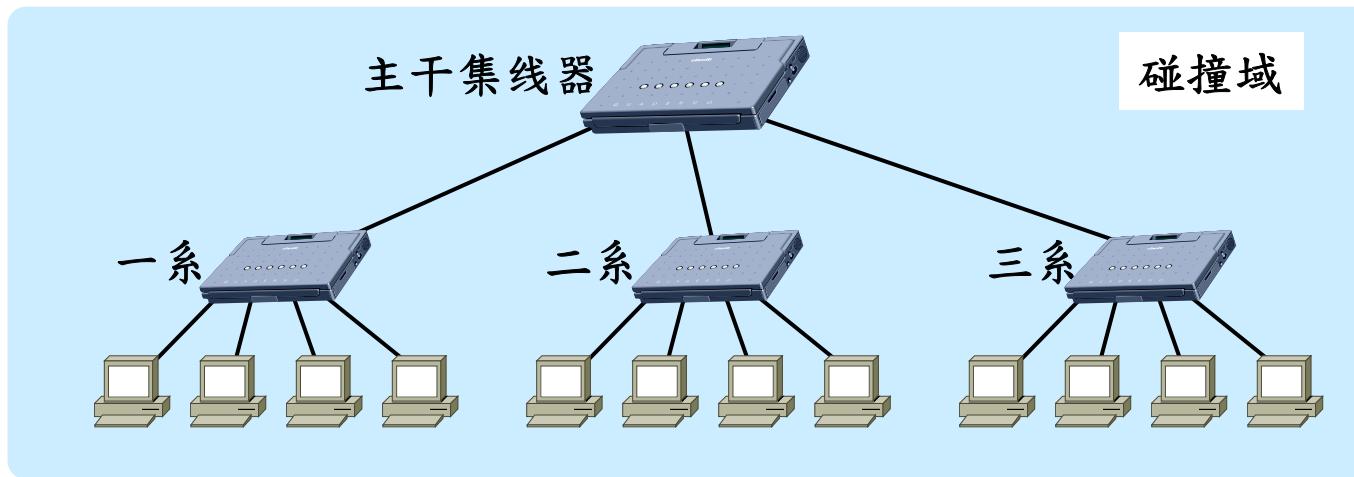
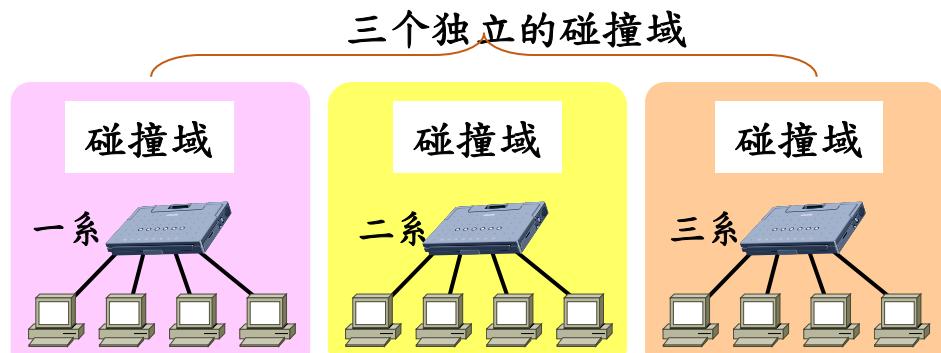
5

交换机



集线器 (Hub)

- 用多个集线器可连成更大的局域网
- 但都在一个碰撞域中
- 半双工：共享式以太网



集线器 (Hub)

- 优点

- 使原来属于不同碰撞域的局域网上的计算机能够进行跨碰撞域的通信。
 - 扩大了局域网覆盖的地理范围。

- 缺点

- 碰撞域增大了，但总的吞吐量并未提高。
 - 如果不同的碰撞域使用不同的数据率，那么就不能用集线器将它们互连起来。

- 集线器工作在物理层。

内容纲要

1

网络布线

2

中继器

3

集线器

4

网桥

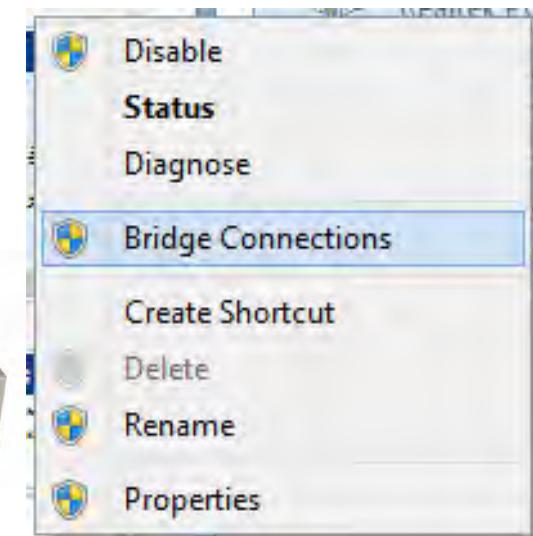
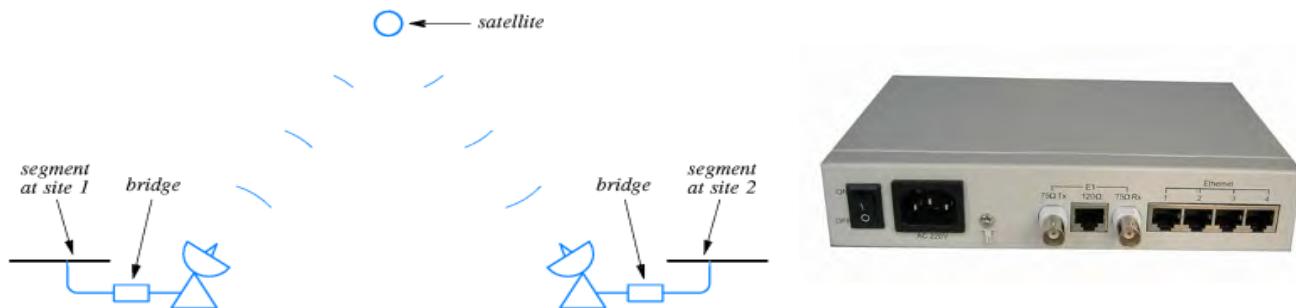
5

交换机



网桥（ Bridges ）

- 网桥（ bridge ）是用于连接两个局域网的互联设备
 - 组成：CPU、内存和两个网络接口的计算机。
 - 网桥并不运行应用软件，CPU从ROM中执行代码。
 - 数据链路层，网卡运行于混杂模式。
 - 网桥对计算机是透明的，不转发干扰。
 - 长距离连接使用网桥硬件必须执行缓冲



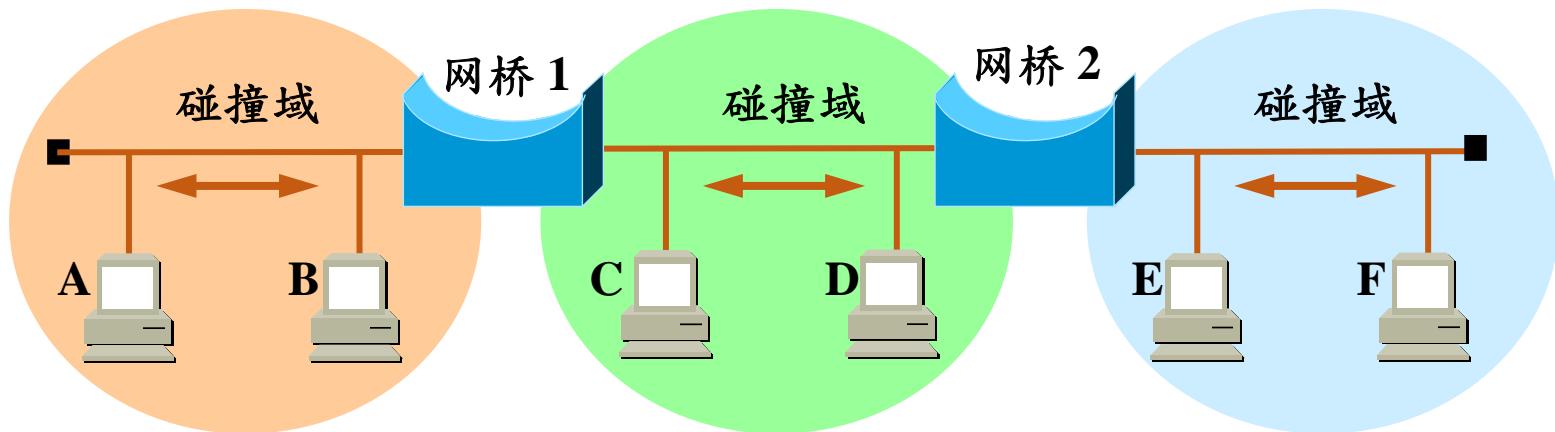
网桥

- 网桥具有过滤帧的功能：接收一帧后，
 - 如果目的站点与源站点在同一个LAN段中，则扔掉此帧；
 - 如果目的与源不在同一个LAN中，则通过某端口转发此帧；
 - 否则，将该帧扩散到除接收端口外的所有其他端口。
- 缺点
 - 存储转发增加了时延。
 - 在MAC子层并没有流量控制功能。
 - 如果传播过多广播则产生网络拥塞，即：广播风暴。



网桥隔离冲突域

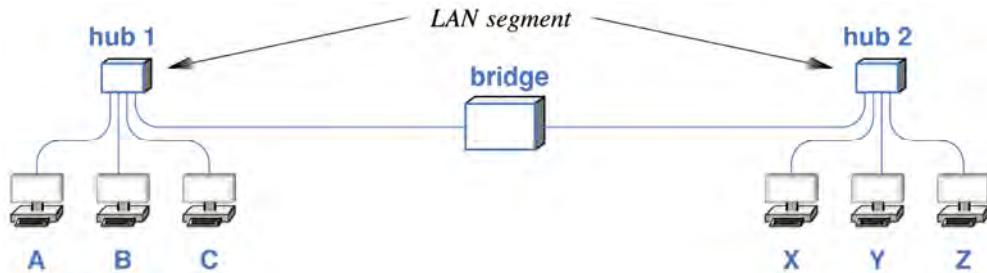
- 网桥工作在数据链路层，不改变它转发的帧的源地址。
- 集线器在转发帧时，不对传输媒体进行检测。
- 网桥在转发帧之前必须执行 CSMA/CD 算法。
 - 若在发送过程中出现碰撞，就必须停止发送和进行退避。



帧过滤 (Frame Filtering)

- 大多数网桥是自适应或自学习网桥
 - 首次启动时，网桥不知道哪些计算机连接到LAN网段。
 - 如果一台计算机没有发送任何帧，网桥无法检测到它的位置。
 - 在稳定状态下，网桥只在必要时转发帧。

课本失误



Event	Segment 1	Segment 2	Frame Sent
Bridge boots	±	±	±
A sends to B	A	±	Both Segments
B sends to A	A, B	±	Segment 1 only
X broadcasts	A, B	X	Both Segments
Y sends to A	A, B	X, Y	Both Segments
Y sends to X	A, B	X, Y	Segment 2 only
C sends to Z	A, B, C	X, Y	Both Segments
Z sends to X	A, B, C	X, Y, Z	Segment 2 only

Figure 17.4 Example of a learning bridge with computers A, B, and C on one segment and computers X, Y, and Z on another.

网桥转发表

- 网桥转发表中写入：地址、接口，帧进入网桥的时间。
 - 这是因为以太网的拓扑可能经常会发生变化
 - 站点也可能会更换适配器（这就改变了站点的地址）
 - 另外，以太网上的工作站并非总是接通电源的。
 - 把每个帧到达网桥的时间登记下来，就可以在转发表中只保留网络拓扑的最新状态信息。这样就使得网桥中的转发表能反映当前网络的最新拓扑状态。

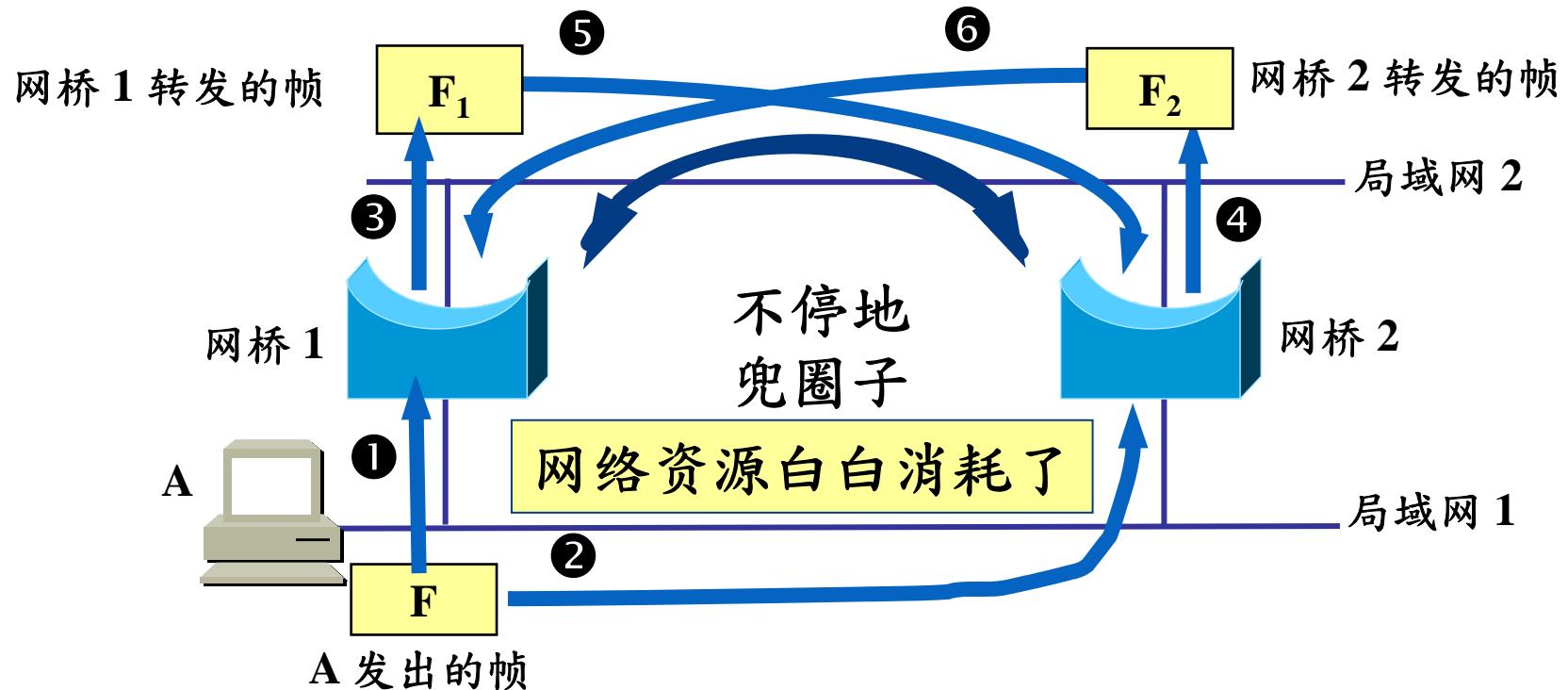


网桥转发表

- 网桥收到一帧后先进行自学习。
 - 查找转发表中与收到帧的源地址有无相匹配的项目。
 - 如没有，就在转发表中增加一个项目（源地址、进入的接口和时间）。如有，则把原有的项目进行更新。
- 转发帧：在转发表中查找收到帧目的地址有无匹配项。
 - 如没有，则通过所有其他（除入口）接口按进行转发。
 - 如有，则按转发表中给出的接口进行转发。
 - 若转发表中给出的接口是该帧进网桥的接口，则丢弃该帧。

网桥环 (Cycle of Bridges)

- 网桥环是指产生转发的帧在网络中不断地兜圈子



分布式生成树（ Distributed Spanning Tree ）

- 相互连接的网桥在彼此通信后，能找出原有网络拓扑的一个子集。
 - 在这个子集里，整个连通的网络中不存在回路，即在任何两个站之间只有一条路径。
 - 为了避免产生转发的帧在网络中不断地兜圈子。
- 为了得出能够反映网络拓扑发生变化时的生成树，在生成树上的根网桥每隔一段时间还要对生成树的拓扑进行更新。



内容纲要

1

网络布线

2

中继器

3

集线器

4

网桥

5

交换机



交换机 (Switch)

- 以太网交换机逻辑上是多接口的网桥

- 交换机工作在数据链路层。
 - 交换机包含处理器和一个中央互连
 - 处理器查找输入帧的地址，使用该互连将帧传送到正确的输出端口。

- 交换机隔离冲突域

- 交换机同时连通许多对的接口，使都能像独占通信媒体那样，进行无碰撞地传输数据。

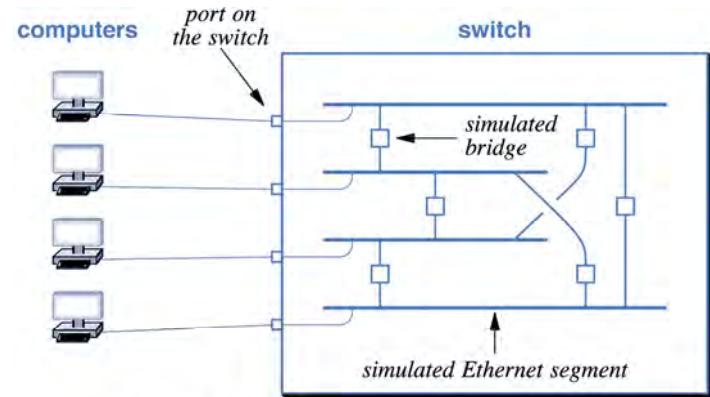
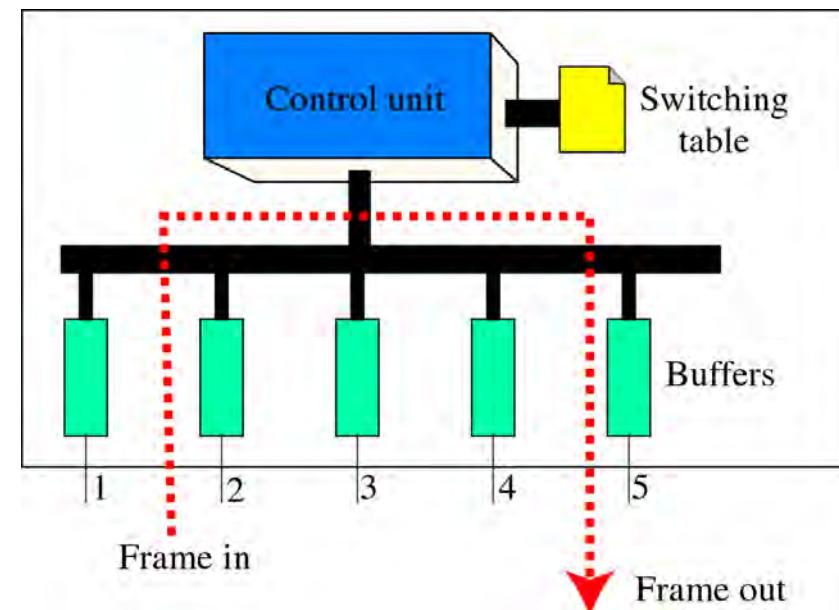


Figure 17.6 Conceptual organization of a switched LAN.



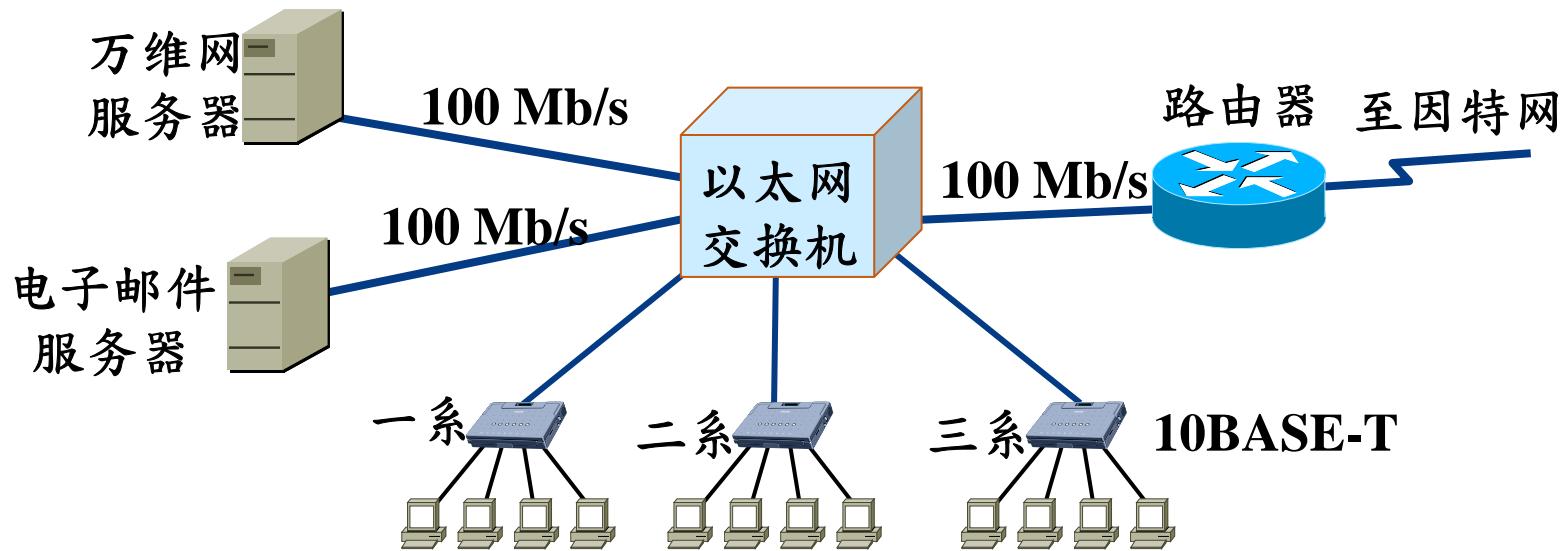
交换机

- 以太网交换机通常都有十几个接口，每个接口全双工
- 由于使用了专用的交换结构芯片，其交换速率较高。
- 交换机的最大优点：独占传输媒体的带宽
 - 普通共享式以太网，所有用户带宽总和不超过总带宽。
 - 使用以太网交换机，用户独占而不是共享传输媒体的带宽。



网络部署：结合交换机和集线器

- 一般不将一台计算机连接到交换机上的每个端口，而是将集线器连接到每个端口，然后将每个计算机连接到其中一个集线器。



第 2~7 层交换机

- 二层交换机：根据物理地址转发帧
- 三层交换机
 - 完成二层端口交换，和部分路由器的路由功能。
- 四层交换机
 - 可根据端口号区分报文的数据类型。
- 七层交换机
 - 防火墙



计算机网络

Computer Network

7

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

计算机网络

Computer Network

8

远距离数字连接

理论课程

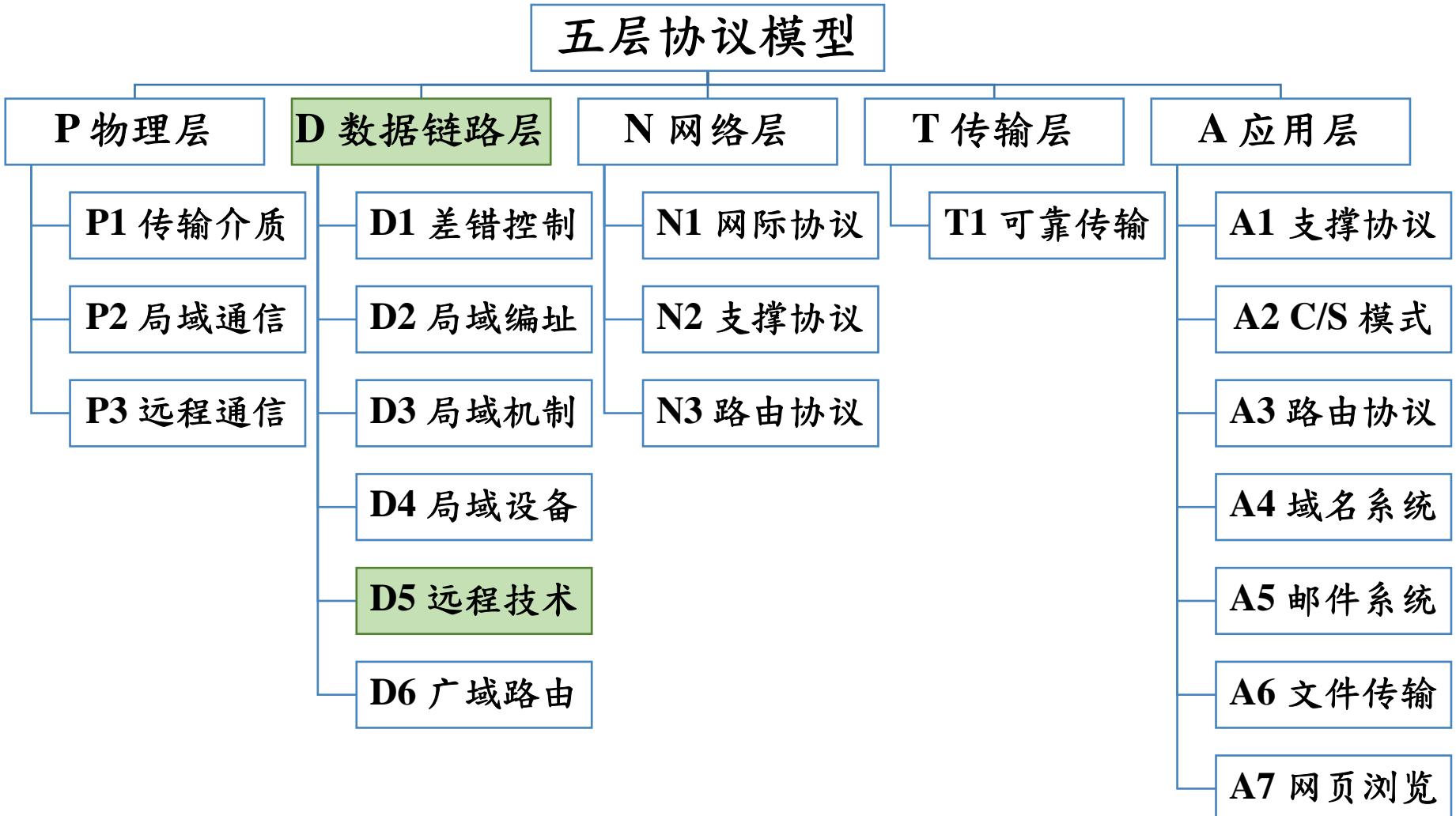


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 远程数字连接的技术
 - 通过DSU/CSU连接到数字电路
 - 上行和下行，非对称模式
 - 窄带和宽带
 - ISDN、ADSL、CATV、光纤到户、SONET的原理和速率
 - 高容量电路标准（E、T、C等）的速率等级

主要内容

- 网络技术的过去与未来
 - 广域网结构：虚电路、数据报
 - APARNET、PSTN、X.25、帧中继、SMDS的基本原理
- 网络所有权、服务模式和性能
 - VPN
 - 网络性能度量：延迟、吞吐率、抖动、服务质量（QoS）



对应课本章节

- PART III Packet Switching And Network Technologies
 - Chapter 19 Networking Technologies Past And Present
 - Chapter 30 Network Security
 - 30.17 Virtual Private Networks (VPNs)
 - 30.18 The Use of VPN Technology For Telecommuting
- PART V Other Networking Concepts & Technologies
 - Chapter 28 Network Performance (QoS and DiffServ)

内容纲要

1

远程数字连接的技术

2

网络技术过去与现在

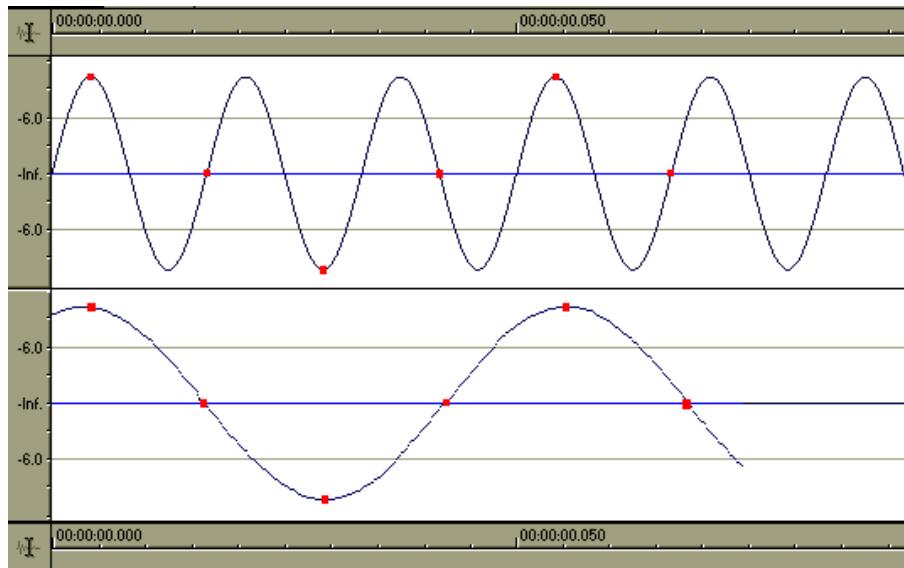
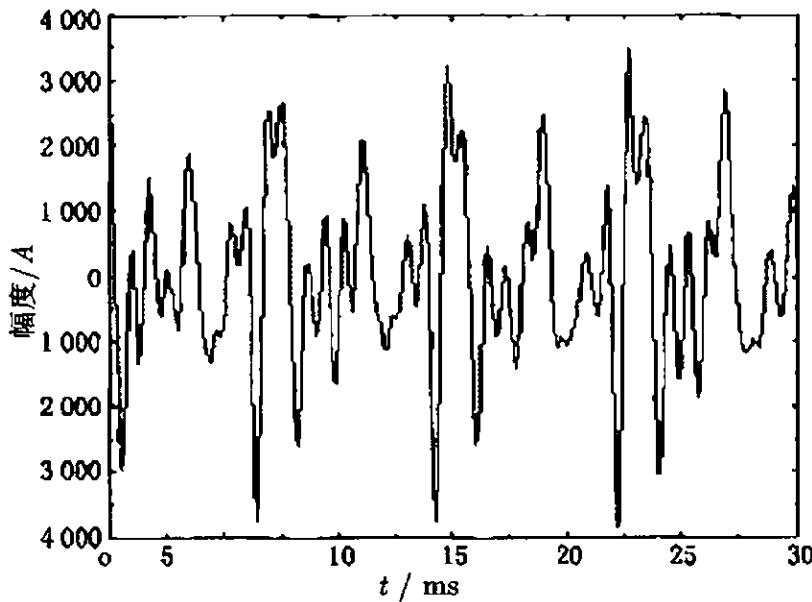
3

网络所有权、服务模式和性能



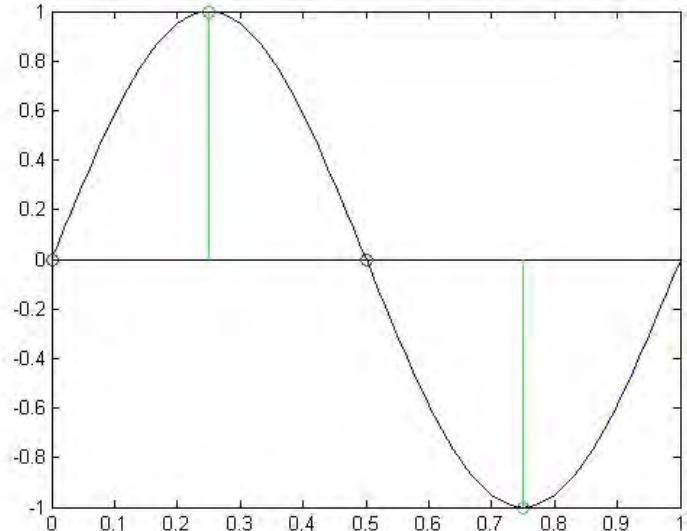
数字电话 (Digital Telephony)

- 研究数字通信的动机：数字通信避免了噪声问题。
- 数字音频：模拟音频信号的数字版本。
- 将模拟信号转换成数字形式的过程称为数字化。



Nyquist 采样定理

- 恢复一个正（余）弦信号的曲线，只需两个点
 - 相邻两个零点位置（红）或者相邻波峰和波谷的位置（绿）
 - 只要按照正（余）弦信号的规则，就能够根据这些特殊点还原出正（余）弦信号
 - 一定是特殊点才能恢复
- 不管信号多复杂，总可以分解为若干个正（余）弦信号的和，对应信号的频率分量。



脉冲编码调制 (PCM)

- Nyquist采样定理

- 如果一个连续信号用大于两倍的最高有效频率采样，信号可以从样本重建。

- 脉冲编码调制 (Pulse Code Modulation , PCM)

- PCM采样信号间隔 $125\mu\text{s}$ ，并将每个样本分为0~255的整数
 - 时间上的采样，数量上的编号

- PCM最初是为了在电话局之间的中继线上传送多路电话。

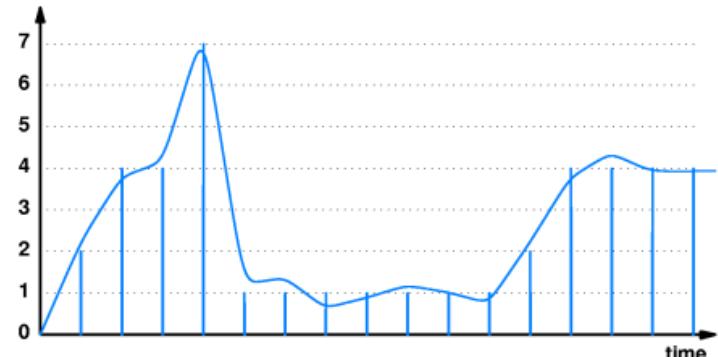


Figure 12.1 An illustration of digitization using eight values. Each vertical line represents an integer value chosen for one sample.

同步通信（Synchronous Communication）

- 在数字化语音系统中传输数据
 - 语音系统采用同步或时钟技术，数据网络采用异步技术。
 - 以精确速率移动数据
- 电话系统精心设计以传输额外的信息
 - 数据随着数字化语音，确保连续传输。
- 接收设备使用附加信息来同步时钟，确保数据以相同的速度离开网络。



DSU/CSU和NIU

- 数字电路租用公用载波

- 数据服务单元（ DSU ）

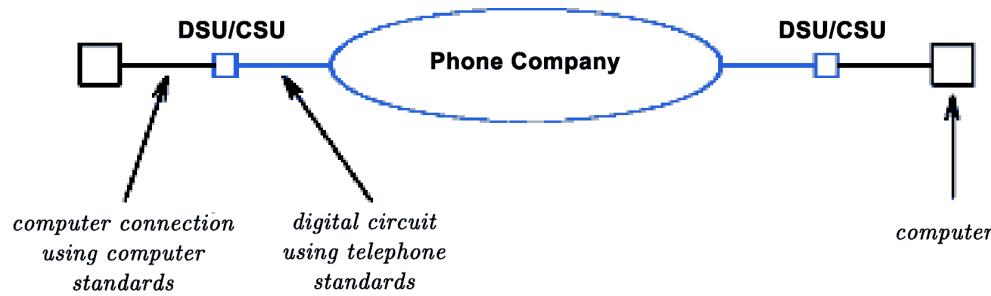
- 能够把局域网通信系统的数据帧转化成适合广域网使用的数据帧，或反向转化。

- 信道服务单元（ CSU ）

- 能够对电信线路进行保护与故障诊断。

- 网络接口单元（ NIU ）

- 控制计算机与通信网络进行交互的一种接口设备。



互联网接入技术：上行和下行

- 互联网接入技术是指连接到ISP的数据通信系统
- 多数互联网用户遵循非对称模式：接收数据比发送多
 - 下行（**downstream**）是指从互联网ISP传取数据到用户
 - 上行（**upstream**）是指从用户传输数据到ISP

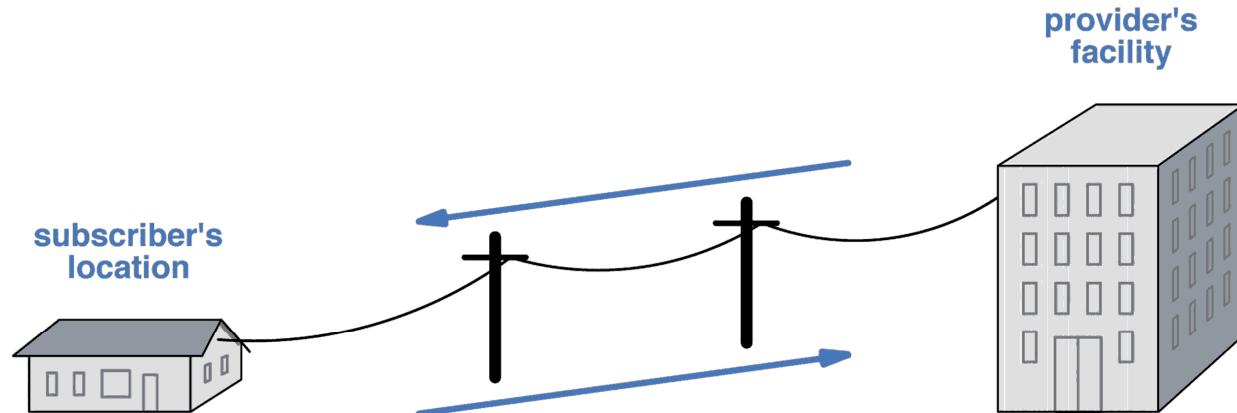


Figure 12.1 Definition of upstream and downstream directions as used in access technologies.

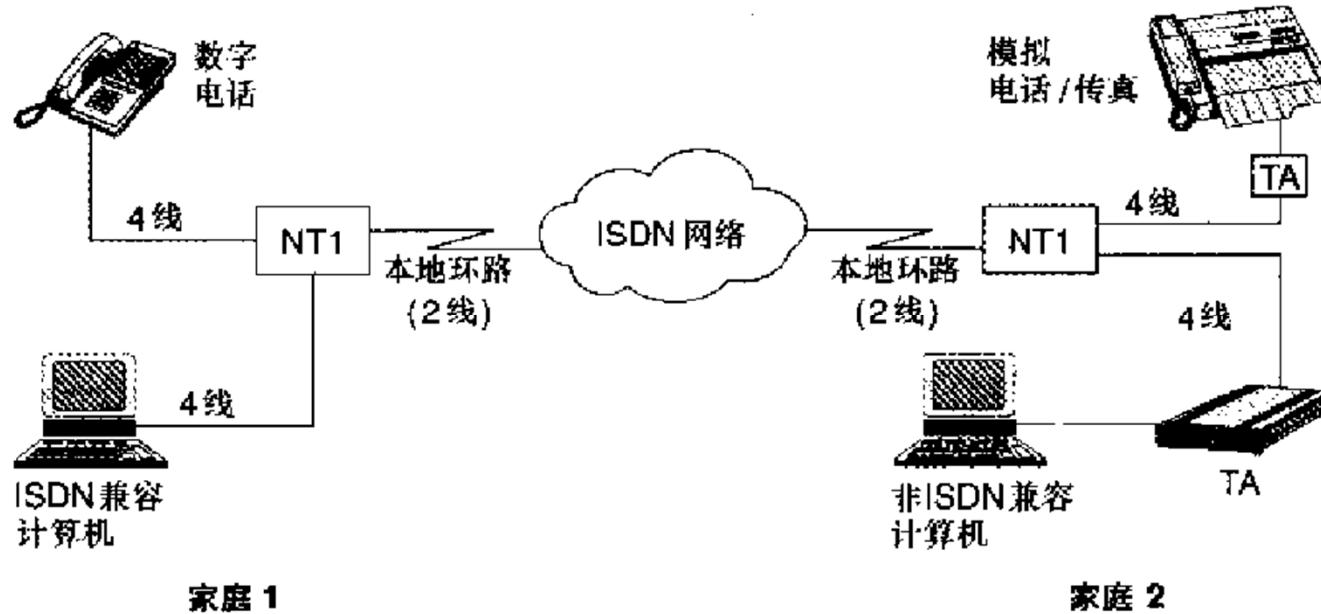
窄带和宽带接入技术

- 在网络方面，网络带宽指的是数据速率
- 宽带和窄带之间的确切边界是模糊的
 - 窄带技术（Narrowband）：在 $< 128 \text{ kbps}$ 的数据传输
 - 如，嘈杂的电话线拨号的最大数据速率 56 kbps 是窄带
 - 宽带技术（Broadband）：提供高数据速率
 - 许多人认为，宽带技术提供 $> 1 \text{ Mbps}$ 这并非总是如此

窄带	宽带
拨号电话连接	DSL技术
使用调制解调器的租用电路	电缆调制解调器技术
部分T1数据电路	无线接入技术
ISDN和其他电信公司的数据服务	在T1速度或更高的数据传输电路

本地环路和 ISDN

- 本地环路 (Local loop)
 - 电话公司中心局和用户之间的物理连接



综合业务数字网（ ISDN ）

- 综合业务数字网（ Integrated Services Digital Network , ISDN ）
 - B信道： 64 kbps ，实现数字化的语音、数据和视频压缩
 - D通道： 16 Kbps ，信令，分组数据，其它，作为控制信道

名称	捆绑通道	速率	图示
基本速率接 口 （ BRI ）	2B+D	64kbps	
初等速率接 口 （ PRI ）	23B+D	北美 DS-1 1.544Mbps	
	30B+2D	欧洲 E-1 2.048Mbps	



数字用户线路技术

- 非对称数字用户线路
 - 英文 : Asymmetric Digital Subscriber Line , ADSL
 - 采用频分复用 (频分多路复用) : ADSL 调制解调器
 - 上行和下行带宽不对称
 - 将本地环路带宽为三个区域 :
 - 一个对应于传统的模拟电话服务 , 称为普通旧电话服务 (POTS)
 - 两个提供数据通信的区域
- 离散多音调 DMT (Discrete Multi-Tone) 调制技术 。
 - 这里的 “多音调” 就是 “多载波” 或 “多子信道” 的意思 。



DMT 技术

- DMT 调制技术采用频分复用的方法
 - 把 $0 \sim 4 \text{ kHz}$ 低端频谱留给传统电话使用
 - 把 $40 \text{ kHz} \sim 1.1 \text{ MHz}$ 的高端频谱划分为许多的子信道。
 - 其中 25 个子信道用于上行信道，而 249 个子信道用于下行信道。
 - 每个子信道占据 4.3125 kHz 带宽，用不同载波数字调制。
 - 相当于在一对用户线上使用许多小的调制解调器并行地传送数据。

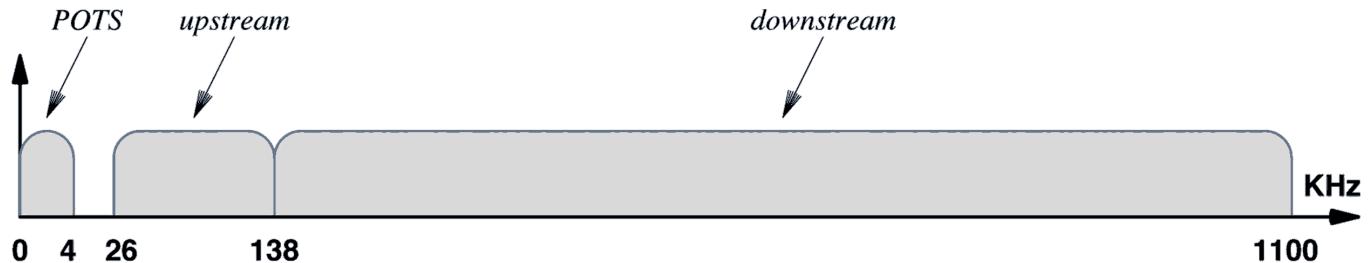


Figure 12.5 An illustration of how ADSL divides the available bandwidth of the local loop.

ADSL 的极限传输距离

- ADSL极限传输距离及数据率与用户线的线径有关
 - 用户线越细，信号传输时的衰减就越大
 - 最高数据传输速率与实际用户线上的信噪比密切相关。
- 举例
 - 0.5 毫米线径的用户线，传输速率为 1.5~2.0 Mb/s 时可传送 5.5 公里；速率提高到 6.1 Mb/s，传输距离缩短为 3.7 公里。
 - 如果把用户线的线径减小到0.4毫米，那么在6.1 Mb/s的传输速率下就只能传送2.7公里

ADSL 的数据率

- 由于用户线的具体条件相差很大，因此 ADSL 采用自适应调制技术使用户线能够传送尽可能高的数据率。
 - 距离、线径、受到相邻用户线的干扰程度等都不同
 - ADSL 启动时，用户线两端 ADSL 调制解调器即测试可用频率、各子信道受干扰情况，及在每个频率上的传输质量。
- 不能保证固定的数据率，质量很差的线甚至无法开通
- 通常下行数据率在 32 kb/s 到 6.4 Mb/s 之间，而上行数据率在 32 kb/s 到 640 kb/s 之间。



ADSL 的数据率

- 通过提高调制效率得到了更高的数据率。
 - ADSL2 要求至少应支持下行 8 Mb/s、上行 800 kb/s。
 - ADSL2+ 则将频谱范围从 1.1 MHz 扩展至 2.2 MHz
 - 下行速率可达 16 Mbps (最大传输速率可达 25 Mbps)
 - 上行速率可达 800 kbps

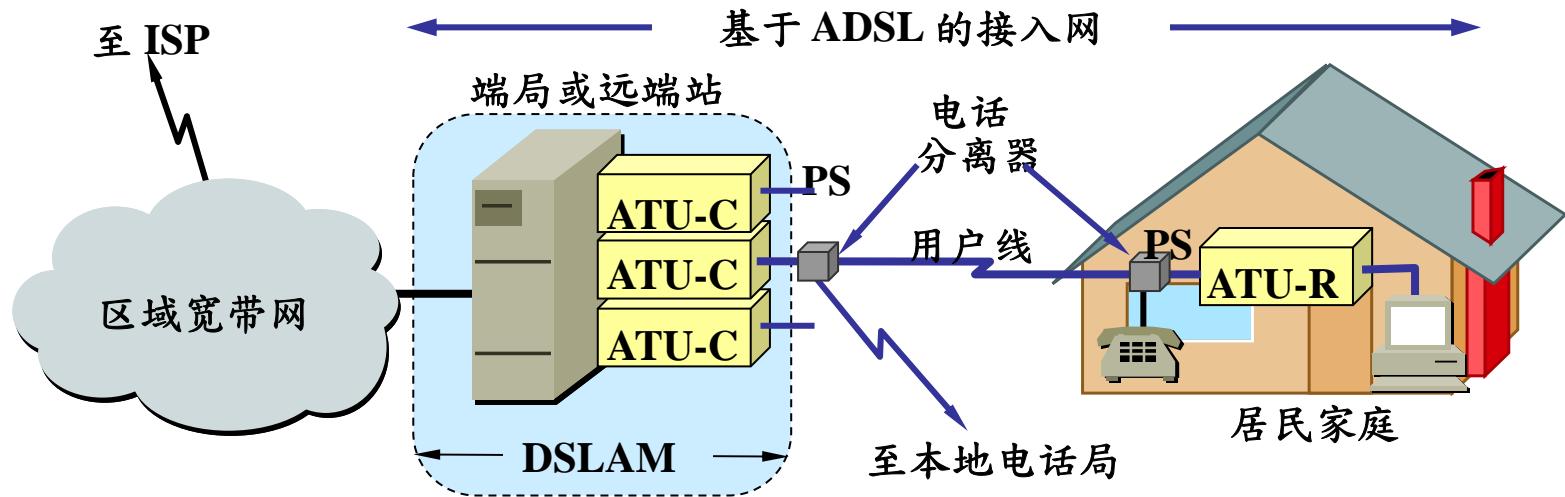
表 14.4 DSL 交叉参考

名称	描述	速率	模式
DSL	数字用户线	192Kbps	双工
HDSL	高数据/位速率 DSL	1. 544Mbps 2. 048Mbps	双工
SDSL	单数据线 DSL	1. 544Mbps 2. 048Mbps	双工
ADSL	非对称 DSL	1. 5 到 9Mbps 16 到 640Kbps	顺流 逆流
VDSL	超高速 DSL	1. 3 到 52Mbps 1. 5 到 23Mbps	顺流 逆流



ADSL 的组成

- 数字用户线接入复用器 (DSL Access Multiplexer)
- 接入端接单元 ATU (Access Termination Unit)
 - C 代表端局 Central Office , R 代表远端 Remote
- 电话分离器 PS (POTS Splitter)



电缆调制解调器技术

- 社区天线电视（CATV）
- 采用频分复用在同轴电缆传输电视信号
- 同轴电缆比双绞线高带宽，不易受到电磁干扰
- 理论上，电缆系统支持52 Mbps下行和512 kbps上行
 - 在实践中，速率少得多
- 缺点：与其他用户共享带宽

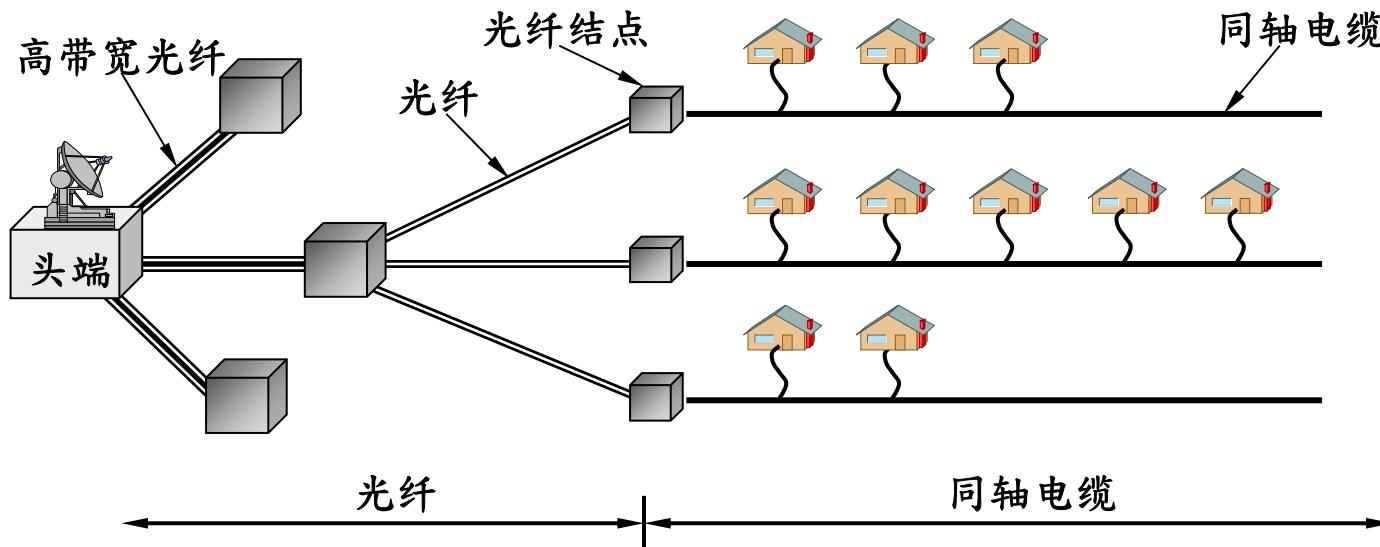


光纤同轴混合网（HFC）

- 光纤同轴混合网（ Hybrid Fiber Coax ）
 - 在目前覆盖面很广的有线电视网 CATV 的基础上开发的一种居民宽带接入网。
 - 可传送 CATV ，还提供电话、数据和其他宽带交互型业务。
- CATV 网
 - 树形拓扑结构的同轴电缆网络
 - 采用模拟技术的频分复用对电视节目进行单向传输
 - HFC 网需要对 CATV 网进行改造

光纤同轴混合网（HFC）

- 用户接口盒 UIB (User Interface Box) 要提供三种连接
 - 使用同轴电缆连接到机顶盒，然后再连接到用户的电视机。
 - 使用双绞线连接到用户的电话机。
 - 使用电缆调制解调器连接到用户的计算机。



采用光纤的接入技术

- FTTx 是一种实现宽带居民接入网的方案。
 - FTTH：光纤一直铺到家庭可能是居民接入网最终解决方法
 - FTTB：光纤进入大楼后就转换为电信号，然后用电缆或双绞线分配到各用户。
 - FTTC：从路边到用户可用星形结构双绞线作为传输媒体。

名称	全称	说明
FTTC	Fiber To The Curb	到小区边界外
FTTB	Fiber To The Building	允许高上行
FTTH	Fiber To The Home	光纤到户，更高上行，视频信道
FTTP	Fiber To The Premises	FTTB和FTTH的通称

在互联网核心的大容量连接

- 接入技术处理最后一英里问题
 - 最后一英里被定义为到一个典型住宅用户或小企业的连接
- 核心是指互联网骨干的连接，核心技术是指高速技术

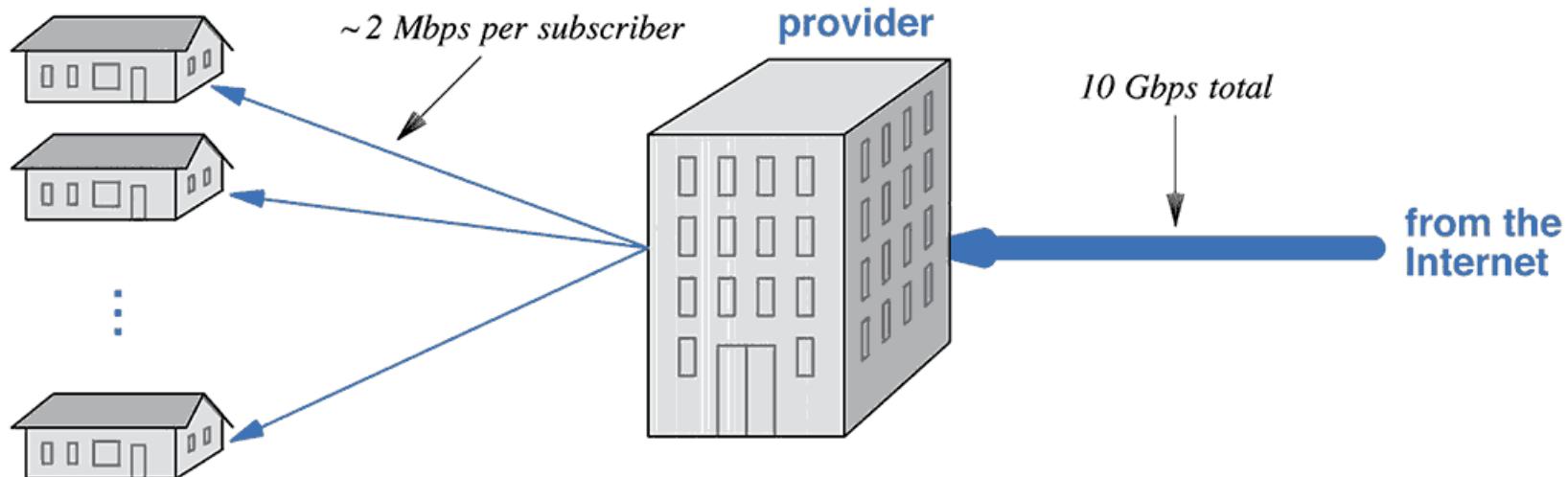


Figure 12.10 Aggregate traffic from the Internet to a provider assuming the provider has 5,000 customers each downloading 2 Mbps.

在互联网核心的大容量连接

- 提供速度在10 Gbps的长距离移动数据，在于从电话公司租用的点对点数字电路高容量数字电路
- 每月付费，用于传输数据
 - 费用取决于电路的数据速率和跨越的距离
- 电话公司有权安装穿过市政街道的电线
- 一个电路可以两楼之间延伸，或从一个城市到另一个城市



数据线路的电话标准

- 由于历史原因，PCM 有两个互不兼容的国际标准
 - 北美的 24 路 PCM（简称为 T1），1.544 Mb/s
 - 欧洲的 30 路 PCM（简称为 E1），2.048 Mb/s
 - 我国采用的是欧洲的 E1 标准。
- DS-n 表示一个标准，而 T-n 表示符合标准的电路
- 当需要有更高的数据率时，可采用复用的方法。

名称	比特率	语音线路	地区
基本速率	0.064 Mbps	1	
T1	1.544 Mbps	24	北美
T2	6.312 Mbps	96	北美
T3	44.736 Mbps	672	北美
E1	2.048 Mbps	30	欧洲
E2	8.448 Mbps	120	欧洲
E3	34.368 Mbps	480	欧洲



高容量电路（STS标准）

- 电话公司使用干线（trunk）来表示高容量电路，并为数字中继电路创造了一系列标准
 - 同步传输信号（Synchronous Transport Signal，STS）标准指定高速连接的电气信号
 - OC（Optical Carrier）标准指光信号在光纤中传播
 - C后缀表示级联（concatenated）

铜线名	光纤名	比特率	语音电路
STS-1	OC-1	51.840 Mbps	810
STS-3	OC-3	155.520 Mbps	2430
STS-12	OC-12	622.080 Mbps	9720
STS-24	OC-24	1,244.160 Mbps	19440
STS-48	OC-48	2,488.320 Mbps	38880
STS-192	OC-192	9,953.280 Mbps	155520



同步光纤网 SONET

- 同步光纤网 SONET (Synchronous Optical Network) 的各级时钟都来自一个非常精确的主时钟。
- 第 1 级同步传送信号 STS-1 (Synchronous Transport Signal) 的传输速率是 51.84 Mb/s。
- 光信号则称为 OC-1 (Optical Carrier)

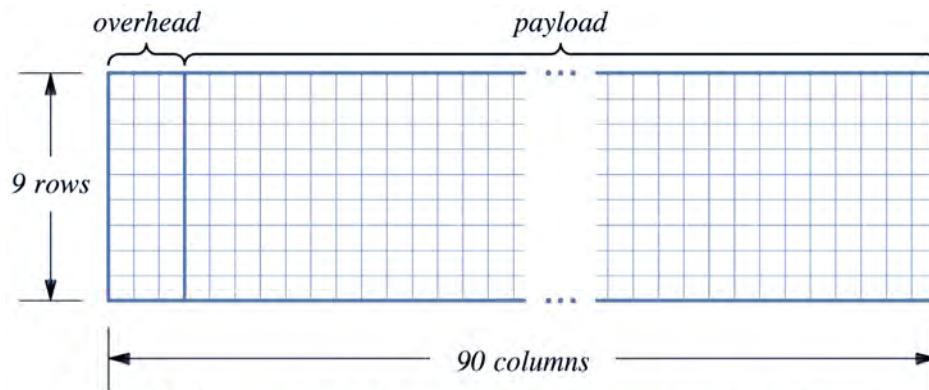


Figure 12.13 Illustration of a SONET frame when used over an STS-1 circuit.

内容纲要

1

远程数字连接的技术

2

网络技术过去与现在

3

网络所有权、服务模式和性能



广域网技术实例

- ARPANET
- 公用电话交换网PSTN
- 公用分组交换网X.25
- Frame Relay (帧中继)
- SMDS (交换多兆位数据服务 Switched Multi-megabit Data Service)
- ATM异步传输模式 (Asynchronous Transfer Mode)



广域网结构

- 虚电路：面向连接，类似电话系统

- 原理

- 建立虚电路（填表）、数据转发（查表）、释放虚电路（删表）

- 特点

- 存在虚电路建立过程
 - 数据转发沿着同一条路径
 - 报文的投递可靠
 - 报文中不需要目的地址，只需要虚电路号
 - 虚电路必需进行释放

广域网结构

- 数据报：无连接，类似电报系统

- 原理

- 路由器为每个入站的报文单独选择一条输出线路

- 特点

- 不需要虚电路建立过程
 - 路由器必须为每个输入报文单独进行路由选择
 - 报文投递是不可靠的
 - 每个报文必须包含目的地址



两者比较

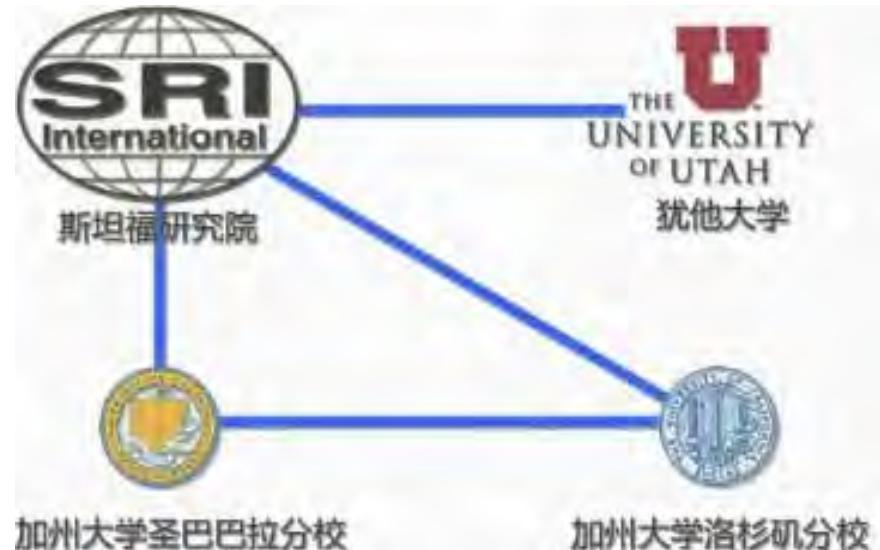
- 从广域网内部来看

- 交换机的内存空间与线路带宽的权衡
- 虚电路建立时间和路由选择时间的比较

项目	数据报	虚电路
电路建立	不需要	需要
地址	每个报文都必须有完整的源和目的地址	每个报文只需要一个虚电路号
状态信息	子网不存储状态信息	每条虚电路都占用子网的表空间
路由选择	每个报文单独进行	在建立虚电路时进行路由选择
路由器失效的影响	除在崩溃时丢失路由器中的报文，对其他报文没有影响	所有经过失效路由器的虚电路都要被中止
拥塞控制	难	容易
用户服务	“端到端”(end-to-end) 控制	“跳到跳”(hop-by-hop) 控制

ARPANET

- 美国高级研究计划署网络（ ARPANET ） (1969-1990)
 - 第一个分组交换广域网
 - 美国国防部高级研究计划署（ Advanced Research Projects Agency ）
 - 最初： 4 个站点
 - 租用串行数据线， 56Kbps
 - Born: 1969; Obsolete: 1990



PSTN

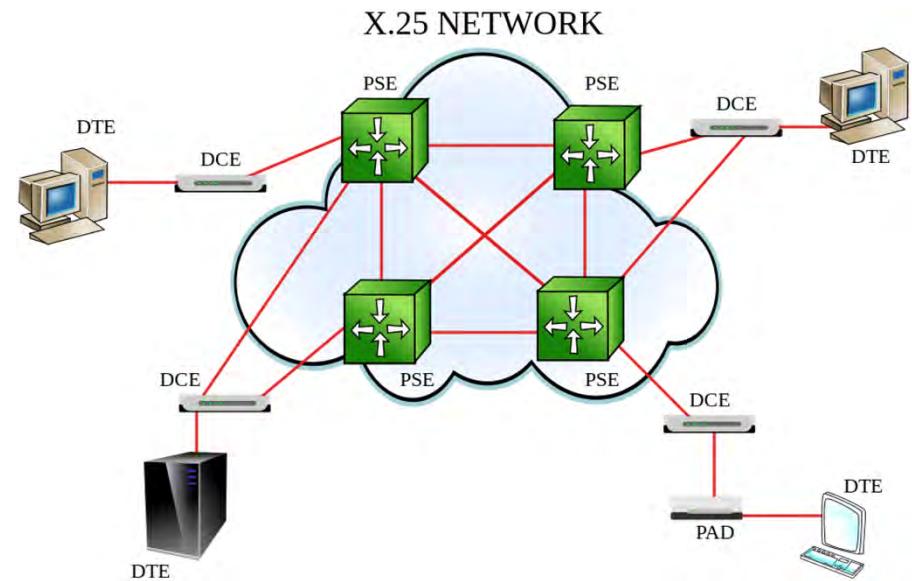
- **PSTN (1876-Now)**

- **Public Switched Telephone Network**，公共交换电话网
- 目前世界上最大的网络，拥有用户数量大约是8亿
- 以电路交换技术为基础；传输模拟话音的通信网络。
- 组成
 - 本地回路：模拟线路；干线：数字化；电话交换机：数字化程控
- 两台计算机想通过PSTN进行通信时，必须引入Modem

X.25

- CCITT X.25 (1976-Now)

- CCITT国际电话电报咨询委员会(Consultative Committee for International Telephone and Telegraph)
- X.25是关于DTE和DCE之间的接口
 - X.25建议：一个DTE如何连接到有关分组交换网上
- 每一个X.25网络由两个或两个以上的X.25分组交换机通过专线互联。



X.25的特点

- X.25是面向连接的，它支持交换虚电路服务
 - 交换虚电路SVC
 - 永久虚电路PVC
- X.25提供差错控制
- X.25提供流量控制

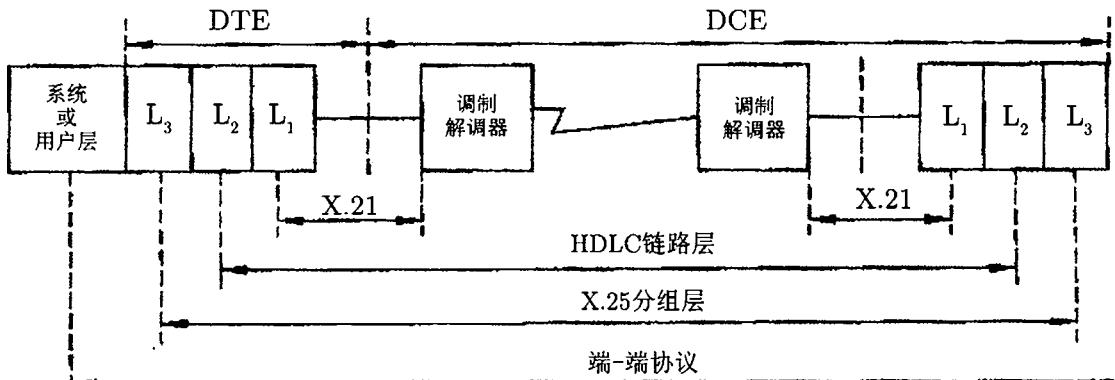


图 15.13 DTE / DCE 接口的协议层次

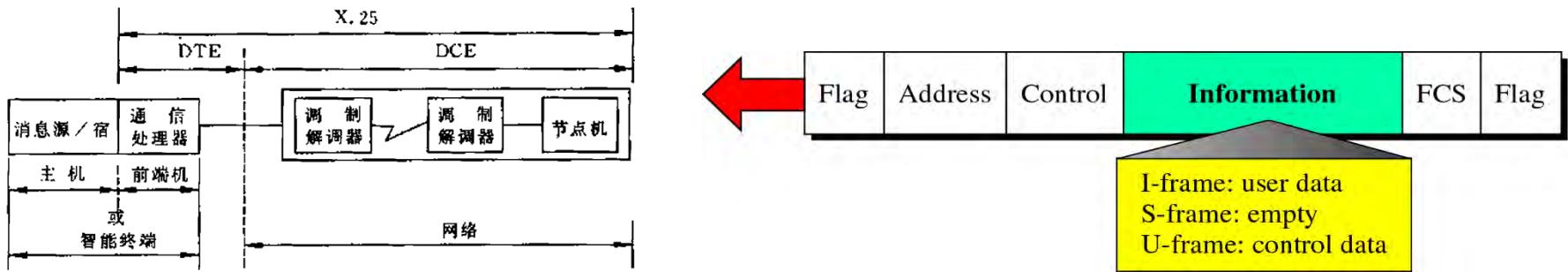
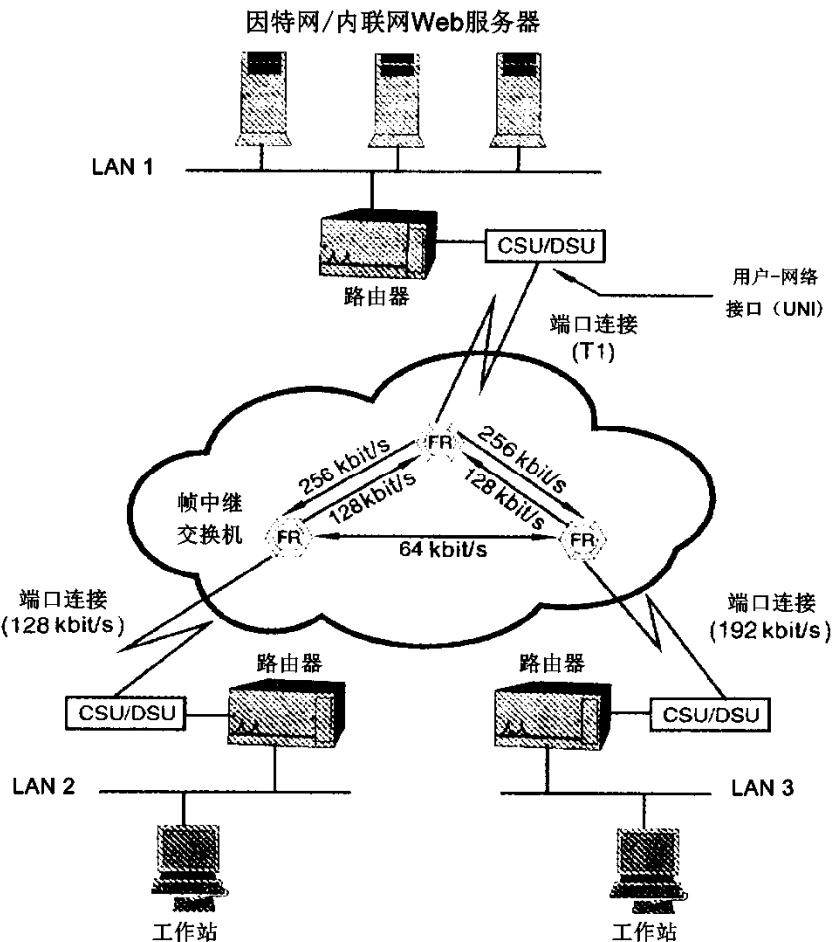


图 15.12 X.25 环境下的 DTE 与 DCE

帧中继 (Frame Relay)

- Frame Relay (1972-Now)

- 用于桥接LAN网段的FR服务
- 帧中继用于接收和传送数据块每块可包含8KB的数据。
- 为处理一个网段的数据，帧中继必须在高数据率操作（1.5Mbps或56kbps）。



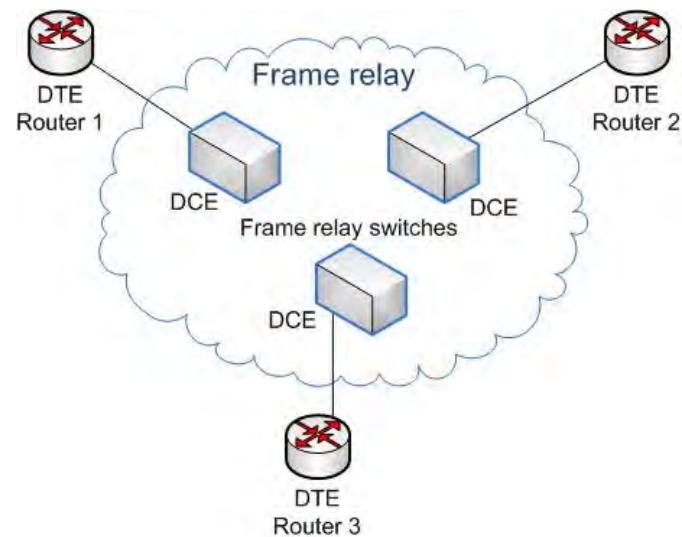
FR的特点

- 优点

- FR支持较高速率（T1或T3）
- FR只包含物理层和数据链路层，比X.25开销小
- FR允许支持突发数据，且帧长度可达9000字节

- 缺点

- 不提供差错控制功能
- 对多媒体数据传输的支持不够

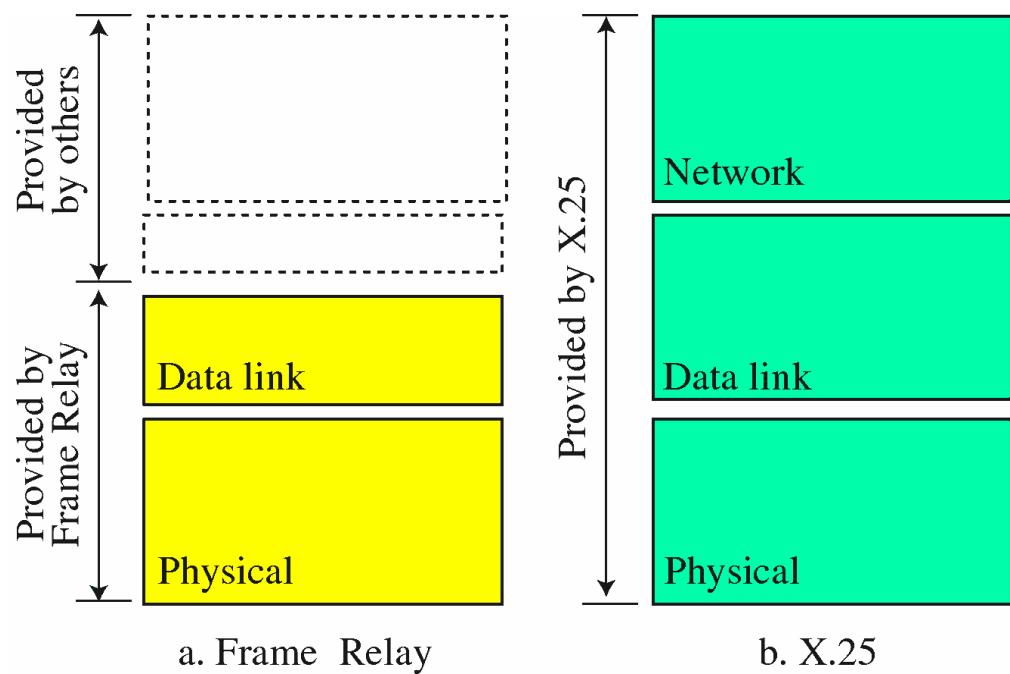


X.25与FR的比较

	X.25	FR
传输线路	缓慢，模拟，可靠	快速，数字，不可靠
计算机	缓慢，昂贵的	快速，廉价
协议简易度	复杂	比较简单
智能程度	网络保证数据传输的可靠性，端用户对数据的处理相对简单	网络不保证数据传输的可靠性，端用户对数据的处理相对复杂
分层	物理层、数据链路层、网络层	物理层、数据链路层
连接建立	在网络层	无
流量和差错控制	数据链路层：逐跳的 网络层：端到端的	无
数据传输率	固定	可变
多路复用	网络层	数据链路层
拥塞控制	不需要	需要

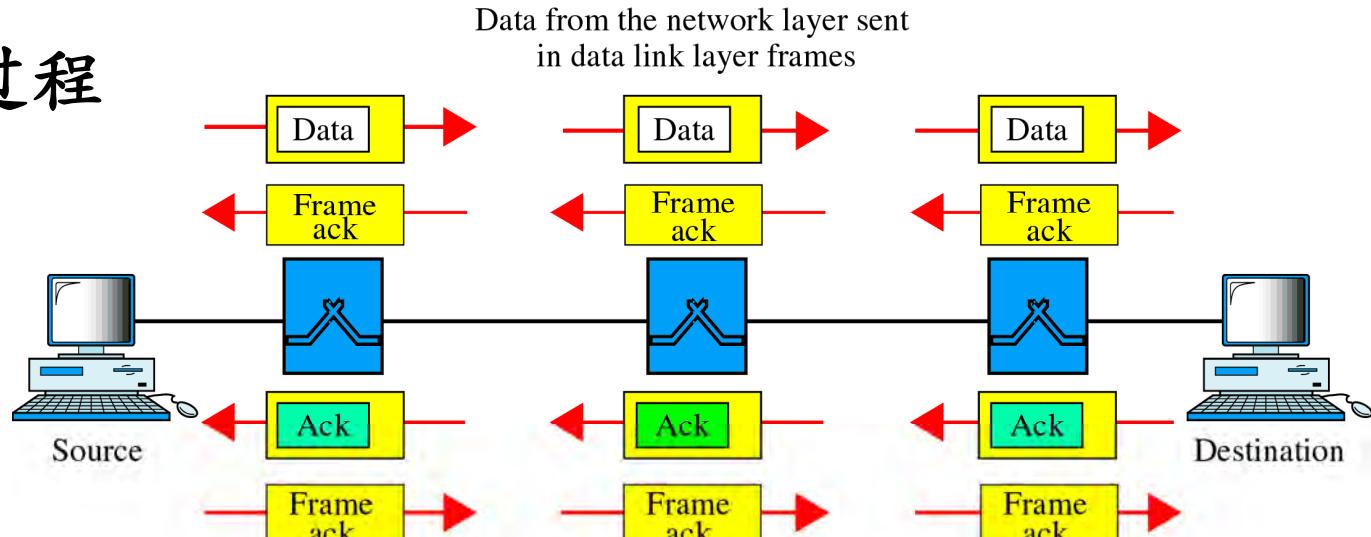
X.25与FR的比较

- FR是轻型化的X.25，与X.25相比
 - 保留了X.25的物理层功能
 - 保留了X.25部分数据链路层功能，并将多路复用功能放在第二层实现
 - 丢弃了X.25第三层

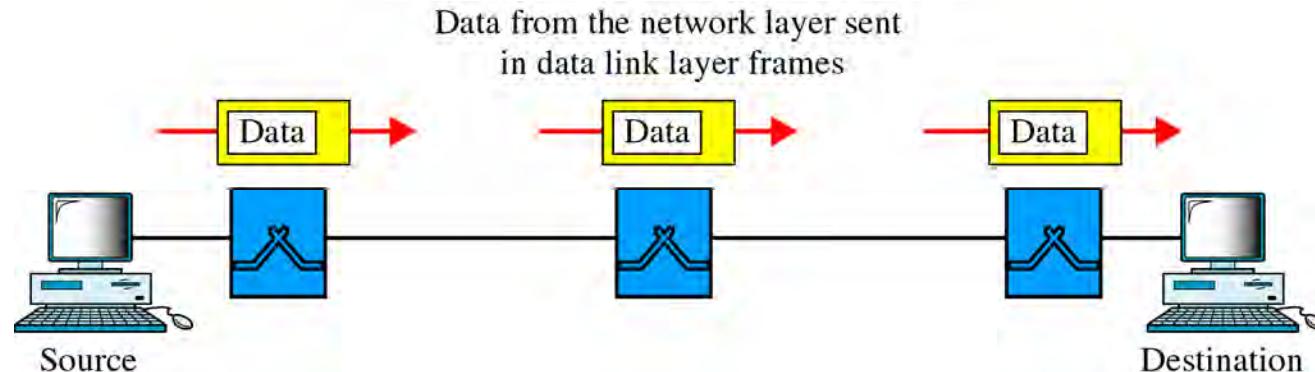


X.25和FR的比较：通信过程

- X.25的通信过程



- FR的通信过程



交换式多兆位数据服务（SMDS）

- 交换式多兆位数据服务SMDS

- 用于连接多个局域网
- 其设计是针对突发数据通信的。
- 标准速率是45Mbps，也支持低于45Mbps的速率
- 提供无连接的数据传输服务
- 已被SONET取代

服务范例比较

- 各种技术的连接类型和使用场景

技术	面向连接	无连接	用于LAN	用于WAN
以太网		*	*	
令牌环		*	*	
FDDI		*	*	
帧中继	*			*
SMDS		*		*
ATM	*		*	
LocalTalk		*	*	

内容纲要

1

远程数字连接的技术

2

网络技术过去与现在

3

网络所有权、服务模式和性能



网络所有权

- 私有网络
 - 网络的使用仅限于公司或个人拥有者
 - 对技术决策和策略有完全的控制权，保证网络与组织外的计算机隔离，安装和维护昂贵
- 公有网络
 - 由服务提供商拥有和运营，任何用户可以使用
 - 灵活性和能够使用先进的网络，而不需维护技术专长。
- 大多数公共网络提供私人通信。



虚拟专用网（VPN）

- 虚拟专用网（Virtual Private Networks）

- 在公用网络上建立专用网络，进行加密通讯。
 - 有的公司没有分布各地的部门，但有很多流动员工在外地工作。他们提供远程接入 VPN 和公司保持联系
 - 在外地工作的员工拨号接入因特网，而员工计算机中的 VPN 软件可在和公司的主机之间建立 VPN 隧道，通信的内容是保密的，像是使用公司内部的本地网络。
 - VPN 系统使用加密保证绝对隐私，即使局外人确实设法获得一个包的副本，外人将无法解释的内容。



性能度量

- 关键量度

- 延迟（时延）：传播时延、接入时延、交换时延、队列时延、服务器时延
- 吞吐率（容量）：
 - 网络可以支持的最大传输速率
▪ 单位：bps, kbps, Mbps, Gbps；注意：bit per second
 - 传输延迟是信号在信道的持续时间，吞吐率是信号输入信道的速率
- 抖动（变化量）：
 - 评估时延的变化量
 - 处理方法：设计无抖动的等时网络；采用补偿抖动的协议（RTP）



服务质量 (QoS)

- 网络服务的等级
- 服务提供商与用户的契约
 - 层级服务：按服务等级计算金额
 - 例如：电信宽带分4Mbps，6Mbps等
 - 服务保证
 - 证券交易：时延不超过10ms
 - 公司需要备份数据中心：吞吐率不少于1Gbps
- 测量服务质量的工具
 - 简单工具：Ping，原理：Echo协议



计算机网络

Computer Network

8

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



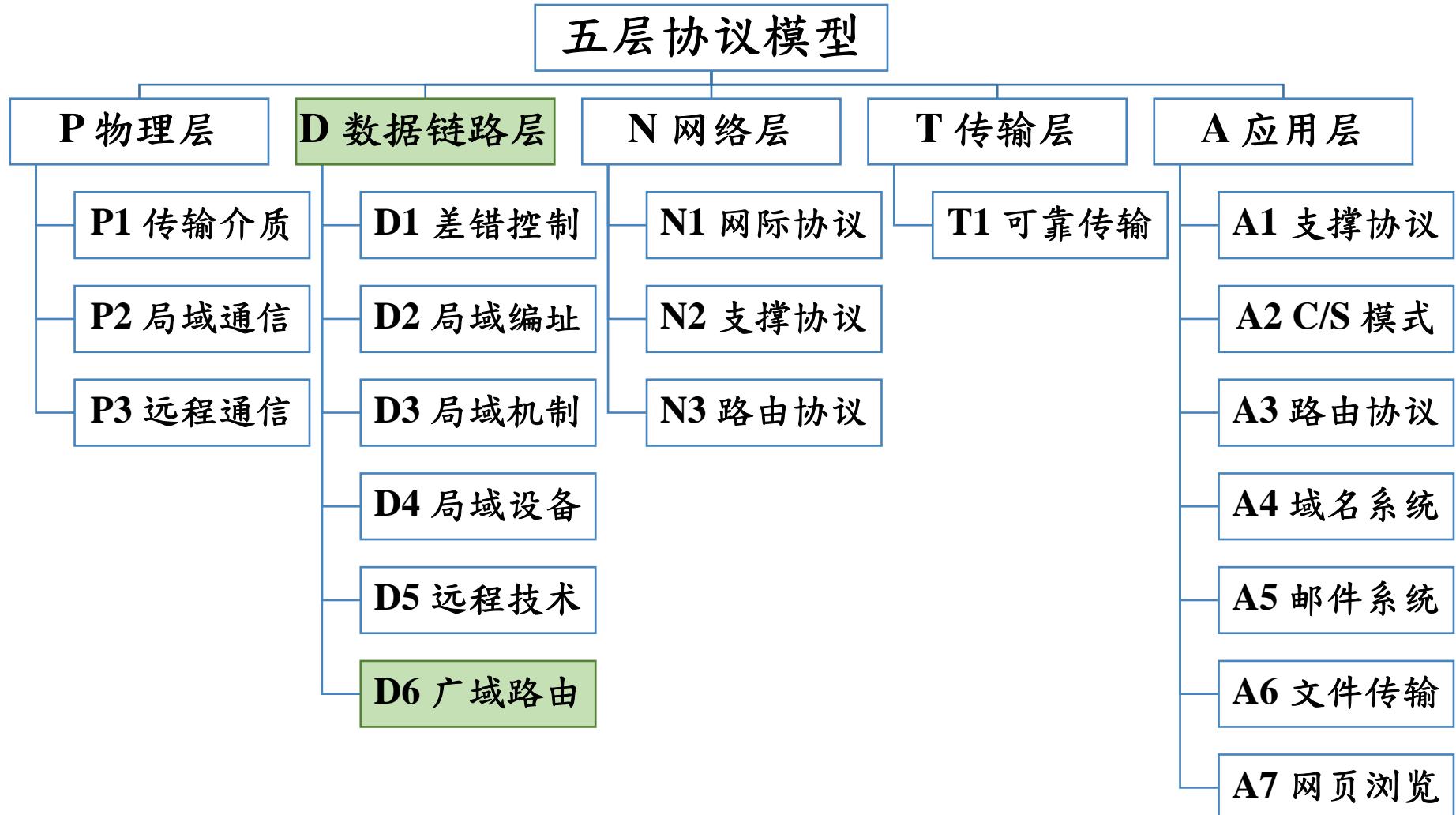
信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

9

广域网技术 和协议分层



主要内容



主要内容

- 广域网技术与路由
 - 广域网分层地址
 - 存储转发和路由、路由算法：LSR和DVR
 - 下一跳和路由表
- 协议与分层
 - OSI/ISO协议（七层）、TCP/IP协议（四层）、五层协议
 - 分层的优点、各层次的作用
 - 帧在各层次间的封装和拆封

对应课本章节

- PART III Packet Switching And Network Technologies
 - Chapter 18 WAN Technologies And Dynamic Routing
- PART IV Internetworking
 - Chapter 20 Internetworking: Concepts, Architecture, and Protocols

内容纲要

1

广域网技术与路由

2

协议与分层



大网络和广域

- 网络技术分为三大类
 - 局域网（ Local Area Network , LAN ）
 - 城域网（ Metropolitan Area Network , MAN ）
 - 广域网（ Wide Area Network , WAN ）

表 1-3 多个处理机互连的系统按其大小的分类

处理机之间的典型距离	处理机所在的范围	实例
0.1 m	印制板	数据流计算机
1 m	系统	多处理机
10 m	房间	
100 m	建筑物	局域网、校园网、企业网
1 km	校园	
10 km	城市	城域网
100 km	国家	广域网
1000 km	国家，洲	广域网，互连的广域网



广域网遇到的新问题

局域网	广域网
距离近，通信随机发生 主机少，设备性能接近	距离远，不确定因素多 主机多，设备性能各异

- 解决方案：高层“分组交换机”

- 中央处理器：转发到下一站
- 存储器：帧，转发信息

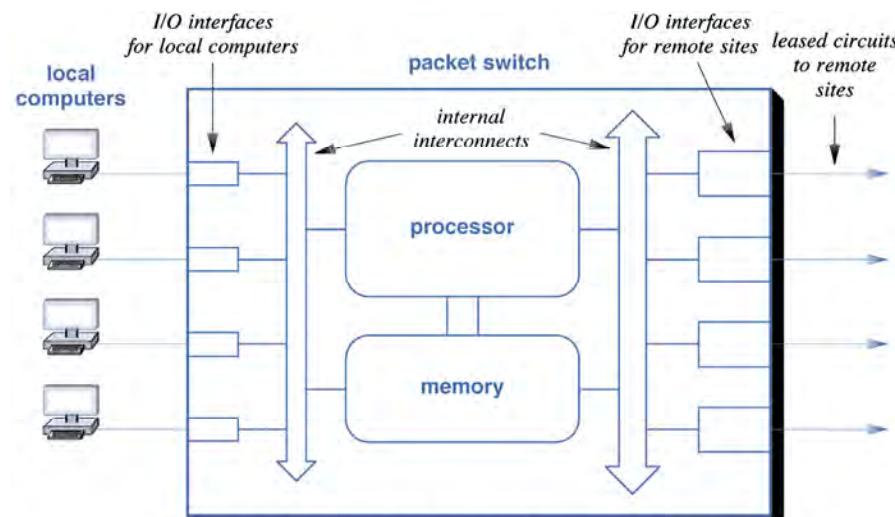


Figure 18.1 Illustration of traditional packet switch architecture.

交换机的机制：存储转发

- 存储转发（ Store and Forward ）
- 存储
 - 存储操作发生于分组到达时
 - 交换机中的I/O硬件将数据包的副本缓存在存储器中
- 转发
 - 转发操作发生于分组到达并等待在内存中。
 - 处理器检查包，解析目的地址，将数据包发送到通向目的地的I/O接口上



广域网的寻址方案

- 主机多使得广域网需要分层寻址方案
 - 站点号+主机号 : (site, computer at the site)
 - 第一部分标识分组交换机 : 每个分组交换机被分配一个唯一的号码
 - 第二部分标识特定计算机



Figure 18.4 Example of an address hierarchy where each address identifies a packet switch and a computer attached to the switch.

下一跳转发：层次寻址

- 数据包到达时，交换机必须选择一个传出路径来转发
- 交换机检查数据包的目的地址，提取交换机号
 - 本地：交换机将数据包直接发送到计算机
 - 如果目的地址中的数字与分组交换机自身的ID相同，则该分组将用于本地分组交换机上的计算机
 - 否则：数据包必须转发到另一个交换机
- 只基于交换机ID转发
 - 这意味着交换机只需要知道哪个传出链路

下一跳转发：层次寻址

- 交换机不需要保有所有可达的计算机的完整信息，也没有一个交换机需要计算整个路由
- 交换机只需要计算数据包的下一跳
- 该过程称为下一跳转发

Algorithm 18.1

Given:

A packet that has arrived at packet switch Q

Perform:

The next-hop forwarding step

Method:

```
Extract the destination address from the packet;  
Divide the address into a packet switch number, P, and a  
computer identification, C;  
if (P == Q) { /* the destination is local */  
    Forward the packet to local computer C;  
} else {  
    Select a link that leads to another packet switch, and for-  
ward  
    the packet over the link;  
}
```

Algorithm 18.1 The two steps a packet switch uses to forward a packet when using next-hop forwarding.



下一跳转发：转发表

- 转发表：存储目的交换机和对应下一跳的表
 - 列出所有可能的分组交换机，并为每个给出下一跳
 - 这些“转发表”本应称为路由表（**routing table**）
 - 将数据包转发到下一跳的过程称为路由（**routing**）

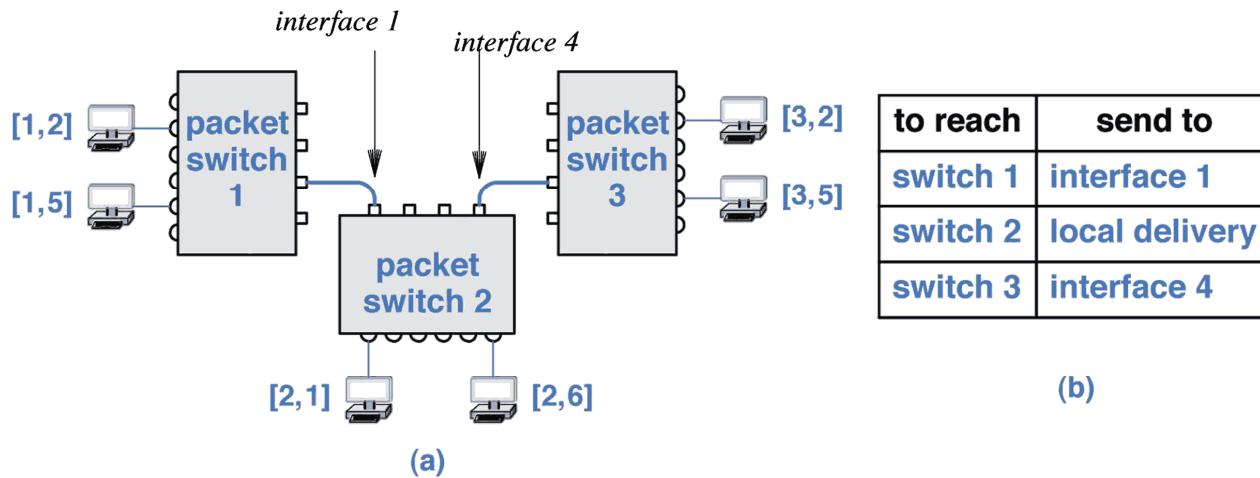


Figure 18.5 (a) A network of three packet switches, and (b) the next-hop forwarding table for switch 2.

下一跳转发：层次地址的优点

- 只用层次地址的一部分来转发包的优点
 - 转发表所需的计算时间减少。
 - 因为转发表可以被组织成一个数组，使用索引，而不是搜索
 - 转发表所需的存储空间减少。
 - 包含每个包交换的一个条目，而不是每个目标计算机的一个条目表
 - 存储空间可能大大减少。广域网每个分组交换机连着许多计算机。
- 地址的第二部分作用
 - 一旦数据包到达最终交换机，用该部分选择特定的计算机

源独立 (Source Independence)

- 源独立性是指，下一跳转发不依赖于
 - 数据分组的原始源
 - 数据分组到达前的特定路径
- 下一跳仅取决于数据分组的目的地
- 源独立令计算机网络中的转发机制显得紧凑和高效



广域网的设备

- 广域网使用路由器连接成网络
 - 第2层交换机（Switch）主要发挥交换功能
 - 路由器（Router）主要发挥路由功能
 - 许多广域网租用专用数据电路，其他形式也可用
- 广域网不需要对称

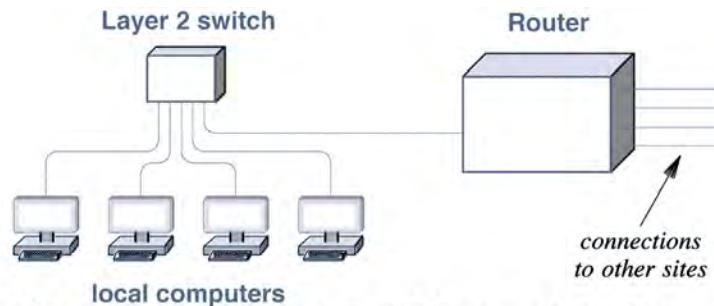


Figure 18.2 Illustration of a modern WAN site with local communication handled by a separate LAN.

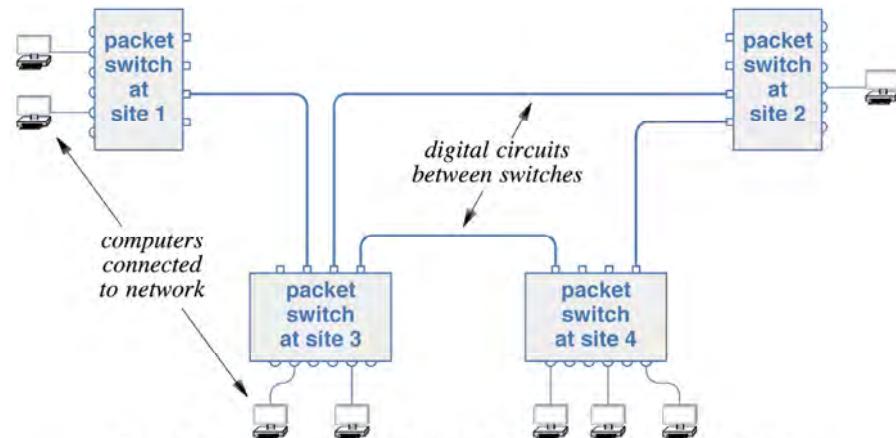


Figure 18.3 An example WAN formed by interconnecting packet switches.



路由表计算

- 静态路由（ Static routing ）

- 交换机启动时，计算并安装路由；路由不会改变
 - 优点：简单和低开销；
 - 缺点：缺乏灵活性，当通信中断时，无法更改静态路由

- 动态路由（ Dynamic routing ）

- 交换机启动，生成初始路由表；网络改变时，程序改变表。
 - 大型网络的设计留有处理偶然硬件故障的冗余
 - 大多数广域网使用动态路由



广域网动态路由更新

- 路由表中的值必须保证
 - 通用通信：转发表须包含每个目标地址的有效下一跳路由
 - 最优路线：转发表中下一跳值必须指向目标的最短路径
- 网络故障进一步使得转发复杂化
 - 网络管理器不能只配置静态转发表
 - 在交换机的软件应不断测试失败，自动重新配置转发表



路由表的制作

- 图抽象出细节，允许路由软件处理问题的本质

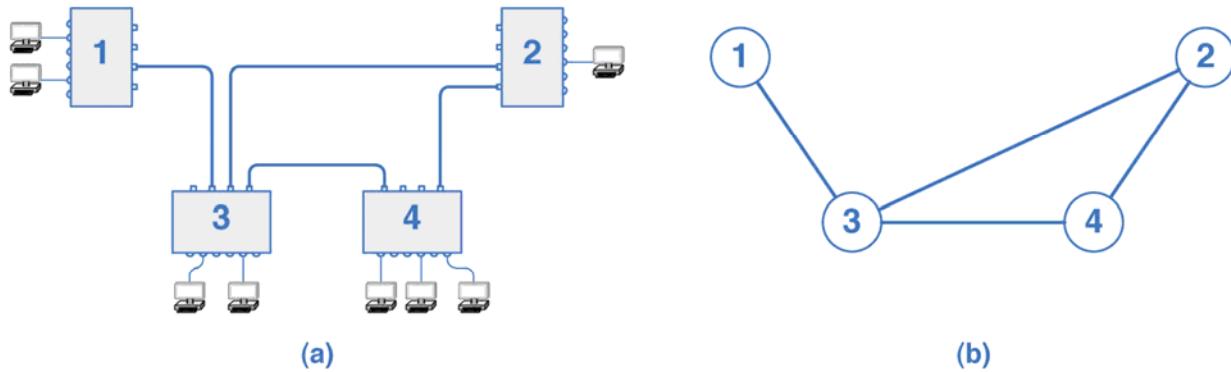


Figure 18.6 Illustration of a WAN and the corresponding graph.

to reach	next hop
1	±
2	(1,3)
3	(1,3)
4	(1,3)

node 1

to reach	next hop
1	(2,3)
2	±
3	(2,3)
4	(2,4)

node 2

to reach	next hop
1	(3,1)
2	(3,2)
3	±
4	(3,4)

node 3

to reach	next hop
1	(4,3)
2	(4,2)
3	(4,3)
4	±

node 4

Figure 18.7 A forwarding table for each node in the graph of Figure 18.6b.

使用默认路由 (Default Routes)

- 默认路由

- 将转发表中具有相同下一跳值的条目列表用单个条目替换
- 只允许一个默认条目

- 默认路由是可选的

- 仅当多个目的地具有相同的下一跳值时才存在默认条目

to reach	next hop
1	±
*	(1,3)

node 1

to reach	next hop
2	±
4	(2,4)
*	(2,3)

node 2

to reach	next hop
1	(3,1)
2	(3,2)
3	±
4	(3,4)

node 3

to reach	next hop
2	(4,2)
4	±
*	(4,3)

node 4



路由表计算

- 广域网需要进行分布式路由计算
 - 而非一个集中的程序计算所有最短路径
 - 每个交换机必须计算自己的转发表
 - 本地所有交换机必须参与分布式路由计算
- 两种形式
 - 链路状态路由（Link-State Routing）
 - 距离矢量路由（Distance-Vector Routing，DVR）

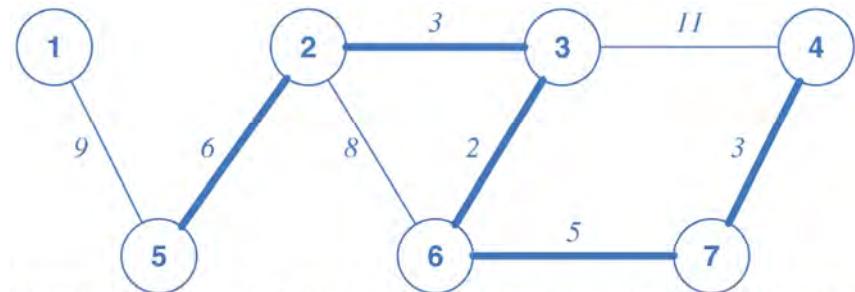


Figure 18.9 An example graph with a weight assigned to each edge and a shortest path between nodes 4 and 5 shown darkened.

链路状态路由（LSR）

- 工作方式是Dijkstra算法，又称为最短路径优先路由
- 分组交换机定期发送横跨链路状态的网络消息
 - 例如，分组交换机5和9测量它们之间的链路并发送状态消息如“5和9之间的链接是通的”
 - 每个状态消息都广播到所有交换机
- 每个交换机收集传入状态消息并利用它们建立网络图
- 然后使用算法产生一个转发表，通过选择本身作为源



链路状态路由（LSR）

- LSR算法能够适应硬件故障
- 如果分组交换机之间的链路失败
 - 连接的交换机检测故障，广播指定链接已关闭的状态消息
 - 所有分组交换机接收广播，更改图的副本以反映链接状态的变化，重新计算最短路径
- 当一个链接再次可用
 - 连接到链路的分组交换机检测到它正在工作
 - 开始发送状态消息，报告其可用性

距离矢量路由 (DVR)

- 如同LSR，网络中的每一个链路都分配一个权重
 - 两个交换机间的目的地的距离被定义为两者之间的路径的权重的总和
- DVR安排交换机定期交换信息，但消息不被广播
- 发送一个完整的清单，目的地和到达各项的当前成本
- 当发送一个DVR的消息，交换机发送一系列单独语句
 - 形式（目的地，距离）：“我到达目的地X的距离是y”
 - 每个交换机周期性地发送一个消息给邻居



距离矢量路由 (DVR)

- DVR维持一个扩展的转发表，为每个目的地存储距离
 - 每个交换机必须保留一个目的地列表以及当前到达目的地下一跳的列表，以便在转发表中找到目的地列表和下一跳
- 当消息从邻居N到达交换机时，检查消息中的每个项
 - 如果邻居对某些目标拥有较短的路径，则更改其转发表
 - 例子：如果邻居N广告路径到目的地D的成本为5，和当前路径通过邻居K成本为100，D的当前下一跳将被N替换，而达到D的成本将是5加上达到N的成本



路由的问题

- 理论上，LSR或DVR计算最短路径，最终会收敛
 - 表示所有分组交换机中的转发表一致
- 如果LSR消息丢失，两个交换机的最短路径不一致
- DVR消息丢失的问题会更严重
 - 主要原因之一DVR协议具有来自反冲洗的问题
 - 例如，假设一个交换机告诉邻居“我可以以成本3到达目的地D”
 - 如果连接导致目标D失败交换机将从其转发表中删除D项（或标记无效），但交换机已经告诉邻居存在路由，将创建一个路由循环



内容纲要

1

广域网技术与路由

2

协议与分层



计算机网络体系结构的形成

- 协议（ protocol ），即：网络协议
 - 参与通信的各方在交换消息时，必须遵守指定的消息格式，以及消息所需的行动规则。
 - 实施这些规则的软件称为协议软件（ protocol software ）。
- 分层模型（ layering model ）
 - 将通信问题划分成模块，为每个模块设计一个单独的协议。
 - 这使得每个协议更容易设计，分析，实现和测试。
 - 每个协议应该处理其他协议无法处理的通信问题的一部分。



协议系列（ Protocol Suites ）

- 协议系列（ suites ）或族（ family ）。
 - 协议的设计应为完整的、相互配合的集合
 - 不是孤立地创建每个协议
 - 协议组合应处理所有可能的硬件故障或其他异常情况。
 - 每个协议解决通信问题的一部分，一起解决整个通信问题。
- 本次课学习每个层的目的和协议用于通信是足够的
- 每个层的细节在整本书其它课程展开



两种国际标准

- 标准

- 只要遵循标准，一个系统就可以和位于世界上任何地方的、也遵循这同一标准的任何系统进行通信。
- 法律上的(*de jure*)国际标准：ISO/OSI 模型
- 事实上的(*de facto*)标准：TCP/IP 模型

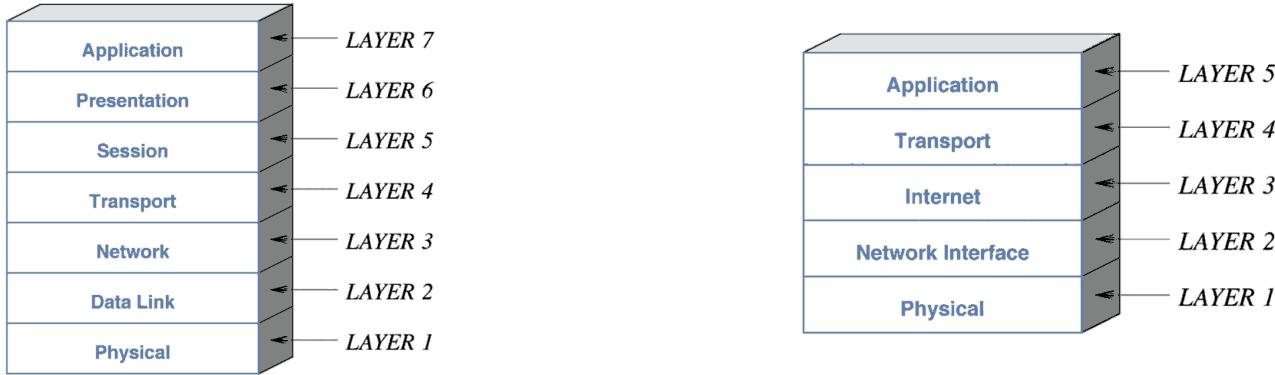


Figure 1.1 The layering model used with the Internet protocols (TCP/IP).

Figure 1.4 The OSI seven-layer model standardized by ISO.

分层的好处

- 分层的好处

- 各层之间是独立的，灵活性好，结构上可分割开。
- 易于实现和维护。
- 能促进标准化工作。

- 层数要适当

- 层数太少会使每一层的协议太复杂。
- 层数太多会在描述和综合各层功能时遇到较多的困难。

协议的复杂性

- 协议必须把所有不利的条件事先都估计到，而不能假定一切都是正常的和非常理想的。
- 看一个计算机网络协议是否正确，不能光看在正常情况下是否正确，而且还必须非常仔细地检查这个协议能否应付各种异常情况。



OSI/ISO参考模型

- OSI/ISO参考模型

- 英文 : Open Systems Interconnection Reference Model
- (1) 物理层 ; (2) 数据链路层 ; (3) 网络层 ;
- (4) 传输层 ; (5) 会话层 ; (6) 表示层 ; (7) 应用层

- 在市场化方面 OSI 失败的原因

- 专家们在完成 OSI 标准时没有商业驱动力 ;
- 定周期太长 , 按 OSI 标准生产的设备无法及时进入市场 ;
- 层次划分不合理 , 有些功能在多个层次中重复出现。



TCP/IP协议族

- TCP/IP协议族

- (1) 网络接口层：最下面的网络接口层并没有具体内容；
 - (2) 网际层；(3) 传输层；(4) 应用层

- 不足

- 网际层之下的网络接口层较为笼统，没有具体内容

五层协议体系结构

- 往往折中综合 OSI 和 TCP/IP 的优点，采用五层协议
 - 应用层(application layer)
 - 传输层(transport layer)
 - 网络层(network layer)
 - 数据链路层(data link layer)
 - 物理层(physical layer)



分层的协议

- 承上启下
 - 两个对等实体间的通信使得本层能够向上一层提供服务。
 - 要实现本层协议，还需要使用下层所提供的服务。
 - 本层的服务用户只能看见服务而无法看见下面的协议。
 - 下面的协议对上面的服务用户是透明的。
- 协议是水平的：协议是控制对等实体之间通信的规则。
- 服务是垂直的：服务是由下层向上层通过层间接口提供的。

ISO-OSI层次模型的软件功能

- 分层的数据结构
 - 数据头
 - 抽象的概念，不一定在开始位置
 - 有效数据
 - 有可能经过一定的修正
- 层次间的调用关系
 - 底层为上层提供服务访问点(SAP)作为接口
 - 上层通过统一的SAP底层功能

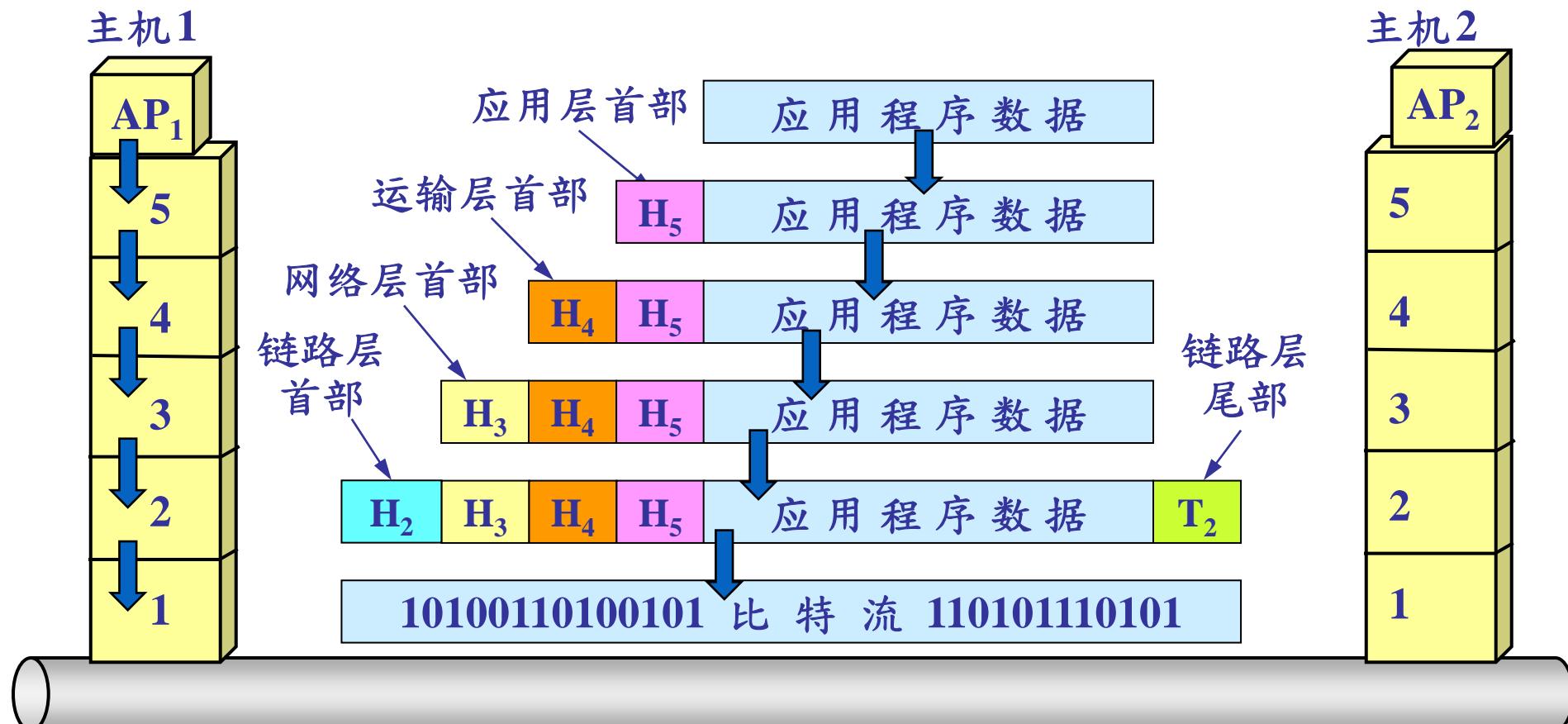
数据在各层次之间的传输

- 加入或剥去首部（尾部）的层次
 - 发送计算机上的给定层中的软件向传出数据添加信息。
 - 接收计算机上同一层的软件使用附加信息处理输入的数据。

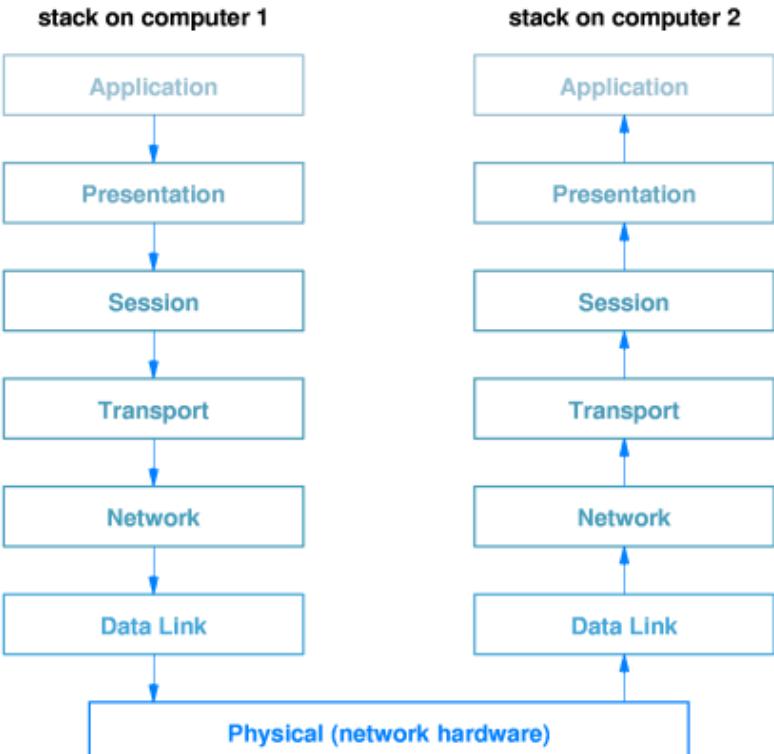


数据在各层次之间的传输

- 加入或剥去首部（尾部）的层次



OSI七层模型



OSI七层模型的作用

- (1) 物理层
 - 完成对bit和载波之间的转换
 - 处理与物理传输介质相关的接口
- (2) 数据链路层
 - 成帧（包括查错控制）
 - 介质访问控制子层(MAC)
 - 逻辑链路控制子层(LLC)

OSI七层模型的作用

- (3) 网络层

- 主机到主机间尽力而为的通信
- 路由寻径：维护路由表和根据路由表查询转发
- 通过询问和差错报告，确保网络连接

- (4) 传输层

- 进程间端到端的通信
- 提供传输的可靠性



OSI七层模型的作用

- (5) 会话层
 - 会话建立、撤销
 - 传输同步
 - 面向连接的交互活动管理：口令认证，数据传输规范
- (6) 表示层
 - 信息压缩
 - 信息加密、解密
 - 与标准格式的转换

OSI七层模型的作用

- (7) 应用层

- 提供最通用的应用程序（电子邮件、BBS、Web、文件传输等）
- 完成用户信息或者软件转换信息的交互

协议分层

- 因为每个堆栈都是独立设计的，协议不能从一个给定的堆栈与另一个协议交互。
- 计算机可以同时运行多个堆栈。

Vendor (供应商)	Stack (栈)
Novell Corporation	Netware
Banyan System Corporation	VINES
Apple Computer Corporation	AppleTalk
Digital Equipment Corporation	DECNET
IBM	SNA
(many vendors)	TCP/IP

计算机网络

Computer Network

9

谢谢观看

理论教程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

10

IP编址和报文

理论课程

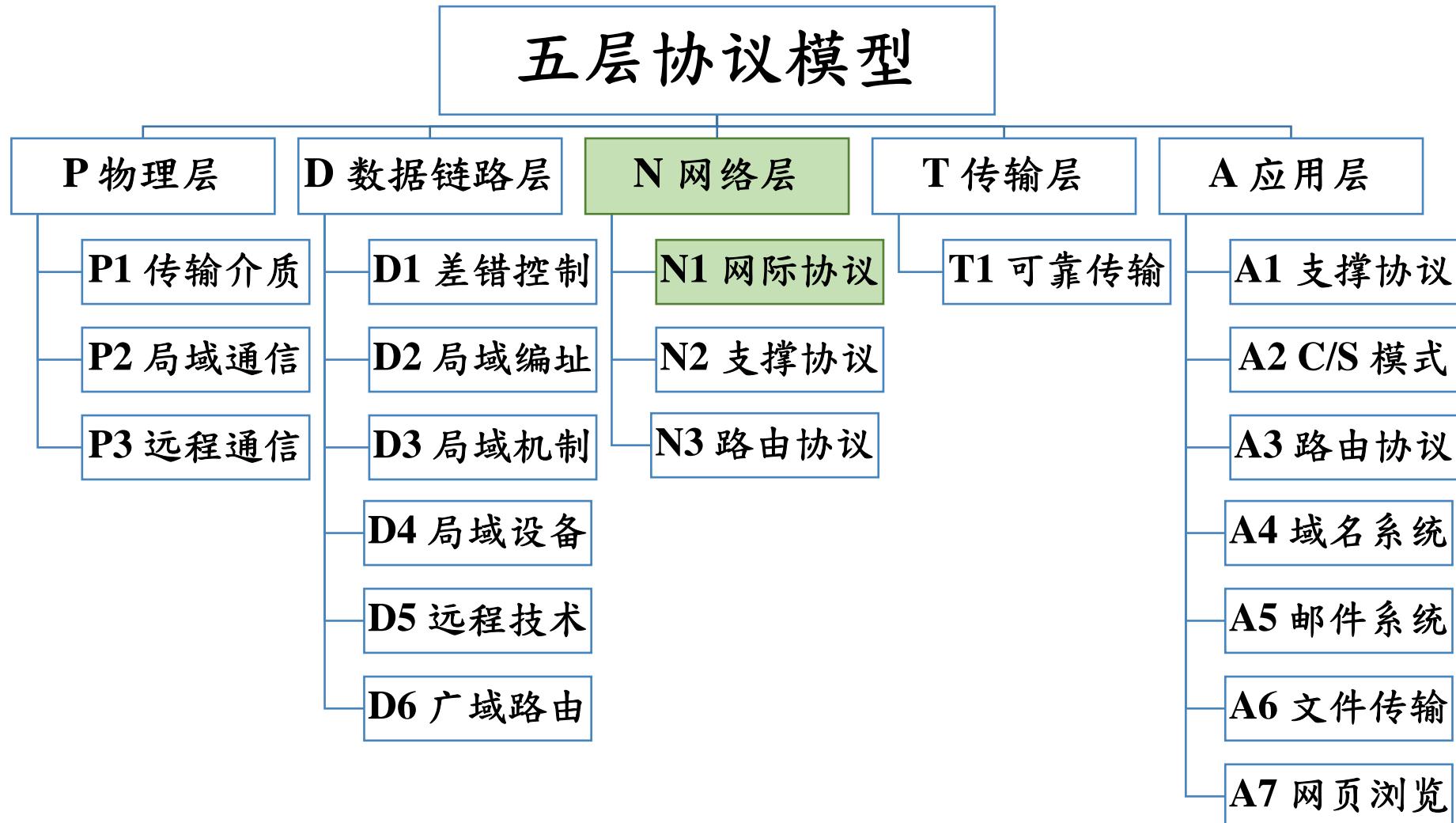


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 世界互联的概念和结构、虚拟网络
- IP（互联网协议）
 - IP编址方案、点分十进制表示法、CIDR表示法
 - 有类地址、无类地址
 - 子网划分（原则）、子网掩码
 - 特殊IP地址
- IP数据报的报头格式：组成和作用（不要求顺序）



主要内容

- IP数据报和数据报转发
 - IP路由表和路由转发的原理
- IP封装、分段与重组
 - IP封装：报文跨互联网传输时的数据链路层行为
 - MTU、IP分段与重组



对应课本章节

- PART IV Internetworking
 - Chapter 20 Internetworking: Concepts, Architecture, and Protocols
 - Chapter 21 IP: Internet Addressing
 - Chapter 22 Datagram Forwarding

内容纲要

1

互联网络

2

互联网协议地址

3

子网划分

4

IP数据报格式

5

IP数据报转发



世界互联的动机

- 世界互联的困难
 - 底层网络（机制、帧格式）、主机性能各异
- 统一网络技术 还是 多个网络技术、统一网络服务？
 - 网络间的不兼容使得仅通过导线连接不同网络是不可能的
- 网络互联（ Internetworking ）
 - 由此产生的连通物理网络的系统称为互联网（ internetwork ，或者小写 internet ）
 - 当前正在使用的互联网络：因特网（ 首字母大写 Internet ）



路由

- 路由 (Routing)

- 为单网络中、多网络间或跨多网络的流量选择路径的过程。
 - 两个局域网；局域网和广域网；或两个广域网
- 广义上，路由在许多类型的网络中执行，包括：
 - 电路交换网络，例如公共交换电话网（PSTN）。
 - 计算机网络，例如因特网。
- 狹义上，路由通常指IP路由。
- 路由：大型网络中的结构化寻址
- 桥接：局域网中的非结构化寻址



路由器

- 路由器（ Router ）
 - 路由器是在计算机网络之间转发数据包的网络设备。
- 网关（ Gateway ）
 - 允许数据从一个离散网络流向另一个离散网络的网络硬件。
 - 被配置来执行网关任务的计算机或计算机程序。
- 在RFC文档中，路由器和网关是同一个概念。
- 包含：处理器和内存以及所连接网络的单独I/O接口



虚拟网络

- 互联网提供了表面上单一的无缝通信系统
- 硬件和软件的结合提供了一个统一的网络系统的错觉
 - 互联网软件隐藏细节物理网络连接物理地址路由信息
 - 应用不需要知道底层物理硬件或路由器的存在

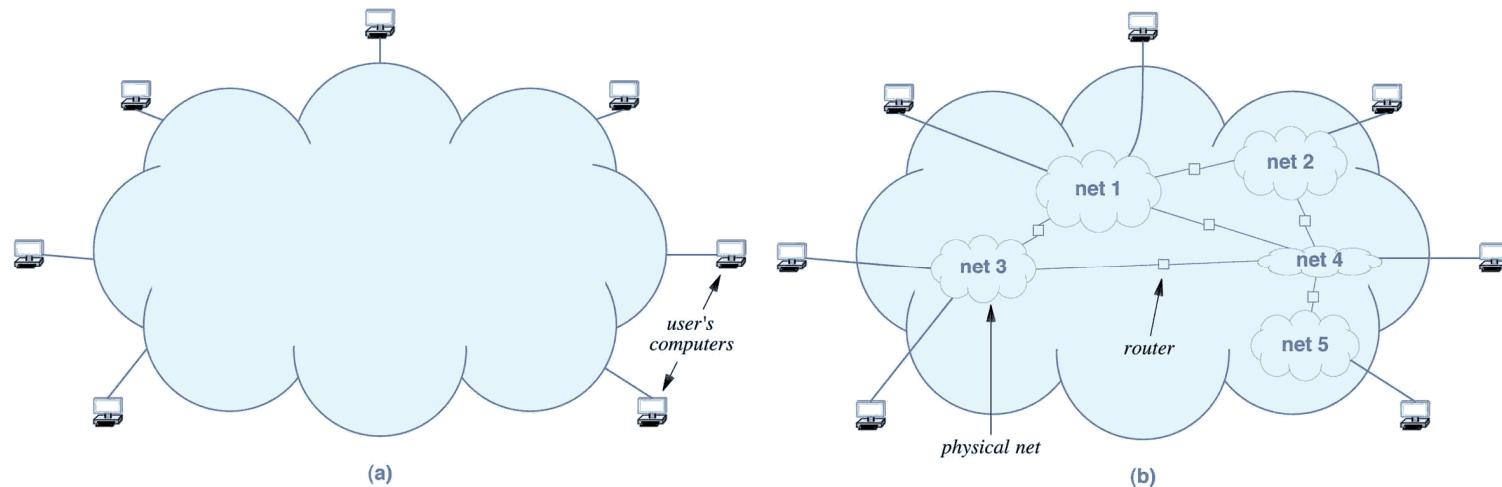


Figure 20.3 The Internet concept. (a) The illusion of a single network provided to users and applications, and (b) the underlying physical structure with routers interconnecting networks.

Internet协议（IP）

- Internet协议（Internet Protocol）

- 协议族中用于跨网络边界中继数据报的主要通信协议。

- 版本

- 实验版本：IPv0~IPv3；历史版本：IPv5，IPv8
 - 第一个主要版本：IPv4；继任版本：IPv6
 - 愚人节笑话（1994）版：IPv9；保留版本：IPv15



内容纲要

1

互联网络

2

互联网协议地址

3

子网划分

4

IP数据报格式

5

IP数据报转发



虚拟互联网的地址：从硬件到软件

- 因特网是由设计师想象的完全由协议软件实现的
 - 物理层：异构网络的编址“各自为政”
 - 软件层：需要一个编址来隐藏异构的物理细节
- 与物理层独立、统一的编址

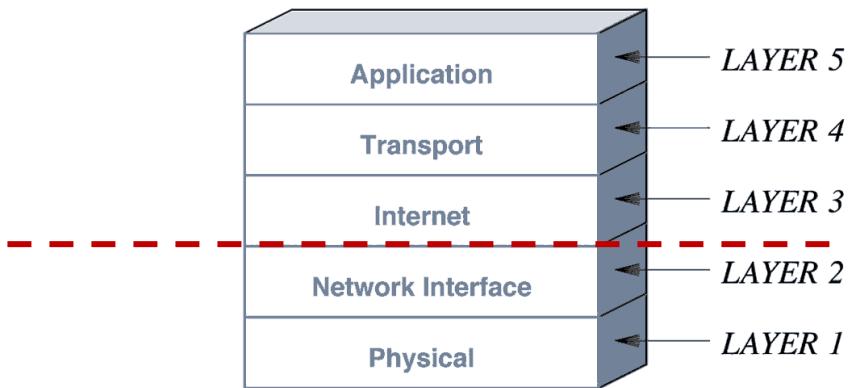


Figure 1.1 The layering model used with the Internet protocols (TCP/IP).

IP地址（IPv4地址）

- Internet协议地址（IP地址）

- 分配给连接到使用IP协议进行通信的每个设备的数字标签。
 - 主要功能：主机或网络接口标识；位置寻址

- 地址所有权

- 互联网名称与数字地址分配公司（ICANN）是处理地址分配和裁决争端的机构
 - ICANN授权注册商分配个人前缀
 - ISP向用户提供地址获取前缀（公司通常与ISP联系）



IP地址编址

- 点分十进制记数法（ Dotted decimal notation ）
 - 每8位作为无符号十进制值（ 0~255 ）并用点分隔
- IP地址（ 或互联网地址 ）：全球唯一的32位数字
 - 网络号，标识主机附加的物理网络，全球协调
 - 主机号，标识网络上的特定计算机，局域网内协调

二进制：11000000 00000101 00110000 00000011

十进制： 192 . 5 . 48 . 3



IP地址编址方法

- 分类IP地址（1981~1993）
 - 将IP地址空间分为5大类，是最基本的编址方法。
- 子网划分（1985~）
 - 主机号的部分位用于表示子网号，对分类地址方法的改进。
- 无分类IP地址（1993~）
 - 灵活调整网络大小。



传统分类地址：ABCDE类

- 机制：根据最前的位内容来确定具体属于哪一类
- 存续时间：1981-1993

类	位前缀	网络位	主机位	网络容量	网络中主机容量	起始地址	结束地址	子网掩码
A	0	8	24	128 (2^7)	$16,777,216$ (2^{24})	0.0.0.0	127.255.255.255	255.0.0.0
B	10	16	16	$16,384$ (2^{14})	$65,536$ (2^{16})	128.0.0.0	191.255.255.255	255.255.0.0
C	110	24	8	$2,097,152$ (2^{21})	256 (2^8)	192.0.0.0	223.255.255.255	255.255.255.0
D	1110			多播地址		224.0.0.0	239.255.255.255	无
E	1111			保留地址		240.0.0.0	255.255.255.255	无

特殊的IP地址

- 主机号全0代表网络，主机号全1代表广播
- 一些IP地址是保留的，不分配给主机

名词	英文名词	规则	示例
网络地址	Network address	主机号全0	59.0.0.0
直接广播地址	Directed Broadcast Address	主机号全1	59.255.255.255
有限广播地址	Limited Broadcast Address	地址全1	255.255.255.255
本机地址	This Computer address	全0	0.0.0.0
回送地址	Loopback address	127.0.0.0/8	127.0.0.1

地址掩码 (Address Masks)

- 地址掩码指示IP地址中主机所在子网地址的位掩码
- 由连续的 N 位1和连续的 $32-N$ 位0构成，共有33种
- 作用：求取网络地址（网络号）

$$N = (D \& M)$$

名称	变名	二进制表示				点分十进制表示
网络地址	N	10000000	00001010	00000000	00000000	128.10.0.0
网络掩码	M	11111111	11111111	00000000	00000000	255.255.0.0
目标地址	D	10000000	00001010	00000010	00000011	128.10.2.3

无类域间路由（ CIDR ）表示法

- 无类域间路由（ Classless Inter-Domain Routing ）
 - 用于分配地址与路由汇总时表示地址块
 - 形式：网络号/网络号长度 $ddd.ddd.ddd.ddd/m$
 - d为网络号，m为掩码中1的个数（不一定是8的倍数）
 - 示例：192.5.48.64/26（正确）；192.168.1.1/24（错误）

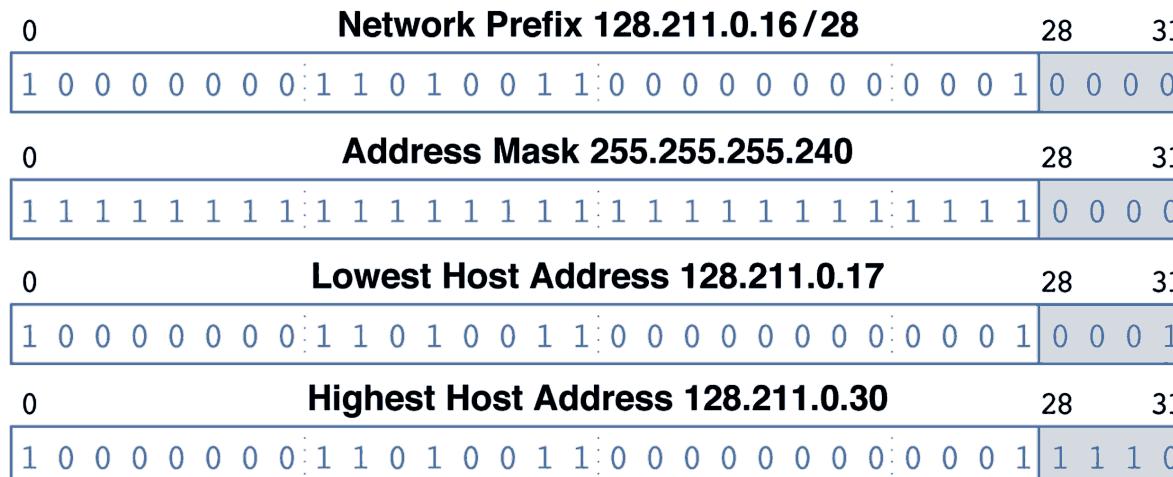


Figure 21.6 Illustration of CIDR addressing for an example /28 prefix.

内容纲要

1

互联网络

2

互联网协议地址

3

子网划分

4

IP数据报格式

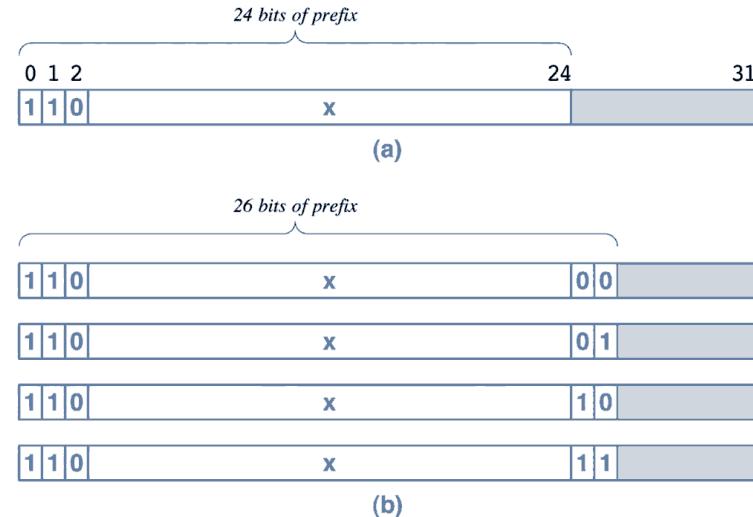
5

IP数据报转发



子网划分

- 在 ARPANET 的早期，IP 地址的设计确实不够合理。
 - IP 地址空间利用率有时很低。
 - 每个物理网络分配一个网络号会使路由表太大从而网络性能变坏
- 两级 IP 地址变为三级 IP 地址
 - 1985 年起增加 “子网号字段”
 - 将网络进一步划分成子网，可以便于管理
 - 从主机号借若干位作为子网号，主机号相应减少位
 - IP 地址 ::= {<网络号>, <子网号>, <主机号>}

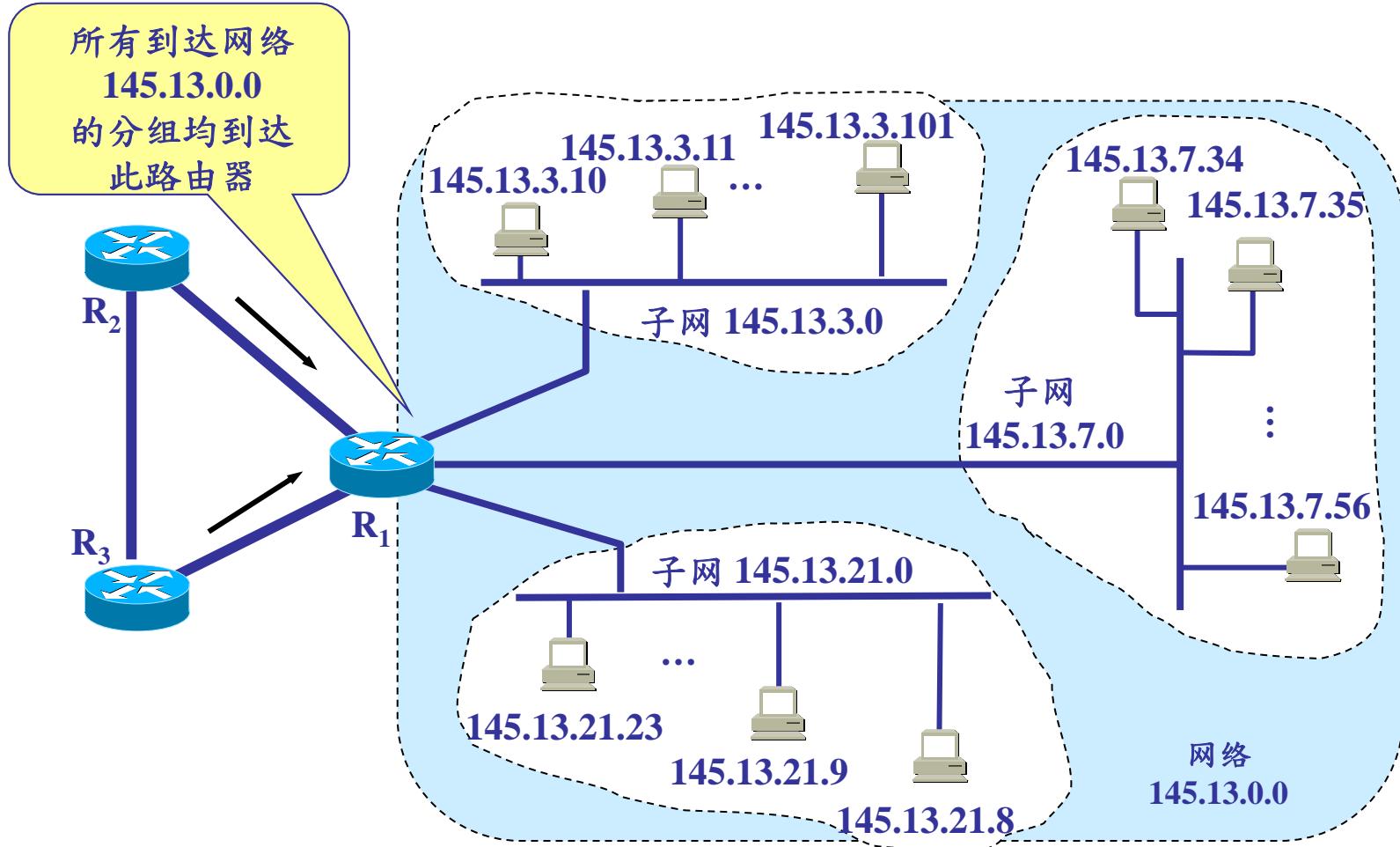


划分子网的基本思路

- 划分子网是单位内部事务，单位对外仍然表现为没有划分子网的网络。
 - 从一个 IP 数据报的首部并无法判断源主机或目的主机所连接的网络是否进行了子网划分。
- 从其他网络发送给本单位某个主机的 IP 数据报，仍根据其目的网络号，找到本单位的路由器。
- 此路由器收到数据报后，提取子网号找到目的子网。
- 最后将 IP 数据报直接交付目的主机。



划分子网的基本思路



子网掩码 (subnet mask)

- 分组转发算法必须做相应的改动
 - 路由器在和相邻路由器交换路由信息时，必须把自己所在网络（或子网）的子网掩码告诉相邻路由器。
 - 路由器的路由表中的每一个项目，除了要给出目的网络地址外，还必须同时给出该网络的子网掩码。
 - 若一个路由器连接在两个子网上就拥有两个网络地址和两个子网掩码。
- 使用子网掩码可以找出 IP 地址中的子网部分。
- 子网划分应剔除主机号或网络号全0全1两种情形
 - RFC 1878 废止：现代软件将能够利用所有可定义的网络。



路由器与IP寻址原理

- 每个路由器有两个或多个IP地址
 - 路由器与多个物理网络有连接，每个IP地址包含特定网络的前缀
- IP地址不标识特定计算机
 - 标识计算机和网络之间的连接
 - 多个网络连接的计算机必须为每个连接分配一个IP地址

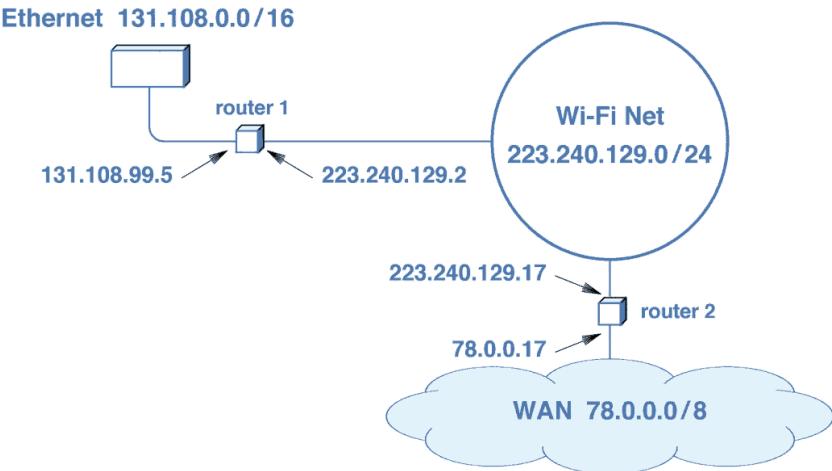


Figure 21.8 An example of IP addresses assigned to two routers.

多穴主机（ Multi-Homed Hosts ）

- 多穴有时用于提高可靠性
 - 如果一个网络失效，主机仍可以通过第二连接到达互联网
 - 连接到多个网络可以直接发送流量，避开拥塞的路由器
- 多宿主主机有多个IP地址，每个网络连接一个地址
 - 有多重连接的原因：负载平衡和速度冗余提高网络可靠性



内容纲要

1

互联网络

2

互联网协议地址

3

子网划分

4

IP数据报格式

5

IP数据报转发



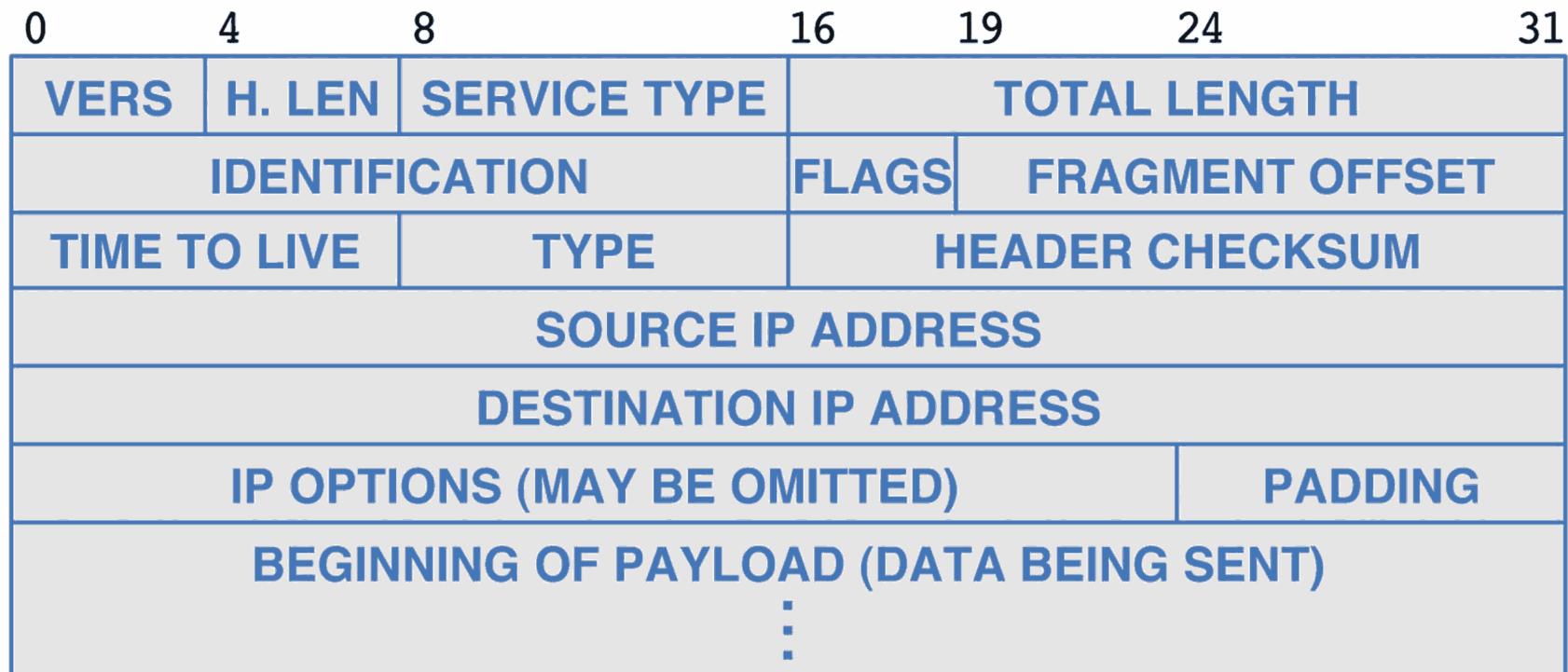
虚拟分组（ Virtual Packet ）

- 网络互联协议定义独立于底层硬件的“分组”格式
 - 为解决异构问题，其结果是一个通用的，虚拟的数据包。
- 底层硬件不理解，路由器和主机（软件）理解
 - 底层硬件不理解或不识别Internet包格式。
 - 因特网中每个主机或路由器都包含理解因特网数据包的协议软件。
- IP数据报（ datagram ）是Internet数据分组的术语。



IP数据报 (IP Datagram)

- IPv4数据报总长（包括头部）为20B至64KB。



IP报文头格式

- IP报文头格式的组成（基本长度：20B）

- 版本：4bits，取值：4或6
- 报头长度：4bits，单位为4Bytes
- 服务类型：8bits，未实际使用
- 报文总长度：16bits，单位为字节
- 标识：16bits，IP软件在存储器中的计数器在产生一个数据报后自增1，并将值赋给标识字段。标识在分片时复制。
- 分片标志：3bits，高到低位：无意义、不分片、还有分片
- 片偏移：13bits，分片在原始报文的位置，单位为8Bytes

IP报文头格式

- IP报文头格式的组成（基本长度：20B）
 - 生存时间（TTL）：8bits，单位为秒，路由器减去在其环节所消耗时间，直至零丢弃。
 - 协议类型：8bits，可能的取值有：ICMP、IGMP、TCP、UDP、OSPF等，用于将数据交给第四层的那个软件。
 - 报头校验和：16bits，检验报头的完整性，不含数据部分。
 - 源IP地址：32bits
 - 目标IP地址：32bits
 - 选项内容：1~40bits，用来支持排错、测量和安全等措施。
 - 填充部分：长度可变，为了使报文头部是4Bytes的整数倍。



内容纲要

2 互联网协议地址

3 子网划分

4 IP数据报格式

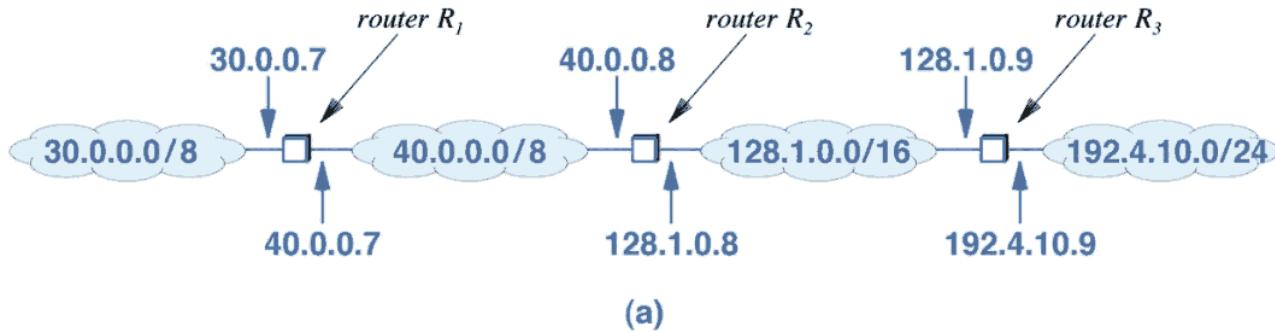
5 IP数据报转发

6 IP封装、分片与重组



路由：转发IP报文

- 路由表的组成：目标网络号、子网掩码、下一跳
 - 下一跳：直接传送标志、子网IP地址
 - 默认路由：0.0.0.0/0



Destination	Mask	Next Hop
30.0.0.0	255.0.0.0	40.0.0.7
40.0.0.0	255.0.0.0	deliver direct
128.1.0.0	255.255.0.0	deliver direct
192.4.10.0	255.255.255.0	128.1.0.9

(b)

IP子网掩码与数据转发

- 通过子网掩码进行计算的路由表匹配
 - 获得IP报文的目标IP地址D
 - 用D顺序逐条匹配路由表各个条目T[0], T[1], T[2]
 - 如果 $D \& T[i].m == T[i].d$, 则下一跳为T[i].n
 - D : 目标端地址
 - T[i] : 路由表中第*i*条目标
 - d 子网网络号 ; m : 子网掩码 ; n : 下一跳IP地址
- 最长前缀匹配 (Longest Prefix Match)
 - 设路由表有以下两个网络前缀 : 128.10.0.0/16; 128.10.2.0/24
 - 对于报文的IP地址128.10.2.3 , 选择128.10.2.0/24



内容纲要

2 互联网协议地址

3 子网划分

4 IP数据报格式

5 IP数据报转发

6 IP封装、分片与重组



尽力而为的传输

- IP要适应不同硬件的需要，但底层硬件可能不起作用
- IP提供一种尽力而为的传输 (Best-Effort Delivery)
 - IP数据报可能会丢失、重复、延迟、乱序，或数据损坏。
 - 需要高层协议软件来处理上述每一个错误



数据报传输与帧

- IP 软件选择下一站，并通过物理网络发送
 - 网络硬件不理解数据报文格式和因特网地址
 - 每个网络格式都有自己的硬件地址
 - 发送方和接收方必须就帧型字段中所使用的值协商一致。
- 发送者必须指定下一接收主机的物理地址
- 封装（ Encapsulation ）

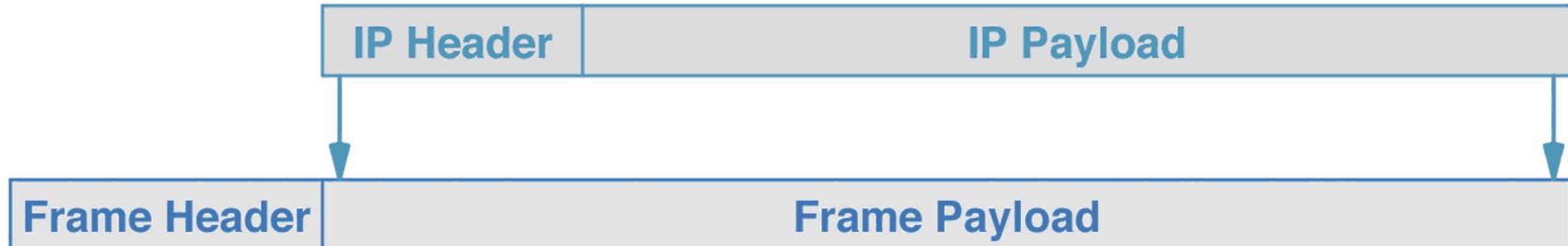
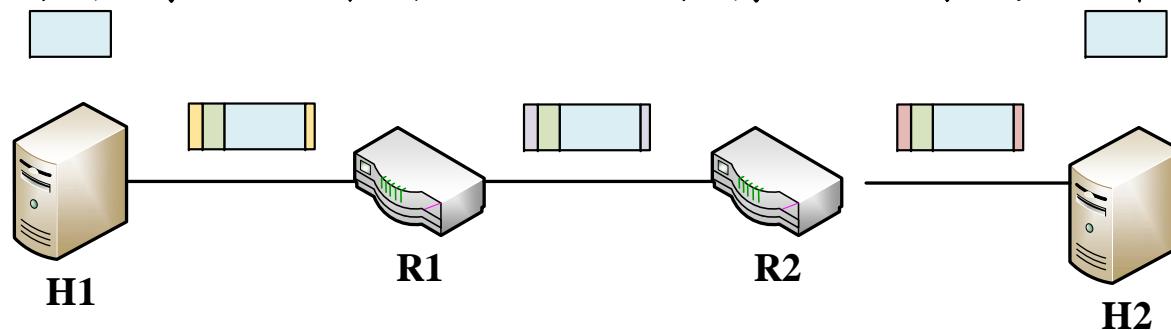


Figure 22.4 Illustration of an IP datagram encapsulated in a frame.

跨互联网传输

- 帧到达下一跳，接收方软件提取IP数据报并丢弃帧头
 - 若还需转发，则再封装
 - 帧头部不积累，主机和路由器不存储额外的头部



项目	值 (H1-R1)	值 (R1-R2)	值 (R2-H2)
源MAC	MAC(H1)	MAC(R1,R)	MAC(R2,R)
目的MAC	MAC(R1,L)	MAC(R2,L)	MAC(H2)
源IP	IP(H1)	IP(H1)	IP(H1)
目的IP	IP(H2)	IP(H2)	IP(H2)

最大传输单元（MTU）

- 最大传输单元：数据链路层帧支持的最大传输字节数
- 当前链路MTU小于IP报文长时，分成较小分片传输
 - 路由器将数据报分成更小的碎片称为分片。
 - 原始数据报首部被复制为各数据报片的首部，但必须修改有关字段的值。
 - 每个分片以IP数据报格式独立发送。

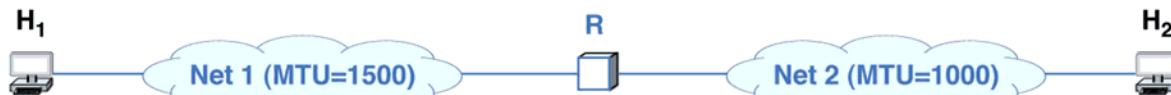


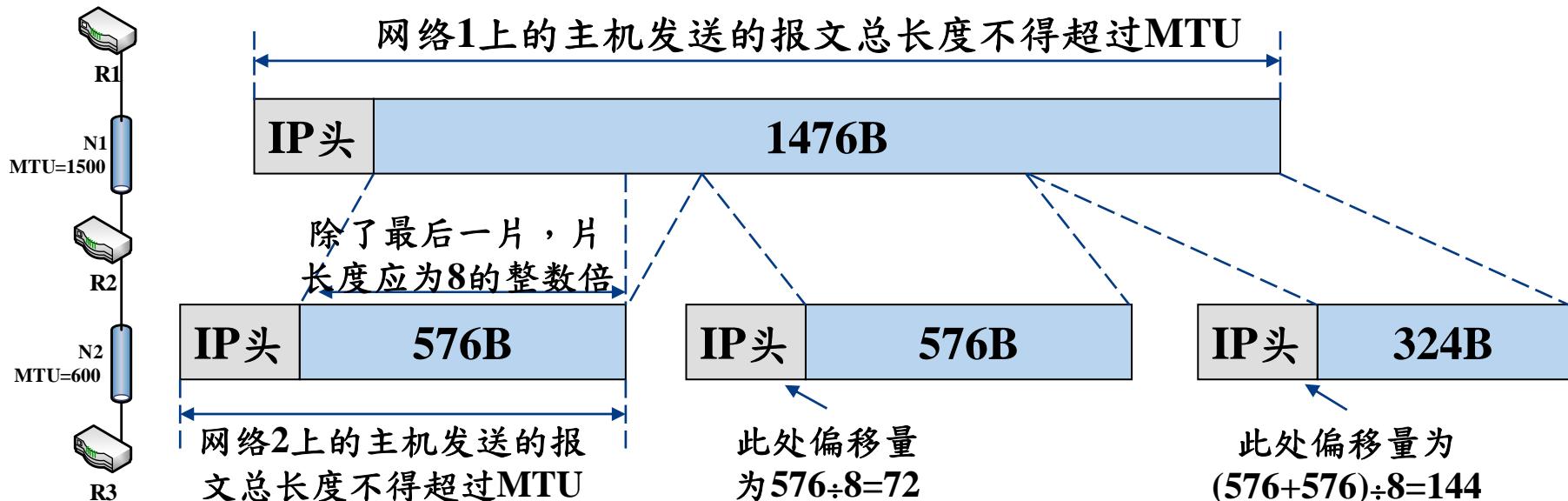
Figure 22.6 Illustration of a router that connects two networks with different MTUs.

Protocol	MTU
Token ring (16Mbps)	17914
Token ring (4Mbps)	4464
FDDI	4352
Ethernet	1500
X.25	576
PPP	296

IP报文的分片策略

- IP报文传输的分片原则

- 各片尽可能大，尽量少分片，但每片不能超过MTU
- IP头部固有长度（基本长度20字节），也在帧的载荷内
- 分片应使得后续片偏移量为8的整数倍



分片信息表示

- IP报文中的相关信息

- IP报文中的**ID**：分片时复制，始终保持初始**ID**不变
- 标志位：若第一次分片，则修改“是否分片”相应位
- 片偏移量：表示当前分片在初始IP包中有效数据的偏移位置（8字节为单位）
 - 标志 **MF(more fragment)**, **MF=1** 表示后面还有分片；**MF=0** 表示这是最后一个分片；**DF(don't fragment)**, **DF=1** 表示不允许分片；**DF=0** 表示允许分片。



IP报文的分片策略

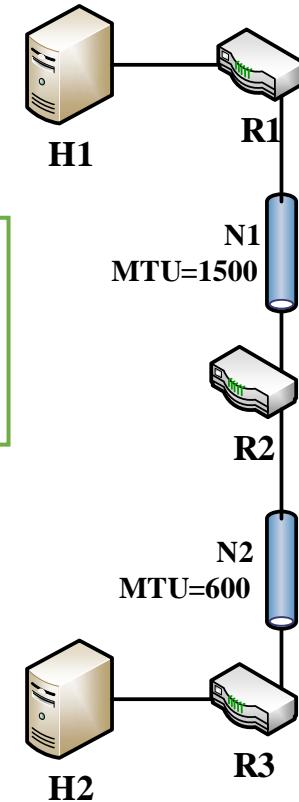
- 例题：某主机H1上的IP数据报文长度为3820字节（含固定报文头20字节），发送给主机H2。途经网络N1、N2的MTU如右图所示，请写出其在N1和N2上的分片情况，DF=0。

- 解：

- R1对报文的分片

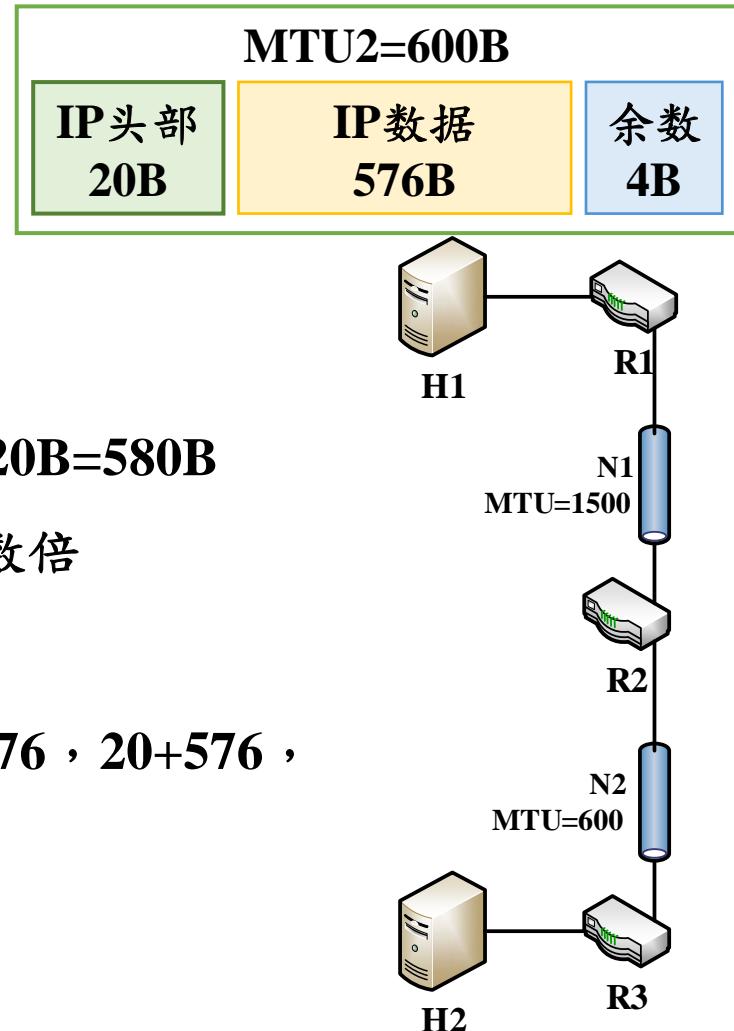


- 分片的数据总长度应为 $3820B - \text{头部}20B = 3800B$
- 每片数据长度不超过MTU – 头部20B = 1480B
- 除最后一片其余各片应为8B的整数倍



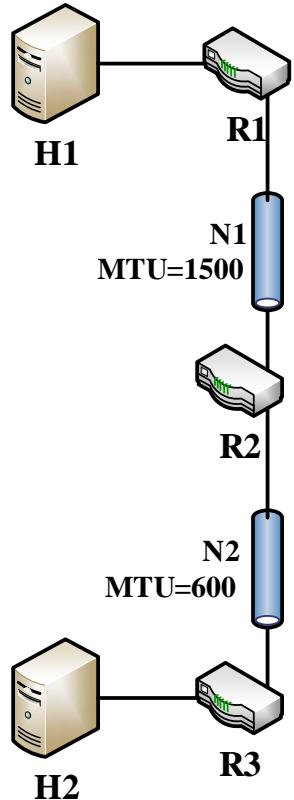
IP报文的分片策略

- R1对报文的分片结果
 - $20+1480$, $20+1480$, $20+840$
- R2对报文的分片情况
 - 分片不重组，分片再分片
 - 每片数据长度不超过MTU—头部 $20B=580B$
 - 除最后一片其余各片应为 $8B$ 的整数倍
- R2对报文的分片结果
 - $20+576$, $20+576$, $20+328$, $20+576$, $20+576$,
 $20+328$, $20+576$, $20+264$



IP报文的分片策略

• 答案



类别	序号	总长度	数据长度	MF	片偏移
原数据报	1	3820	3800	0	0
R1发给R2 (在N1) 的数据报	1	1500	1480	1	0
	2	1500	1480	1	185
	3	860	840	0	370
R2发给R3 (在N2) 的数据报	1	596	576	1	0
	2	596	576	1	72
	3	348	328	1	144
	4	596	576	1	185
	5	596	576	1	257
	6	348	328	1	329
	7	596	576	1	370
	8	284	264	0	442



重组 (Reassembly)

- 报文重组策略

- 源端到目标端数据传输过程中可能有~~多次~~分片
- 所有分片重组在目标端进行，中间路由设备不做分片重组
 - 减少中间节点的数据处理过程
- 碎片还可以再分片 (further fragment a fragment)

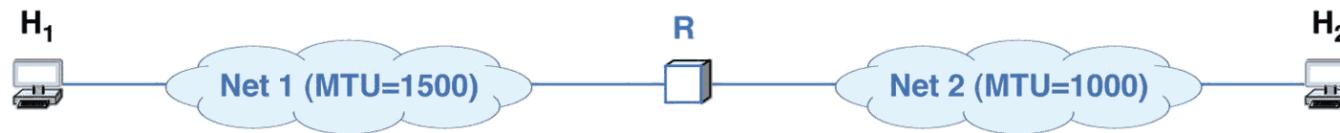


Figure 22.6 Illustration of a router that connects two networks with different MTUs.

Copyright © 2009 Pearson Prentice Hall, Inc.

IP报文丢失问题

- IP报文丢失判断

- 目标端对IP报文分片作重组处理的时候进行丢失判断
- 对应于源端发出的每一个报文，在收到第一个分片的时候，给出一个等待的有限时间T-out，如果T-out之后还没有收到全部分片，则为超时
- 任何一个分片丢失或数据出错，则丢弃整个报文

计算机网络

Computer Network

10

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



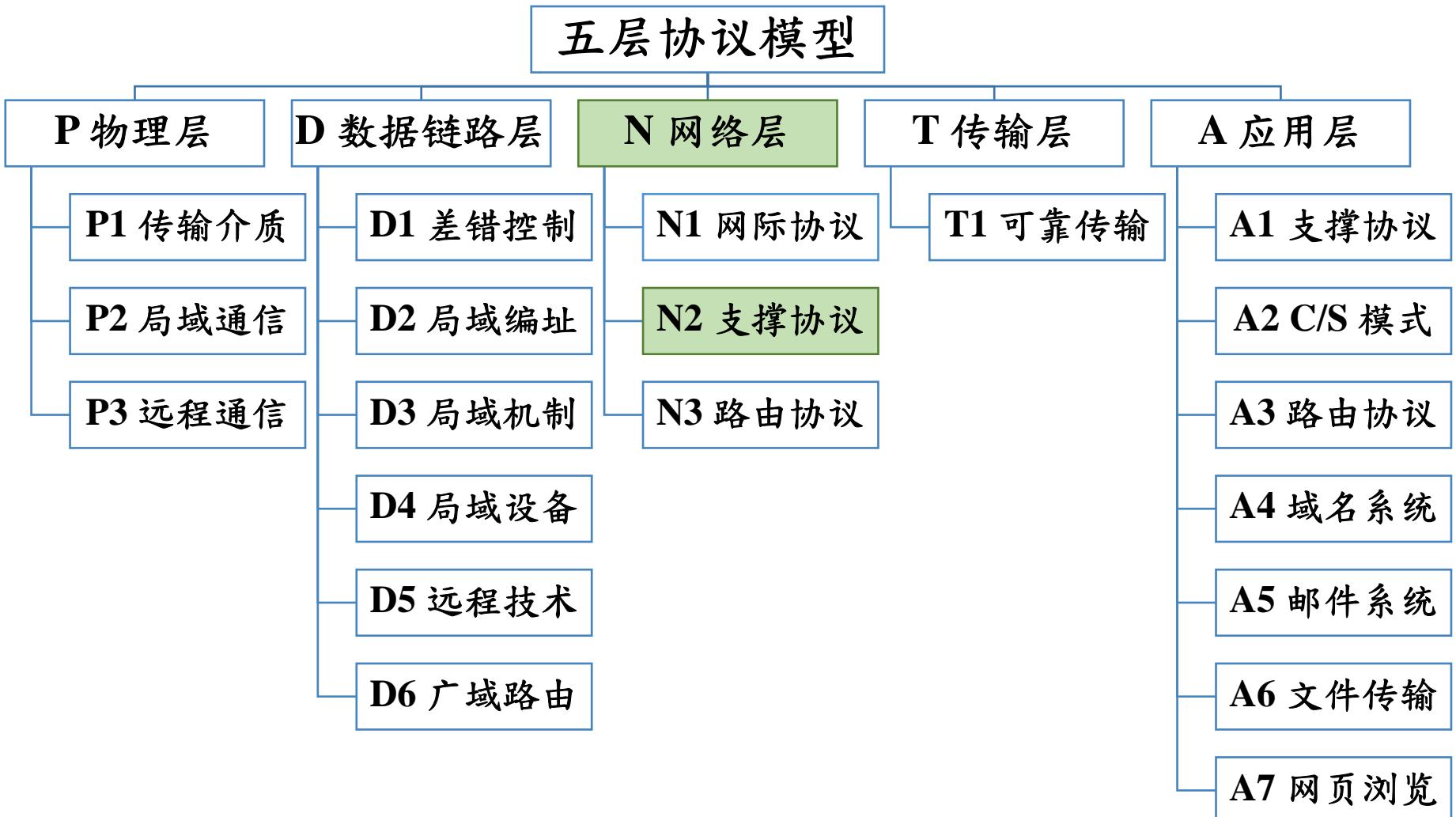
信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

11

IP 支撑协议 和 IPv6



知识框架



主要内容

- ICMP协议
 - ICMP的报文种类、主要功能
 - IP与ICMP的关系
 - ping 命令测试可达性的原理
 - tracert 命令追踪路由的原理
 - 使用ICMP发现MTU
- ARP协议
 - 地址解析，地址解析的方法



主要内容

- 支撑协议与技术
 - DHCP、私有地址和NAT技术
- IPv6
 - IPv4地址的瓶颈
 - 地址格式

对应课本章节

- PART IV Internetworking
 - Chapter 23 Support Protocols And Technologies
 - Chapter 24 The Future IP (IPv6)

内容纲要

1

差错报告机制（ ICMP ）

2

地址解析协议（ ARP ）

3

动态主机配置协议（ DHCP ）

4

网络地址转换（ NAT ）

5

未来的IP：IPv6



Internet控制报文协议（ ICMP ）

- Internet控制报文协议（ ICMP ）
 - Internet Control Message Protocol
 - 目的：提高 IP 数据报交付成功的机会。
 - 协议层：网络层
- 机制
 - 主机或路由器报告差错情况和提供有关异常情况的报告。
- ICMP 报文作为 IP 层数据报的数据，加上数据报的首部，组成 IP 数据报发送出去。

ICMP 错误报文与信息报文

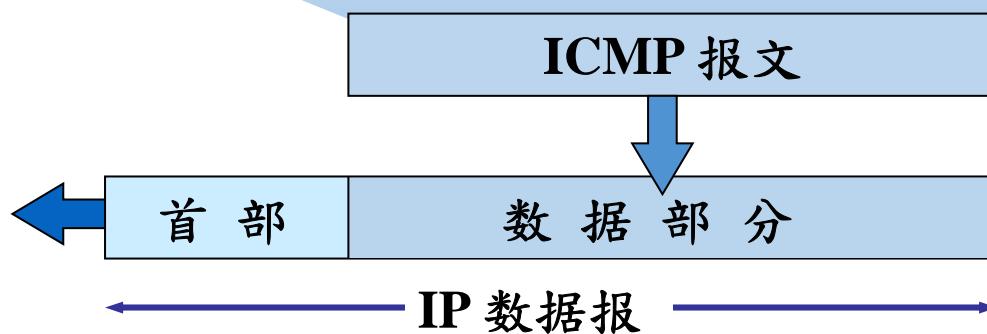
- ICMP 报文的种类
 - 差错报告报文 和 询问报文。

报文种类	类型值	说明
差错 报告 报文 Error Messages	3	目的不可达
	4	源站抑制 (Source Quench)
	11	超时 (Time Exceeded)
	12	参数问题 (Parameter Problem)
	5	重定向 (Redirect)
询问 报文 Informational Messages	8 / 0	回应请求/应答 (Echo Request/Reply)
	13 / 14	时间戳请求/应答 (Timestamp Request/Reply)
	17 / 18	地址掩码请求/应答 (Address Mask Request/Reply)
	10 / 9	路由器请求/通告 (Router Solicitation / Advertisement)



ICMP 报文格式

- ICMP 报文格式



ICMP 询问报文：目的请求

Ethernet Type 2

Destination: 00:50:56:FC:52:95 VMware:FC:52:95 [0-5]
Source: 00:0C:29:17:29:CA VMware:17:29:CA [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Header Length: 5 (20 bytes) [14 Mask 0x0F]

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]
Time To Live: 128 [22]
Protocol: 1 ICMP - Internet Control Message Protocol [23]
Header Checksum: 0x0000 Checksum invalid. Should be: 0x1128 [24-25]
Source IP Address: 192.168.7.4 [26-29]
Dest. IP Address: 123.125.114.144 [30-33]

ICMP - Internet Control Messages Protocol

ICMP Type: 8 Echo Request [34]
ICMP Code: 0 [35]
ICMP Checksum: 0x4D5A [36-37]
Identifier: 0x0001 [38-39]
Sequence Number: 1 [40-41]
ICMP Data Area: abcdefghijklmnopqrstuvwxyz [42-73]

ICMP 询问报文：目的应答

Ethernet Type 2

Destination: 00:0C:29:17:29:CA VMware:17:29:CA [0-5]
Source: 00:50:56:FC:52:95 VMware:FC:52:95 [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Header Length: 5 (20 bytes) [14 Mask 0x0F]

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]
Time To Live: 128 [22]
Protocol: 1 ICMP - Internet Control Message Protocol [23]
Header Checksum: 0xB747 [24-25]
Source IP Address: 123.125.114.144 [26-29]
Dest. IP Address: 192.168.7.4 [30-33]

ICMP - Internet Control Messages Protocol

ICMP Type: 0 Echo Reply [34]
ICMP Code: 0 [35]
ICMP Checksum: 0x555A [36-37]
Identifier: 0x0001 [38-39]
Sequence Number: 1 [40-41]
ICMP Data Area: abcdefghijklmnopqrstuvwxyz [42-73]

ICMP 差错报告报文：超时

Ethernet Type 2

Destination: 00:0C:29:17:29:CA VMware:17:29:CA [0-5]
Source: 00:50:56:FC:52:95 VMware:FC:52:95 [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Header Length: 5 (20 bytes) [14 Mask 0x0F]

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]
Time To Live: 128 [22]
Protocol: 1 ICMP - Internet Control Message Protocol [23]
Header Checksum: 0x8288 [24-25]
Source IP Address: 192.168.7.2 [26-29]
Dest. IP Address: 192.168.7.4 [30-33]

ICMP - Internet Control Messages Protocol

ICMP Type: 11 Time Exceeded [34]
ICMP Code: 0 Time to Live count exceeded [35]
ICMP Checksum: 0xF4FF [36-37]
Unused (must be zero): 0x00000000 [38-41]

ICMP 差错报告报文：超时

Header of packet that caused error follows.

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [42 Mask 0xF0]
Header Length: 5 (*20 bytes*) [42 Mask 0x0F]

...

Fragment Offset: 0 (*0 bytes*) [48-49 Mask 0xFFFF]
Time To Live: 1 [50]
Protocol: 1 *ICMP - Internet Control Message Protocol* [51]
Header Checksum: 0x1C3F [52-53]
Source IP Address: 192.168.7.4 [54-57]
Dest. IP Address: 210.34.0.12 [58-61]

ICMP - Internet Control Messages Protocol

ICMP Type: 8 *Echo Request* [62]
ICMP Code: 0 [63]
ICMP Checksum: 0xF7F7 [64-65]
Identifier: 0x0001 [66-67]
Sequence Number: 7 [68-69]
ICMP Data

Area: [70-133]

ICMP 差错报告报文

- 不应发送 ICMP 差错报告报文的几种情况
 - 对 ICMP 差错报告报文不再发送 ICMP 差错报告报文。
 - 对第一个分片的数据报片的所有后续数据报片都不发送 ICMP 差错报告报文。
 - 对具有多播地址的数据报都不发送 ICMP 差错报告报文。
 - 对具有特殊地址（如 127.0.0.0 或 0.0.0.0）的数据报不发送 ICMP 差错报告报文。



ping 使用 ICMP 消息测试可达性

- PING (Packet Internet Groper) , 因特网包探索器
- 报文类型：ICMP回送请求和回送回复消息。
 - 把包含ICMP回送请求消息的IP数据报发送到指定的目的地。
 - 每当一个回送请求到达，ICMP软件必须发送一个回送应答。
- ping 是应用层直接使用网络层 ICMP 的例子
 - 它没有通过传输层的 TCP 或 UDP 。



tracert 使用 ICMP 追踪路由器

- 每个路由器处理生存周期（TIME TO LIVE）计数器。
 - traceroute 程序发送一系列数据报，等待每一个响应
 - 如果计数器达到零，则路由器丢弃数据报，并将 ICMP 超时错误发送回源。
 - traceroute 不断增加 TTL 值，直到该值足够大到数据报到达其最终目标。
 - 发送 ICMP 回送请求消息；目标主机将生成 ICMP 回送应答。
 - 将数据报发送给不存在的应用程序；目标主机将生成 ICMP 目的地无法到达的消息。



相关命令行程序（Windows为例）

- Ping
 - 因特网分组测程序（ Packet Internet Groper ）
- Tracert
 - 跟踪路由（ Trace Router ）
- Route
 - 路由（ Route ）
- 高级用法，请进入控制台，输入：XXXX /?

检查网络是否连通

C:\Windows\system32>ping www.xmu.edu.cn

正在 Ping www.xmu.edu.cn [210.34.0.12] 具有 32 字节的数据:

来自 210.34.0.12 的回复: 字节=32 时间=1ms TTL=128

210.34.0.12 的 Ping 统计信息:

数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),

往返行程的估计时间(以毫秒为单位):

最短 = 1ms, 最长 = 1ms, 平均 = 1ms

操作网络路由表

```
C:\Windows\system32>route print -4
```

```
=====
```

IPv4 路由表

```
=====
```

活动路由:

网络目标	网络掩码	网关	接口	跃点数
0.0.0.0	0.0.0.0	192.168.7.2	192.168.7.132	10
127.0.0.0	255.0.0.0	在链路上	127.0.0.1	306
127.0.0.1	255.255.255.255	在链路上	127.0.0.1	306
127.255.255.255	255.255.255.255	在链路上	127.0.0.1	306
192.168.7.0	255.255.255.0	在链路上	192.168.7.132	266
192.168.7.132	255.255.255.255	在链路上	192.168.7.132	266
192.168.7.255	255.255.255.255	在链路上	192.168.7.132	266
224.0.0.0	240.0.0.0	在链路上	127.0.0.1	306
224.0.0.0	240.0.0.0	在链路上	192.168.7.132	266
255.255.255.255	255.255.255.255	在链路上	127.0.0.1	306
255.255.255.255	255.255.255.255	在链路上	192.168.7.132	266

```
=====
```

永久路由:

无



搜索目标的跃点数

C:\Windows\system32>tracert www.xmu.edu.cn

通过最多 30 个跃点跟踪
到 www.xmu.edu.cn [210.34.0.12] 的路由：

1	<1 毫秒	1 ms	12 ms	192.168.7.2
2	*	*	*	请求超时。
3	*	*	*	请求超时。
4	*	*	*	请求超时。
5	1 ms	1 ms	1 ms	210.34.0.12

跟踪完成。

使用ICMP发现路径MTU

- 路径MTU发现
 - RFC1191使用分段标志中的“不能分段”来要求中间路由器在发现包太长时返回一个 ICMP出错报文。
 - 尽管大多数的系统不支持路径MTU发现功能，但可以很容易地修改traceroute程序，用它来确定路径MTU。
 - 发送的第一个分组的长度正好与出口MTU相等，每次收到“不能分片”差错时就减小分组的长度。如果路由器发送的 ICMP差错报文是新格式，包含出口的MTU，那么就用该 MTU值来发送，否则就用下一个最小的MTU值来发送。

内容纲要

1

差错报告机制（ ICMP ）

2

地址解析协议（ ARP ）

3

动态主机配置协议（ DHCP ）

4

网络地址转换（ NAT ）

5

未来的IP：IPv6



地址解析（Address Resolution）

- 将IP地址解析为MAC地址的叫做地址解析
 - IP是虚拟的，但数据链路层需要物理地址，最终要换的
- 地址解析协议（Address Resolution Protocol，ARP）
- 概念地址边界
 - IP地址、物理地址

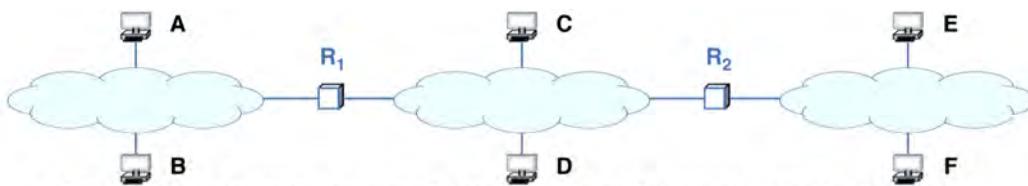


Figure 23.1 An example internet of three networks and computers connected to each.

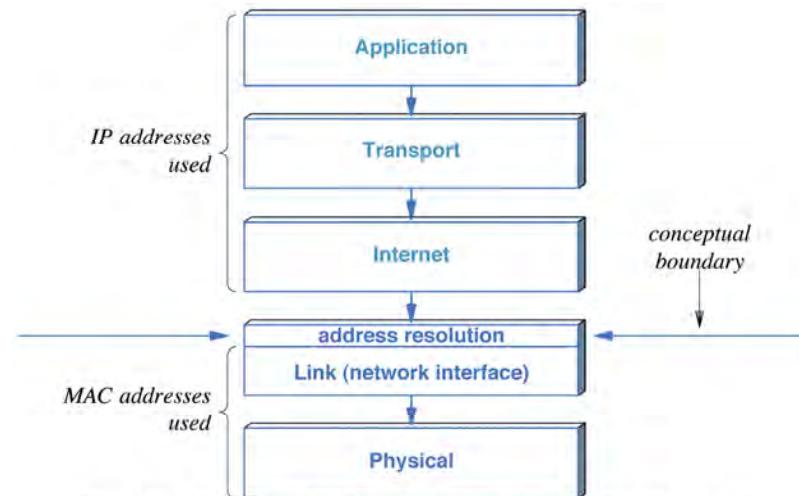
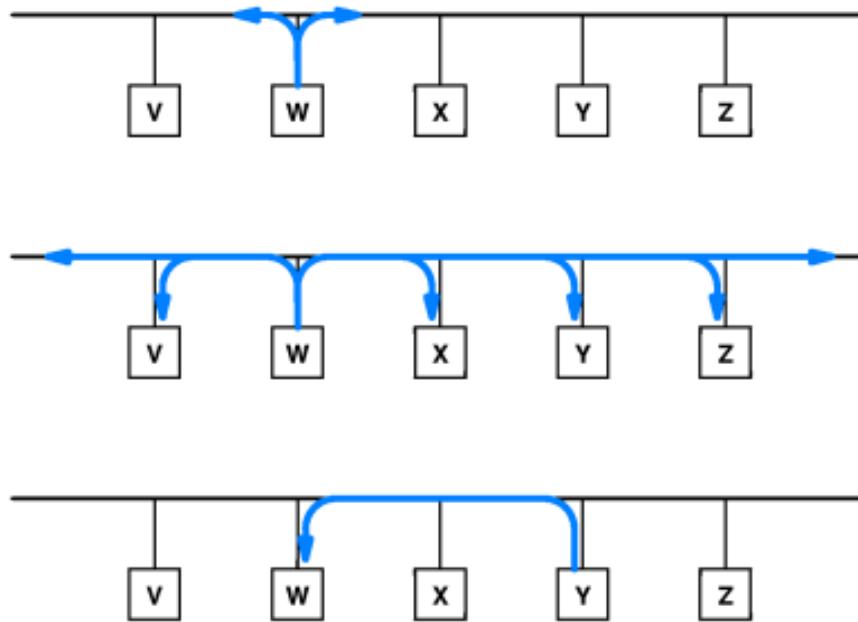


Figure 23.5 Illustration of the boundary between the use of IP addresses and MAC addresses.

地址解析技术

- 不管网络层使用的是什么协议，在实际网络的链路上传送数据帧时，最终还是必须使用硬件地址。
- ARP标准定义了请求与响应



地址解析技术分类

- 查表（Table lookup）

- 存储在内存表。

- 相近形式计算

- 配置使得硬件地址可通过简单的布尔和算术运算得出对应的协议地址。

- 消息交换

- 一台计算机发出某个地址联编的请求消息后，另一台计算机返回一个包含所需信息的应答消息。



使用 ARP 的四种典型情况

- 主机发送数据报到本网络上的另一台主机。
 - 找到目的主机的硬件地址。
- 主机发送数据报到另一个网络上的一台主机。
 - 找到本网络上一个路由器的硬件地址。由该路由器转发。
- 路由器转发数据报到本网络上的一台主机。
 - 找到目的主机的硬件地址。
- 路由器转发数据报到另一个网络上的一台主机。
 - 找到本网络上另一个路由器的硬件地址。由该路由器转发。



ARP 消息格式

- ARP帧格式

0	8	16	24	31			
HARDWARE ADDRESS TYPE		PROTOCOL ADDRESS TYPE					
HADDR LEN	PADDR LEN	OPERATION					
SENDER HADDR (first 4 octets)							
SENDER HADDR (last 2 octets)		SENDER PADDR (first 2 octets)					
SENDER PADDR (last 2 octets)		TARGET HADDR (first 2 octets)					
TARGET HADDR (last 4 octets)							
TARGET PADDR (all 4 octets)							

Figure 23.3 The format for an ARP message when binding an IPv4 address to an Ethernet address.

- ARP 封装

- 帧类型: 0x0806



Figure 23.4 Illustration of ARP encapsulation in an Ethernet frame.

ARP路由表

C:\Windows\system32>arp -a

接口：192.168.33.3 --- 0xd

Internet 地址	物理地址	类型
192.168.33.6	f8-b1-56-b5-39-bc	动态
192.168.33.14	9c-21-6a-f6-82-6d	动态
224.0.0.22	01-00-5e-00-00-16	静态

接口：192.168.1.1 --- 0x12

Internet 地址	物理地址	类型
224.0.0.22	01-00-5e-00-00-16	静态

接口：169.254.0.1 --- 0x13

Internet 地址	物理地址	类型
224.0.0.22	01-00-5e-00-00-16	静态

ARP 消息格式

Packet Info

Packet Number: 1
Flags: 0x00000000
Status: 0x00000000
Packet Length: 64
Timestamp: 14:17:23.430079000 04/11/2014

Ethernet Type 2

Destination: FF:FF:FF:FF:FF:FF Ethernet Broadcast [0-5]
Source: 00:0C:29:17:29:CA VMware:17:29:CA [6-11]
Protocol Type: 0x0806 IP ARP [12-13]

ARP - Address Resolution Protocol

Hardware: 1 Ethernet (10Mb) [14-15]
Protocol: 0x0800 IP [16-17]
Hardware Addr Length: 6 [18]
Protocol Addr Length: 4 [19]
Operation: 1 ARP Request [20-21]
Sender Hardware Addr: 00:0C:29:17:29:CA VMware:17:29:CA [22-27]
Sender Internet Addr: 192.168.7.4 [28-31]
Target Hardware Addr: 00:00:00:00:00:00 Xerox:00:00:00 (ignored) [32-37]
Target Internet Addr: 192.168.7.2 [38-41]

Extra bytes

Number of bytes:
..... 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [42-57]
.. 00 00 [58-59]



ARP 消息格式

Packet Info

Packet Number: 2
Flags: 0x00000000
Status: 0x00000000
Packet Length: 64
Timestamp: 14:17:23.516605000 04/11/2014

Ethernet Type 2

Destination: 00:0C:29:17:29:CA VMware:17:29:CA [0-5]
Source: 00:50:56:FC:52:95 VMware:FC:52:95 [6-11]
Protocol Type: 0x0806 IP ARP [12-13]

ARP - Address Resolution Protocol

Hardware: 1 Ethernet (10Mb) [14-15]
Protocol: 0x0800 IP [16-17]
Hardware Addr Length: 6 [18]
Protocol Addr Length: 4 [19]
Operation: 2 ARP Response [20-21]
Sender Hardware Addr: 00:50:56:FC:52:95 VMware:FC:52:95 [22-27]
Sender Internet Addr: 192.168.7.2 [28-31]
Target Hardware Addr: 00:0C:29:17:29:CA VMware:17:29:CA [32-37]
Target Internet Addr: 192.168.7.4 [38-41]

Extra bytes

Number of bytes:
..... 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 [42-57]
.. 00 00 [58-59]

应当注意的问题

- ARP 解决同一局域网上的IP和硬件地址的映射问题。
 - 如果目标主机和源主机不在同一个局域网，则通过 ARP 找到位于本局域网上的某个路由器的硬件地址，把分组发送给这个路由器转发给下一个网络。直到同一局域网。
 - 从IP地址到硬件地址的解析是对用户是透明的。
 - 只要主机或路由器要和本网络上的另一个已知 IP 地址的主机或路由器进行通信，ARP 协议就会自动地将该 IP 地址解析为链路层所需要的硬件地址。



ARP欺骗

- ARP欺骗的核心思想

- 向目标主机发送伪造的ARP应答，并使目标主机接收应答中伪造的IP地址与MAC地址之间的映射对，以此更新目标主机ARP缓存。

- ARP欺骗的防范

- S代表源主机，被欺骗的目标主机；
D代表目的主机，S本来向它发送数据； A代表攻击者，进行ARP欺骗。

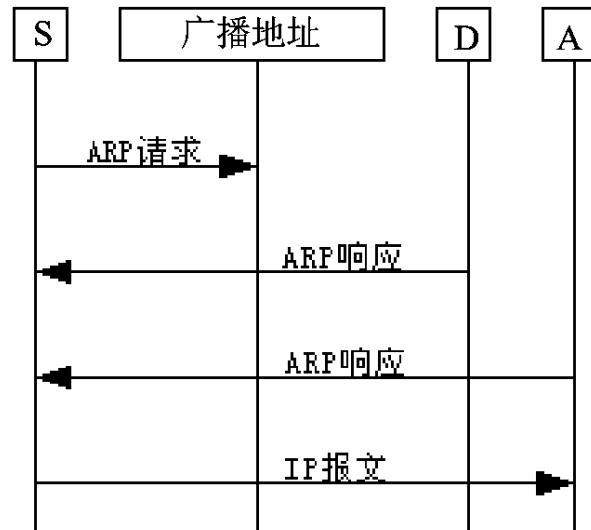


图1 实施ARP欺骗的过程

ARP高速缓存（ ARP Cache ）

- 局域网各主机和路由器的IP 地址到硬件地址的映射表。
 - 当主机 A 欲向本局域网上的某个主机 B 发送 IP 数据报时，就先在其 ARP 高速缓存中查看有无主机 B 的 IP 地址。如有，就可查出其对应的硬件地址，再将此硬件地址写入 MAC 帧，然后通过局域网将该 MAC 帧发往此硬件地址。
 - 为了减少网络上的通信量，主机 A 在发送其 ARP 请求分组时，就将其 IP 地址到硬件地址的映射写入 ARP 请求分组。
 - 当主机 B 收到 A 的 ARP 请求分组时，就将主机 A 的这一地址映射写入主机 B 自己的 ARP 高速缓存中。



内容纲要

1

差错报告机制（ ICMP ）

2

地址解析协议（ ARP ）

3

动态主机配置协议（ DHCP ）

4

网络地址转换（ NAT ）

5

未来的IP：IPv6



IP地址哪里来

- 向ISP购买一个（或段）IP的使用权
- 大量设备如何使用有限的地址上网
 - DHCP服务：“时分多路复用”，轮流使用IP地址
 - NAT、NAPT服务：“频分多路复用”，共用一个IP地址



动态主机配置协议（DHCP）

- 早期：反向地址解析协议（RARP）
- 作用：从服务器获得IP地址。
- 已知条件
 - 本地机器：没有IP（本机IP：0.0.0.0；MAC已知）
 - 目的机器：有IP但不知道（目的IP、MAC，全1广播）
- DHCP获得的IP地址有租期，可附加其他配置
- DHCP提供一个好心的服务（防君子不防小人）



监听结果

- 用Omnipeek软件解析DHCP包

— 拔出网线，开软件，勾选DHCP，插入网线，再解析

ID	Src. Logical	Src. Physic al	Src. Port	Dest. Log.	Dest. Phy.	Dest. Prt.	Summary	Expert
1	0.0.0.0	00:0C: 29:37: 5A:1B	UDP 68	255.25 5.255. 255	FF:FF:F F:FF:FF :FF	UDP 67	C DISCOVER 192.168.7.132 WIN-KG9CLM76UIA	
2	192.168 .7.254	00:50: 56:E2: AF:04	UDP 67	192.16 8.7.13 2	00:0C:2 9:37:5A :1B	UDP 68	R OFFER 192.168.7.132	
3	0.0.0.0	00:0C: 29:37: 5A:1B	UDP 68	255.25 5.255. 255	FF:FF:F F:FF:FF :FF	UDP 67	C REQUEST 192.168.7.132 WIN-KG9CLM76UIA	
4	192.168 .7.254	00:50: 56:E2: AF:04	UDP 67	192.16 8.7.13 2	00:0C:2 9:37:5A :1B	UDP 68	R ACK	DHCP Low Lease Time (30 minutes, threshold=30 minutes)

监听结果节选

Packet #1

Ethernet Type 2

Destination: FF:FF:FF:FF:FF:FF Ethernet Broadcast [0-5]
Source: 00:0C:29:37:5A:1B VMware:37:5A:1B [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Protocol: 17 UDP [23]
Source IP Address: 0.0.0.0 [26-29]
Dest. IP Address: 255.255.255.255 IP Broadcast [30-33]

UDP - User Datagram Protocol

Source Port: 68 bootpc [34-35]
Destination Port: 67 bootps [36-37]

BootP - Bootstrap Protocol

IP Address Known By Client: 0.0.0.0 IP Address Not Known By Client [54-57]
Client IP Addr Given By Srvr: 0.0.0.0 [58-61]
Server IP Address: 0.0.0.0 [62-65]
Gateway IP Address: 0.0.0.0 [66-69]
Client Hardware Addr: 00:0C:29:37:5A:1B VMware:37:5A:1B [70-75]

DHCP - Dynamic Host Configuration Protocol

Requested IP Address

Address: 192.168.7.132 [296-299]

Host Name Address

String: WIN-KG9CLM76UIA [302-316]



厦门大学
XIAMEN UNIVERSITY



信息学院 黄 烽
(国家示范性软件学院) 博士/副教授
School of Informatics Dr. Wei Huang

DHCP的配置

- Windows提供DHCP Client服务

The left screenshot shows the TP-LINK TL-WVR308 web interface. The URL is 192.168.33.14/userRpm/Index.htm. The main menu includes LAN口设置, DHCP服务, 客户端列表, and 静态地址分配. The left sidebar lists system status, setup guides, port settings, WAN/LAN/MAC/switch configurations, wireless, and security features. The central panel is titled '配置参数' (Parameter Configuration) and shows the following settings for the DHCP service:

DHCP服务器:	<input checked="" type="radio"/> 启用 <input type="radio"/> 禁用
地址池起始地址:	192.168.33.9
地址池结束地址:	192.168.33.13
地址租期:	120 分钟 (1-2880)
网关地址:	192.168.33.14 (可选)
缺省域名:	(可选)
首选DNS服务器:	0.0.0.0 (可选)
备用DNS服务器:	0.0.0.0 (可选)

The right screenshot shows the 'Network Connection Details' window in Windows. It displays the following information:

属性	值
连接特定的 DNS 后缀	localdomain
描述	Intel(R) 82574L 千兆网络连接
物理地址	00-0C-29-37-5A-1B
已启用 DHCP	是
IPv4 地址	192.168.7.132
IPv4 子网掩码	255.255.255.0
获得租约的时间	2013年5月19日 10:03:51
租约过期的时间	2013年5月19日 10:34:01
IPv4 默认网关	192.168.7.2
IPv4 DHCP 服务器	192.168.7.254
IPv4 DNS 服务器	192.168.7.2
IPv4 WINS 服务器	192.168.7.2
已启用 NetBIOS over Tc...	是
连接-本地 IPv6 地址	fe80::69a1:1231:cea2:75ef%12
IPv6 默认网关	
IPv6 DNS 服务器	

内容纲要

1

差错报告机制（ ICMP ）

2

地址解析协议（ ARP ）

3

动态主机配置协议（ DHCP ）

4

网络地址转换（ NAT ）

5

未来的IP：IPv6



网络地址转换（NAT）

- NAT应用场景
 - 多台主机上网，但是只有一个公网IP地址
- NAT动机：IP地址紧张，端口号并不紧张

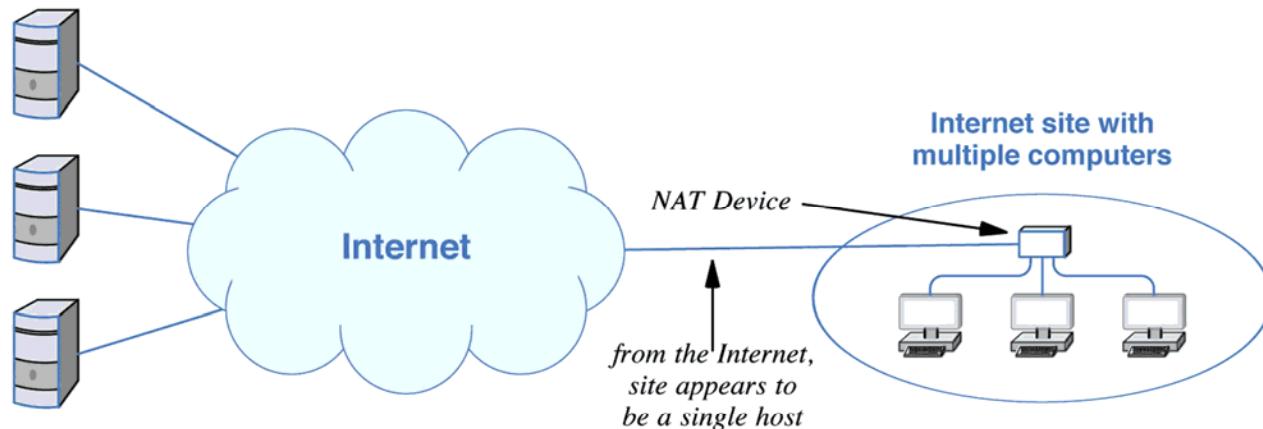


Figure 23.9 The conceptual architecture used with NAT.

Copyright © 2009 Pearson Prentice Hall, Inc.

私有地址

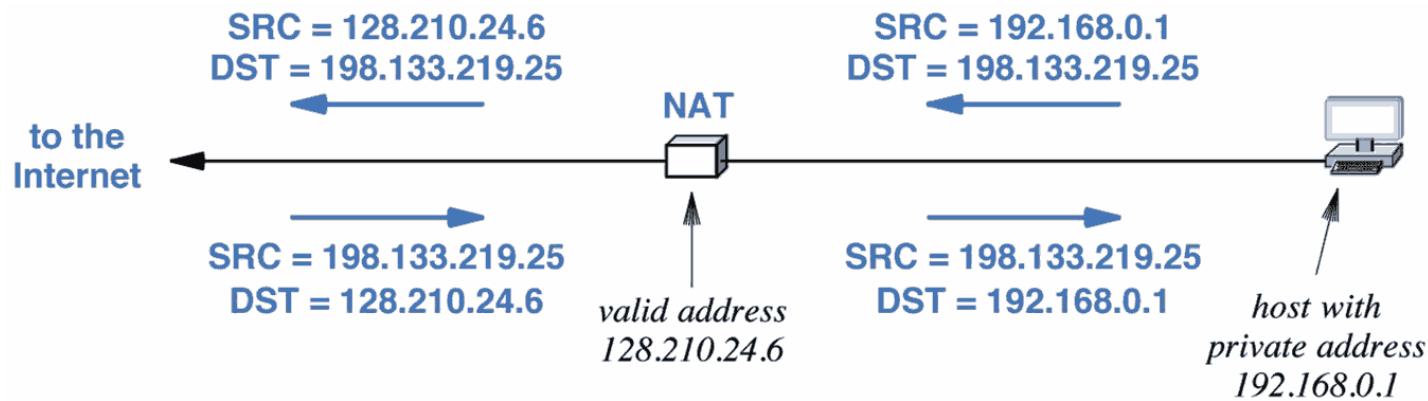
- 目的：虚拟的寻址机制
 - NAT的目的是提供一种错觉。
- NAT使用的私有地址块

网络号	类	个数	说明
10.0.0.0/8	A	1	
169.254.0.0/16	B	1	一般开启DHCP客户端又无法获取到IP时使用
172.16.0.0/12	B	16	
● 192.168.0.0/16	C	256	

NAT的地址转换

- NAT的最基本形式将数据报中的IP源地址由站点替换为Internet，并且将IP目的地址由Internet替换为站点

Direction	Field	Old Value	New Value
out	IP Source	192.168.0.1	128.210.24.6
	IP Destination	198.133.219.25	-- no change --
in	IP Source	198.133.219.25	-- no change --
	IP Destination	128.210.24.6	192.168.0.1



传输层的NAT (NAPT)

- 传输层的特别之处：端口号
- 端口号也参与转换
 - 因为终究是主机上的应用在上网
- NAT有时候也用于负载均衡

Dir.	Fields	Old Value	New Value
out	IP SRC:TCP SRC	192.168.0.1:30000	128.10.24.6:40001
out	IP SRC:TCP SRC	192.168.0.2:30000	128.10.24.6:40002
in	IP DEST:TCP DEST	128.10.19.20:40001	192.168.0.1:30000
in	IP DEST:TCP DEST	128.10.19.20:40002	192.168.0.2:30000

FTP Login (VMWare)

Timestamp: 21:00:57.444125300 04/11/2014

Ethernet Type 2

Destination: 00:50:56:FC:52:95 VMware:FC:52:95 [0-5]
Source: 00:0C:29:17:29:CA VMware:17:29:CA [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]
Time To Live: 128 [22]
Protocol: 6 TCP - Transmission Control Protocol [23]
Header Checksum: 0x0000 Checksum invalid. Should be: 0xB059 [24-25]
Source IP Address: 192.168.7.4 [26-29]
Dest. IP Address: 59.77.7.25 [30-33]

TCP - Transport Control Protocol

Source Port: 4425 netrockey6 [34-35]
Destination Port: 21 ftp [36-37]
Sequence Number: 1304971726 [38-41]
Ack Number: 1171416600 [42-45]
TCP Offset: 5 (20 bytes) [46 Mask 0xF0]

...

FTP Control - File Transfer Protocol

Line 1: USER student<CR><LF> [54-65]

FTP Login (NAT)

Timestamp: 21:00:57.764403200 04/11/2014

Ethernet Type 2

Destination: 3C:E5:A6:D0:***:*** HangzhouH3:D0:***:*** [0-5]

Source: F8:B1:56:B5:***:*** [6-11]

Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]

Time To Live: 128 [22]

Protocol: 6 TCP - Transmission Control Protocol [23]

Header Checksum: 0x0000 Checksum invalid. Should be: 0x0B26 [24-25]

Source IP Address: 59.77.5.*** [26-29]

Dest. IP Address: 59.77.7.25 [30-33]

TCP - Transport Control Protocol

Source Port: 10405 [34-35]

Destination Port: 21 ftp [36-37]

Sequence Number: 2633766987 [38-41]

Ack Number: 300260607 [42-45]

TCP Offset: 5 (20 bytes) [46 Mask 0xF0]

...

FTP Control - File Transfer Protocol

Line 1: USER student<CR><LF> [54-65]



FTP Response (NAT)

Timestamp: 21:00:57.764979200 04/11/2014

Ethernet Type 2

Destination: F8:B1:56:B5:***:*** [0-5]
Source: 3C:E5:A6:D0:***:*** HangzhouH3:D0:***:*** [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]
Time To Live: 63 [22]
Protocol: 6 TCP - Transmission Control Protocol [23]
Header Checksum: 0xB59D [24-25]
Source IP Address: 59.77.7.25 [26-29]
Dest. IP Address: 59.77.5.*** [30-33]

TCP - Transport Control Protocol

Source Port: 21 ftp [34-35]
Destination Port: 10405 [36-37]
Sequence Number: 300260607 [38-41]
Ack Number: 2633767001 [42-45]
TCP Offset: 5 (20 bytes) [46 Mask 0xF0]

...

FTP Control - File Transfer Protocol

Line 1: 331 User name okay, need password.<CR><LF> [54-87]

FTP Response (VMWare)

Timestamp: 21:00:57.444794300 04/11/2014

Ethernet Type 2

Destination: 00:0C:29:17:29:CA VMware:17:29:CA [0-5]
Source: 00:50:56:FC:52:95 VMware:FC:52:95 [6-11]
Protocol Type: 0x0800 IP [12-13]

IP Version 4 Header - Internet Protocol Datagram

...

Fragment Offset: 0 (0 bytes) [20-21 Mask 0x1FFF]
Time To Live: 128 [22]
Protocol: 6 TCP - Transmission Control Protocol [23]
Header Checksum: 0xF389 [24-25]
Source IP Address: 59.77.7.25 [26-29]
Dest. IP Address: 192.168.7.4 [30-33]

TCP - Transport Control Protocol

Source Port: 21 ftp [34-35]
Destination Port: 4425 netrockey6 [36-37]
Sequence Number: 1171416600 [38-41]
Ack Number: 1304971740 [42-45]
TCP Offset: 5 (20 bytes) [46 Mask 0xF0]

...

FTP Control - File Transfer Protocol

Line 1: 331 User name okay, need password.<CR><LF> [54-87]

内容纲要

1

差错报告机制（ ICMP ）

2

地址解析协议（ ARP ）

3

动态主机配置协议（ DHCP ）

4

网络地址转换（ NAT ）

5

未来的IP：IPv6



IPv6的产生

- 2011年2月3日 IPv4的42亿地址分配用尽。
- IP的瓶颈

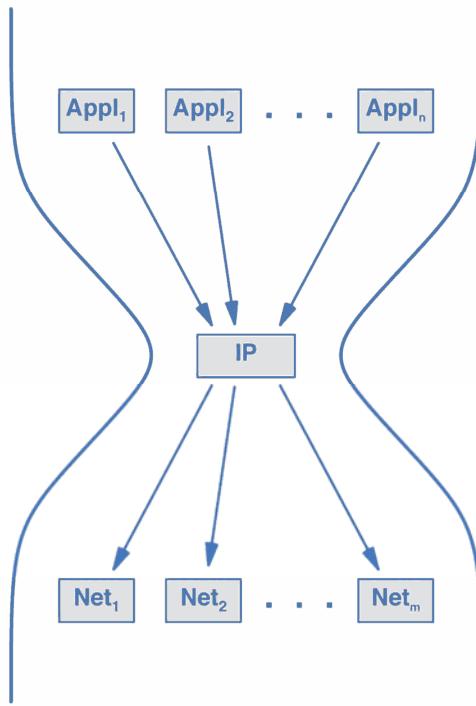


Figure 24.1 The hourglass model of Internet communication with IP at the center.

IPv6的特点

- 地址空间：128位
- 头部格式：新的头部
- 扩展头部：不同信息编码到不同头部中
 - 经济性、可扩展性
- 支持实时业务：允许底层网络建立高质量通路
- 可扩充的协议：允许在数据报添加额外的信息

IPv6数据报格式

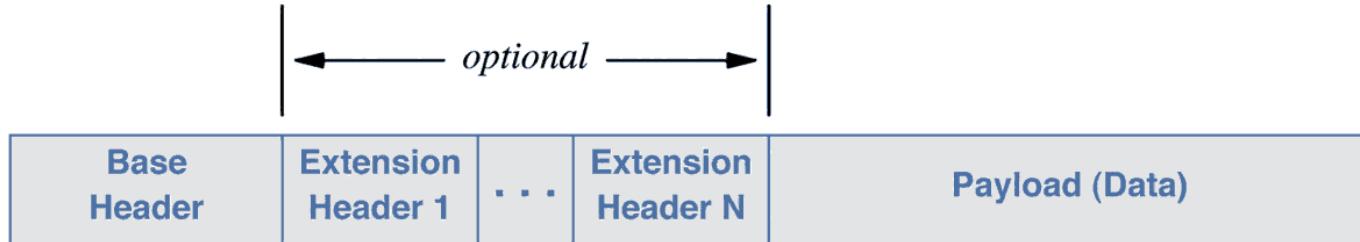


Figure 24.2 The general form of an IPv6 datagram.



Figure 24.3 The format of the base header in an IPv6 datagram.

IPv6地址

- 冒分十六进制数表示法（兼容CIDR表示法）
 - 按16位一组，以冒号分隔每个组
 - 2001:0db8:85a3:0000:0000:8a2e:0370:7334
 - 前导0压缩：0db8写成db8
 - 零压缩：两个冒号代替连续出现两个以上的零，最多1次
 - 2001:db8:85a3::8a2e:370:7334
 - IPv4扩展到IPv6
 - ::ffff:0:0/96前缀



计算机网络

Computer Network

11

谢谢观看

理论教程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

12

传输控制协议

理论课程

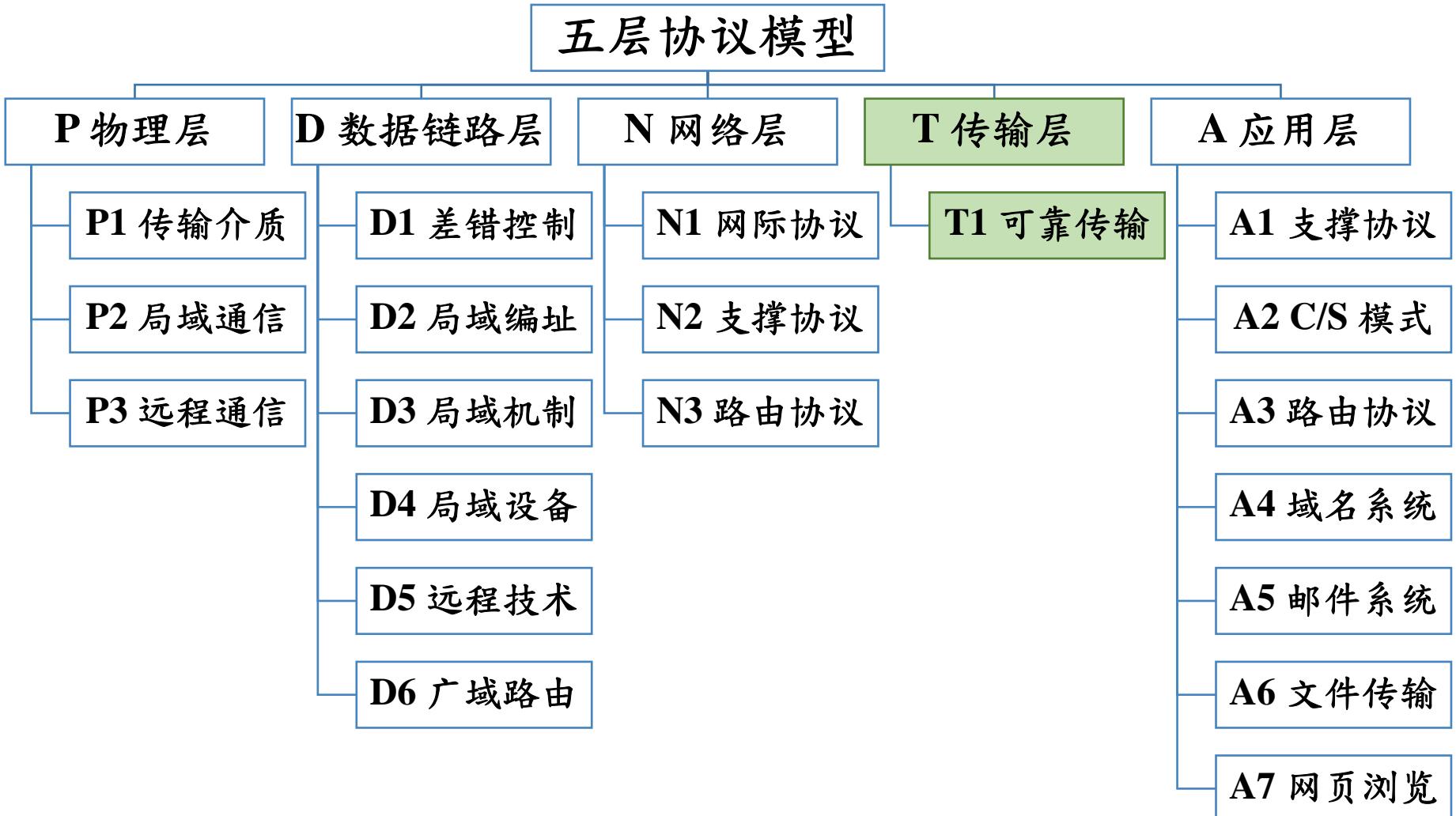


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 传输层
 - 作用
 - 端口号，端口号的分类
- UDP
 - UDP的无连接、尽力交付、面向报文、允许广播
- TCP
 - 特点：面向连接、点对点、可靠、全双工、字节流
 - TCP段格式中的各部分组成

主要内容

- TCP

- 机制

- 应答机制、超时机制、重传机制、窗口机制
 - 流量控制机制：滑动窗口
 - 拥塞控制：慢开始、拥塞避免、快重传、快恢复、随机早期检测
 - TCP的连接建立和解除（三次握手、四次挥手）

- 传输层解决网络层的主要问题：丢包、重复、乱序



对应课本章节

- PART IV Internetworking
 - Chapter 25 UDP: Datagram Transport Service
 - Chapter 26 TCP: Reliable Transport Service

内容纲要

1

传输层概述

2

用户数据报协议

3

传输控制协议

4

TCP的程序示例

5

小结



传输层的必要性

- 主机上多个进程共享主机的网络连接进行通信
 - 唯一标识主机和主机上的连接：IP地址
 - 唯一标识进程：端口号
 - 网络通信本质上是两个进程间的通信，不是主机间通信
- 传输层的作用：提供了进程间的复用和解复用
 - 传输层隐藏了硬件拓扑、路由细节等，使应用程序直接调用其接口，建立一条虚拟的端到端的通信信道



协议端口号

- 端口 (Port)

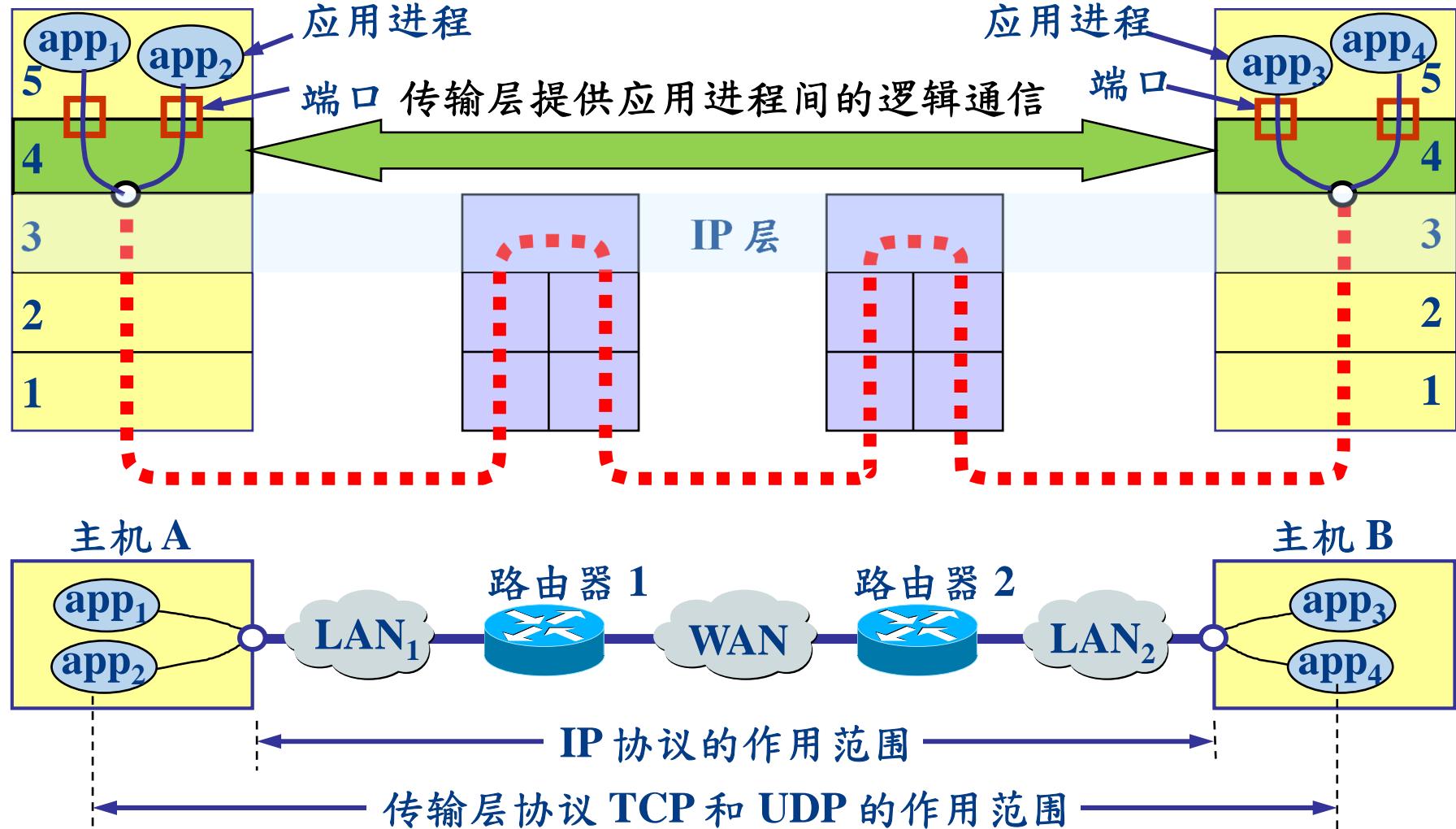
- 软件端口，有别于交换机上的硬件端口
- 端口号 (16bits)，范围：0~65535，
- 作用：用于标识本机的不同进程
- 分类 (参考：www.iana.org)

熟知端口号	登记端口号	客户端口号
0 ~ 1023	1024 ~ 49151	49152 ~ 65535
0 ~ 0x03FF	0x0400 ~ 0xBFFF	0xC000 ~ 0xFFFF

经常使用的端口号

协议	传输层协议	端口号	作用
FTP (数据)	TCP	20	FTP数据传输
FTP (控制)	TCP	21	FTP控制命令传输
Telnet	TCP	23	终端远程登录
SMTP	TCP	25	发送电子邮件
HTTP	TCP	80	网页服务
POP3	TCP	110	接收电子邮件
IMAP	TCP	143	同步邮箱
HTTPS	TCP	443	加密网页服务
RDP	TCP	3389	Windows 远程桌面连接
DNS	TCP/UDP	53	域名解析
DHCP (源)	UDP	68	动态IP获取的客户端端口。
DHCP (目的)	UDP	67	动态IP获取的服务器端端口。

网络进程通信



传输层的两个主要协议

协议	UDP [RFC 768]	TCP [RFC 793]
全称	User Datagram Protocol , 用户数据报协议	Transmission Control Protocol , 传输控制协议
数据单位	UDP 数据报 (Datagram)	TCP 报文段 (segment)
作用	端到端	端到端、字节流
连接	无连接	面向连接
收到确认	接收方不给出确认	收到确认
多方	一对一、一对多、多对多	一对一
长度	任意	每报文不超过64KB
优点	高效	安全
比喻	发电报 (短信)	打电话

内容纲要

1

传输层概述

2

用户数据报协议

3

传输控制协议

4

TCP的程序示例

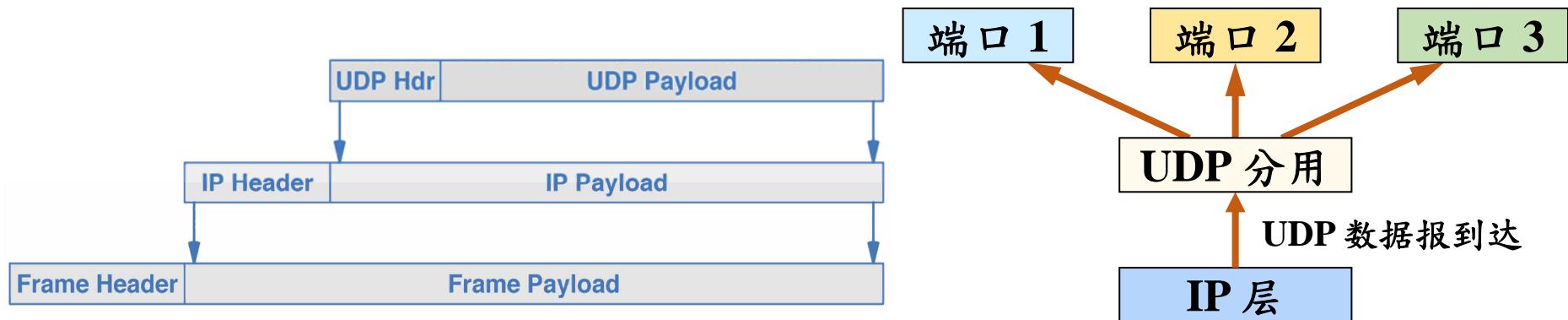
5

小结



UDP概述

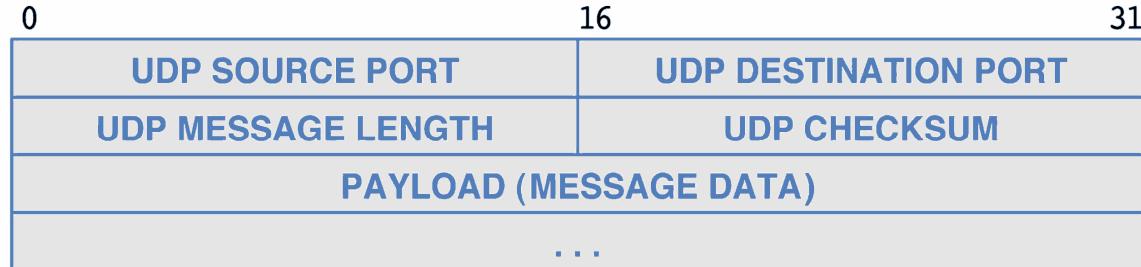
- 用户数据报协议（ User Datagram Protocol , UDP ）
- UDP的作用：不可靠但轻快的传输
 - 在IP服务之上，增加端口的功能和差错检测的功能
 - UDP校验和是可选的（0表示不计算）
- UDP消息可以丢失、重复、延迟、乱序、损坏。



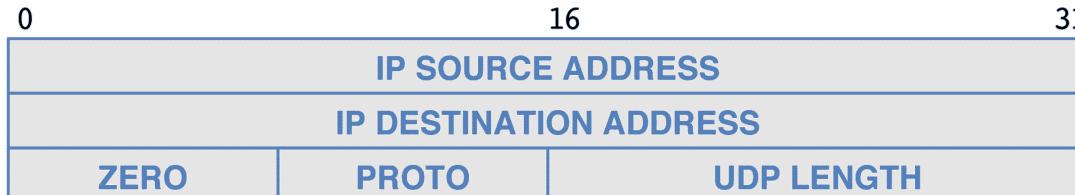
UDP分组结构

- 报文格式

- 源端口 : 16 bits
- 目的端口 : 16 bits



- 数据报文长度（单位：字节）：最小为8，即只有头部
- 校验和：不校正，错误即丢
 - 组成：伪首部（12B）、UDP首部（8B）、数据
 - 伪报文头



UDP的特点

- 无连接：发送数据之前不需要建立连接
 - 连接是指对状态（变量）的保持，不是对网络的保持
 - TCP协议中，通信双方保持序列号等变量，确保通信可靠
- 尽力交付：不保证可靠交付，同时也不使用拥塞控制
 - UDP发送报文前，不判断网络产生拥塞，不调节发送速率
 - 很适合多媒体通信等实时应用的要求
- 支持一对一、一对多的交互通信
- 首部开销小：只有 8 个字节

面向报文的 UDP

- 发送方
 - UDP 对应用程序交来的报文，在添加首部后向下交付 IP 层
 - 既不合并，也不拆分，而是保留这些报文的边界
- 接收方
 - UDP 对 IP 层交来的数据，去除首部后原封交付应用进程
 - 一次交付一个完整的报文
- 应用程序必须选择合适大小的报文。
 - 应用层交给 UDP 多长的报文，UDP 照样发送。



UDP的应用场景

- 常用于丢包损失不大，应用层可以控制丢包的场景。

类别	作用	示例
面向简单事务	查询响应协议	域名系统或网络时间协议
	没有完整协议栈的引导	DHCP和TFTP
提供数据报	构建其他协议	IP隧道，远程过程调用，网络文件系统
大量客户端	流媒体应用	IPTV
实时应用	建立在实时流的协议	IP语音，网络游戏
单向沟通	服务发现和共享信息中的广播信息	广播时间或路由信息协议

向广播地址11000端口发送UDP消息

```
using System;
using System.Net;
using System.Net.Sockets;
using System.Text;
class Program {
    static void Main(string[] args) {
        Socket s = new Socket(AddressFamily.InterNetwork, SocketType.Dgram,
ProtocolType.Udp);
        IPAddress broadcast = IPAddress.Parse("192.168.1.255");
        byte[] sendbuf = Encoding.ASCII.GetBytes("HELLO NETWORK");
        IPEndPoint ep = new IPEndPoint(broadcast, 11000);
        s.SendTo(sendbuf, ep);
        Console.WriteLine("Message sent to the broadcast address");
    }
}
```

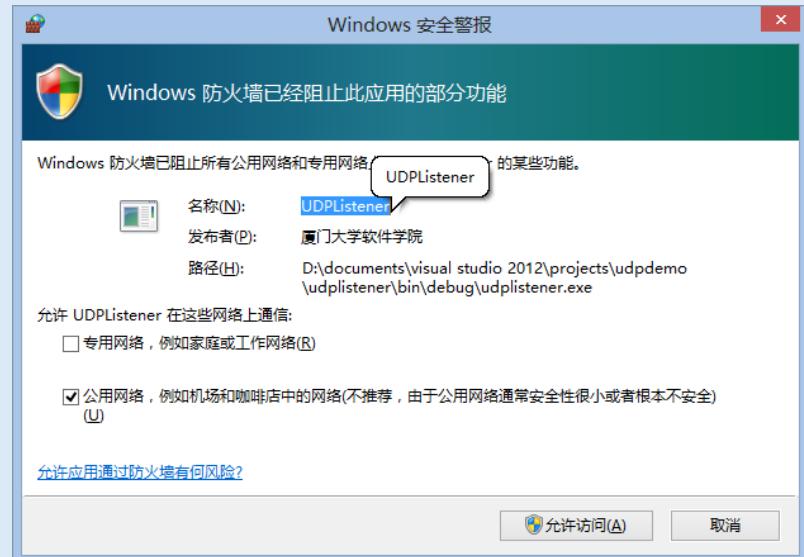
输出：

Message sent to the broadcast address

侦听广播地址11000端口的消息

```
using System.Net;
using System.Net.Sockets;
using System.Text;

public class UDPListener {
    private const int listenPort = 11000;
    private static void StartListener() {
        .....
    }
    public static int Main() {
        StartListener();
        return 0;
    }
}
```



侦听广播地址11000端口的消息（续）

```
private static void StartListener() {  
    bool done = false;  
    UdpClient listener = new UdpClient(listenPort);  
    IPEndPoint groupEP = new IPEndPoint(IPAddress.Any, listenPort);  
    try {  
        while (!done) {  
            Console.WriteLine("Waiting for broadcast");  
            byte[] bytes = listener.Receive(ref groupEP);  
            Console.WriteLine("Received broadcast from {0} :\n {1}\n",  
groupEP.ToString(), Encoding.ASCII.GetString(bytes, 0, bytes.Length));  
        }  
    }  
    catch (Exception e) { Console.WriteLine(e.ToString()); }  
    finally { listener.Close(); }  
}
```

输出(连接成功)：
Waiting for broadcast
Received broadcast from 192.168.1.1:52600 :
HELLO NETWORK

Waiting for broadcast



内容纲要

3 传输控制协议

3-1 介绍

3-2 基本机制

3-3 流量控制

3-4 拥塞控制



可靠传输的需求

- IP协议面临的问题：丢包、重复、乱序、数据损坏等。
- 部分程序需要可靠的传输
 - 传输信道不发生差错，如果出错要“抛出异常”
 - 不管多快的速度发送数据，接收端都来得及处理
- 传输控制协议（Transmission Control Protocol，TCP）
 - TCP提供面向连接、点对点、流接口、完整可靠的全双工通信。



TCP报文段的头部格式

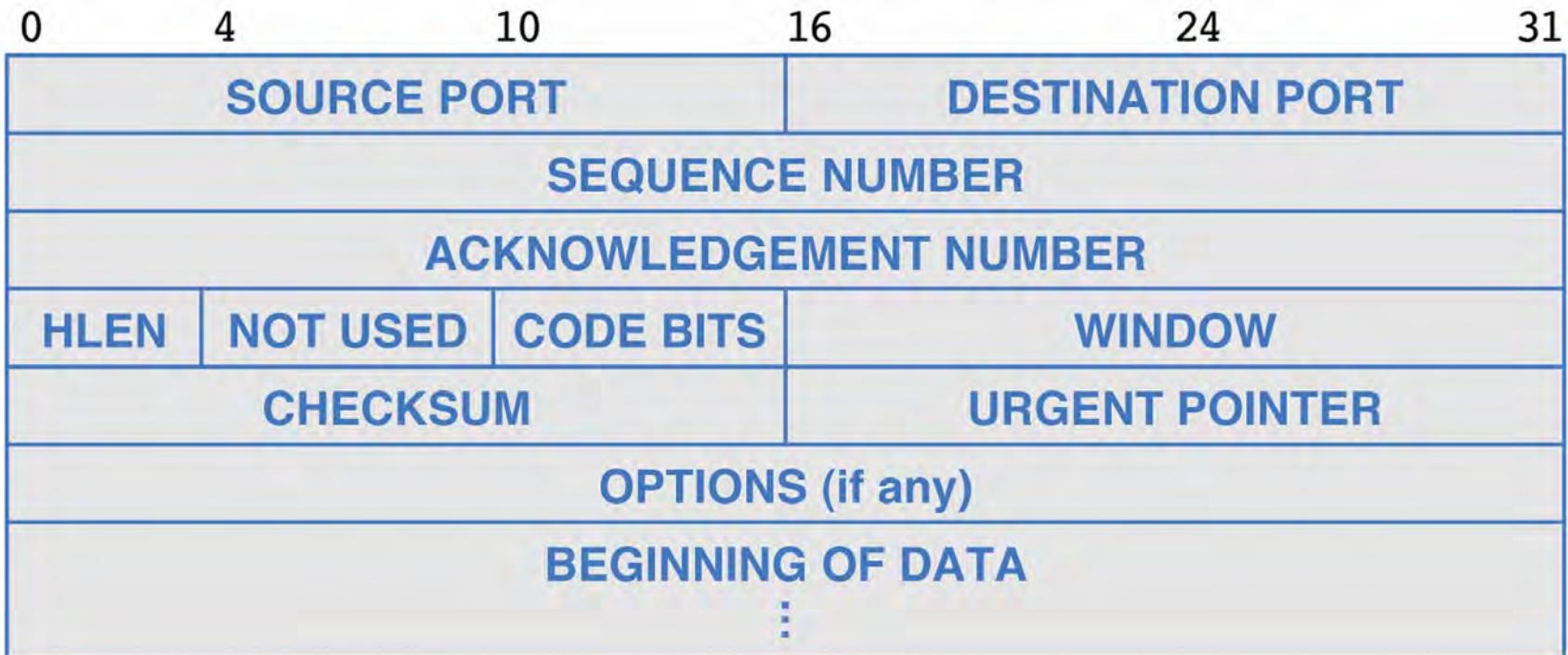


Figure 26.10 The TCP segment format used for both data and control messages.

Copyright © 2009 Pearson Prentice Hall, Inc.



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院) 博士/副教授
School of Informatics Dr. Wei Huang

TCP报文段的头部格式

- TCP报文头数据项：基本信息20B
 - 源端口号：16bits；目标端口号：16bits
 - 发送数据序列号（报文段序号）：32bits
 - TCP将每一个字节按顺序编号（与IP不同！）
 - 指的是本报文段第一个字节的序号
 - 确认序列号：32bits
 - 期望收到下一个字节的序号（而不是已收到的序号），表明之前的字节都已经收到



TCP报文段的头部格式

- TCP报文头数据项：基本信息20B
 - 报头长度：4bits，单位4Bytes
 - 保留位：4bits
 - 标识符：8bits
 - 拥塞窗口下降、ECN回显
 - 紧急指针：优先传输；ACK字段：确认号有效
 - 数据前推：不等待缓存满了再一起推送
 - 连接复位：拒绝连接；序号同步：建立连接时用来同步信号
 - 终止连接：数据已发送完毕，释放信号

TCP报文段的头部格式

- TCP报文头数据项：基本信息20B

- 滑动窗口缓冲区大小：16bits
- 校验和：16bits；紧急指针：16bits
- 选项字段：变长
 - 最大报文段长度
 - 窗口扩大选项
 - 时间戳



TCP报文头部

Timestamp: 22:28:39.376666700 04/21/2015

Ethernet Type 2

Destination: F8:B1:56:**:**:** [0-5]
Source: 9C:21:6A:**:**:** [6-11]
Protocol Type: 0x0800 *IP* [12-13]

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Header Length: 5 (*20 bytes*) [14 Mask 0x0F]
Diff. Services: %00000000 [15]
 0000 00.. Default
 00 Not-ECT
Total Length: 141 [16-17]
Identifier: 186 [18-19]
Fragmentation Flags: %010 [20 Mask 0xE0]
 0.. Reserved
 .1. Do Not Fragment
 ..0 Last Fragment
Fragment Offset: 0 (*0 bytes*) [20-21 Mask 0x1FFF]
Time To Live: 54 [22]
Protocol: 6 *TCP - Transmission Control Protocol* [23]
Header Checksum: 0x28E6 [24-25]
Source IP Address: 111.187.**.** [26-29]
Dest. IP Address: 192.168.**.** [30-33]

TCP报文头部（续）

TCP - Transport Control Protocol

Source Port: 20919 [34-35]
Destination Port: 3389 *ms-wbt-server* [36-37]
Sequence Number: 538319976 [38-41]
Ack Number: 3805334299 [42-45]
TCP Offset: 5 (*20 bytes*) [46 Mask 0xF0]
Reserved: %0000 [46 Mask 0x0F]
TCP Flags:
 %00011000 ...AP... [47]
 0... (*No Congestion Window Reduction*)
 .0... (*No ECN-Echo*)
 ..0. (*No Urgent pointer*)
 1 *Ack*
 1... *Push*
 0.. (*No Reset*)
 0. (*No SYN*)
 0 (No *FIN*)
Window: 1353 [48-49]
TCP Checksum: 0x3B1A [50-51]
Urgent Pointer: 0 [52-53]

No TCP Options
Application Layer

Data Area:

....



TCP报文头部

Timestamp: 22:28:39.376666700 04/21/2015

Ethernet Type 2

Destination: 9C:21:6A:***:***:*** [0-5]
Source: F8:B1:56:***:***:*** [6-11]
Protocol Type: 0x0800 *IP* [12-13]

IP Version 4 Header - Internet Protocol Datagram

Version: 4 [14 Mask 0xF0]
Header Length: 5 (*20 bytes*) [14 Mask 0x0F]
Diff. Services: %00000000 [15]
 0000 00.. Default
 00 Not-ECT
Total Length: 40 [16-17]
Identifier: 25622 [18-19]
Fragmentation Flags: %010 [20 Mask 0xE0]
 0.. Reserved
 .1. Do Not Fragment
 ..0 Last Fragment
Fragment Offset: 0 (*0 bytes*) [20-21 Mask 0x1FFF]
Time To Live: 128 [22]
Protocol: 6 *TCP - Transmission Control Protocol* [23]
Header Checksum: 0x7BEE [24-25]
Source IP Address: 192.168.*.* [26-29]
Dest. IP Address: 111.187.*.* [30-33]

TCP报文头部（续）

TCP - Transport Control Protocol

Source Port: 3389 *ms-wbt-server* [34-35]
Destination Port: 20919 [36-37]
Sequence Number: 3805334299 [38-41]
Ack Number: 538320077 [42-45]
TCP Offset: 5 (*20 bytes*) [46 Mask 0xF0]
Reserved: %0000 [46 Mask 0x0F]
TCP Flags:
 %00010000A.... [47]
 0... (*No Congestion Window Reduction*)
 .0... (*No ECN-Echo*)
 ..0. (*No Urgent pointer*)
 ...1 *Ack*
 0... (*No Push*)
 0.. (*No Reset*)
 0. (*No SYN*)
 0 (*No FIN*)
Window: 62735 [48-49]
TCP Checksum: 0x5635 [50-51]
Urgent Pointer: 0 [52-53] *No TCP Options*

Extra bytes

Number of bytes: 00 00 00 00 00 00 [54-59]

FCS - Frame Check Sequence

FCS: 0x7D0900B8 *Calculated*



廈門大學
XIAMEN UNIVERSITY



信息学院 黄 烽
(国家示范性软件学院) 博士/副教授
School of Informatics Dr. Wei Huang

内容纲要

3 传输控制协议

3-1 介绍

3-2 基本机制

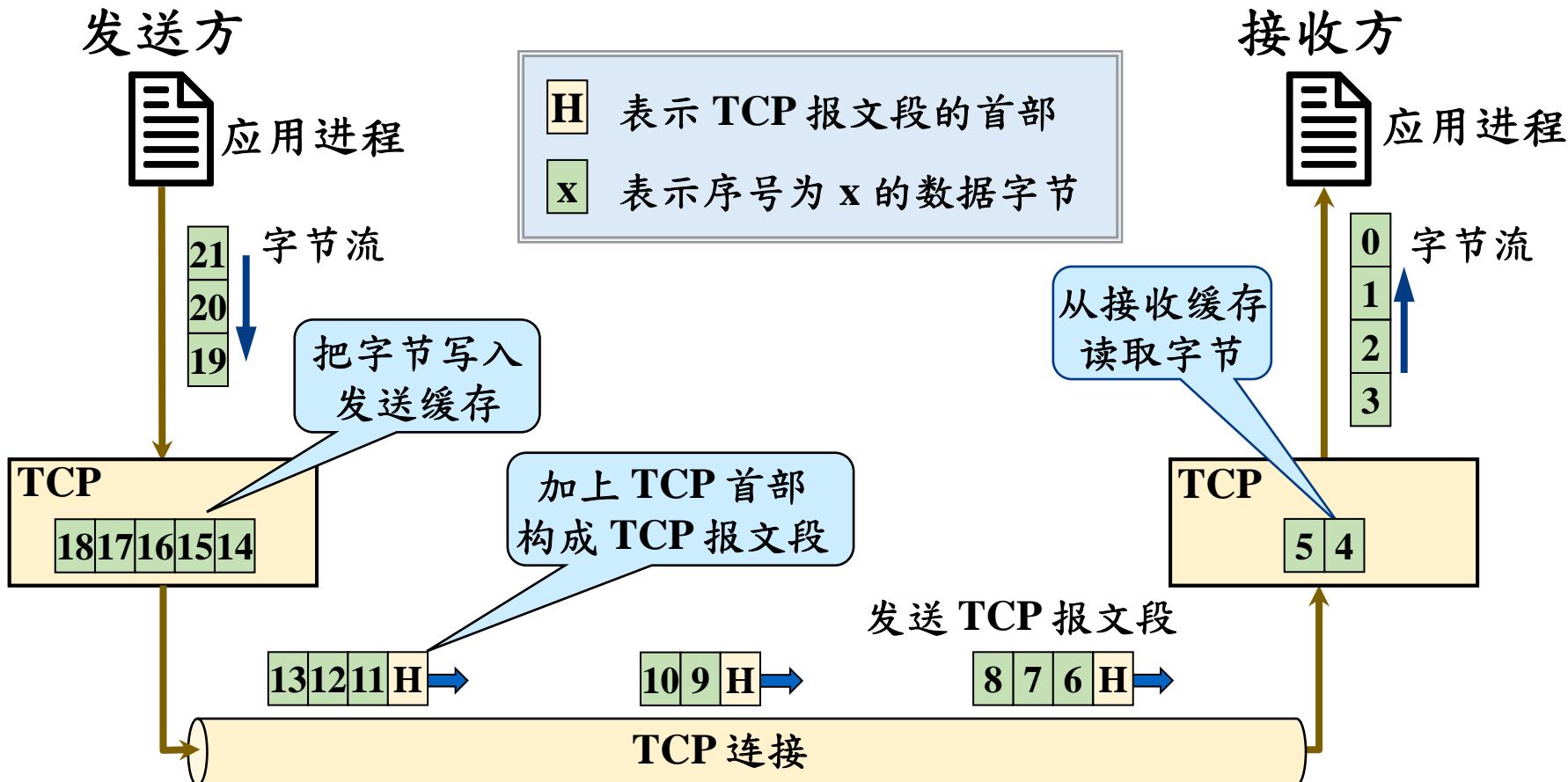
3-3 流量控制

3-4 拥塞控制



TCP流接口

- TCP 的传输是以字节为单位封装在报文段中传输。



TCP虚连接

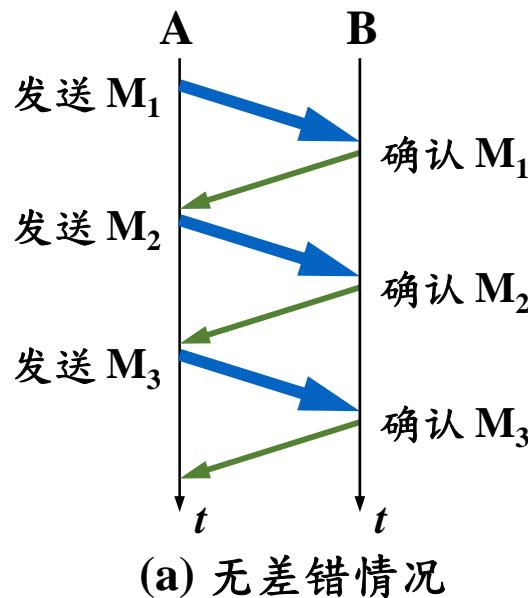
- TCP 连接是一条虚连接而不是一条真正的物理连接。
 - TCP 不关心应用进程发送到TCP 的缓存中的消息长度。
 - TCP 根据对方给出的窗口值和当前网络拥塞的程度来决定一个报文段应包含的字节数。
 - UDP 发送的报文长度是应用进程给出的。
 - TCP 将过长的数据块划分再传送，也可以等待积累有足够的字节后再构成报文段发送出去。



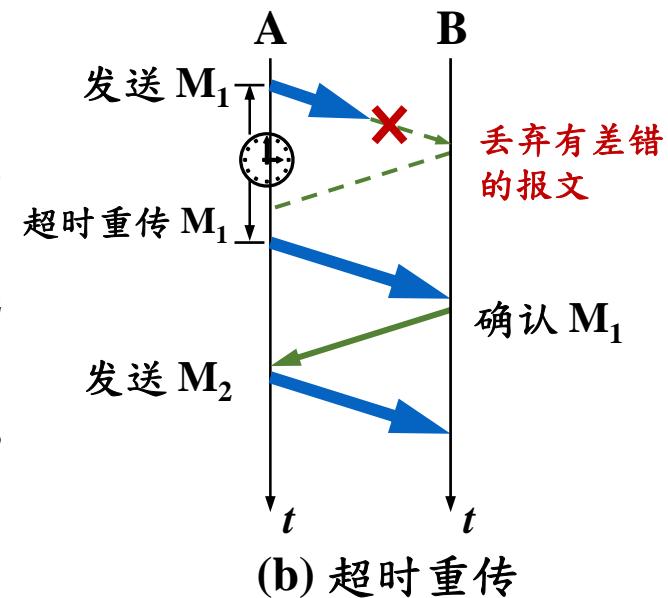
停止-等待协议的工作原理

- 无差错情况：发送、停止、等待
- 有差错情况：设置超时计时器
 - 发送以后，保留副本（万一需要重传）

- 分组编号
- 重传时间比平均往返时间长一些
 - 不确定因素：
拥塞、路径



(a) 无差错情况

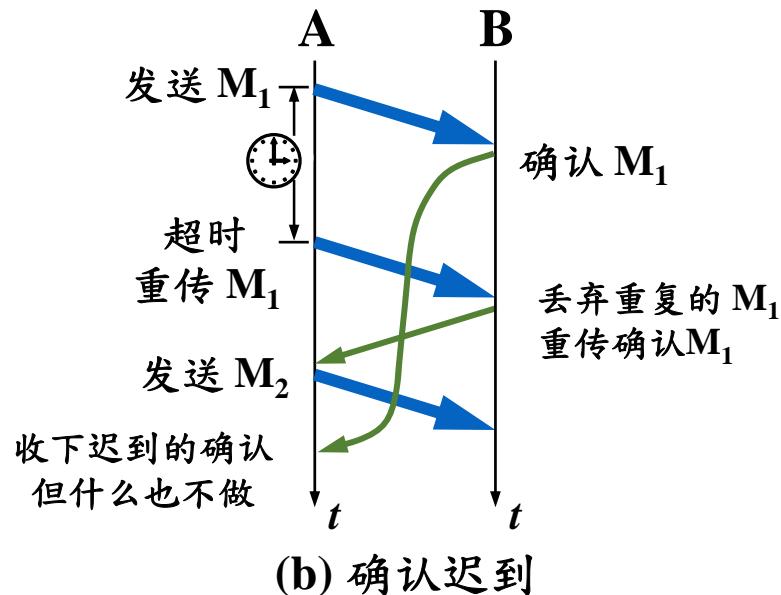
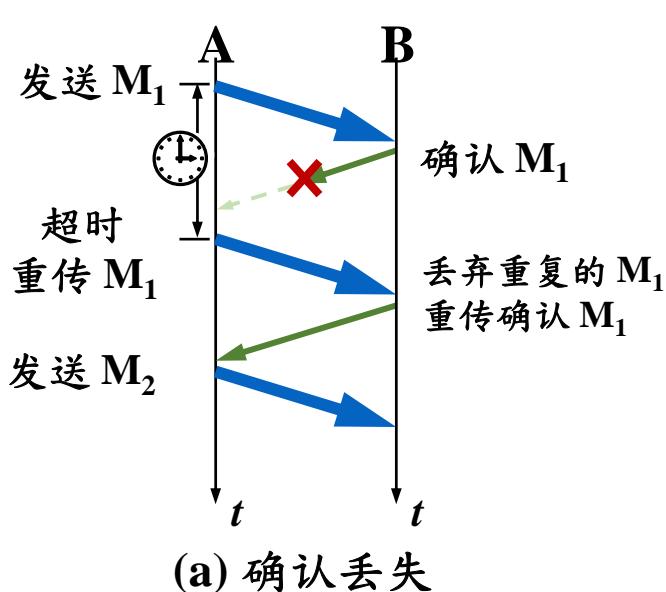


(b) 超时重传

停等协议的自动重传请求机制

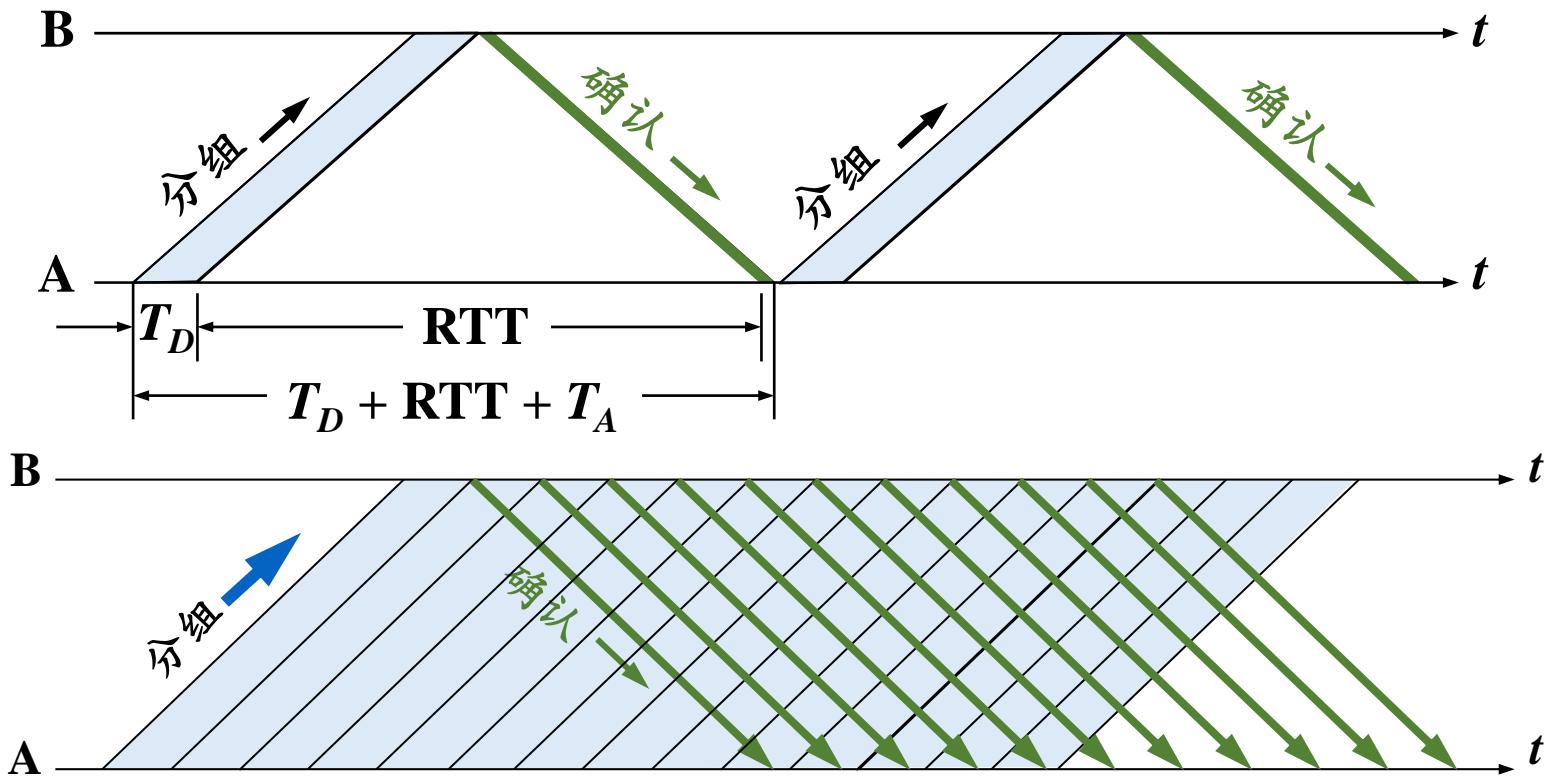
- 确认丢失和确认收到

- 通过自动重传请求，在不可靠的网络上实现可靠的通信
- 不区分分组丢失和确认丢失，
 - 一直未收到确认，则线路太差



连续自动重传请求

- 停止等待协议：优点是简单，缺点是信道利用率太低
- 流水线传输提高了利用率



窗口机制：连续自动重传请求

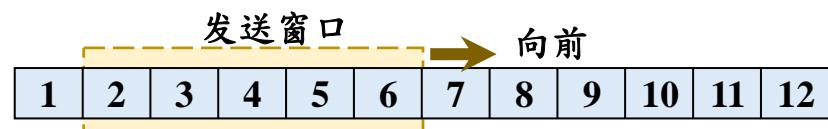
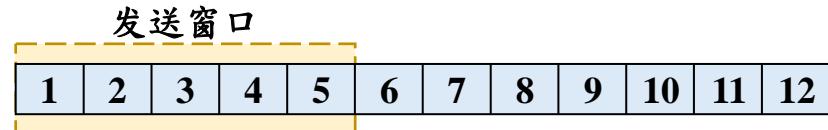
- 发送窗口

- 逐一确认

- 窗口内的分组都连续发送出去，不需要等待确认
 - 每收到一个确认，窗口就继续往前推进一格

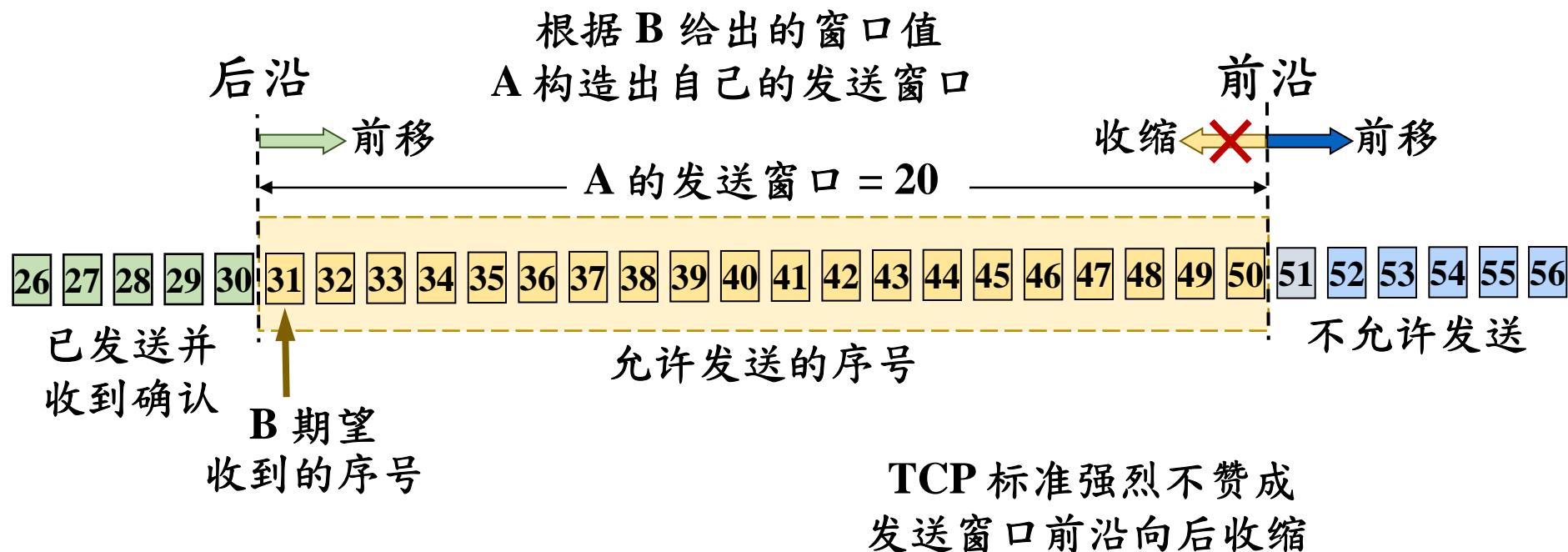
- 累积确认

- 接收方收到几个分组后，对最后一个分组发送确认
 - 例如窗口为5，第3分组丢失，只确认前2个，则3-5需重发



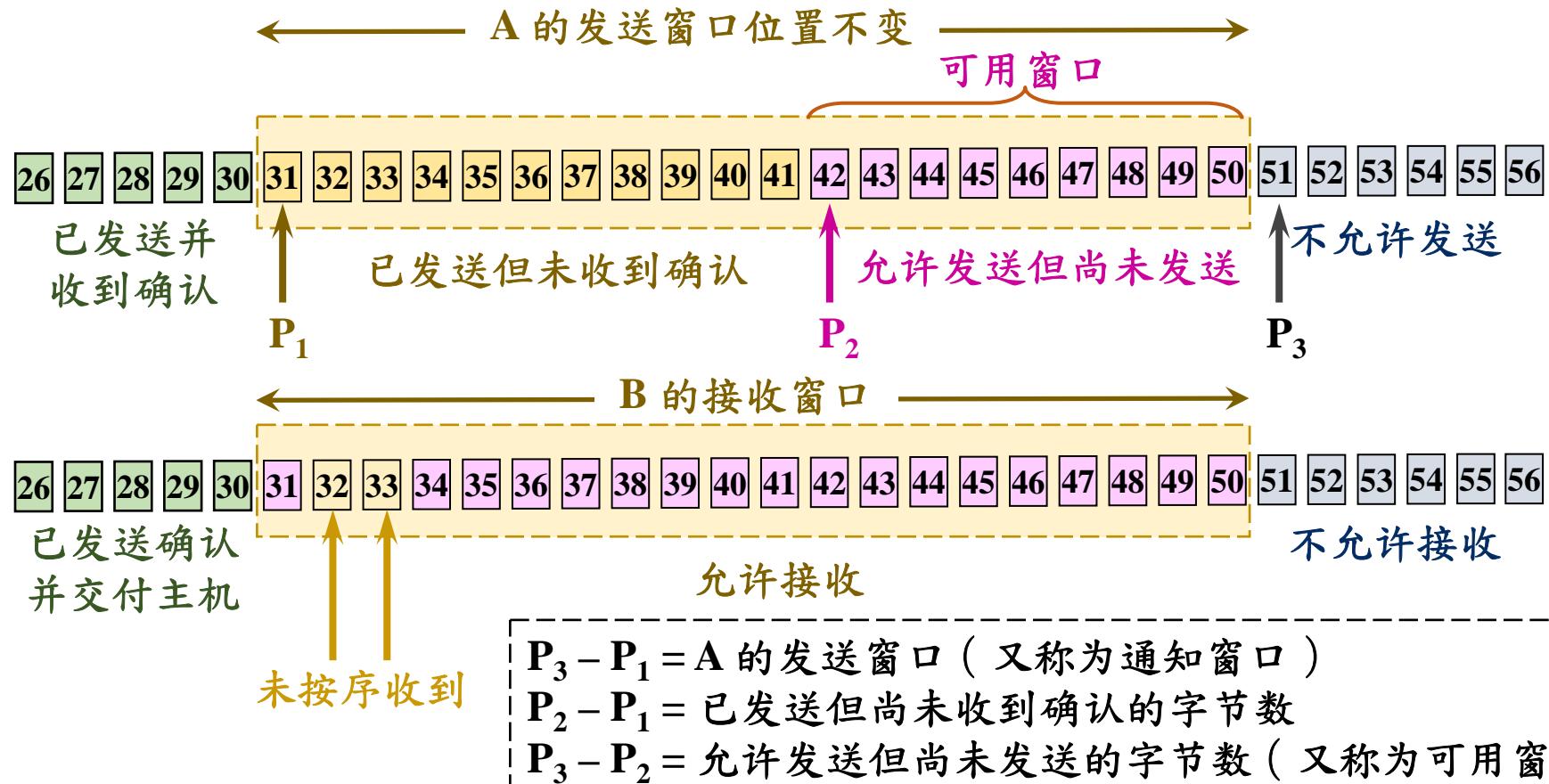
窗口机制：字节为单位的滑动窗口

- 以字节为单位的滑动窗口



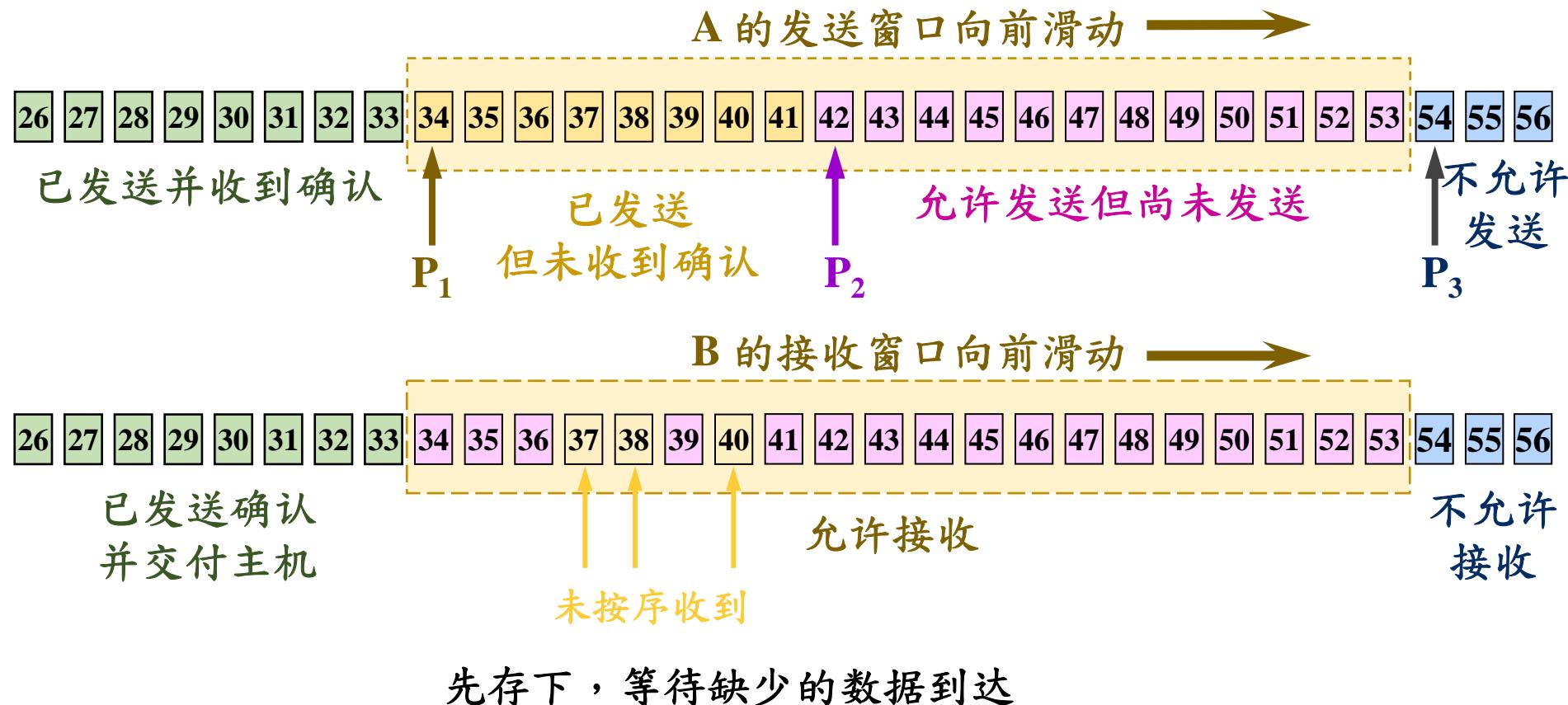
窗口机制：字节为单位的滑动窗口

- A 发送了 11 个字节的数据



窗口机制：字节为单位的滑动窗口

- A 收到新的确认号，发送窗口向前滑动



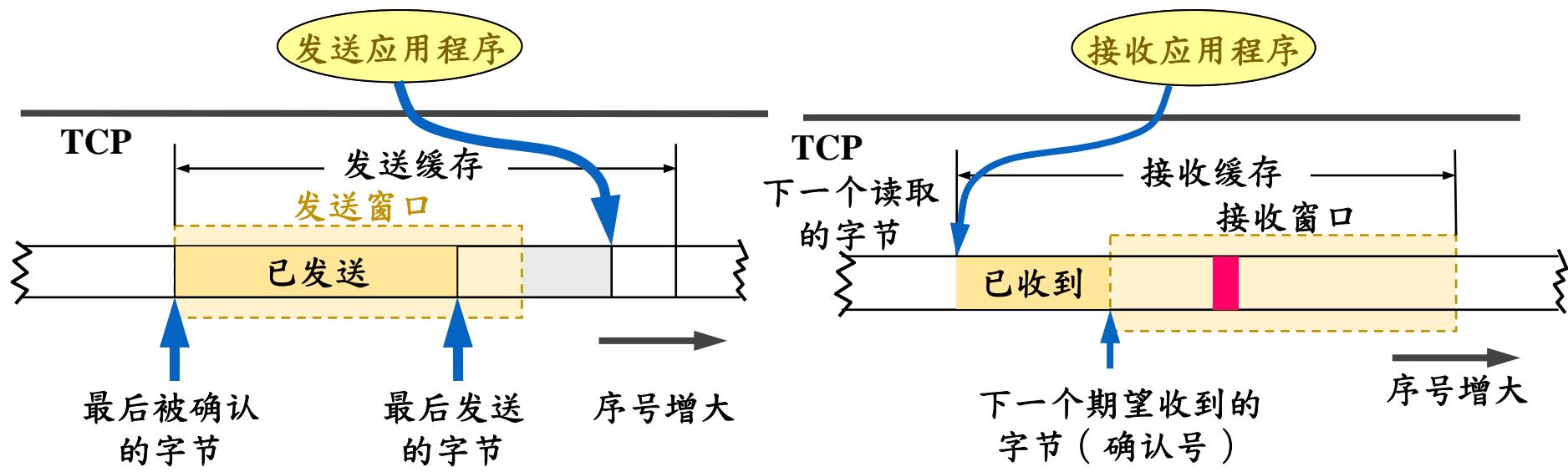
窗口机制：字节为单位的滑动窗口

- A 的发送窗口内的序号都已用完，但还没有再收到确认，必须停止发送。



窗口机制：字节为单位的滑动窗口

- A 的发送窗口内的序号都已用完，但还没有再收到确认，必须停止发送。



窗口机制：发送与接收缓存

- 发送缓存的作用

- 发送应用程序传送给发送方 TCP 准备发送的数据；
 - TCP 已发送出但尚未收到确认的数据。

- 接收缓存的作用

- 按序到达的、但尚未被接收应用程序读取的数据；
 - 不按序到达的数据。



窗口机制

- A 的发送窗口并不总是和 B 的接收窗口一样大（因为有一定的时间滞后）。
- TCP 标准没有规定对不按序到达的数据应如何处理。通常是先临时存放在接收窗口中，等到字节流中所缺少的字节收到后，再按序交付上层的应用进程。
- TCP 要求接收方必须有累积确认的功能，这样可以减小传输开销。

超时重传：时间的选择

- TCP 每发送一个报文段，就对这个报文段设置一次计时器。只要计时器设置的重传时间到但还没有收到确认，就要重传这一报文段。
- 由于 TCP 的下层是一个互联网环境，IP 数据报所选择的路由变化很大。因而运输层的往返时间的方差也很大。
- TCP 保留了 RTT 的一个加权平均往返时间 RTT_s （这又称为平滑的往返时间）。



超时重传：时间的选择

- 第一次测量到 RTT 样本时， RTT_S 值就取为所测量到的 RTT 样本值。以后每测量到一个新的 RTT 样本，就按下式重新计算一次 RTT_S ：

新的 $RTT_S = (1 - \alpha) \times (\text{旧的 } RTT_S) + \alpha \times (\text{新的 RTT 样本})$

- 式中， $0 \leq \alpha < 1$ 。若 α 很接近于零，表示 RTT 值更新较慢。
若选择 α 接近于 1，则表示 RTT 值更新较快。
- RFC 2988 推荐的 α 值为 0.125。

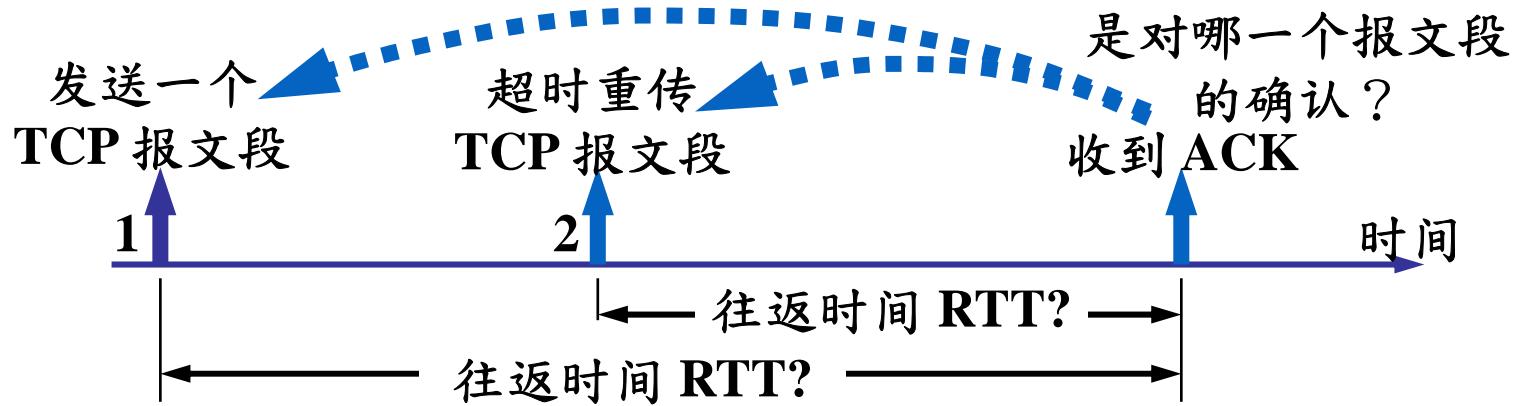


超时重传：时间的选择

- 超时重传时间（ Retransmission Time-Out , RTO ）应略大于上面得出的加权平均往返时间 RTT_S 。
 - $RTO = RTT_S + 4 \times RTT_D$ (RFC 2988)
 - RTT_D 是 RTT 的偏差的加权平均值。
 - 第一次测量时， RTT_D 值取为测量到的 RTT 样本值的一半。在以后的测量中，则使用下式计算加权平均的 RTT_D :
 - 新 $RTT_D = (1 - \beta) \times (\text{旧 } RTT_D) + \beta \times |RTT_S - \text{新 RTT 样本}|$
 - β 是个小于 1 的系数，其推荐值是 $1/4$ ，即 0.25。

超时重传：时间的选择

- 误判重传确认和原报文的确认

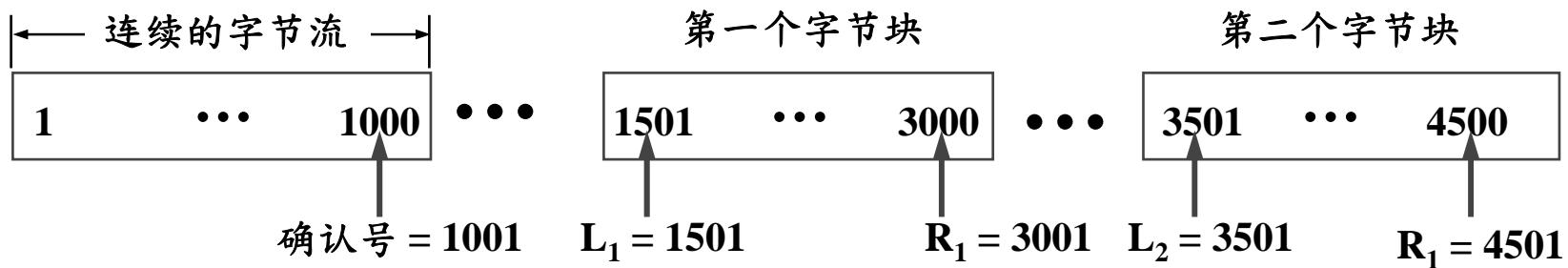


- Karn 算法

- 只要报文段重传了，就不采用其往返时间样本。
- 超时重传时间更新：每重传一次RTO乘以2，不重传时恢复

选择确认 (SACK)

- 接收方收到了和前面的字节流不连续的两个字节块。
- 如果这些字节的序号都在接收窗口之内，那么接收方就先收下这些数据，在选项区域把这些信息准确地告诉发送方，使发送方不再重复已收到的数据。



内容纲要

3 传输控制协议

3-1 介绍

3-2 基本机制

3-3 流量控制

3-4 拥塞控制



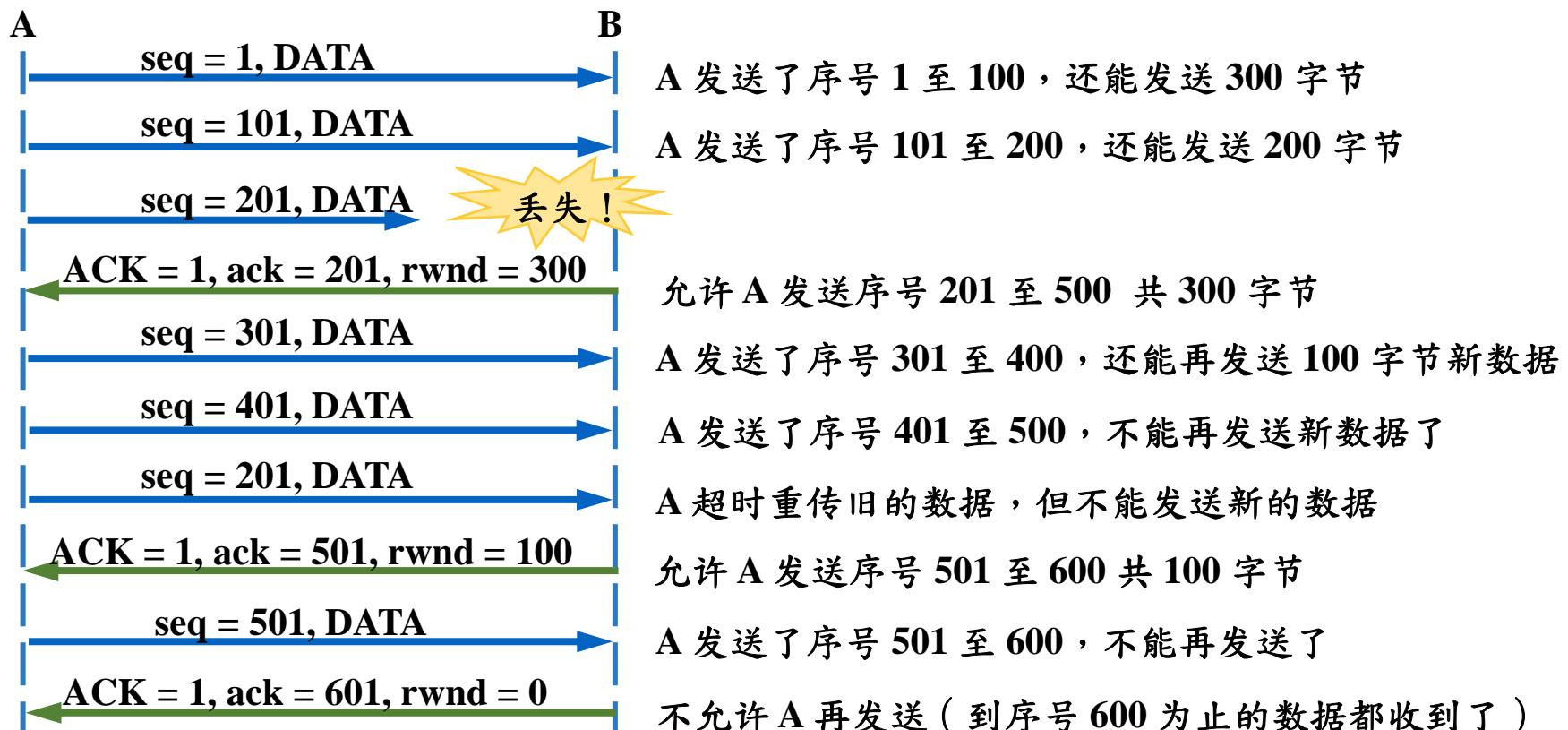
流量控制：滑动窗口

- 但如果发送方把数据发送得过快，接收方就可能来不及接收，这就会造成数据的丢失。
- 流量控制（flow control）的目的是使发送方的发送速率，让接收方来得及接收，且不使网络发生拥塞。
- 利用滑动窗口机制可以很方便地在TCP连接上实现流量控制。
- TCP窗口单位是字节，不是报文段。
 - 发送方不能超过接收方给出的接受窗口值



流量控制：滑动窗口举例

- A 向 B 发送数据。在连接建立时，
B 告诉 A：“我的接收窗口 $rwnd = 400$ （字节）”。



流量控制：滑动窗口举例

- TCP 为每一个连接设有一个持续计时器。
 - 只要一方收到对方的零窗口通知，就启动持续计时器。
- 若持续计时器设置的时间到期，就发送一个零窗口探测报文段（仅携带 1 字节的数据），而对方就在确认这个探测报文段时给出了现在的窗口值。
 - 若窗口仍然是零，则收到这个报文段的一方就重新设置持续计时器。
 - 若窗口不是零，则死锁的僵局就可以打破了。



流量控制：滑动窗口举例

- 传输效率：用不同机制控制 TCP 报文段的发送时机
 - 第一种机制是 TCP 维持一个变量，它等于最大报文段长度 MSS。只要缓存中存放的数据达到 MSS 字节时，就组装成一个 TCP 报文段发送出去。
 - 第二种机制是由发送方的应用进程指明要求发送报文段，即 TCP 支持的推送（ push ）操作。
 - 第三种机制是发送方的一个计时器期限到了，就把当前已有缓存数据装入报文段（但长度不能超过 MSS ）发送出去。



内容纲要

3 传输控制协议

3-2 基本机制

3-3 流量控制

3-4 拥塞控制

3-5 连接管理



拥塞控制：原理

- 在某段时间，若对网络中某资源的需求超过了该资源所能提供的可用部分，网络的性能就要变坏——产生拥塞(congestion)。
- 出现资源拥塞的条件：对资源需求的总和 > 可用资源
- 若网络中有许多资源同时产生拥塞，网络的性能就要明显变坏，整个网络的吞吐量将随输入负荷的增大而下降。



拥塞控制：与流量控制关系

- 拥塞控制

- 前提：网络能够承受现有的网络负荷。
 - 一个全局性的过程，涉及到所有主机、路由器，以及与降低网络传输性能有关的所有因素。

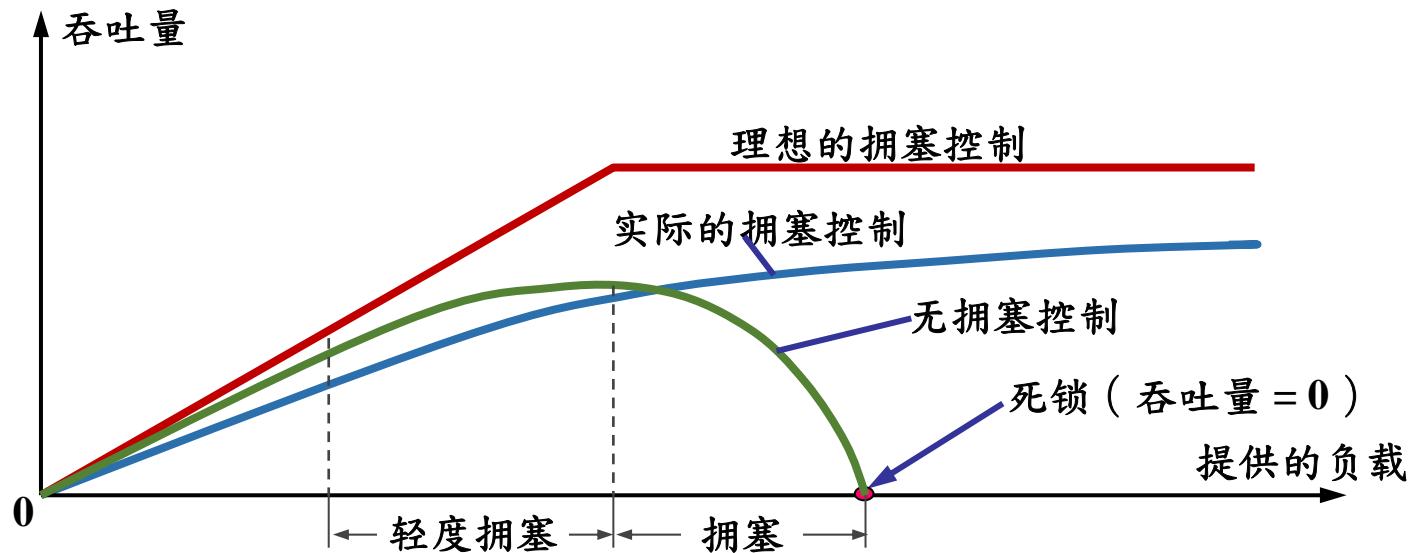
- 流量控制

- 指在给定的发送端和接收端之间的点对点通信量的控制。
 - 抑制发送端发送数据的速率，以便使接收端来得及接收。



拥塞控制：拥塞控制的作用

- 拥塞控制是一个动态的（而不是静态的）问题
 - 网络高速化，这很容易出现缓存不够大而造成分组的丢失。但分组的丢失是网络发生拥塞的征兆而不是原因。
 - 许多情况下，拥塞成为网络性能恶化甚至死锁的原因。



拥塞控制：拥塞控制思路

- **开环控制（Open loop）**

- 设计网络事先将有关拥塞的因素考虑周到，力求网络在工作时不产生拥塞。

- **闭环控制（Close loop）**

- 基于反馈环路的概念，属于闭环控制

- 措施

- 监测网络系统以便检测到拥塞在何时、何处发生
 - 将拥塞发生的信息传送到可采取行动的地方
 - 调整网络系统的运行以解决出现的问题

拥塞控制：拥塞窗口

- 拥塞窗口（congestion window）

- 发送方维持的状态变量。
 - 拥塞窗口的大小取决于网络的拥塞程度，并且动态地在变化。发送方让自己的发送窗口等于拥塞窗口。如再考虑到接收方的接收能力，则发送窗口还可能小于拥塞窗口。

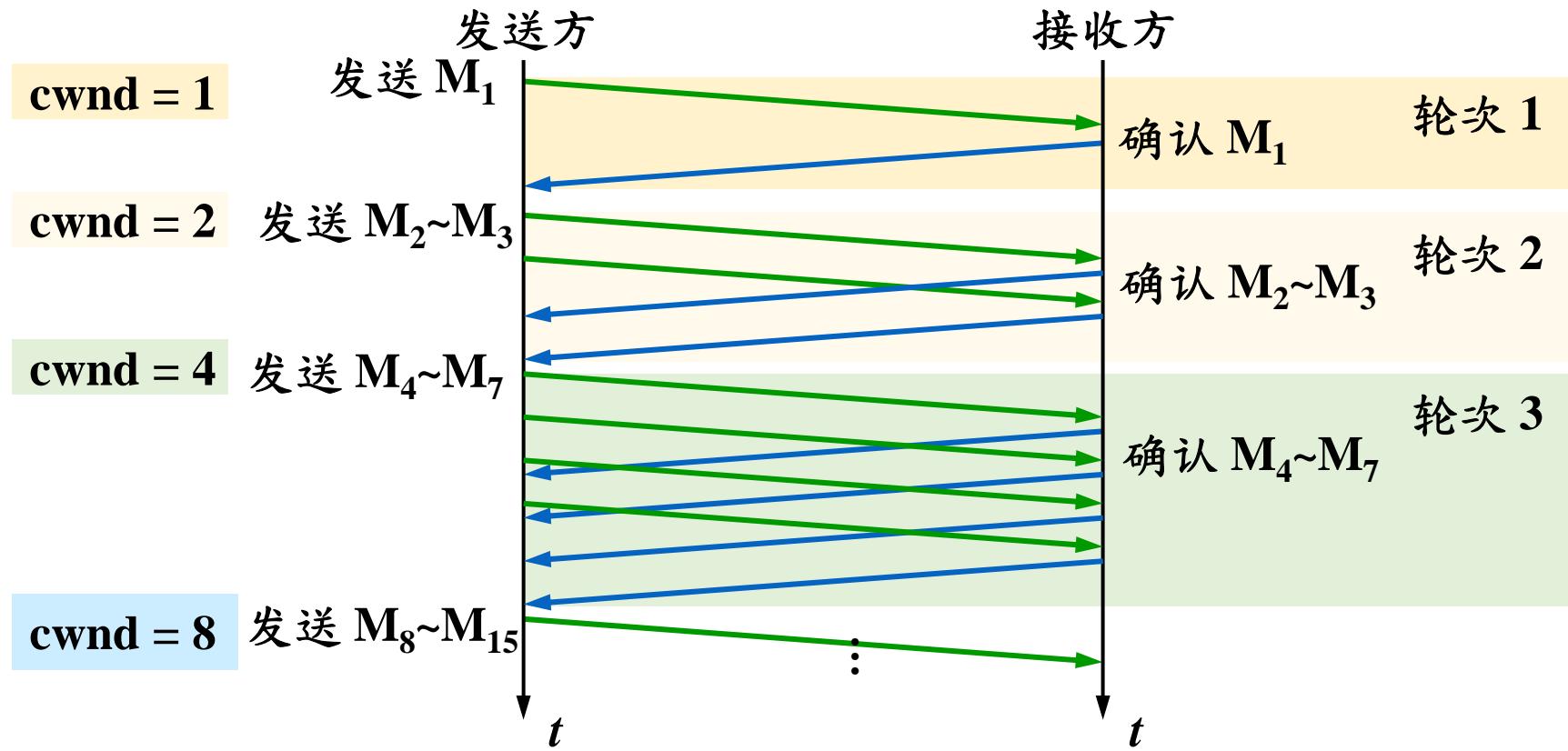
- 发送方控制拥塞窗口的原则是

- 只要网络没有出现拥塞，拥塞窗口就再增大一些，以便把更多的分组发送出去。但只要网络出现拥塞，拥塞窗口就减小一些，以减少注入到网络中的分组数。



拥塞控制：慢开始

- 发送方每收到一个对新报文段的确认（重传的不算在内）就使 cwnd 加 1。



拥塞控制：慢开始和拥塞避免

- 加性增大
 - TCP 初始化时，拥塞窗口置为 1，每收到确认增加1
 - 到达门限值（初始为16）改用拥塞避免算法
- 乘性减小
 - 一旦拥塞，门限值改为窗口大小的一半
- 拥塞避免
 - 在拥塞避免阶段，把拥塞窗口控制为按线性规律增长，使网络较不容易拥塞。（而非完全避免）



拥塞控制：慢开始和拥塞避免

发送端每收到一个确认，就把 cwnd 加 1。于是发送端可以接着发送 M_1 和 M_2 两个报文段。

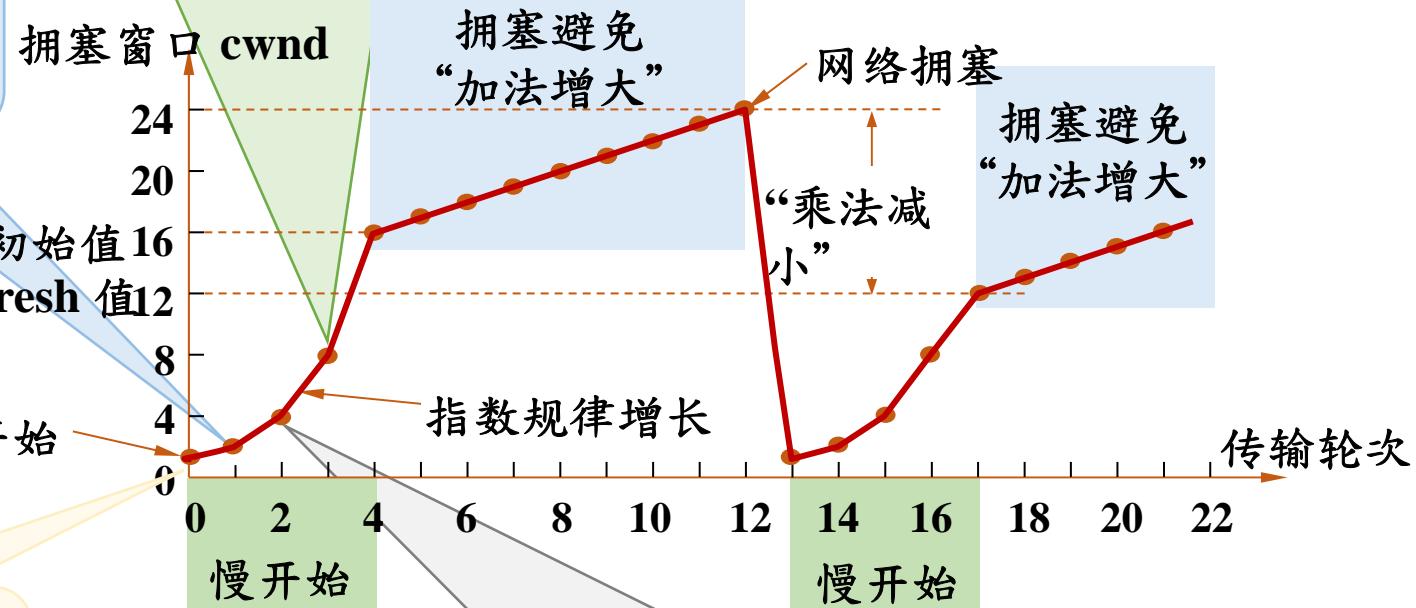
发送端每收到一个对新报文段的确认，就把发送端的拥塞窗口加 1，因此拥塞窗口 cwnd 随着传输轮次按指数规律增长。

ssthresh 的初始值
新的 ssthresh 值

12

慢开始

在执行慢开始算法时，拥塞窗口 cwnd 的初始值为 1，发送第一个报文段 M_0 。

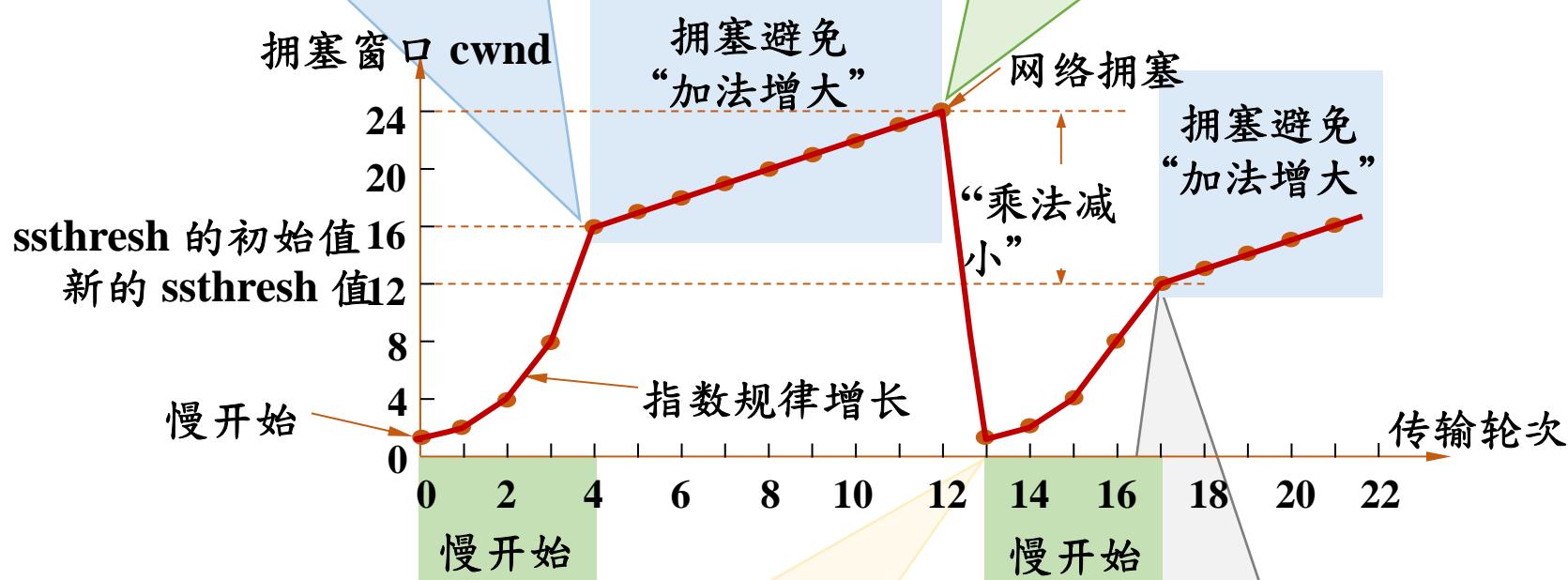


接收端共发回两个确认。发送端每收到一个对新报文段的确认，就把发送端的 cwnd 加 1。现在 cwnd 从 2 增大到 4，并可接着发送后面的 4 个报文段。

拥塞控制：慢开始和拥塞避免

当拥塞窗口 $cwnd$ 增长到慢开始门限值 $ssthresh$ 时（即当 $cwnd = 16$ 时），就改为执行拥塞避免算法，拥塞窗口按线性规律增长。

假定拥塞窗口的数值增长到 24 时，网络出现超时，表明网络拥塞了。



更新后的 $ssthresh$ 值变为 12（即发送窗口数值 24 的一半），拥塞窗口再重新设置为 1，并执行慢开始算法。

当 $cwnd = 12$ 时改为执行拥塞避免算法，拥塞窗口按线性规律增长，每经过一个往返时延就增加一个 MSS 的大小。

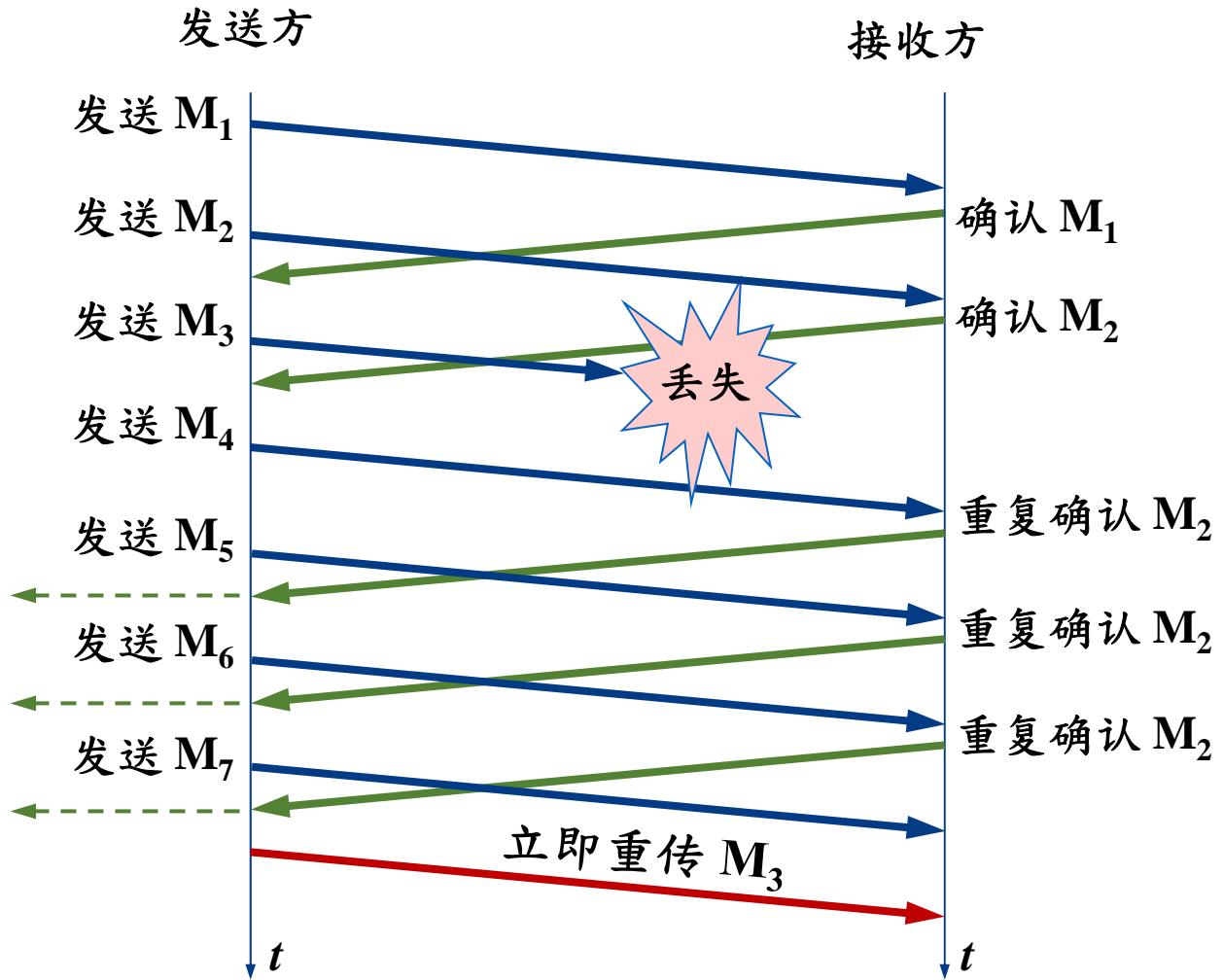
拥塞控制：快重传和快恢复

- 快重传算法首先要求接收方每收到一个失序的报文段后就立即发出重复确认。这样做可以让发送方及早知道有报文段没有到达接收方。
- 发送方只要一连收到三个重复确认就应当立即重传对方尚未收到的报文段。
- 不难看出，快重传并非取消重传计时器，而是在某些情况下可更早地重传丢失的报文段。



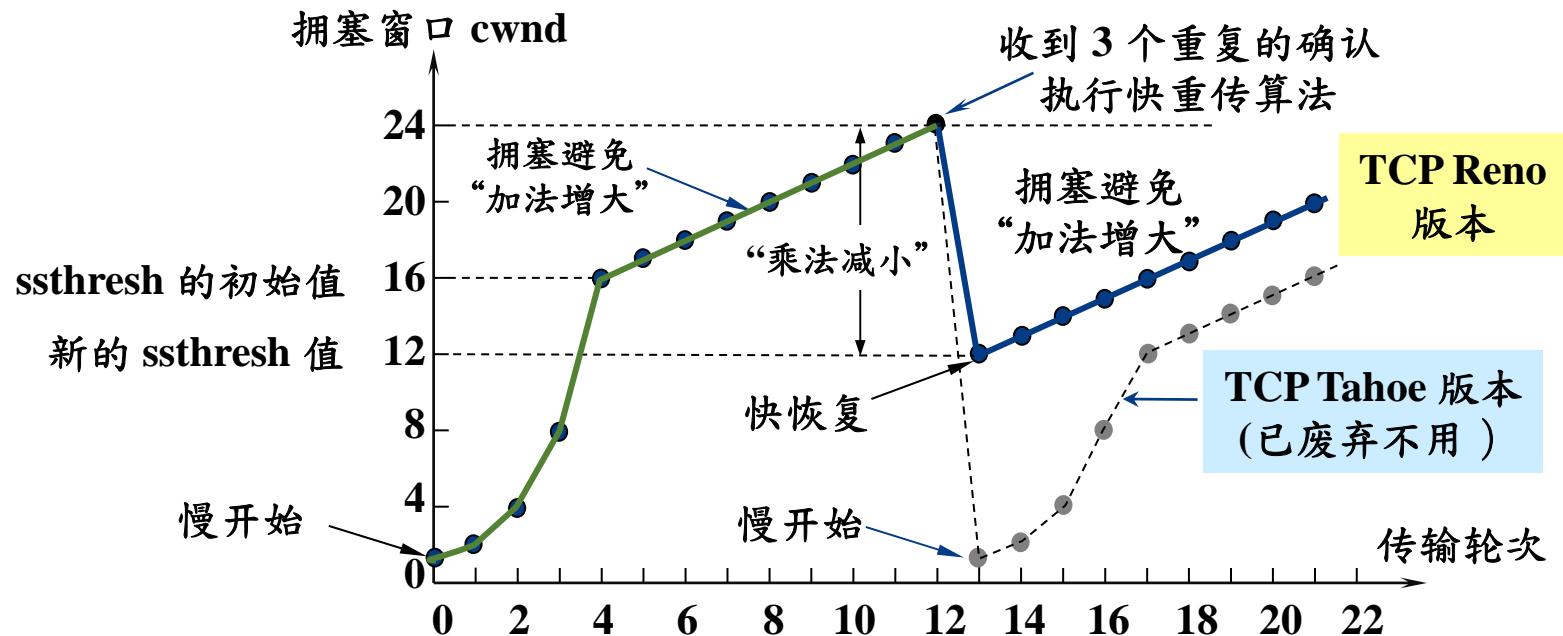
拥塞控制：快重传和快恢复

收到三个连续的
对 M_2 的重复确认
立即重传 M_3



拥塞控制：快重传和快恢复

- 当发送端收到连续三个重复的确认时，就执行“乘法减小”算法，把慢开始门限 ssthresh 减半。但接下去不执行慢开始算法。



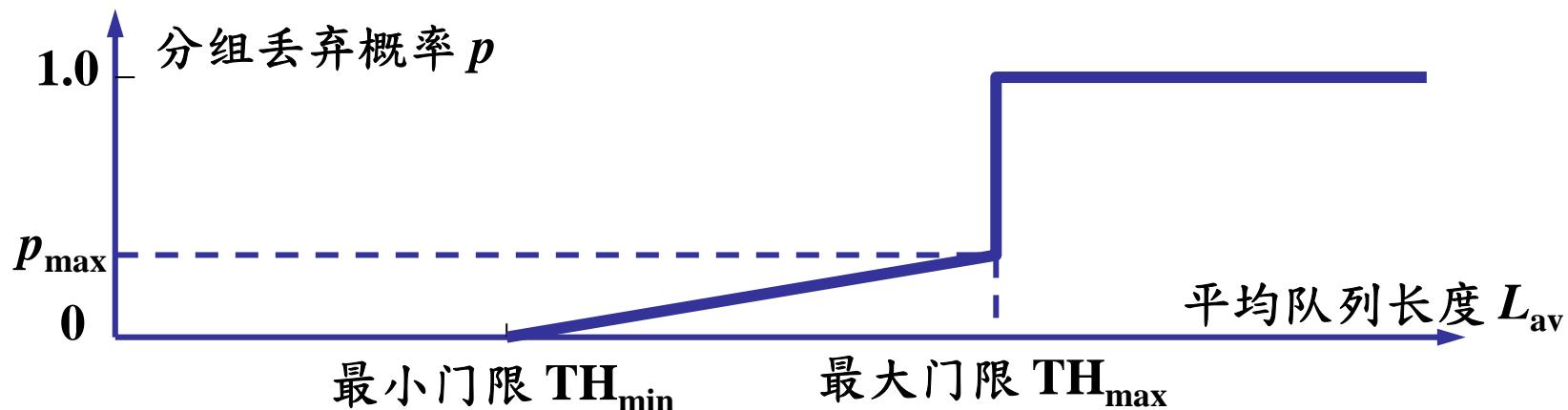
拥塞控制：快重传和快恢复

- 发送方的发送窗口的上限值应当取为接收方窗口 **rwnd** 和拥塞窗口 **cwnd** 这两个变量中较小的一个
- 即应按以下公式确定：
 - 发送窗口的上限值=**Min [rwnd, cwnd]**
 - 当 **rwnd < cwnd**，接收方的接收能力限制发送窗口的最大值。
 - 当 **cwnd < rwnd**，网络的拥塞限制发送窗口的最大值。



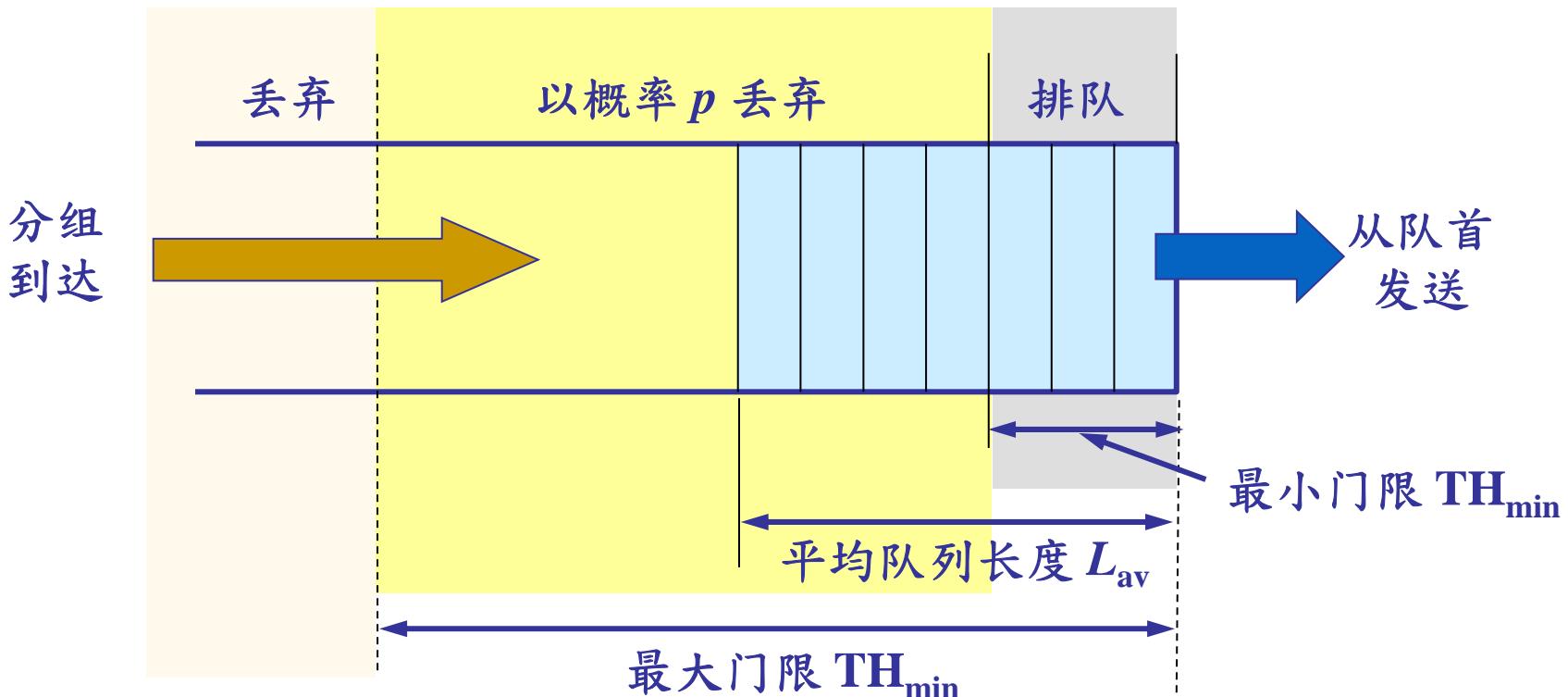
拥塞控制：随机早期检测RED

- 随机早期检测 RED (Random Early Detection)
 - 使路由器的队列维持两个参数，即队列长度最小门限 TH_{min} 和最大门限 TH_{max} 。
 - RED 对每一个到达的数据报都先计算平均队列长度 L_{AV} 。



拥塞控制：随机早期检测RED

- 不使用瞬时队列长度是因为突发数据不太可能使队列溢出



内容纲要

3 传输控制协议

3-2 基本机制

3-3 流量控制

3-4 拥塞控制

3-5 连接管理



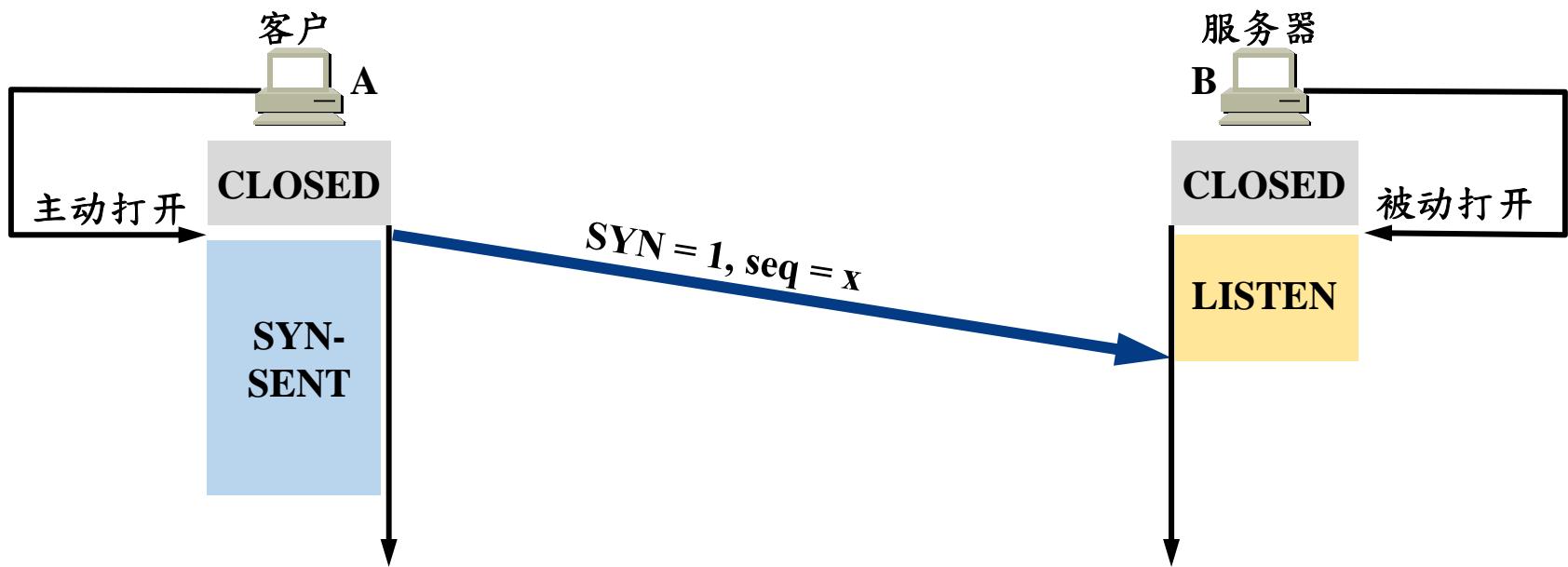
传输连接管理

- 三个阶段：连接建立、数据传送和连接释放。
- 通信双方：客户端、服务器端。
 - 主动发起连接的应用进程叫做客户（Client）
 - 被动等待连接建立的进程叫做服务器（Server）
- 要解决三个问题：
 - 要使每一方能够确知对方的存在。
 - 要允许双方协商一些参数（如：最大窗口大小等）。
 - 能够对运输实体资源（如缓存大小等）进行分配。



传输连接管理：连接建立

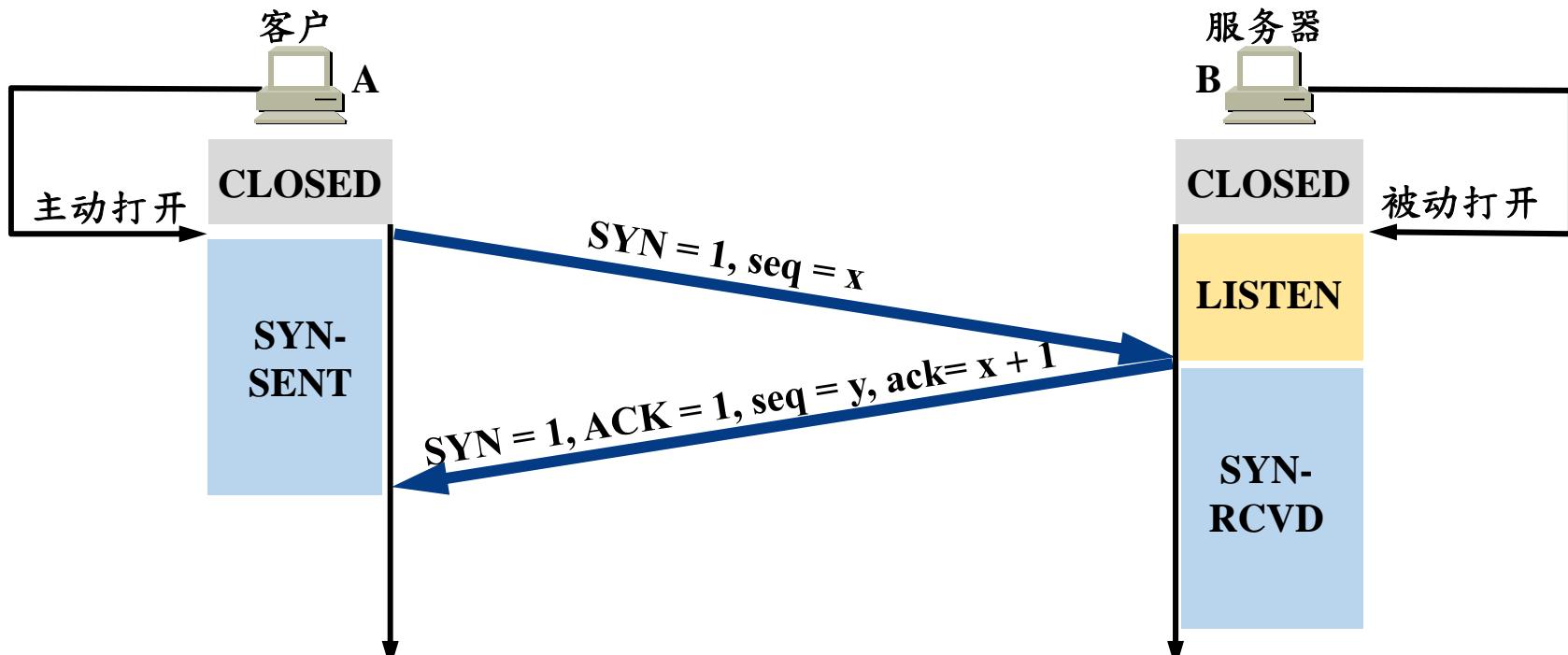
- 三次握手建立 TCP 连接



- A 的 TCP 向 B 发出连接请求报文段，其首部中的同步位 $SYN = 1$ ，并选择序号 $seq = x$ ，表明传送数据时的第一个数据字节的序号是 x 。

传输连接管理：连接建立

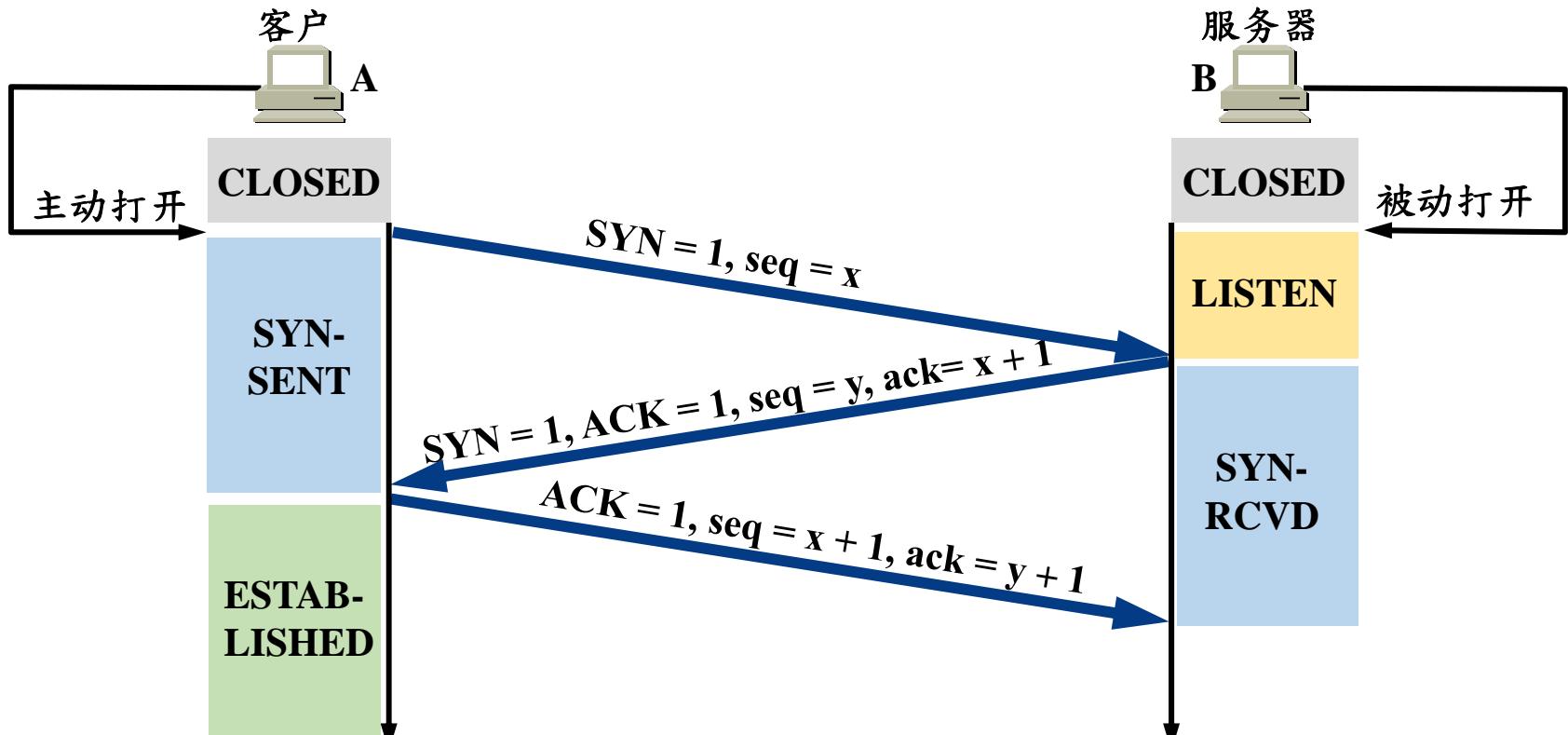
- 三次握手建立 TCP 连接



- B 的 TCP 收到连接请求报文段后，如同意，则发回确认。
- B 在确认报文段中应使 $SYN = 1$ ，使 $ACK = 1$ ，其确认号 $ack = x + 1$ ，自己选择的序号 $seq = y$ 。

传输连接管理：连接建立

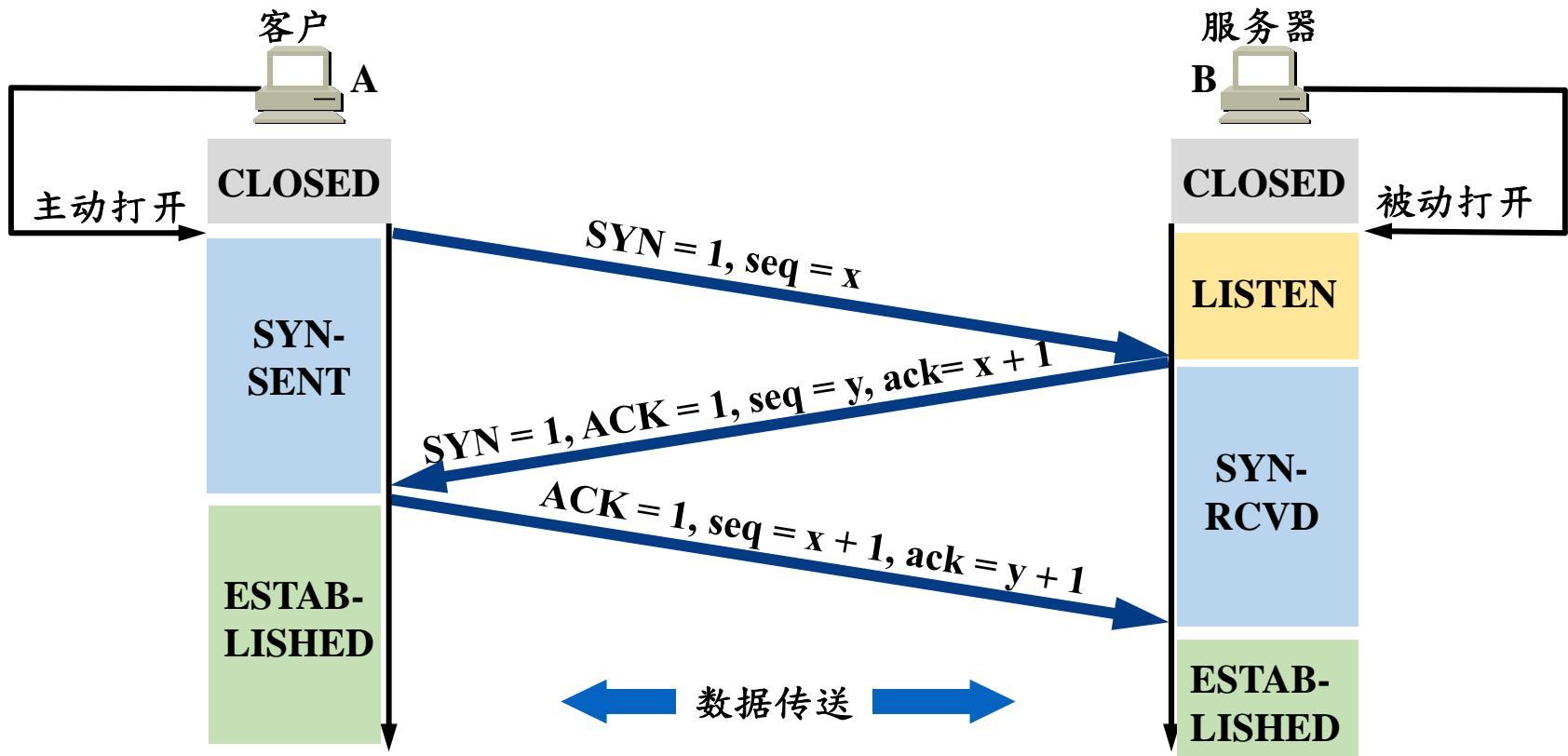
- 三次握手建立 TCP 连接



- A 收到此报文段后向 B 确认， $ACK=1$ ，确认号 $ack = y + 1$ 。
- A 的 TCP 通知上层应用进程，连接已经建立。

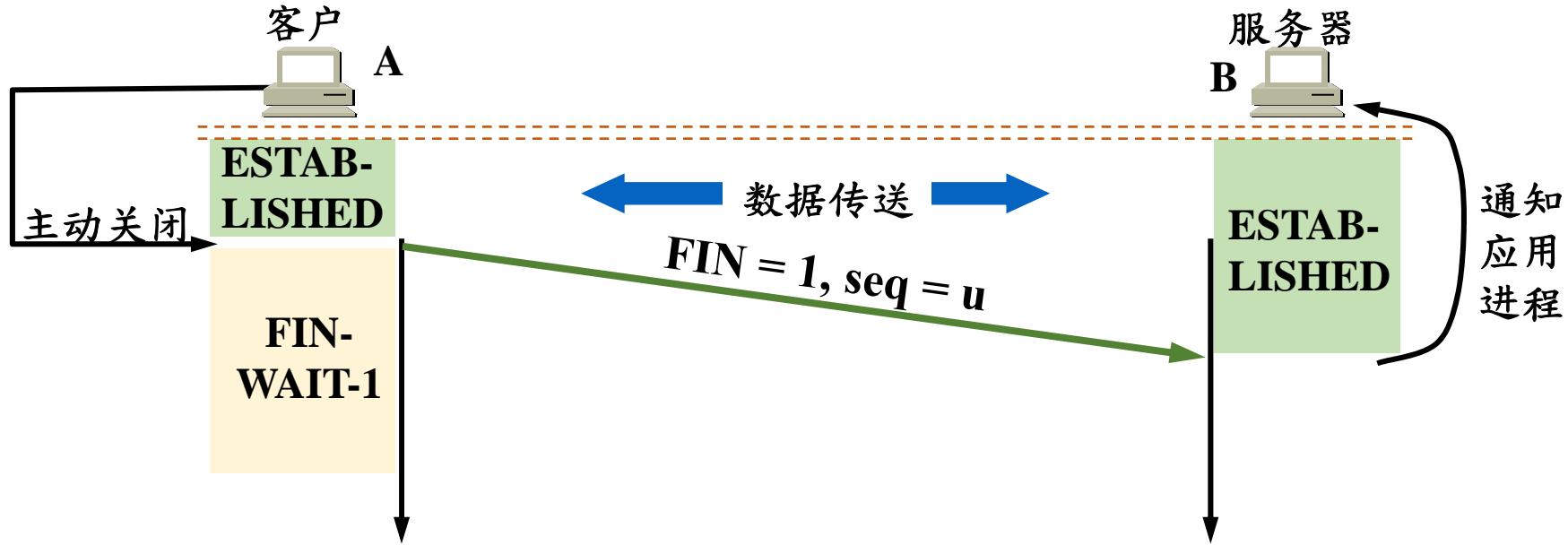
传输连接管理：连接建立

- 三次握手建立 TCP 连接



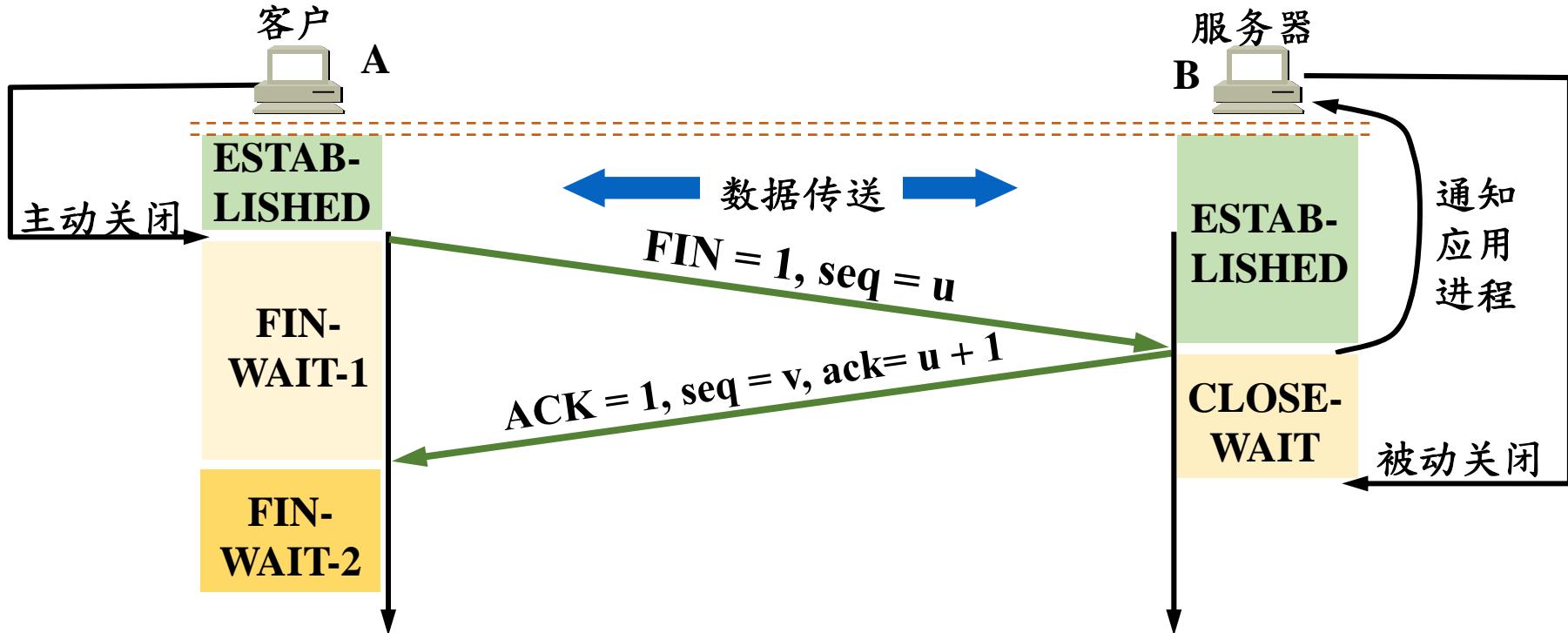
- B收到A确认后，通知其上层应用进程TCP连接已经建立。

传输连接管理：连接解除



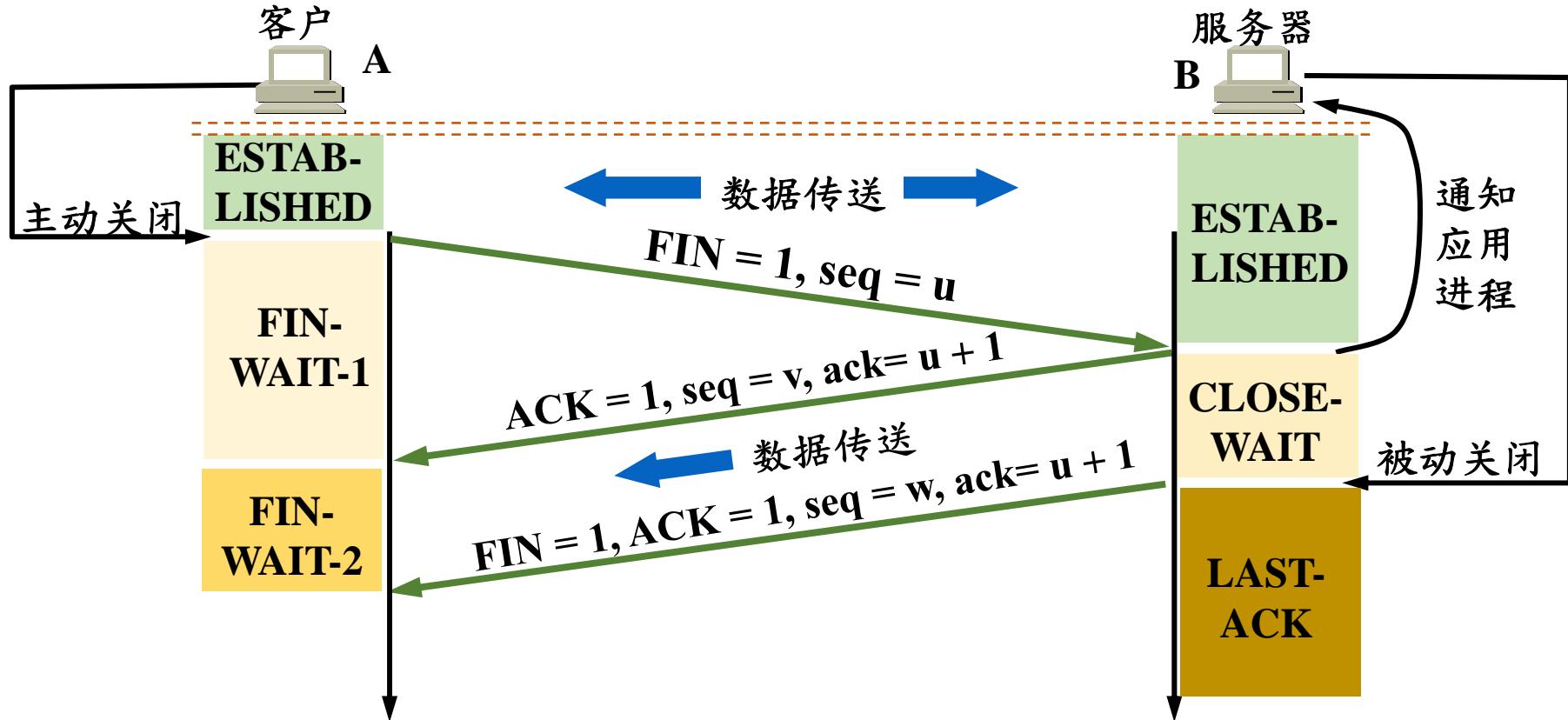
- 数据传输结束后，通信的双方都可释放连接。现在 A 的应用进程先向其 TCP 发出连接释放报文段，并停止再发送数据，主动关闭 TCP连接。
- A 把连接释放报文段首部的 $FIN = 1$ ，其序号 $seq = u$ ，等待 B 的确认。

传输连接管理：连接解除



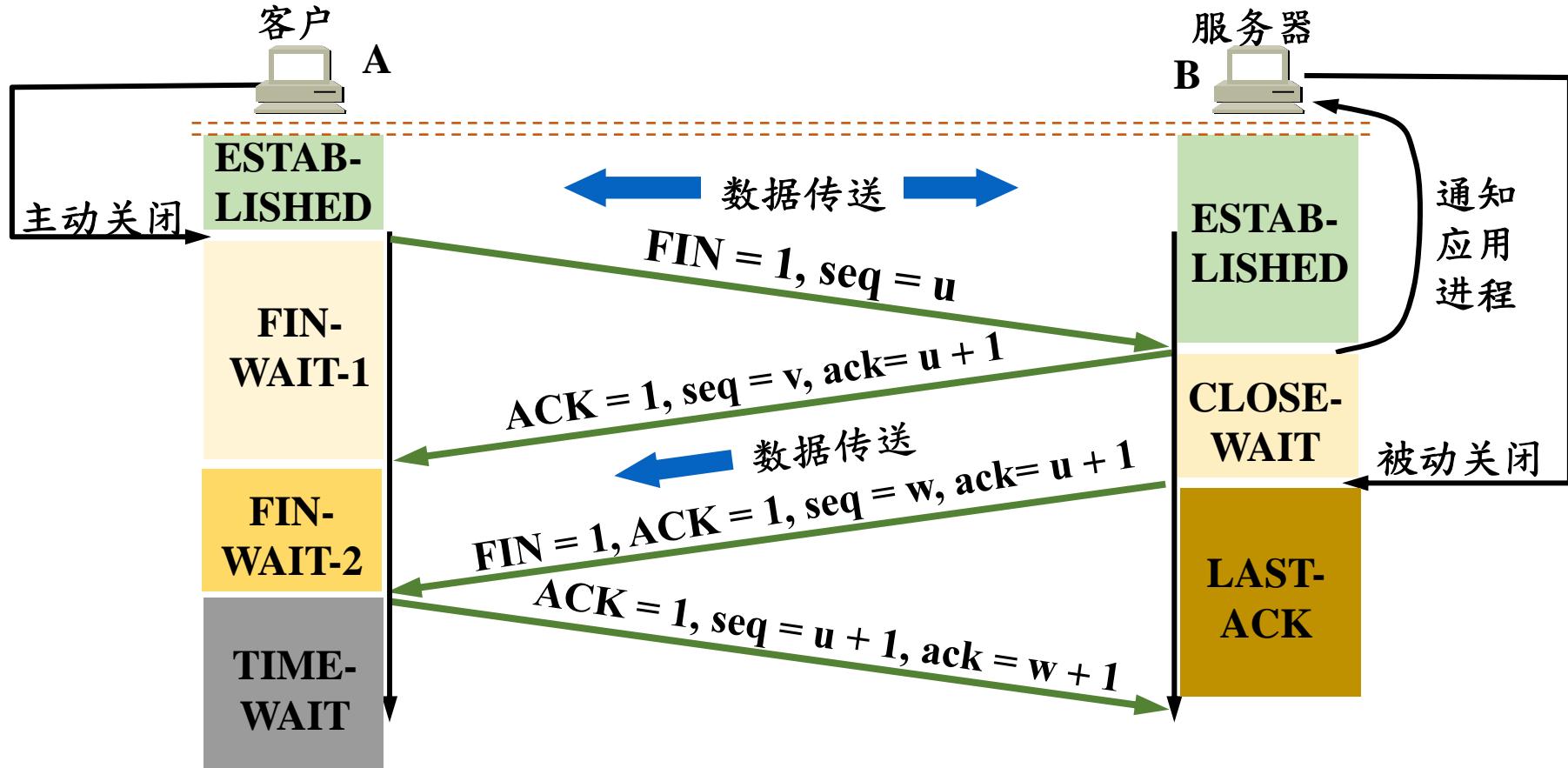
- B 发出确认，确认号 $ack=u+1$ ，该报文段的序号 $seq=v$ 。
- TCP 服务器进程通知高层应用进程。
- 从 A 到 B 这个方向的连接就释放了，TCP 连接处于半关闭状态。B 若发送数据，A 仍要接收。

传输连接管理：连接解除



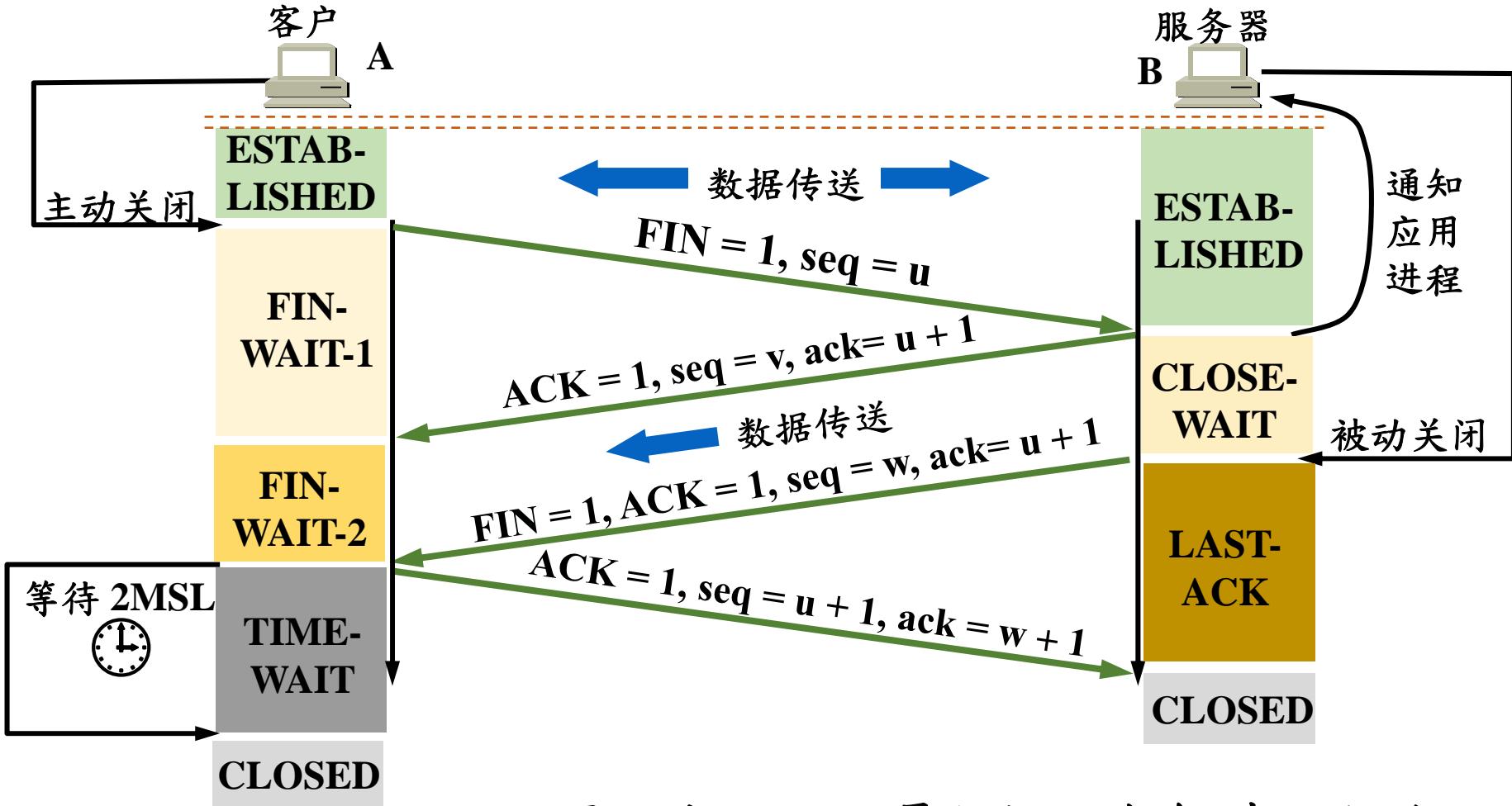
- 若 B 已经没有要向 A 发送的数据，其应用进程就通知 TCP 释放连接。

传输连接管理：连接解除



- A 收到连接释放报文段后，必须发出确认。
- 在确认报文段中 $ACK=1$ ，确认号 $ack=w+1$ ，序号 $seq=u+1$

传输连接管理：连接解除



- 必须经过 2MSL (最长报文寿命) 真正释放。

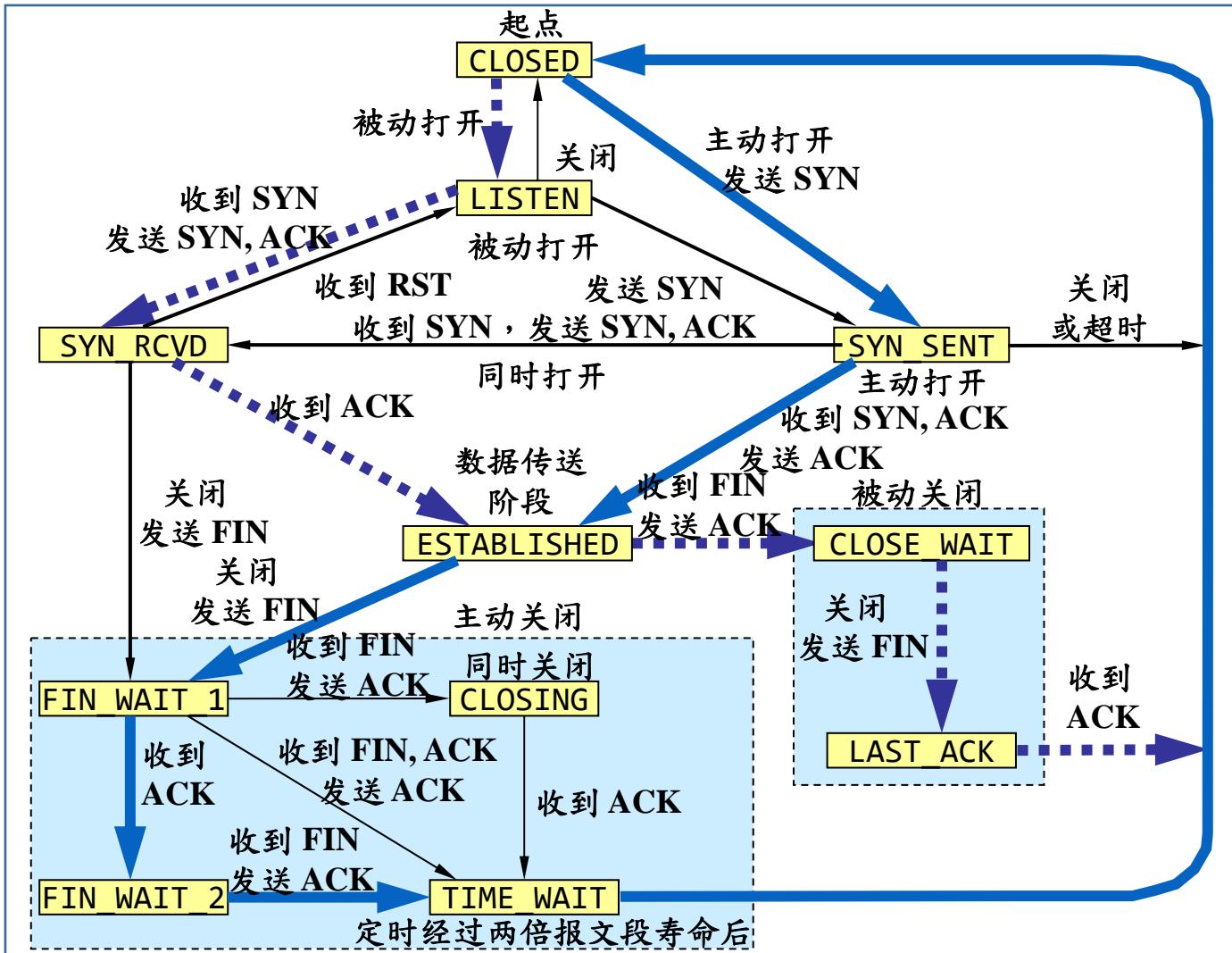
传输连接管理：有限状态机

- 图中每一个方框都是 TCP 可能具有的状态。
- 方框中的大写英文字符串是标准 TCP 连接状态名。
- 状态之间的箭头表示可能发生的状态变迁。
 - 粗实线箭头表示对客户进程的正常变迁。
 - 粗虚线箭头表示对服务器进程的正常变迁。
 - 另一种细线箭头表示异常变迁。
- 箭头旁边的字，表明引起这种变迁的原因，或表明发生状态变迁后又出现什么动作。



传输连接管理：有限状态机

- 有限状态机



内容纲要

1

传输层概述

2

用户数据报协议

3

传输控制协议

4

TCP的程序示例

5

小结



连接时间服务器的TCP端口

```
using System;
using System.Net.Sockets;
using System.Text;
public class TcpTimeClient {
    private const int portNum = 13;
    private const string hostName = "time.nist.gov";
    public static int Main(String[] args) {
        try {
            TcpClient client = new TcpClient(hostName, portNum);
            NetworkStream ns = client.GetStream();
            byte[] bytes = new byte[1024];
            int bytesRead = ns.Read(bytes, 0, bytes.Length);
            Console.WriteLine(Encoding.ASCII.GetString(bytes, 0,
bytesRead));
            client.Close();
        }
    }
}
```

连接时间服务器的TCP端口（续）

```
    }
    catch (Exception e) {
        Console.WriteLine(e.ToString());
    }
    return 0;
}
```

}

Output on success:

56399 13-04-17 06:07:47 50 0 0 658.3 UTC(NIST) *

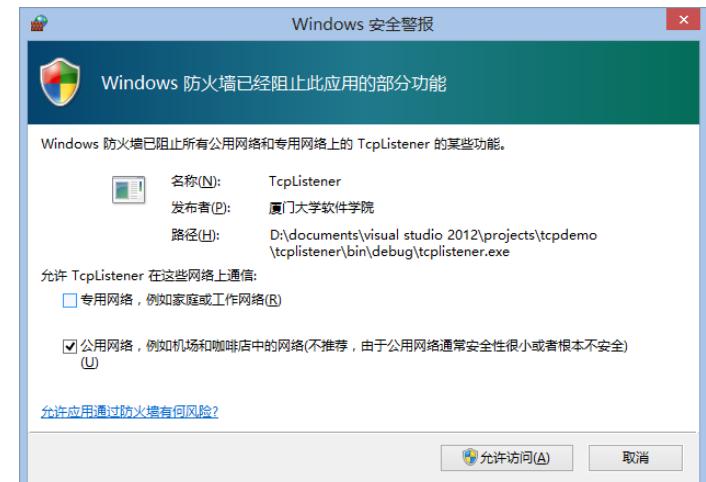
Output on failure:

System.Net.Sockets.SocketException (0x80004005): 由于连接方在一段时间后没有正确答复或连接的主机没有反应，连接尝试失败。 65.55.21.14:13

在 System.Net.Sockets.TcpClient..ctor(String hostname, Int32 port)

监视端口13以提供时间服务

```
using System;
using System.Net.Sockets;
using System.Text;
public class TcpTimeServer {
    private const int portNum = 13;
    public static int Main(String[] args) {
        bool done = false;
        TcpListener listener = new TcpListener(portNum);
        listener.Start();
        while (!done) {
        .....
        }
        listener.Stop();
        return 0;
    }
}
```



监视端口13以提供时间服务（续）

```
while (!done) {  
    Console.WriteLine("Waiting for connection...");  
    TcpClient client = listener.AcceptTcpClient();  
  
    Console.WriteLine("Connection accepted.");  
    NetworkStream ns = client.GetStream();  
  
    byte[] byteTime =  
Encoding.ASCII.GetBytes(DateTime.Now.ToString());  
    try {  
        ns.Write(byteTime, 0, byteTime.Length);  
        ns.Close();  
        client.Close();  
    }  
    catch (Exception e)
```

监视端口13以提供时间服务（续）

```
{  
    Console.WriteLine(e.ToString());  
}  
}
```

Output:

Waiting for connection...Connection accepted.
Waiting for connection...

内容纲要

1

传输层概述

2

用户数据报协议

3

传输控制协议

4

TCP的程序示例

5

小结



计算机网络

Computer Network

12

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



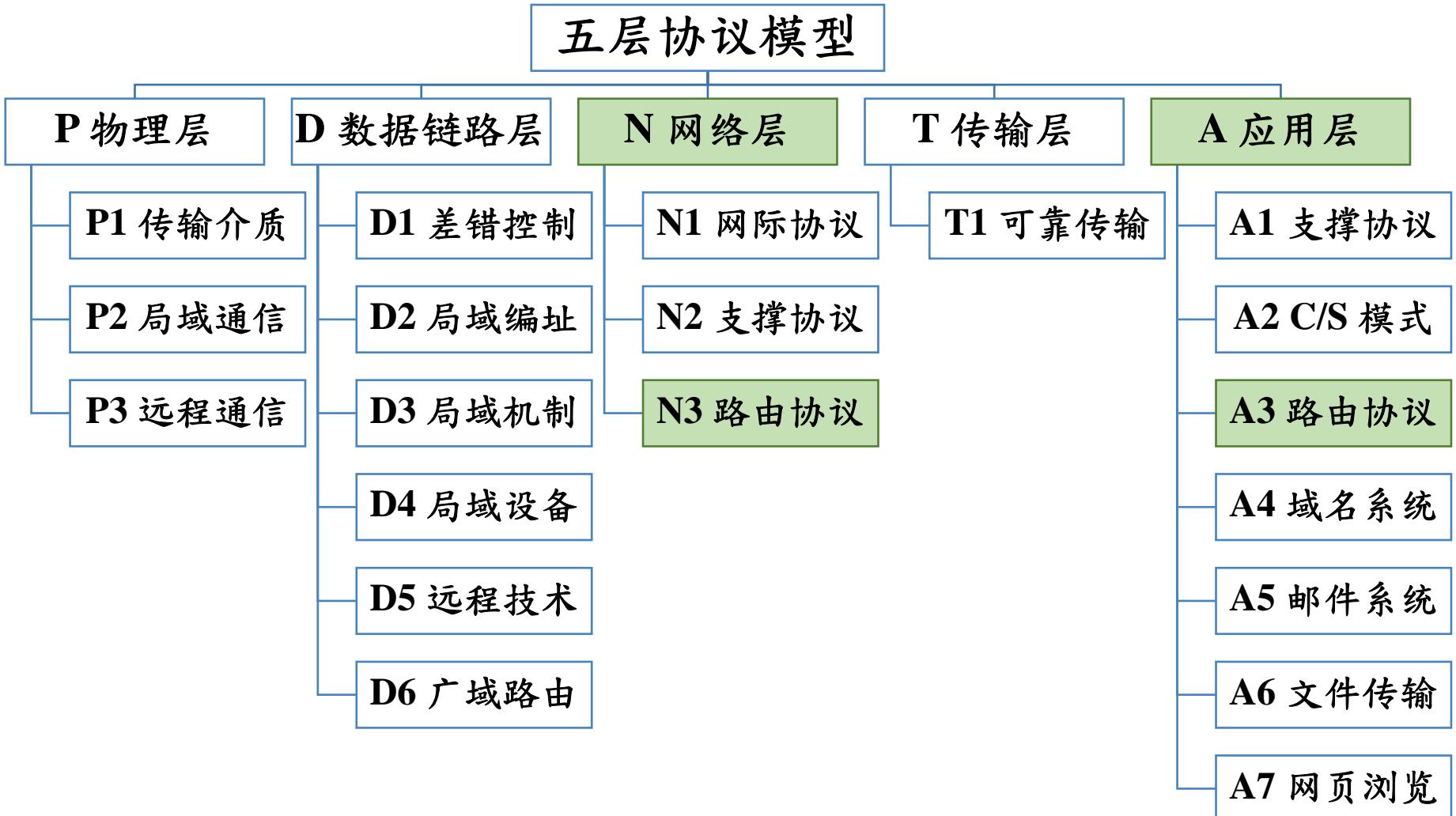
信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

13

因特网路由



知识框架



主要内容

- 静态路由与动态路由
- 自治系统（AS）的概念
- 内部网关协议（IGP）
 - RIP协议的工作原理和特点
 - OSPF协议的工作原理和特点
- 外部网关协议（EGP）
 - BGP协议



对应课本章节

- PART IV Internetworking
 - Chapter 27 Internet Routing And Routing Protocols

内容纲要

1

路由协议的基本概念

2

自治系统

3

路由信息协议

4

开放最短路径优先协议

5

边界网关协议



路由协议

- 路由协议（ Routing Protocol ）
- 定义
 - 一种指定数据报文转送方式的网络协议。
- 原理
 - 路由协议创建了路由表，描述了网络拓扑结构。
 - 路由协议与路由器协同工作，执行路由选择和数据包转发功能。

路由选择的复杂性

- 路由选择的复杂性
 - 它是网络中的所有结点共同协调工作的结果。
 - 路由选择的环境往往是不断变化的，而这种变化有时无法事先知道。
- 不存在一种绝对的最佳路由算法。
 - 所谓“最佳”只能是相对于某一种特定要求下得出的较为合理的选择而已。



路由算法的自适应性分类

- 静态路由选择策略（非自适应路由选择）
 - 简单和开销较小，但不能及时适应网络状态的变化。
- 动态路由选择策略（自适应路由选择）
 - 能较好地适应网络状态的变化，但实现较复杂，开销较大。



内容纲要

1

路由协议的基本概念

2

自治系统

3

路由信息协议

4

开放最短路径优先协议

5

边界网关协议



分层次的路由选择协议

- 因特网采用分层次的路由选择协议。
- 不应该让所有的路由器记录所有的网络的到达方式
 - 路由表非常大
 - 处理花时间，路由器间交换路由信息所需的带宽大。
 - 隐私考虑
 - 许多单位不愿意外界了解自己单位网络的布局细节和本部门所采用的路由选择协议，但同时还希望连接到因特网上。



自治系统 AS (Autonomous System)

- 自治系统 (Autonomous System)
 - 定义：自治系统是在单一的技术管理下的一组路由器
 - 域内路由选择：使用一种 AS 内部的路由选择协议和共同度量以确定分组在该 AS 内的路由
 - 现在对自治系统 AS 的定义是强调下面的事实：尽管一个 AS 使用了多种内部路由选择协议和度量，但重要的是一个 AS 对其他 AS 表现出的是一个**单一的和一致的**路由选择策略。
 - 域间路由选择：使用一种 AS 之间的路由选择协议用以确定分组在 AS 之间的路由



路由选择协议分层

- 内部网关协议（Interior Gateway Protocol，IGP）
 - 在一个自治系统内部使用的路由选择协议。
 - 目前这类路由选择协议使用得最多
 - 示例：RIP 和 OSPF 协议。
- 外部网关协议（External Gateway Protocol，EGP）
 - 自治系统边界网关使用的路由选择协议
 - 若源站和目的站处在不同的自治系统中，当数据报传到一个自治系统的边界时，就需要使用一种协议传到另一个自治系统中。
 - 示例：BGP-4。

RFC对路由和网
关应视为同义词



内容纲要

1

路由协议的基本概念

2

自治系统

3

路由信息协议

4

开放最短路径优先协议

5

边界网关协议

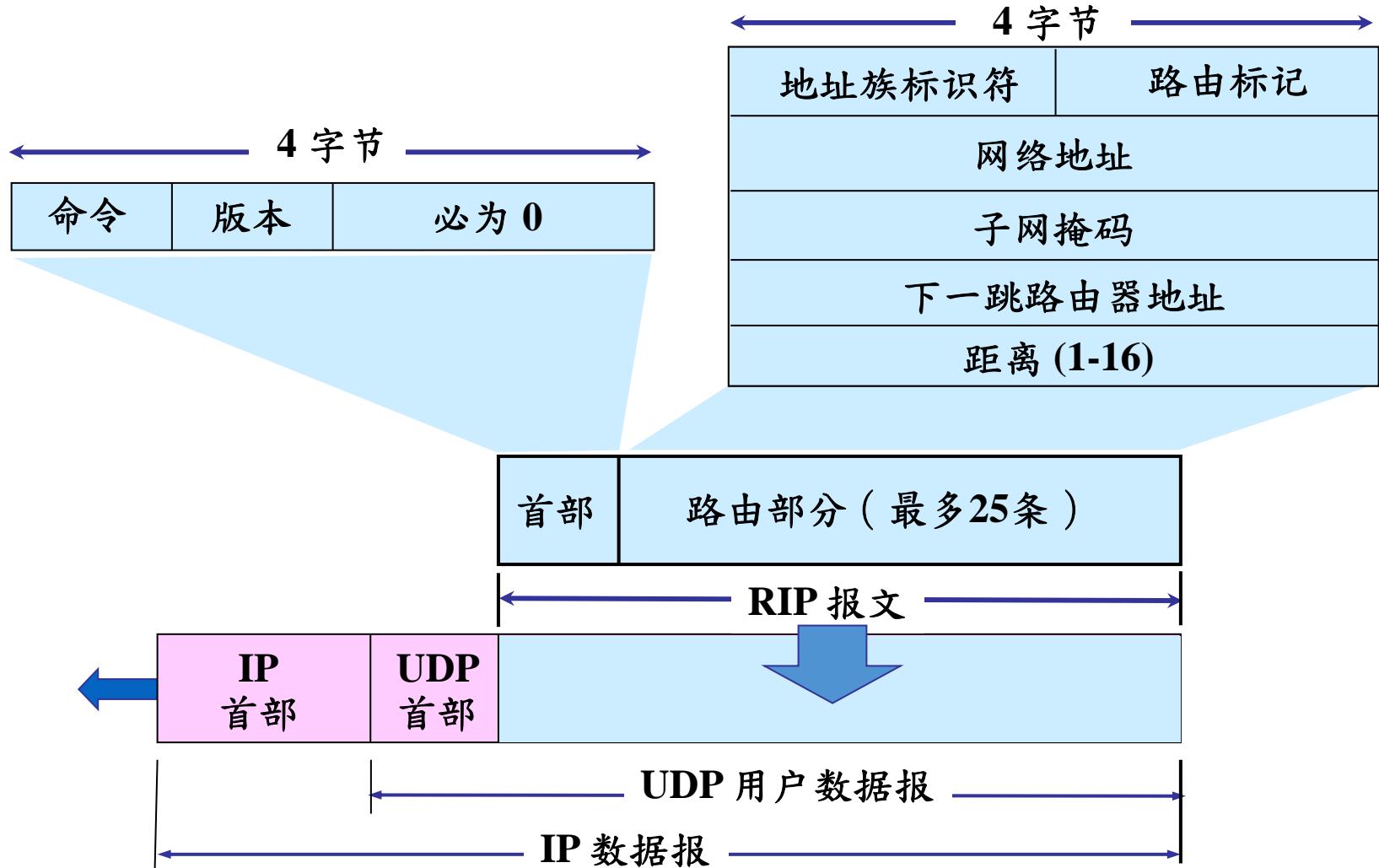


内部网关协议 RIP

- 路由信息协议（ Routing Information Protocol , RIP ）
 - 最先得到广泛使用的内部网关协议。
- 工作原理：分布式的基于距离向量的路由选择协议
 - 网络中每个路由器维护从自己到其它目的网络的距离记录。
- 要点
 - 仅和相邻路由器交换信息。
 - 交换的信息是自己的路由表。
 - 按固定间隔时间交换路由信息。



RIP2 协议的报文格式



RIP2 报文

- RIP2 报文中的路由部分由若干个路由信息组成。
 - 每个路由信息需要用 20 个字节。
 - 地址族标识符字段用来标志所使用的地址协议。
- 路由标记填入自治系统的号码
 - 这是考虑使RIP可能收到本自治系统以外的路由选择信息。
- 随后填入某个网络地址、该网络的子网掩码、下一跳路由器地址以及到此网络的距离。



RIP距离的定义

- 距离也称为跳数（ hop count ）
 - 从路由器到直接连接的网络的距离定义为 1。
 - RIP 选择一个具有最少路由器的路由（即最短路由），哪怕还存在另一条高速（低时延）但路由器较多的路由。
 - 每经过一个路由器，跳数就加 1。
 - 距离为 16 时即相当于不可达。（只适用于小型互联网）
- 好的路由就是它通过的路由器的数目少
 - 不能在两个网络之间同时使用多条路由。

路由表的建立

- 路由器刚开始工作时
 - 记录到直连网络的距离（记为1）。
- 路由器更新路由信息
 - 和有限的相邻路由器交换并更新路由信息。
- 收敛过程块
 - 经过若干次更新后，所有的路由器最终知道到达本自治系统中任何网络的最短距离和下一跳路由器的地址。
 - 在自治系统中所有的结点都得到正确的路由选择信息。



距离向量算法

- 算法基础：Bellman-Ford 算法

- 要点：设 X 是结点 A 到 B 的最短路径上的一个结点。若把路径 A 到 B 拆成两段路径 A 到 X 和 X 到 B，则每段路径 A 到 X 和 X 到 B 也都分别是结点 A 到 X 和结点 X 到 B 的最短路径。

- 收到相邻路由器（其地址为 X）的一个 RIP 报文

- 第一步，修改此 RIP 报文中的所有项目
 - 把“下一跳”字段中的地址都改为 X
 - 把所有的“距离”字段的值加 1。

距离向量算法

- 收到相邻路由器（其地址为 X）的一个 RIP 报文
 - 第二步，对修改后的 RIP 报文中的每个项目，重复：

目的网络是否在路由表中	下一跳地址和已有记录相同	距离比已有记录小	处理
否			添加
是	是		替换
	否	是	更新
	否	新路更新，	

则把此相邻路由器距离置为 16（不可达）。

路由器之间交换信息

- RIP协议工作过程

- 让互联网的所有路由器都和相邻路由器不断交换路由信息
 - 并不断更新其路由表，使得每个路由器到每个目的网络的路由都是最短的（即跳数最少）。

- 结果

- 虽然所有的路由器最终都拥有了整个自治系统的全局路由信息，但由于每一个路由器的位置不同，它们的路由表当然也应当是不同的。



RIP 协议的优缺点

- 优点

- 实现简单，开销较小。

- 缺点

- 限制了网络的规模，最大距离为 15（16 表示不可达）。
 - 当网络出现故障时，要经过比较长的时间才能将此信息传送到所有的路由器。
 - 路由器之间交换的路由信息是路由器中的完整路由表，因而随着网络规模的扩大，开销也就增加。



RIP 协议的演示

正常情况



左边数字1表示
从本路由器到网 1

中间数字1表示
距离是 1

右侧为数字表示下一跳路由器号；
或为减号，表示直接交付

1 1 -

R_1 说：“我到网 1 的距离是 1，是直接交付。”

RIP 协议的演示

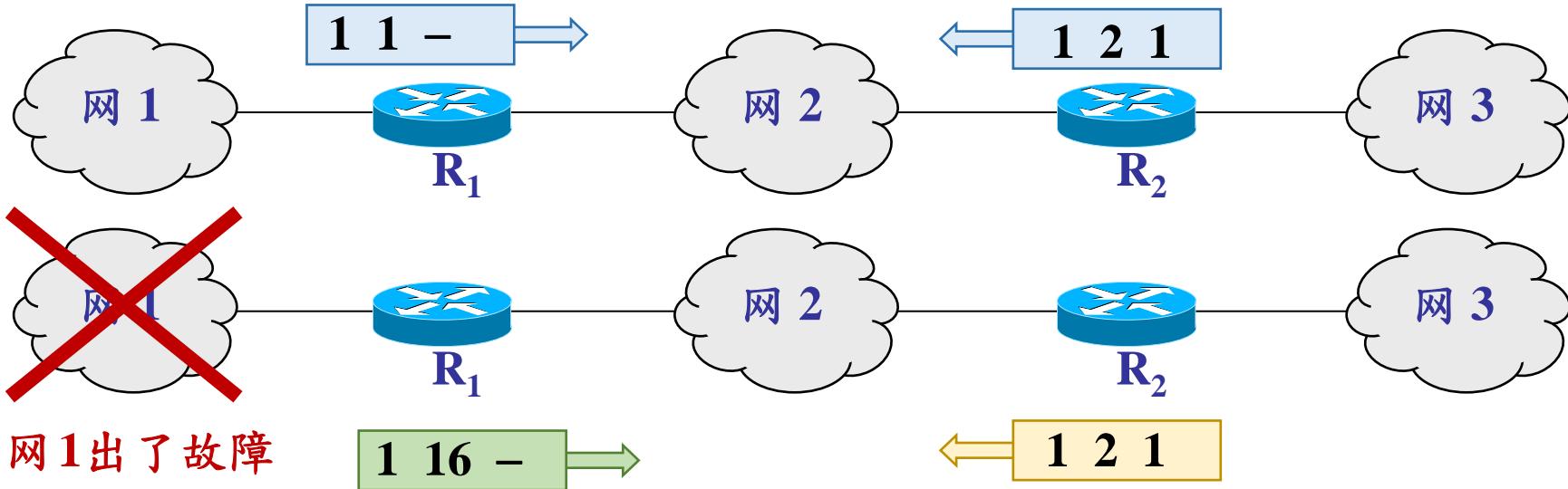
正常情况



R_2 说：“我到网 1 的距离是 2，是经过 R_1 。”

RIP 协议的演示

正常情况

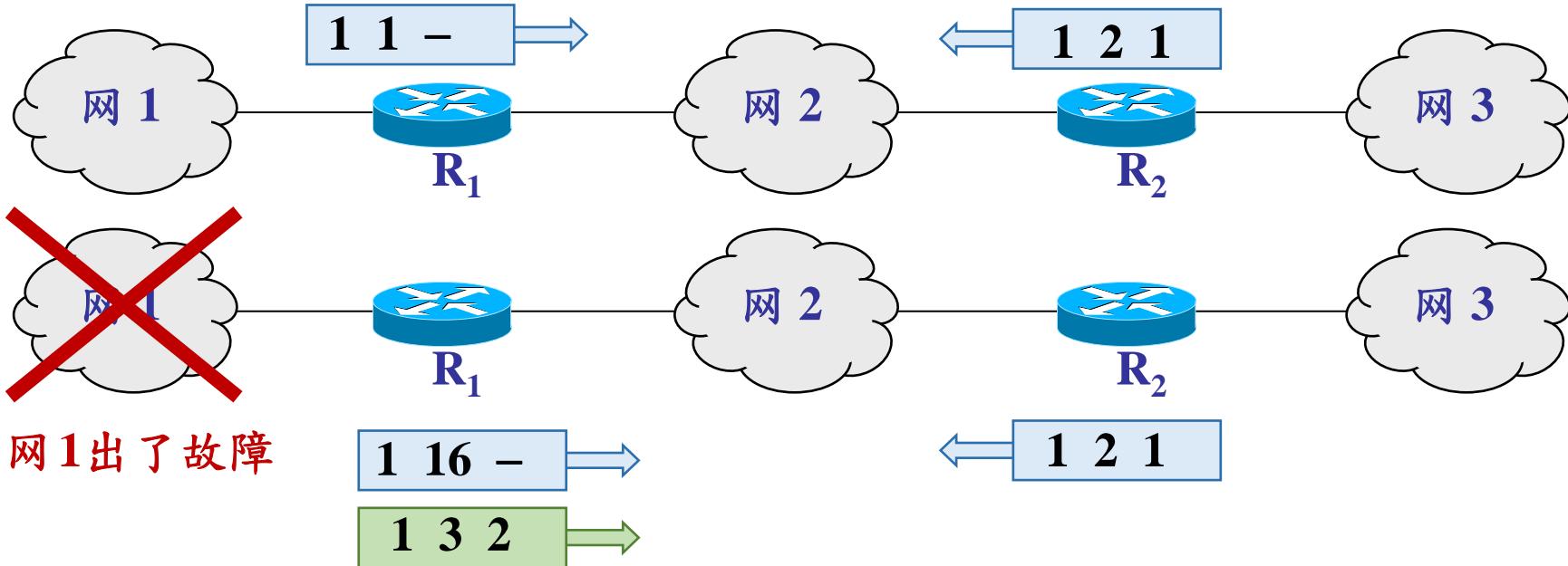


R₁ 说：“我到网 1 的距离是 16（表示无法到达），是直接交付。”

但 R₂ 在收到 R₁ 的更新报文之前，还发送原来的报文，因为这时 R₂ 并不知道 R₁ 出了故障。

RIP 协议的演示

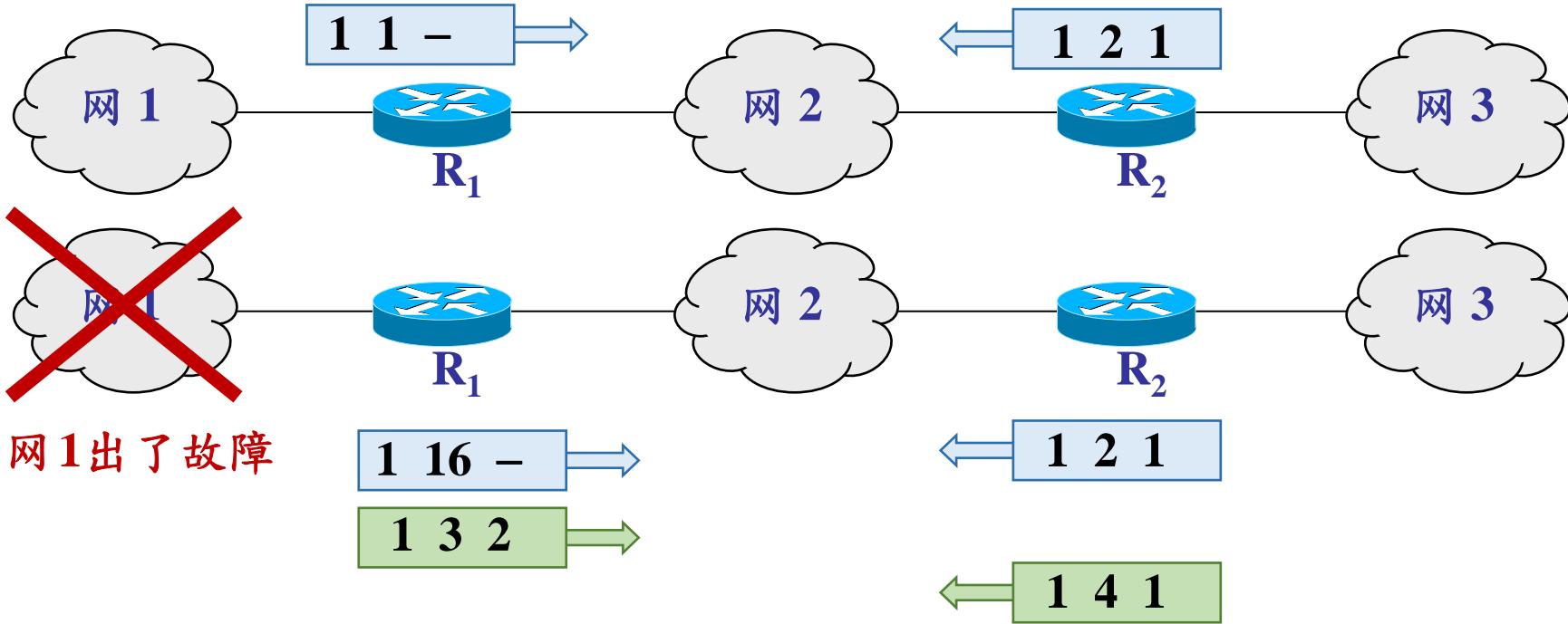
正常情况



R₁ 收到 R₂ 的更新报文后，误认为可经过 R₂ 到达网1，于是更新自己的路由表，说：“我到网1的距离是3，下一跳经过R₂”。然后将此更新信息发送给 R₂。

RIP 协议的演示

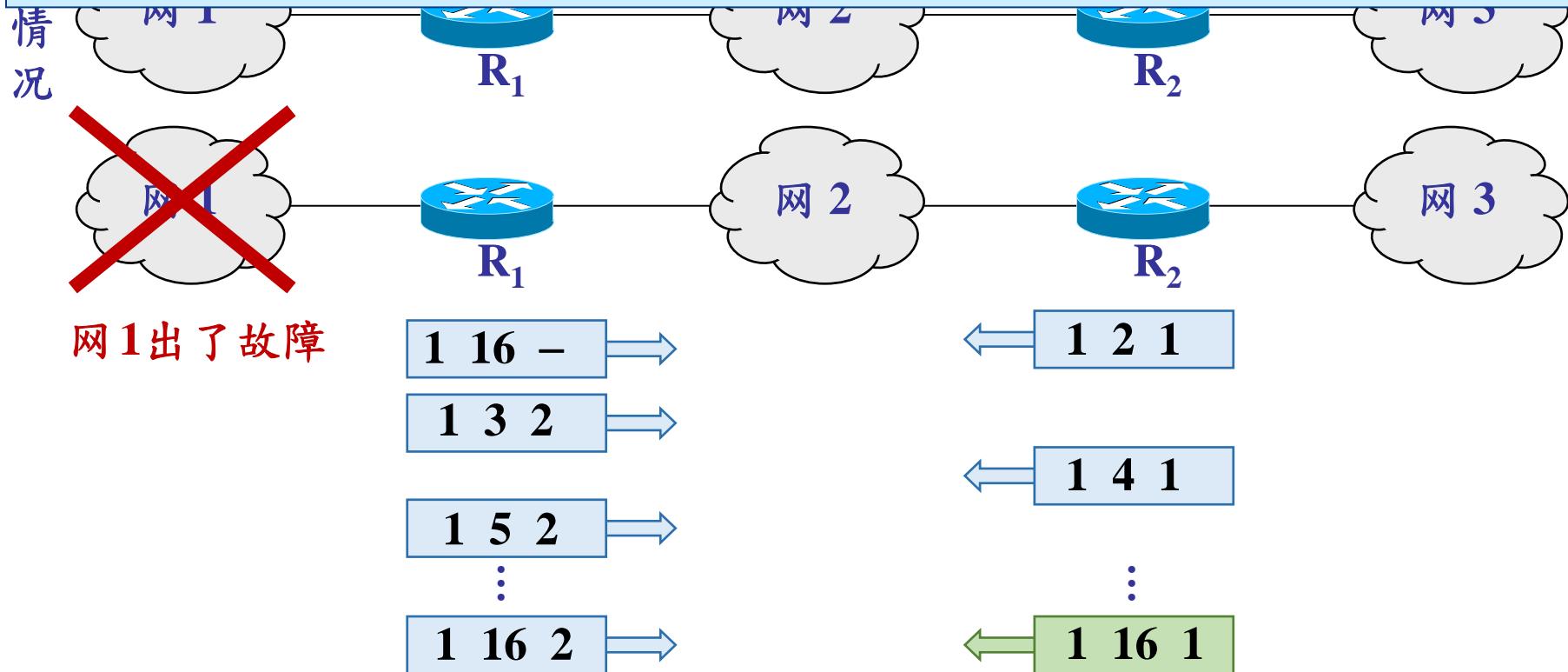
正常情况



R_2 以后又更新自己的路由表为 “ $1, 4, R_1$ ”，表明 “我到网 1 距离是 4，下一跳经过 R_1 ”。

RIP 协议的演示

这就是好消息传播得快，而坏消息传播得慢。网络出故障的传播时间往往需要较长的时间(例如数分钟)。这是 RIP 的一个主要缺点。



这样不断更新下去，直到 R_1 和 R_2 到网 1 的距离都增大到 16 时， R_1 和 R_2 才知道网 1 是不可达的。

内容纲要

1

路由协议的基本概念

2

自治系统

3

路由信息协议

4

开放最短路径优先协议

5

边界网关协议



内部网关协议 OSPF

- **开放最短路径优先 (Open Shortest Path First, OSPF)**
 - 开放：不受某一家厂商控制，而是公开发表的。
 - 最短路径优先：因为使用了 Dijkstra 提出的最短路径算法
 - 协议名不表示其他的路由选择协议不是“最短路径优先”。
 - 采用分布式的链路状态协议。



三个要点

- 向本自治系统中所有路由器发送信息，这里使用的方法是洪泛法。
- 发送的信息就是与本路由器相邻的所有路由器的链路状态，但这只是路由器所知道的部分信息。
 - “链路状态”就是说明本路由器都和哪些路由器相邻，以及该链路的“度量”(metric)。
- 只有当链路状态发生变化时，路由器才用洪泛法向所有路由器发送此信息。

链路状态数据库 link-state database

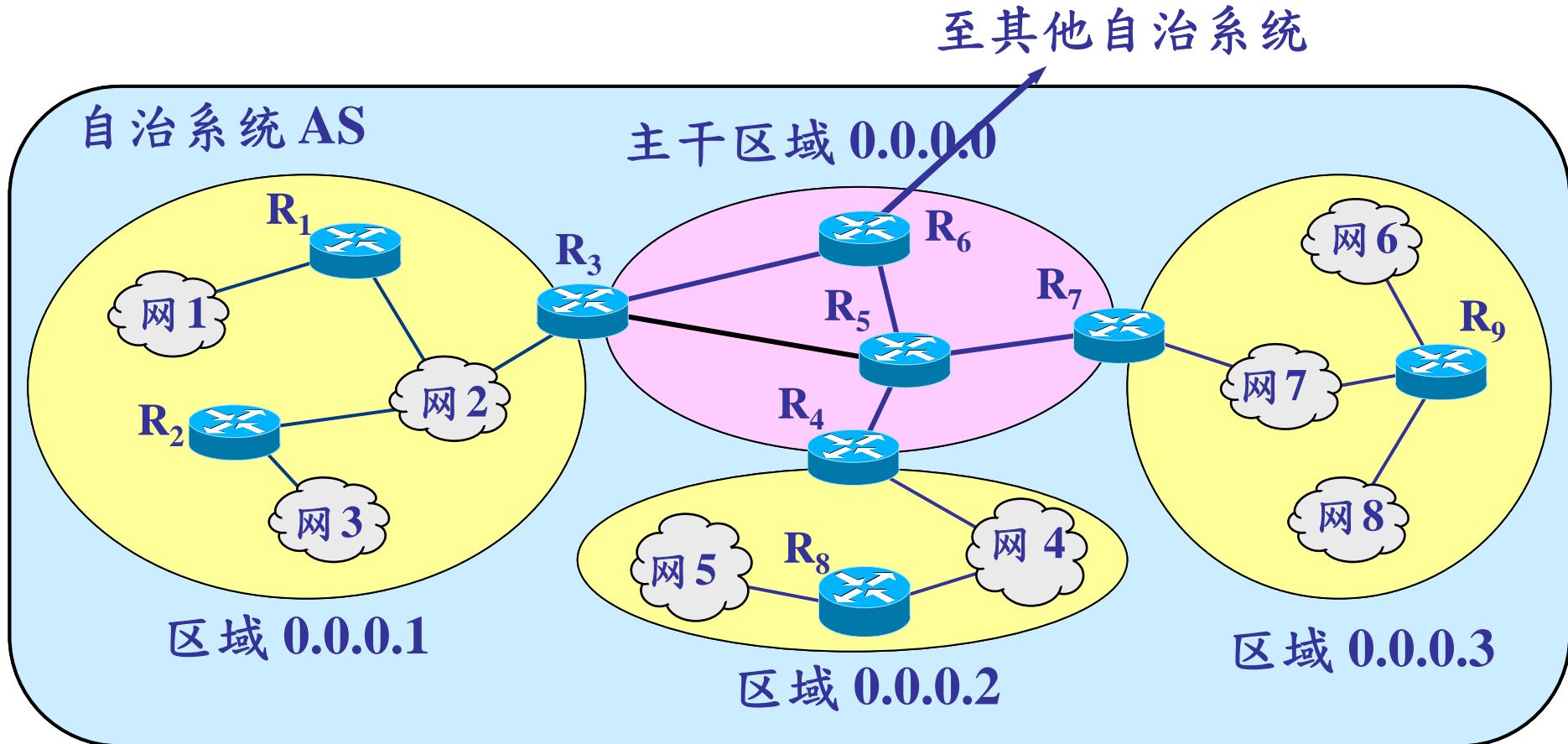
- 由于各路由器之间频繁地交换链路状态信息，因此所有的路由器最终都能建立一个链路状态数据库。
- 这个数据库实际上就是全网的拓扑结构图，它在全网范围内是一致的（这称为链路状态数据库的同步）。
- OSPF 的链路状态数据库能较快地进行更新，使各个路由器能及时更新其路由表。OSPF 的更新过程收敛得快是其重要优点。



OSPF 的区域(area)

- 为了使 OSPF 能够用于规模很大的网络，OSPF 将一个自治系统再划分为若干个更小的范围，叫作区域。
- 每一个区域都有一个 32 位的区域标识符（用点分十进制表示）。
- 区域也不能太大，在一个区域内的路由器最好不超过 200 个。

OSPF 划分为两种不同的区域

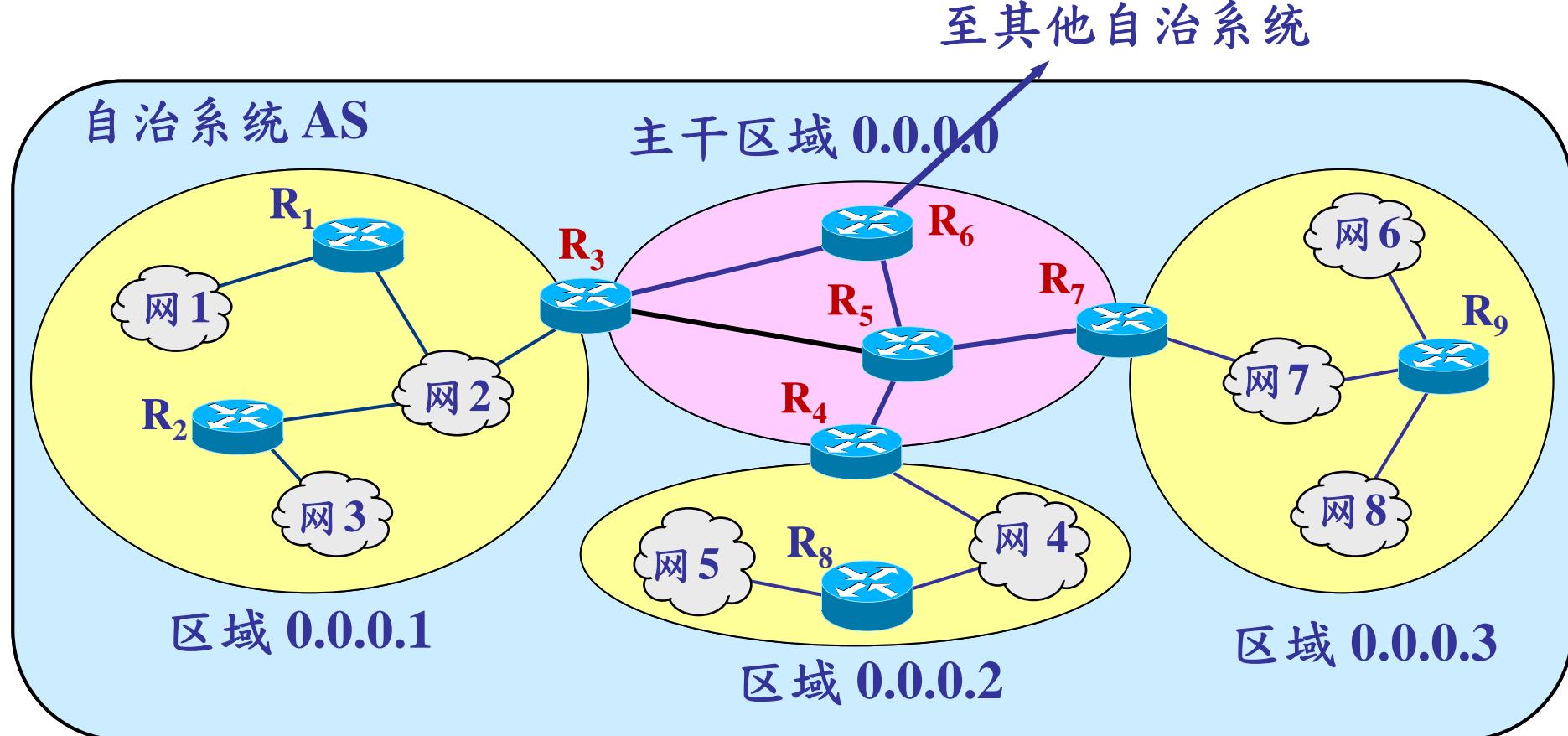


划分区域

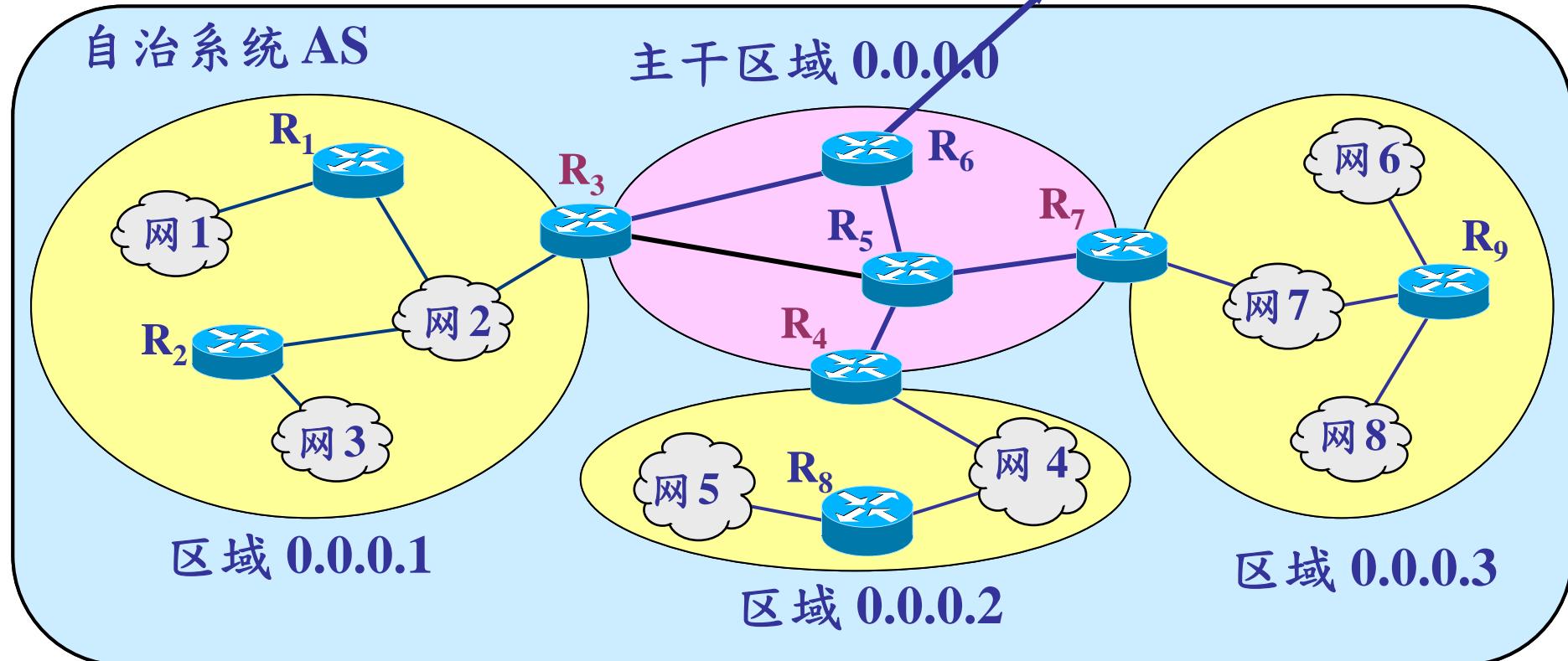
- 划分区域的好处就是将利用洪泛法交换链路状态信息的范围局限于每一个区域而不是整个的自治系统，这就减少了整个网络上的通信量。
- 在一个区域内部的路由器只知道本区域的完整网络拓扑，而不知道其他区域的网络拓扑的情况。
- OSPF 使用层次结构的区域划分。在上层的区域叫作主干区域(**backbone area**)。主干区域的标识符规定为 0.0.0.0。主干区域的作用是连通其他在下层的区域。



主干路由器



区域边界路由器



OSPF 直接用 IP 数据报传送

- OSPF 不用 UDP 而是直接用 IP 数据报传送。
- OSPF 构成的数据报很短。这样做可减少路由信息的通信量。
- 数据报很短的另一好处是可以不必将长的数据报分片传送。分片传送的数据报只要丢失一个，就无法组装成原来的数据报，而整个数据报就必须重传。



OSPF 的其他特点

- OSPF 对不同的链路可根据 IP 分组的不同服务类型 TOS 而设置成不同的代价。因此，OSPF 对于不同类型的业务可计算出不同的路由。
- 如果到同一目的网络有多条相同代价的路径，则可将通信量分配给这几条路径，称：多路径间的负载平衡。
- 所有在 OSPF 路由器之间交换的分组都具有鉴别功能。
- 支持可变长度的子网划分和无分类编址 CIDR。
- 每一个链路状态都带上一个 32 位的序号，序号越大状态就越新。



OSPF 分组

位 0 8 16 31

版本	类型	分组长度
路由器标识符		
区域标识符		
检验和		鉴别类型
鉴别		鉴别

24 字节

OSPF 分组首部

类型 1 至类型 5 的 OSPF 分组



OSPF 的五种分组类型

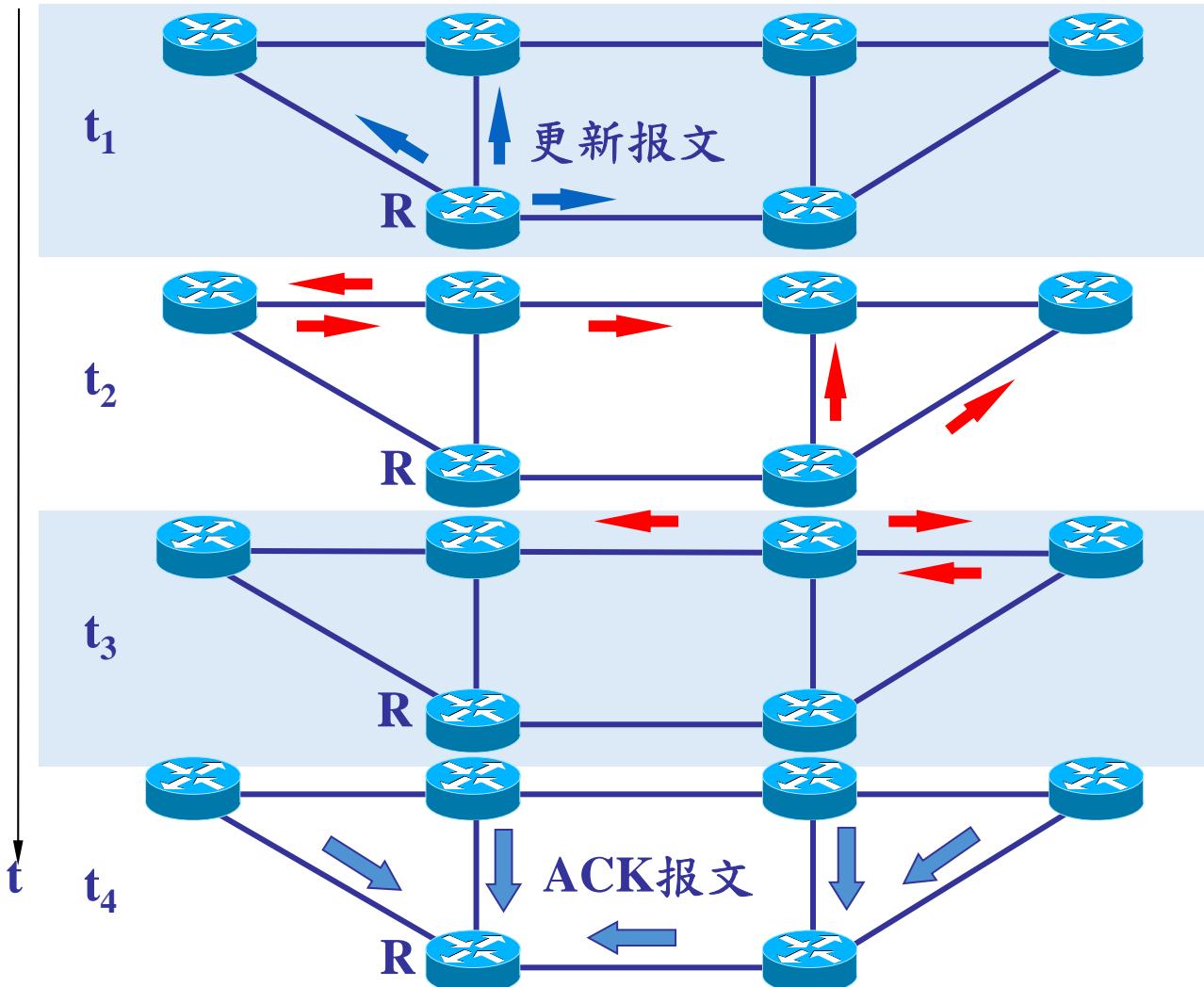
- 问候(Hello)分组
- 数据库描述(Database Description)分组
- 链路状态请求(Link State Request)分组
- 链路状态更新(Link State Update)分组
 - 用洪泛法对全网更新链路状态
- 链路状态确认(Link State Acknowledgment) 分组



OSPF的基本操作



OSPF 使用的是可靠的洪泛法



OSPF 的其他特点

- OSPF 还规定每隔一段时间，如 30 分钟，要刷新一次数据库中的链路状态。
- 由于一个路由器的链路状态只涉及到与相邻路由器的连通状态，因而与整个互联网的规模并无直接关系。因此当互联网规模很大时，OSPF 协议要比距离向量协议 RIP 好得多。
- OSPF 没有“坏消息传播得慢”的问题，据统计，其响应网络变化的时间小于 100 ms。



指定的路由器 (designated router)

- 多点接入的局域网采用了指定的路由器的方法，使广播的信息量大大减少。
- 指定的路由器代表该局域网上所有的链路向连接到该网络上的各路由器发送状态信息。



内容纲要

1

路由协议的基本概念

2

自治系统

3

路由信息协议

4

开放最短路径优先协议

5

边界网关协议



外部网关协议 BGP

- BGP 是不同自治系统的路由器之间交换路由信息的协议。
- BGP 较新版本是 2006 年 1 月发表的 BGP-4（BGP 第 4 个版本），即 RFC 4271 ~ 4278。
- 可以将 BGP-4 简写为 BGP。



BGP 使用的环境却不同

- 因特网规模太大，使自治系统之间路由选择非常困难。
- 自治系统之间路由选择，寻找最佳路由是很不现实的。
 - 当一条路径通过几个不同 AS 时，要想对这样的路径计算出有意义的代价是不太可能的。
 - 比较合理的做法是在 AS 之间交换“可达性”信息。
- 因此，边界网关协议 BGP 只能是力求寻找一条能够到达目的网络且比较好的路由（不能兜圈子），而并非要寻找一条最佳路由。



BGP 发言人 (BGP speaker)

- 每一个自治系统的管理员要选择至少一个路由器作为该自治系统的“BGP 发言人”。
- 一般说来，两个 BGP 发言人都是通过一个共享网络连接在一起的，而 BGP 发言人往往就是 BGP 边界路由器，但也可以不是 BGP 边界路由器。

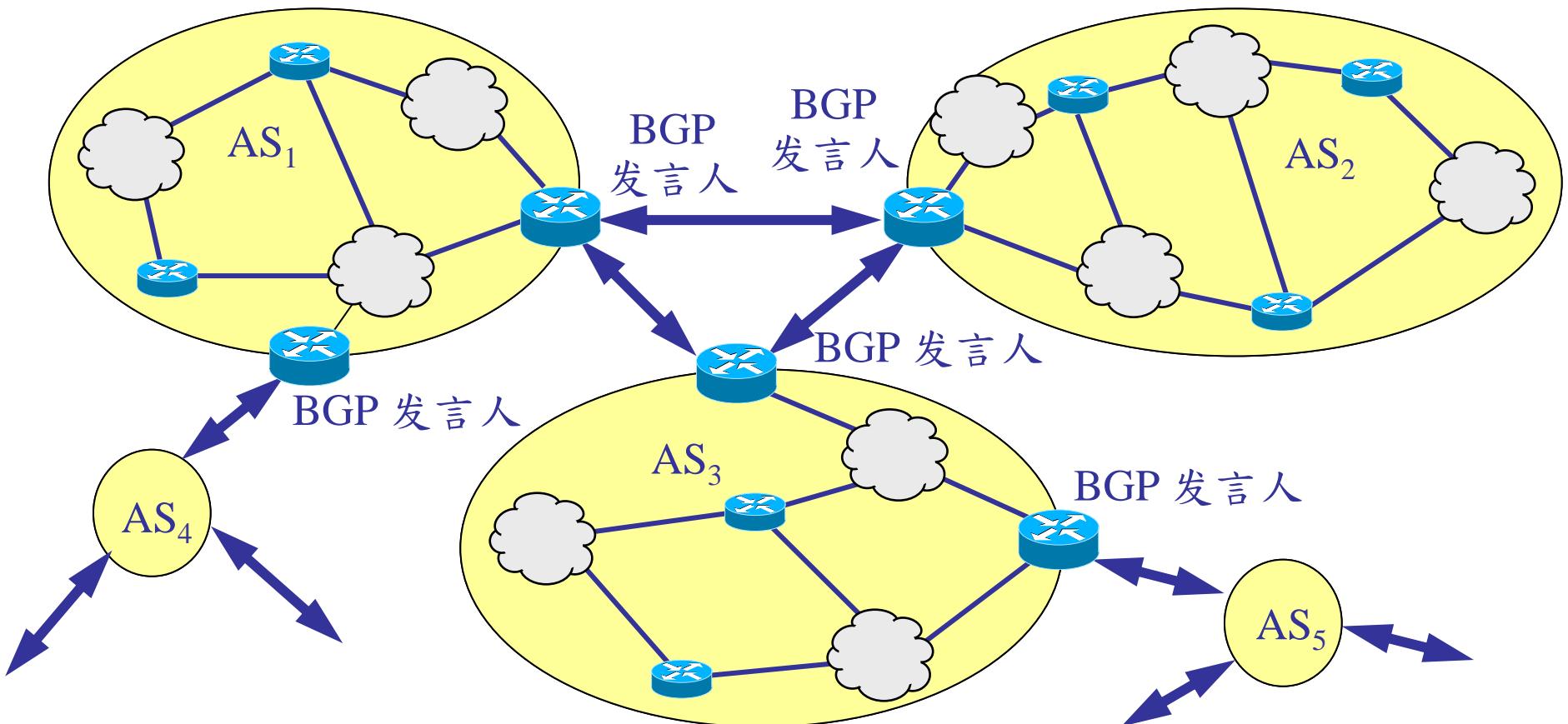


BGP 交换路由信息

- 一个 BGP 发言人与其他自治系统中的 BGP 发言人要交换路由信息，就要先建立 TCP 连接，然后在此连接上交换 BGP 报文以建立 BGP 会话(session)，利用 BGP 会话交换路由信息。
- 使用 TCP 连接能提供可靠的服务，也简化了路由选择协议。
- 使用 TCP 连接交换路由信息的两个 BGP 发言人，彼此成为对方的邻站或对等站。

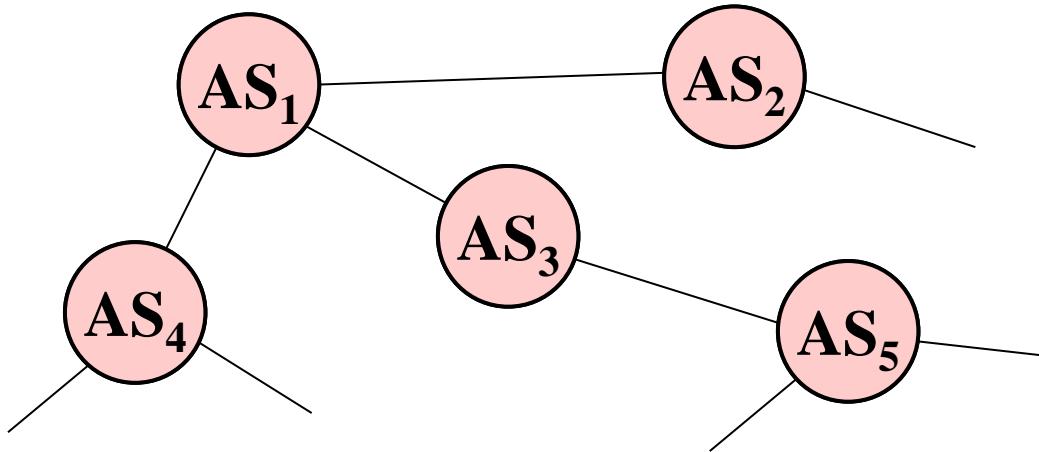


BGP 发言人和自治系统 AS 的关系



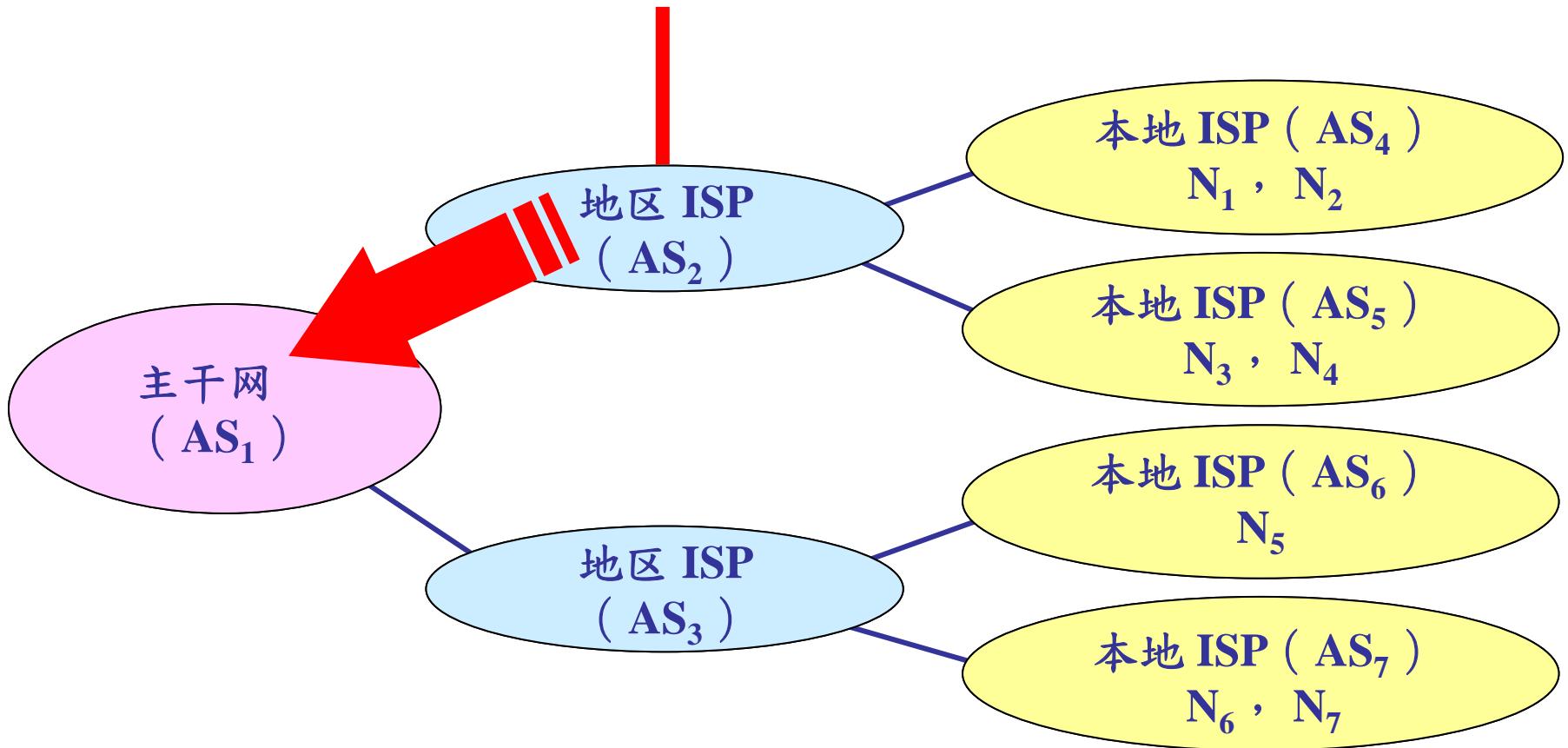
AS 的连通图举例

- BGP 所交换的网络可达性的信息就是要到达某个网络所要经过的一系列 AS。
- 当 BGP 发言人互相交换了网络可达性的信息后，各 BGP 发言人就根据所采用的策略从收到的路由信息中找出到达各 AS 的较好路由。



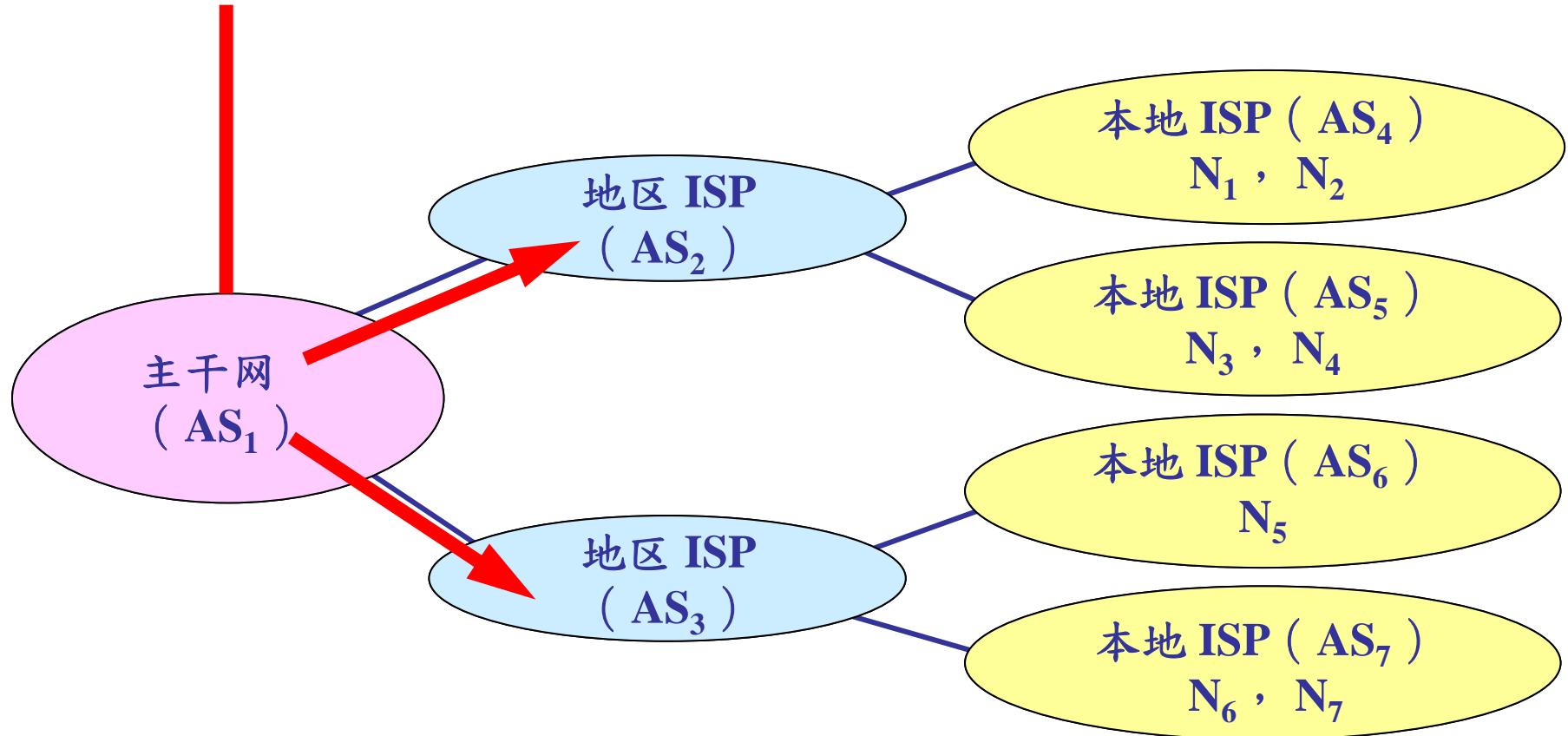
BGP 发言人交换路径向量

自治系统 AS_2 的 BGP 发言人通知主干网的 BGP 发言人：“要到达网络 N_1, N_2, N_3 和 N_4 可经过 AS_2 。”



BGP 发言人交换路径向量

主干网还可发出通知：“要到达网络 N_5, N_6 和 N_7 可沿路径（ AS_1, AS_3 ）。”



BGP 协议的特点

- BGP 协议交换路由信息的结点数量级是自治系统的量级，这要比这些自治系统中的网络数少很多。
- 每一个自治系统中 BGP 发言人（或边界路由器）的数目是很少的。这样就使得自治系统之间的路由选择不致过分复杂。



BGP 协议的特点

- BGP 支持 CIDR，因此 BGP 的路由表也就应当包括目的网络前缀、下一跳路由器，以及到达该目的网络所要经过的各个自治系统序列。
- 在BGP 刚刚运行时，BGP 的邻站是交换整个的 BGP 路由表。但以后只需要在发生变化时更新有变化的部分。这样做对节省网络带宽和减少路由器的处理开销方面都有好处。

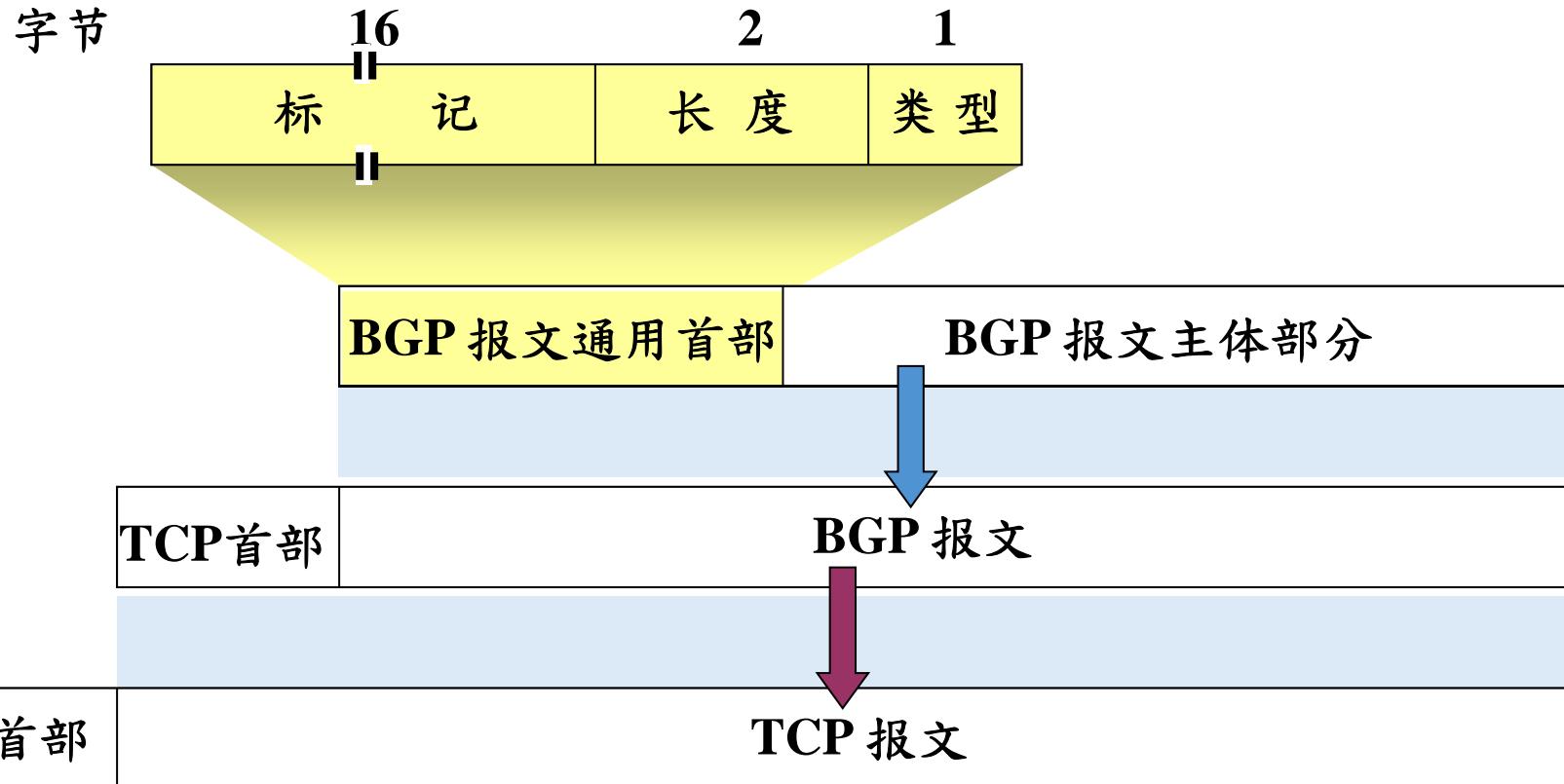


BGP-4 共使用四种报文

- 打开(open)：与相邻的另一个BGP发言人建立关系。
- 更新(update)：发送某一路由的信息，以及列出要撤消的多条路由。
- 保活(keepalive)报文：确认打开报文和周期性地证实邻站关系。
- 通知(notification)报文：发送检测到的差错。
- RFC 2918 中增加了 Route-refresh 报文：请求对等端重新通告。



BGP 报文具有通用的首部

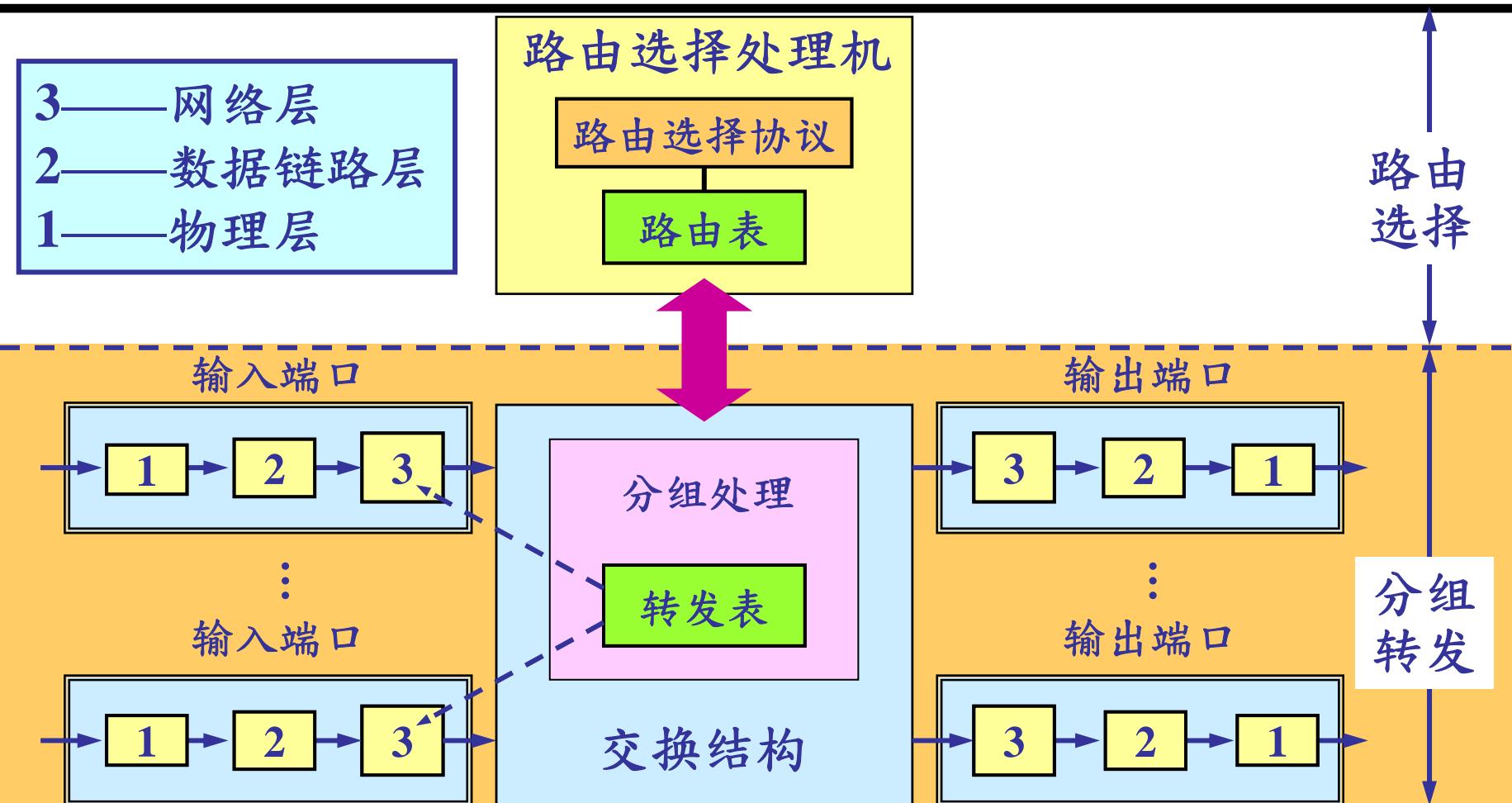


路由器在网际互连中的作用

- 路由器是一种具有多个输入端口和多个输出端口的专用计算机，其任务是转发分组。也就是说，将路由器某个输入端口收到的分组，按照分组要去的目的地（即目的网络），把该分组从路由器的某个合适的输出端口转发给下一跳路由器。
- 下一跳路由器也按照这种方法处理分组，直到该分组到达终点为止。



典型的路由器的结构



“转发”和“路由选择”的区别

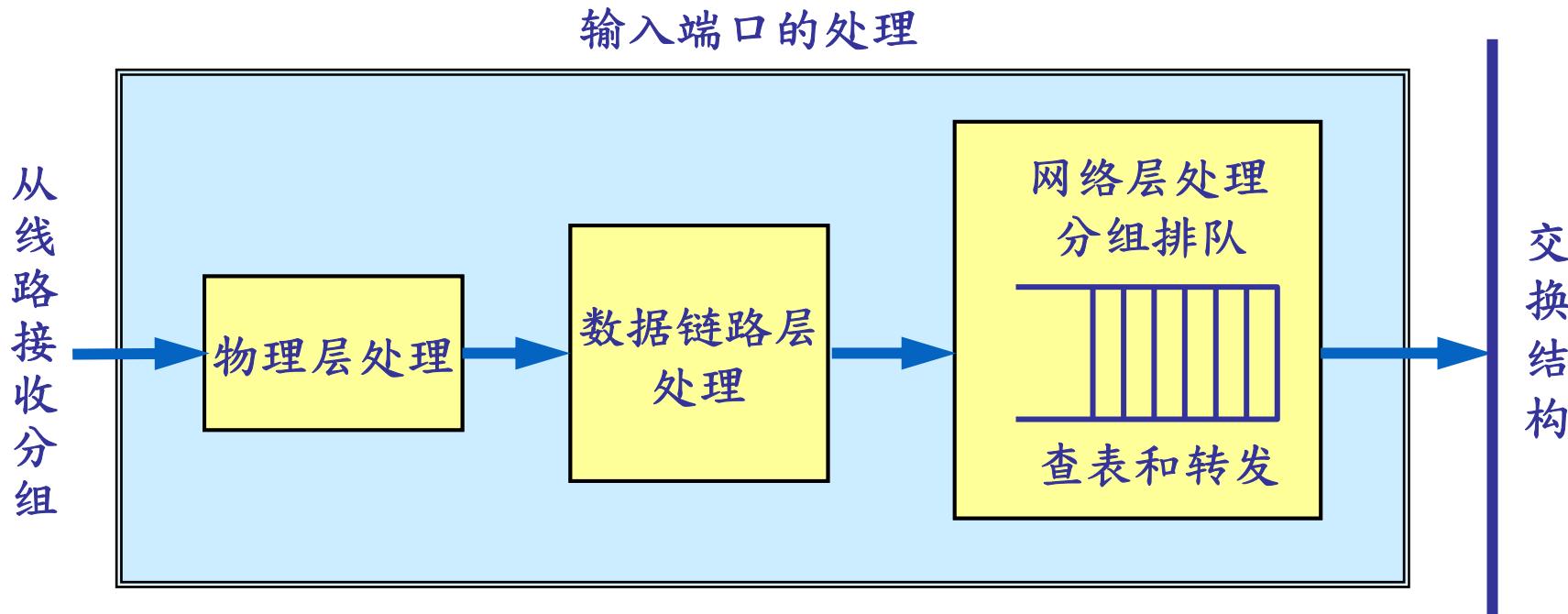
- 转发 (forwarding) : 从路由表得出的
 - 路由器根据转发表将 IP 数据报从合适的端口转发出去。
- 路由选择 (routing) : 根据路由选择算法得出
 - 则是按照分布式算法，根据从各相邻路由器得到的关于网络拓扑的变化情况，动态地改变所选择的路由。
- 在讨论路由选择的原理时，往往不去区分转发表和路由表的区别



输入端口对分组的处理

- 输入端口对线路上收到分组的处理

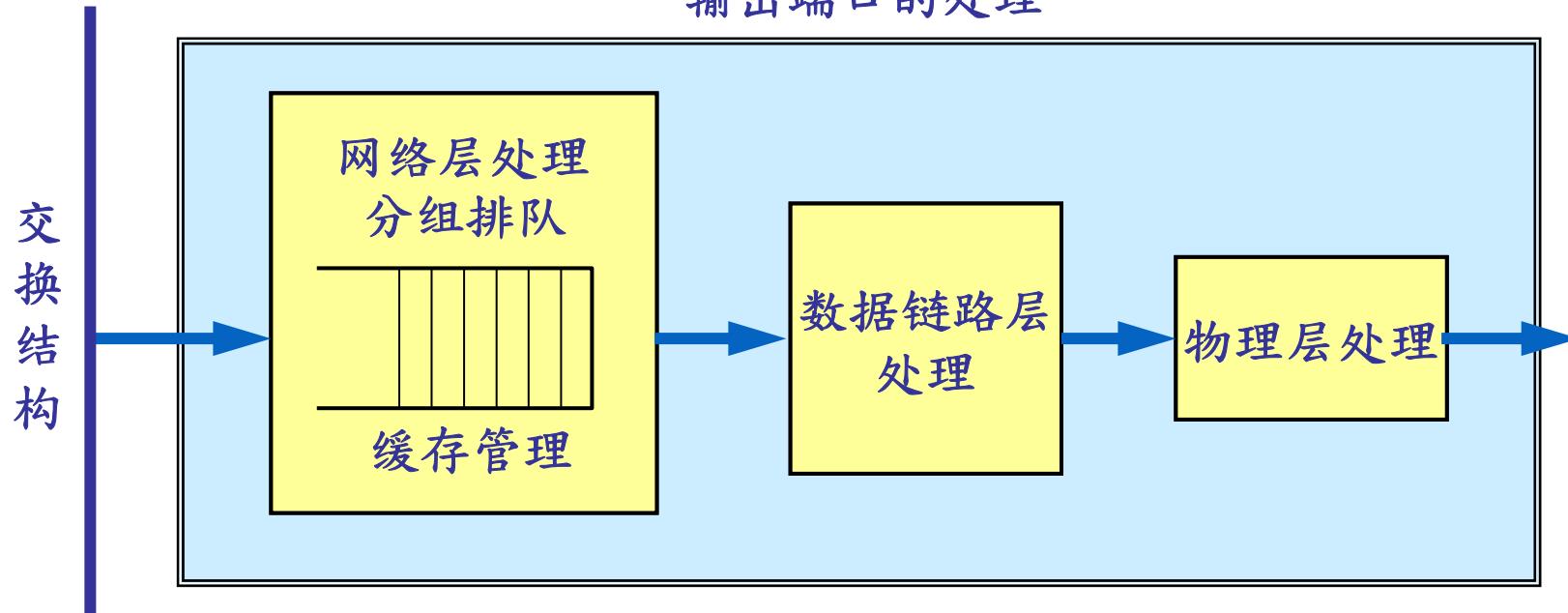
- 数据链路层剥去帧头部和尾部后，将分组送到网络层的队列中排队等待处理。这会产生一定的时延。



输出端口将分组发送到线路

- 输出端口将交换结构传送来的分组发送到线路
 - 交换结构传送过来的分组先进行缓存。数据链路层处理模块将分组加上链路层的首部和尾部，交给物理层后发送到外部线路。

输出端口的处理

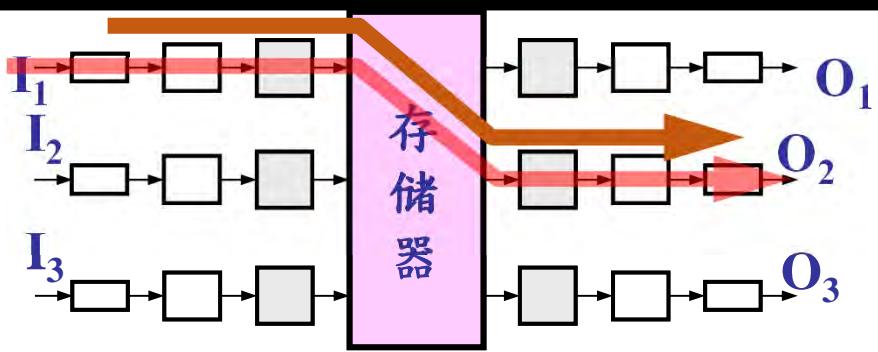


分组丢弃

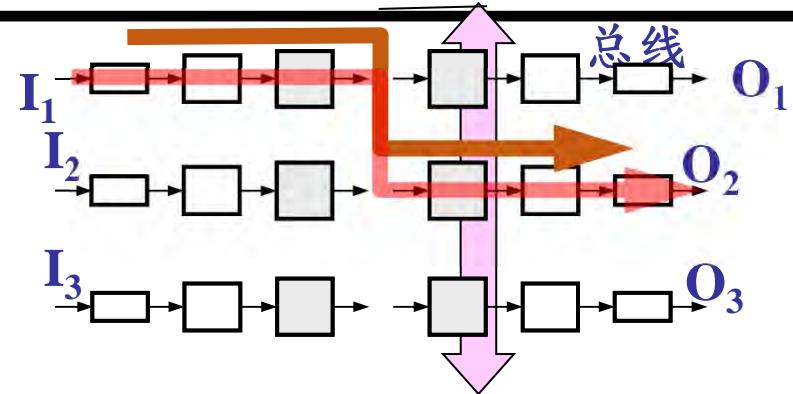
- 若路由器处理分组的速率赶不上分组进入队列的速率，则队列的存储空间最终必定减少到零，这就使后面再进入队列的分组由于没有存储空间而只能被丢弃。
- 路由器中的输入或输出队列产生溢出是造成分组丢失的重要原因。



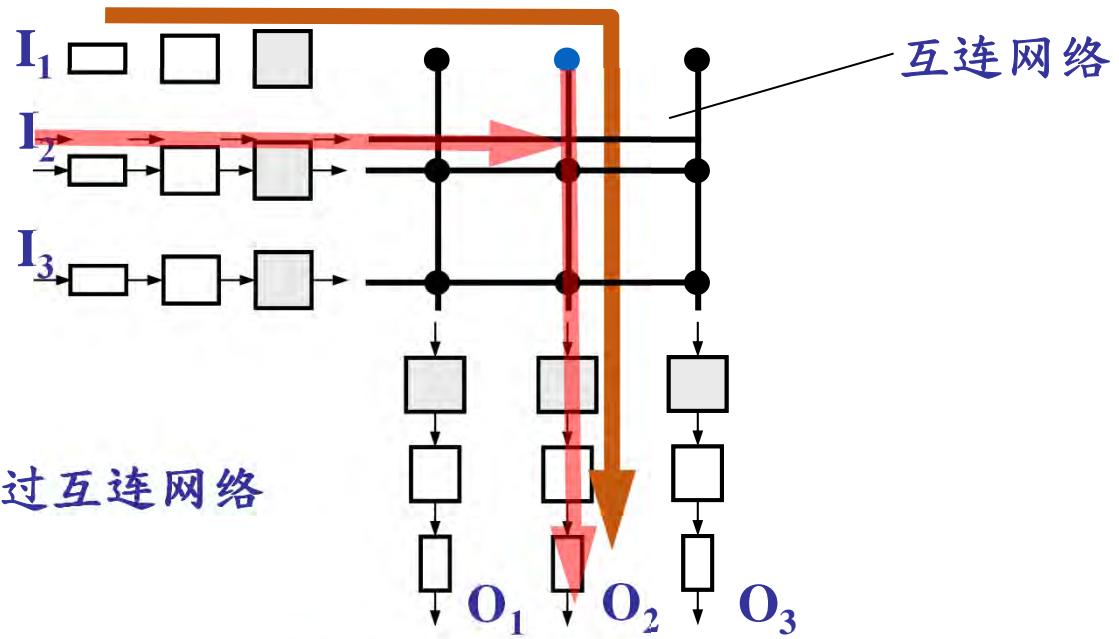
交换结构



(a) 通过存储器



(b) 通过总线



(c) 通过互连网络

计算机网络

Computer Network

13

谢谢观看

理论教程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

14

客户服务器模式 和套接字API

理论课程

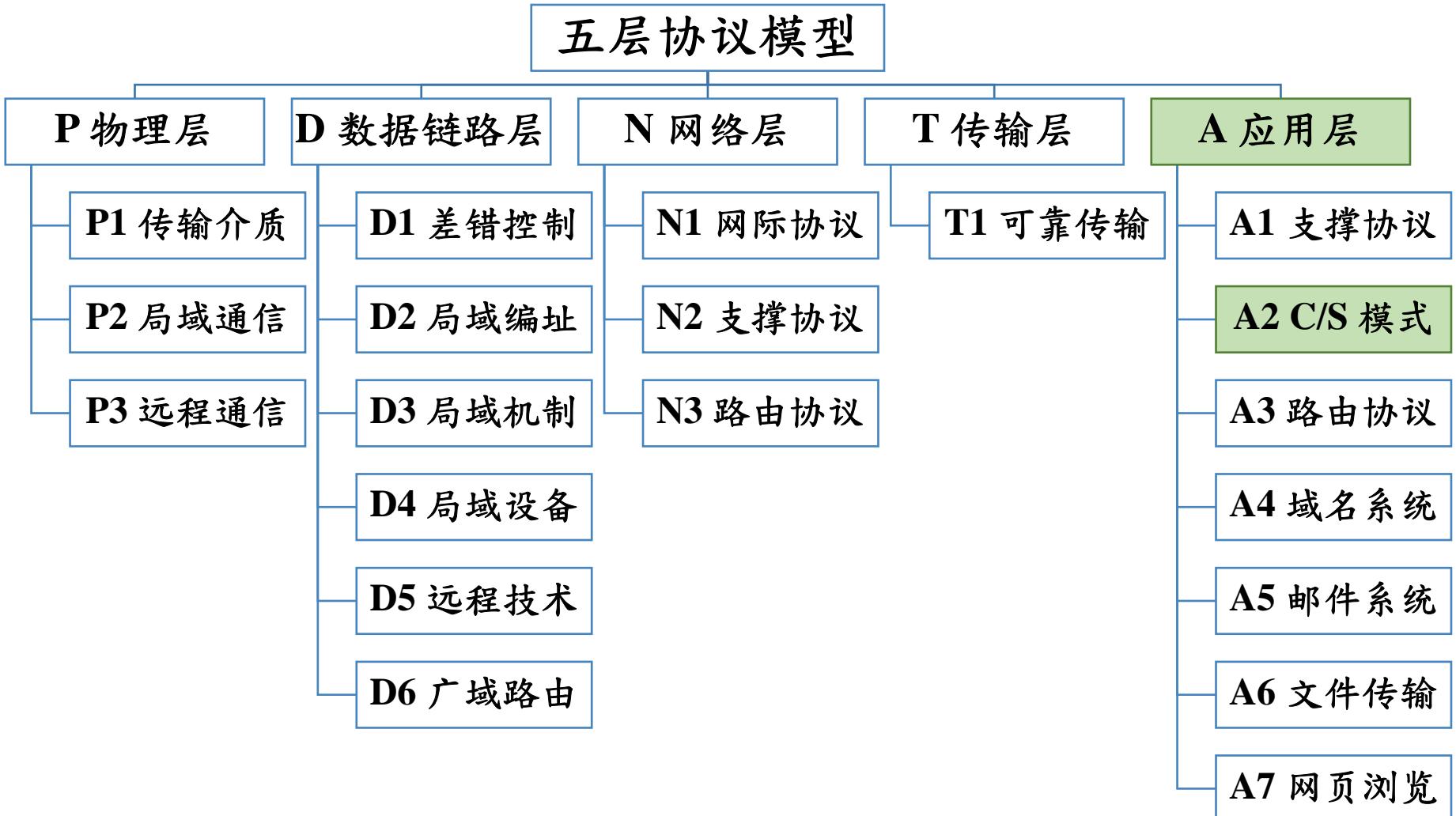


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 客户端—服务器端（C/S）交互模式工作原理
- 并发的概念
- Socket结构、半相关与全相关
- 服务器与用户、服务器端与客户端，二者区别
- Socket API主要函数（Windows或Linux系统下）
- 流模式的客户端、服务器端Socket API调用流程
- 报文模式的客户端、服务器端Socket API调用流程



对应课本章节

- PART I Introduction And Internet Applications
 - Chapter 3 Internet Applications And Network Programming

内容纲要

1

客户端和服务器端

2

网间进程通信

3

Socket API

4

客户服务器端程序实例

5

小结



客户和服务器

- 应用层的许多协议都是基于客户/服务器方式。
 - 将一项任务划分为服务的请求者和服务的提供者两部分。
- 客户和服务器都是指通信中所涉及的两个应用进程。
 - 客户（Client）和服务（Server）默认指客户端和服务器端。
 - 客户端是访问服务器提供的服务的程序。
 - 服务器是为客户程序或设备提供功能的计算机程序或设备。
- 当传入消息与应用程序指定的端口相匹配时，协议软件将该消息传递给应用程序。



客户和服务器

- 客户端或服务器端是指一个程序。
 - 客户和服务器默认指客户端和服务器端。
- 客户软件和服务器软件是指一个软件。
 - 软件由多个程序和数据共同完成功能。
 - 这些程序有客户端程序，也有服务器端程序。
- 客户设备和服务器设备是指一个硬件。
 - 硬件上可以运行多个软件。
 - 这些软件有客户软件，也有服务器软件。



客户端和服务器端工作模式

- 客户端

- 主动打开：客户端使用远程主机的地址和在该机上特定服务器程序的著名端口地址通道打开通信。
- 主动结束：虽然请求的响应可重复几次，但整个过程是有限的，最终结束。在那一刻，客户端关闭通信通道。

- 服务器端

- 被动打开：服务器像被动打开的大门，响应客户端的请求。
- 响应请求：当请求到达时，它会作出迭代的或同步的响应。



客户端软件（ Client software ）

- 客户端软件（ Client software ）

- 任意的应用程序，需要远程访问时成为一个临时的客户端，但也可以在本地执行其他计算程序。
- 直接由用户调用，并只执行一次会话。
- 主动与服务器接触。
- 可以在需要的时候访问多个服务，但是在一个时间中，主动联系一个远程服务器。
- 本地用户的主机上运行，不需特殊硬件或复杂的操作系统。



服务器端软件（ Server software ）

- 服务器端软件（ Server software ）

- 一个专用于提供一个服务的专用程序，同时可以处理多个远程客户端。
- 系统启动时自动调用，并通过多个会话连续执行。
- 在共享计算机上运行（不在用户的个人计算机上）。
- 被动地等待从任意远程客户的联系
- 接受来自任意客户的联系，但提供单一服务。
- 需要强大的硬件和复杂的操作系统。



客户和服务器的区别

- 轻重区别

- 服务器是一个系统（硬件和软件），它响应计算机网络上的请求以提供或帮助提供网络服务。
- 客户端是一个计算机硬件或软件访问服务器提供的服务的一块。
- 客户-服务器（C/S）模式是一种分布式的应用结构计算，在任务或工作负载的资源或服务的提供者（服务器）和服务请求者（客户）之间。



一台计算机上的多种服务

- **TCP / UDP复用与解复用**

- 一个够强大的计算机系统可同时运行多个客户端和服务器。
- 计算机只需要单一的物理连接连到互联网。
- TCP程序通过TCP报头中由源主机指出的端口地址，了解到发送主机希望目标主机哪一应用程序接收数据报。
- 相应地，UDP类似。

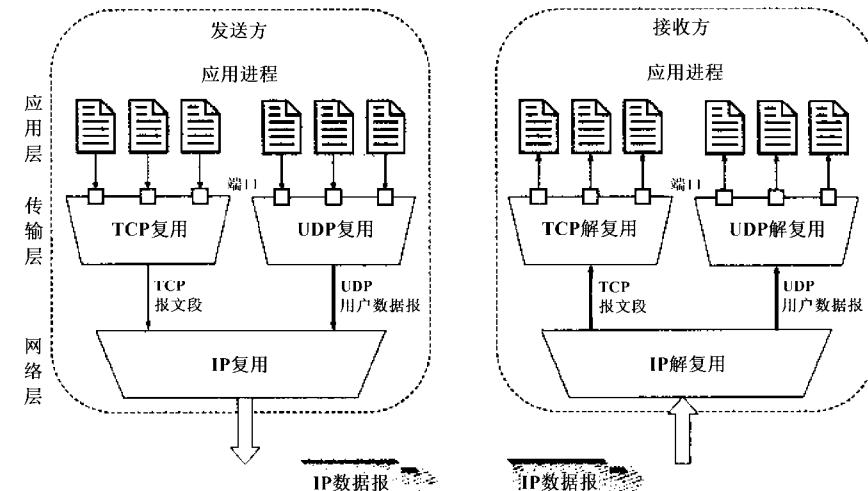


图 7-7 端口在进程之间的通信中所起的作用



一台计算机上的多种服务

- 为了在通信时不致发生混乱，就必须把端口号和主机的IP地址结合在一起使用。
- TCP连接由它的两个端点来标识
 - 每个端点由IP地址和端口号决定的，称为Socket（插口）。
- UDP无连接但也需要Socket
 - 虽然在相互通信的两个进程之间没有一条虚连接，但发送端UDP一定有一个发送端口而在接收端UDP也一定有一个接收端口。



UDP报文队列

- UDP与应用层之间的端口都是用报文队列来实现。
- 操作系统为该进程创建：入队列和出队列

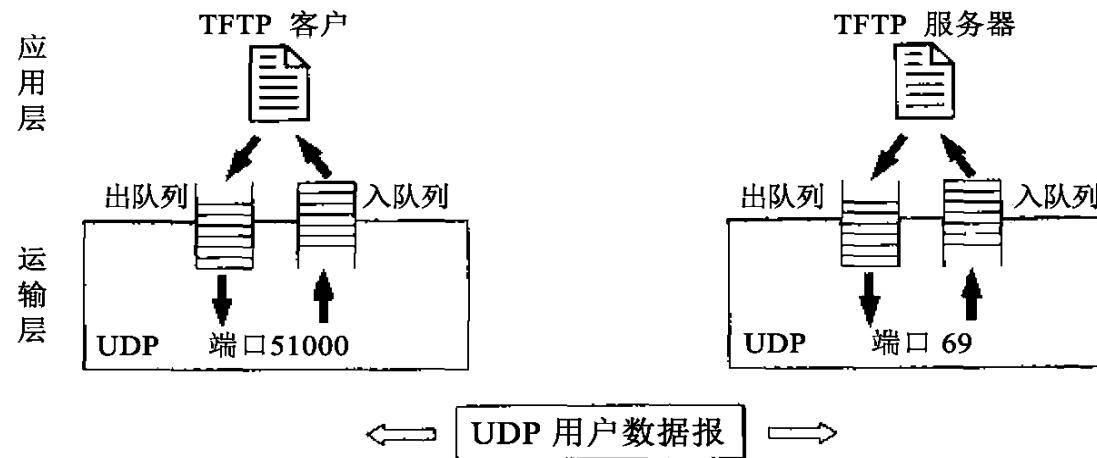
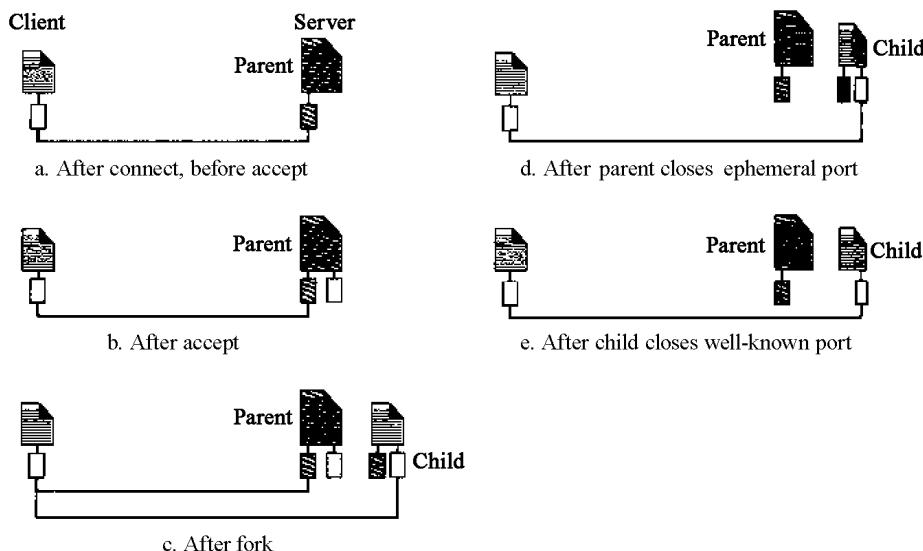


图7-9 UDP中的队列



并发 (Concurrency)

- 并发是客户端-服务器交互模式的基础。
 - 允许多个应用在同一时间执行的计算机系统称为支持并发。
 - 一个拥有多个线程的程序称为并发程序。
 - 因为并发服务器同时为多个客户提供服务，而不需要每个客户机等待上一个客户端来完成。



内容纲要

1

客户端和服务器端

2

网间进程通信

3

Socket API

4

客户服务器端程序实例

5

小结



系统调用和应用编程接口

- 大多数操作系统使用系统调用(system call)的机制在应用程序和操作系统之间传递控制权。
- 对程序员来说，每一个系统调用和一般程序设计中的函数调用非常相似，只是系统调用是将控制权传递给了操作系统。
- Application Program Interface (API)



单机进程通信

- 进程通信的概念最初来源于单机系统。
- 由于每个进程都在自己的地址范围内运行，为保证两个相互通信的进程之间既互不干扰又协调一致工作，操作系统为进程通信提供了相应设施：
 - 如UNIX BSD中的管道（ pipe ）、命名管道（ named pipe ）和软中断信号（ signal ），UNIX system V的消息（ message ）、共享存储区（ shared memory ）和信号量（ semaphore ）等，但都仅限于用在本机进程之间通信。

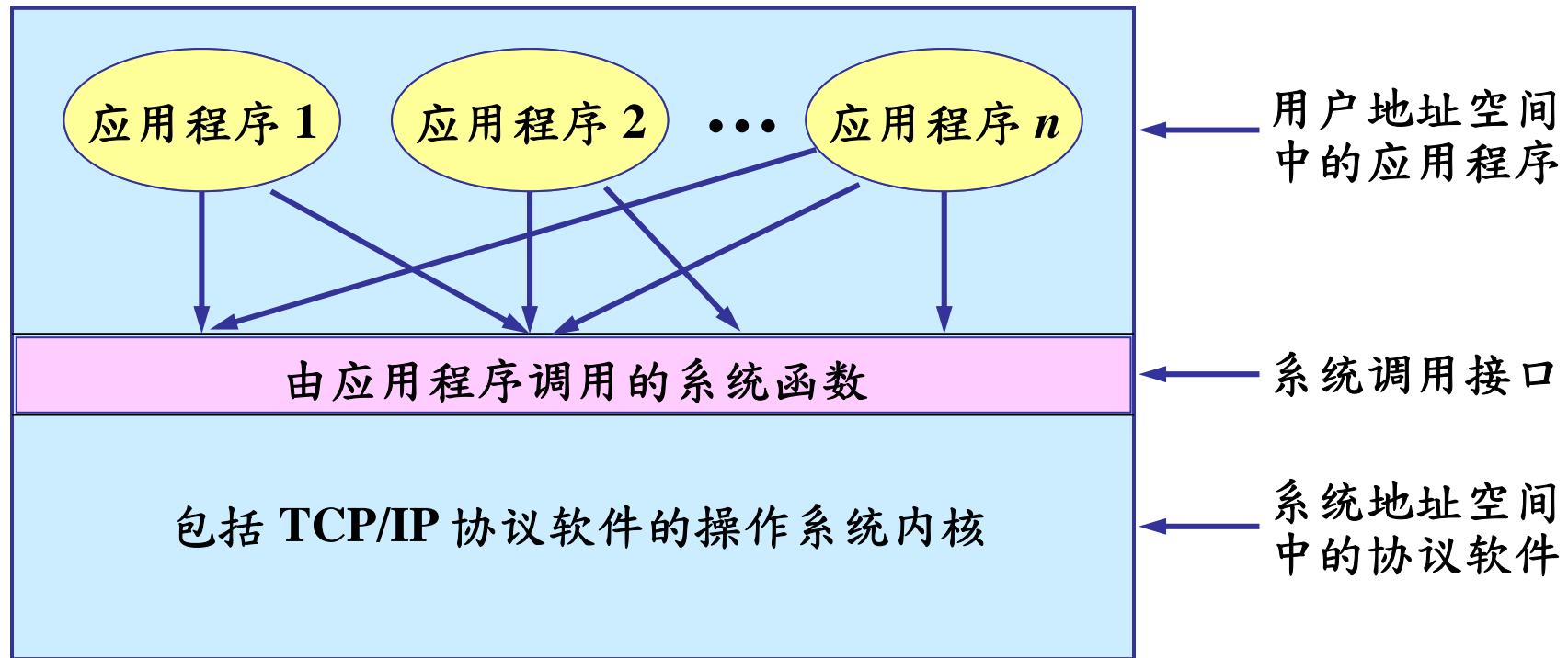


网间进程通信

- 网间进程通信要解决不同主机进程间的相互通信问题。
 - 可以把同机进程通信看作是其中的特例。
 - 首先要解决的是网间进程标识问题。同一主机上，不同进程可用进程号（process ID）唯一标识。但在网络环境下，各主机独立分配的进程号不能唯一标识该进程。
 - 其次，操作系统支持的网络协议众多，不同协议的工作方式不同，地址格式也不同。因此，网间进程通信还要解决多重协议的识别问题。



多个应用进程使用系统调用的机制



协议端口

- TCP/IP协议提出了协议端口（ protocol port，简称端口 ）的概念，用于标识通信的进程。
 - 端口是一种抽象的软件结构（包括数据结构和I/O缓冲区）。
 - 网络通信的最终地址就不仅是主机地址，还包括端口号。
 - 每个端口有端口号（ Port Number ），区别不同端口。
 - TCP有一个255号端口，UDP也有255号端口，常用端口号要记住
 - TCP/IP协议实现中，端口间操作类似一般I/O操作，进程获取端口，相当于获取本地I/O文件，可用一般读写原语访问。



套接字（Socket）

- 当应用进程需要使用网络进行通信时就发出系统调用，请求操作系统为其创建“套接字”，以便把网络通信所需要的系统资源分配给该应用进程。
- 操作系统为这些资源的总和用一个叫做套接字描述符的号码来表示，并把此号码返回给应用进程。应用进程所进行的网络操作都必须使用这个号码。
- 通信完毕后，应用进程通过一个关闭套接字的系统调用通知操作系统回收与该“号码”相关的所有资源。



主要的Socket API函数名

Name	Used By	Meaning
accept	server	Accept an incoming connection
bind	server	Specify IP address and protocol port
close	either	Terminate communication
connect	client	Connect to a remote application
getpeername	server	Obtain client's IP address
getsockopt	server	Obtain current options for a socket
listen	server	Prepare socket for use by a server
recv	either	Receive incoming data or message
recvmsg	either	Receive data (message paradigm)
recvfrom	either	Receive a message and sender's addr.
send (write)	either	Send outgoing data or message
sendmsg	either	Send an outgoing message
sendto	either	Send a message (variant of sendmsg)
setsockopt	either	Change socket options
shutdown	either	Terminate a connection
socket	either	Create a socket for use by above

连接建立阶段

- 当套接字被创建后，它的端口号和 IP 地址都是空的，因此应用进程要调用 **bind**（绑定）来指明套接字的本地地址。在服务器端调用 **bind** 时就是把熟知端口号和本地IP地址填写到已创建的套接字中。这就叫做把本地地址绑定到套接字。
- 服务器在调用 **bind** 后，还必须调用 **listen**（监听）把套接字设置为被动方式，以便随时接受客户的服务请求。



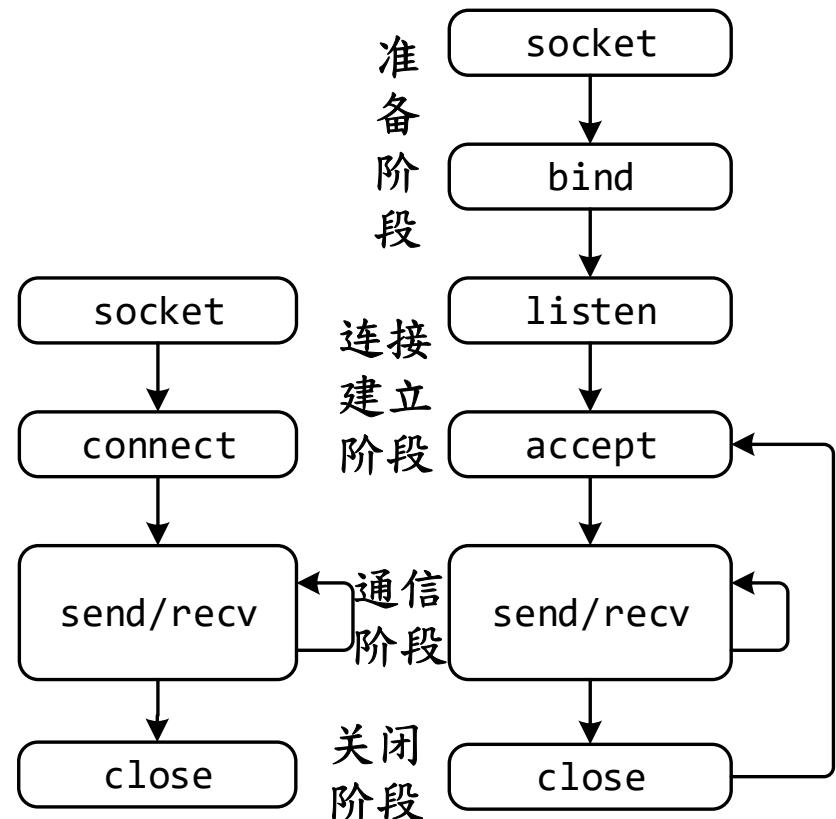
连接建立阶段

- UDP服务器由于只提供无连接服务，不使用 listen 系统调用。
- 服务器紧接着就调用 accept（接受），以便把远地客户进程发来的连接请求提取出来。系统调用 accept 的一个变量就是要指明从哪一个套接字发起的连接。

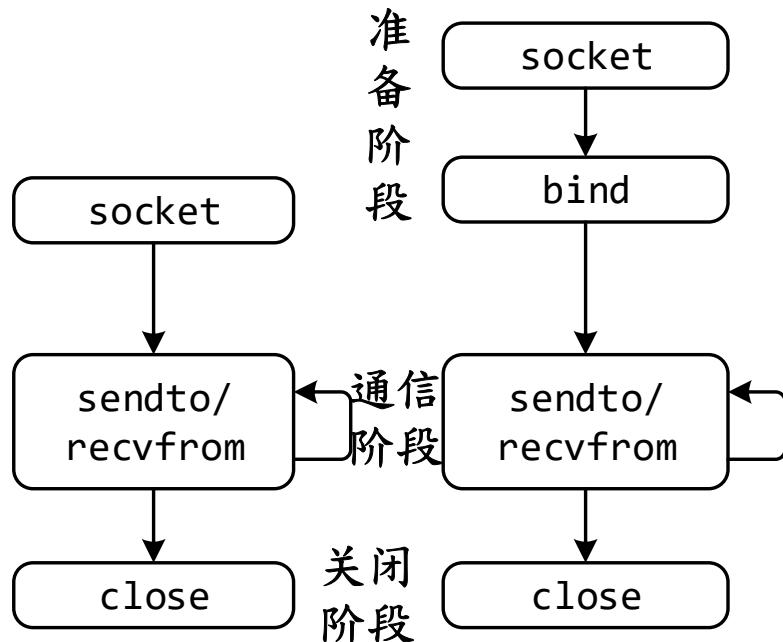


C/S中的套接字

• TCP面向连接



UDP无连接



内容纲要

1

客户端和服务器端

2

网间进程通信

3

Socket API

4

客户服务器端程序实例

5

小结



半相关和全相关

- 半相关（ half-association ）
 - 网络中用三元组在全局唯一标志一个进程：
 - （协议，本地地址，本地端口号）指定连接的每半部分
 - 全相关（ association ）
 - 一个完整的网间进程通信需要由两个进程组成，并且只能使用同一种高层协议。
 - 一个完整的网间通信需要一个五元组来标识
- （协议，本地地址，本地端口号，远程地址，远程端口号）。



Socket

- 本地套接字地址

- 本地IP地址

- 本地端口号

- 远程套接字地址

- 远程IP地址

- 远程端口号

- 协议

- TCP, UDP, 原始(raw) IP



系统要求

- WinSock 2 Library

- 本节内容源自MSDN

Item	Value
Minimum supported client	Windows 2000 Professional [desktop apps only]
Minimum supported server	Windows 2000 Server [desktop apps only]
Header	Winsock2.h
Library	Ws2_32.lib
DLL	Ws2_32.dll

函数 socket

- 函数 socket 创建一个绑定到特定传输服务提供者的插口。

```
SOCKET WSAAPI socket(  
    _In_    int af,  
    _In_    int type,  
    _In_    int protocol  
) ;
```

af [in]

The address family specification.

type [in]

The type specification for the new socket.

protocol [in]

The protocol to be used. The possible options for the protocol parameter are specific to the address family and socket type specified.



函数 socket

- 参数 1：地址族

Af	Meaning
AF_UNSPEC 0	The address family is unspecified.
AF_INET 2	The Internet Protocol version 4 (IPv4) address family.
AF_IPX 6	The IPX/SPX address family. This address family is only supported if the NWLink IPX/SPX NetBIOS Compatible Transport protocol is installed.
AF_APPLETALK 16	The AppleTalk address family. This address family is only supported if the AppleTalk protocol is installed.
AF_NETBIOS 17	The NetBIOS address family. This address family is only supported if the Windows Sockets provider for NetBIOS is installed.
AF_INET6 23	The Internet Protocol version 6 (IPv6) address family.
AF_IRDA 26	The Infrared Data Association (IrDA) address family. This address family is only supported if the computer has an infrared port and driver installed.
AF_BTH 32	The Bluetooth address family.

函数 socket

• 参数 2：插口类型

Type	Meaning
SOCK_STREAM 1	A socket type that provides sequenced, reliable, two-way, connection-based byte streams with an OOB data transmission mechanism. This socket type uses the Transmission Control Protocol (TCP) for the Internet address family (AF_INET or AF_INET6).
SOCK_DGRAM 2	A socket type that supports datagrams, which are connectionless, unreliable buffers of a fixed (typically small) maximum length. This socket type uses the User Datagram Protocol (UDP) for the Internet address family (AF_INET or AF_INET6).
SOCK_RAW 3	A socket type that provides a raw socket that allows an application to manipulate the next upper-layer protocol header. To manipulate the IPv4 header, the IP_HDRINCL socket option must be set on the socket. To manipulate the IPv6 header, the IPV6_HDRINCL socket option must be set on the socket.
SOCK_RDM 4	A socket type that provides a reliable message datagram. An example of this type is the Pragmatic General Multicast (PGM) multicast protocol implementation in Windows, often referred to as reliable multicast programming. This type value is only supported if the Reliable Multicast Protocol is installed.
SOCK_SEQPACKET 5	A socket type that provides a pseudo-stream packet based on datagrams.

函数 socket

• 参数 3：协议

Protocol	Meaning
0	If a value of 0 is specified, the caller does not wish to specify a protocol and the service provider will choose the protocol to use.
IPPROTO_ICMP 1	The Internet Control Message Protocol (ICMP). This is a possible value when the af parameter is AF_UNSPEC, AF_INET, or AF_INET6 and the type parameter is SOCK_RAW or unspecified.
IPPROTO_IGMP 2	The Internet Group Management Protocol (IGMP). This is a possible value when the af parameter is AF_UNSPEC, AF_INET, or AF_INET6 and the type parameter is SOCK_RAW or unspecified.
BTHPROTO_RFCOMM 3	The Bluetooth Radio Frequency Communications (Bluetooth RFCOMM) protocol. This is a possible value when the af parameter is AF_BTH and the type is SOCK_STREAM.
IPPROTO_TCP 6	The Transmission Control Protocol (TCP). This is a possible value when the af parameter is AF_INET or AF_INET6 and the type parameter is SOCK_STREAM.
IPPROTO_UDP 17	The User Datagram Protocol (UDP). This is a possible value when the af parameter is AF_INET or AF_INET6 and the type parameter is SOCK_DGRAM.
IPPROTO_ICMPV6 58	The ICMPv6. This is a possible value when the af parameter is AF_UNSPEC, AF_INET, or AF_INET6 and the type parameter is SOCK_RAW or unspecified.
IPPROTO_RM 113	The PGM protocol for reliable multicast. This is a possible value when the af parameter is AF_INET and the type parameter is SOCK_RDM.



函数 socket

- 返回值
 - If no error occurs, socket returns a descriptor referencing the new socket. Otherwise, a value of INVALID_SOCKET (Preferred Style) is returned, and a specific error code can be retrieved by calling WSAGetLastError.
- 高级内容
 - Sockets without the overlapped attribute can be created by WSASocket.
 - Connection-oriented sockets such as SOCK_STREAM provide full-duplex connections, and must be in a connected state before any data can be sent or received on it. A connection to another socket is created with a connect call. Once connected, data can be transferred using send and recv calls. When a session has been completed, a closesocket must be performed.

函数 bind

- 函数 bind 将本地地址关联到插口上。

```
int bind(  
    _In_    SOCKET s,  
    _In_    const struct sockaddr *name,  
    _In_    int namelen  
);
```

s [in]

A descriptor identifying an unbound socket.

name [in]

A pointer to a sockaddr structure of the local address to assign to the bound socket.

namelen [in]

The length, in bytes, of the value pointed to by the name parameter.



结构体 sockaddr

- The sockaddr structure varies depending on the protocol selected.

```
struct sockaddr {  
    ushort   sa_family;  
    char     sa_data[14];  
};  
  
struct sockaddr_in {  
    short    sin_family;  
    u_short  sin_port;  
    struct   in_addr sin_addr;  
    char     sin_zero[8];  
};
```

The in_addr structure represents an IPv4 Internet address.

函数 bind

```
// Declare variables
SOCKET ListenSocket;
struct sockaddr_in saServer;
hostent* localHost;
char* localIP;

// Create a listening socket
ListenSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);

// Get the local host information
localHost = gethostbyname("");
localIP = inet_ntoa (*((struct in_addr *)*localHost->h_addr_list));

// Set up the sockaddr structure
saServer.sin_family = AF_INET;
saServer.sin_addr.s_addr = inet_addr(localIP);
saServer.sin_port = htons(5150);

// Bind the listening socket using the
// information in the sockaddr structure
bind( ListenSocket,(SOCKADDR*) &saServer, sizeof(saServer) );
```

网络字节顺序 (htons)

- 字节顺序

- 大端序 (Big-endian) 和小端序 (Little-endian byte order)
 - 不同计算机存放多字节值的顺序不同，有的在起始地址存放低位字节（小数在前），有的存高位字节（大数在前）。

- 网络字节顺序

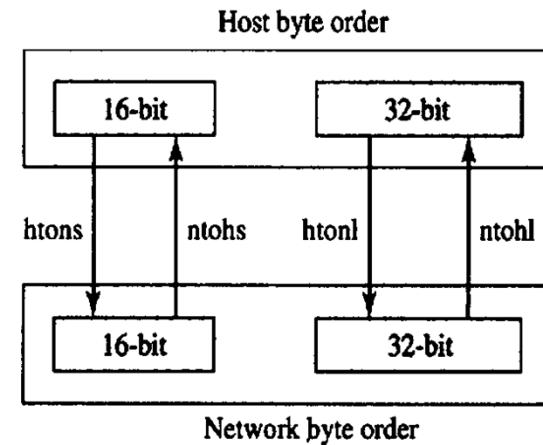
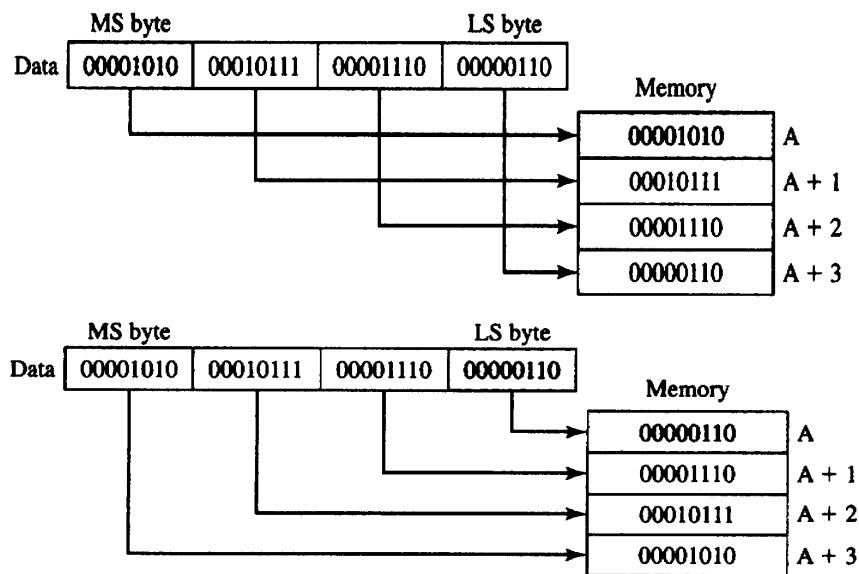
- 网络协议需指定字节顺序，保证数据的正确性
 - TCP/IP 使用 **16位和32位整数的大数在前** 格式

- 主机中的字节顺序



网络字节顺序

- 上图大尾序：高地址为尾
- 下图小尾序：低地址为尾
- 转换：**(host, network) to (h, n)** (**short, long**)



函数 connect

- 函数 connect 建立到一个给定套接字的链接。

```
int connect(  
    _In_    SOCKET s,  
    _In_    const struct sockaddr *name,  
    _In_    int namelen  
);
```

s [in]

A descriptor identifying an unconnected socket.

name [in]

A pointer to a sockaddr structure to which the connection should be established.

namelen [in]

The length, in bytes, of the sockaddr structure pointed to by the name parameter.

函数 connect

```
// The sockaddr_in structure specifies the address family,
// IP address, and port of the server to be connected to.
sockaddr_in clientService;
clientService.sin_family = AF_INET;
clientService.sin_addr.s_addr = inet_addr("127.0.0.1");
clientService.sin_port = htons(27015);

// Connect to server.
iResult = connect(ConnectSocket, (SOCKADDR *) & clientService, sizeof
(clientService));
if (iResult == SOCKET_ERROR) {
    wprintf(L"connect function failed with error: %ld\n",
WSAGetLastError());
    iResult = closesocket(ConnectSocket);
    if (iResult == SOCKET_ERROR)
        wprintf(L"closesocket function failed with error: %ld\n",
WSAGetLastError());
    WSACleanup();
    return 1;
}
wprintf(L"Connected to server.\n");
```



函数 listen

- 函数 listen 将套接字置于侦听传入连接的状态中。

```
int listen(  
    _In_  SOCKET s,  
    _In_  int backlog  
);
```

s [in]

A descriptor identifying a bound, unconnected socket.

backlog [in]

The maximum length of the queue of pending connections. If set to SOMAXCONN, the underlying service provider responsible for socket s will set the backlog to a maximum reasonable value. There is no standard provision to obtain the actual backlog value.

函数 accept

- 函数 accept 允许套接字上的传入连接尝试。

```
SOCKET accept(  
    _In_      SOCKET s,  
    _Out_     struct sockaddr *addr,  
    _Inout_   int *addrlen  
) ;
```

s [in]

A descriptor that identifies a socket that has been placed in a listening state with the listen function.

addr [out]

An optional pointer to a buffer that receives the address of the connecting entity, as known to the communications layer.

addrlen [in]

An optional pointer to an integer that contains the length of structure pointed to by the addr parameter.

函数 accept

- 返回值

- 如果没有发生错误，则接受返回SOCKET类型的值，该套接字是新套接字的描述符。此返回值是实际连接的套接字的句柄。
- 否则，一个价值INVALID_SOCKET返回一个特定的错误代码可以通过调用WSAGetLastError检索。

函数 accept

```
// Listen for incoming connection requests.  
// on the created socket  
if (listen(ListenSocket, 1) == SOCKET_ERROR) {  
    wprintf(L"listen failed with error: %ld\n", WSAGetLastError());  
    closesocket(ListenSocket);  
    WSACleanup();  
    return 1;  
}  
  
// Create a SOCKET for accepting incoming requests.  
SOCKET AcceptSocket;  
wprintf(L"Waiting for client to connect...\n");  
// Accept the connection.  
AcceptSocket = accept(ListenSocket, NULL, NULL);  
if (AcceptSocket == INVALID_SOCKET) {  
    wprintf(L"accept failed with error: %ld\n", WSAGetLastError());  
    closesocket(ListenSocket);  
    WSACleanup();  
    return 1;  
} else  
    wprintf(L"Client connected.\n");
```

函数 send

- 函数 send 将数据发送到一个已连接的套接字。

```
int send(  
    _In_ SOCKET s,  
    _In_ const char *buf,  
    _In_ int len,  
    _In_ int flags  
);
```

s [in]

A descriptor identifying a connected socket.

buf [in]

A pointer to a buffer containing the data to be transmitted.

len [in]

The length (bytes) of the data in buffer pointed to by the buf para.

flag [in]

A set of flags that specify the way in which the call is made.

函数 send

- 参数 flag
 - Constructed by using the bitwise OR operator with any of the following.

Value	Meaning
0	Default.
MSG_DONTROUTE	Specifies that the data should not be subject to routing. A Windows Sockets service provider can choose to ignore this flag.
MSG_OOB	Sends OOB data (stream-style socket such as SOCK_STREAM only).

- 返回值
 - If no error occurs, send returns the total number of bytes sent, which can be less than the number requested to be sent in the len parameter. Otherwise, a value of SOCKET_ERROR is returned, and a specific error code can be retrieved by calling WSAGetLastError.

函数 send

```
// Send an initial buffer
iResult = send( ConnectSocket, sendbuf, (int)strlen(sendbuf), 0 );
if (iResult == SOCKET_ERROR) {
    wprintf(L"send failed with error: %d\n", WSAGetLastError());
    closesocket(ConnectSocket);
    WSACleanup();
    return 1;
}
printf("Bytes Sent: %d\n", iResult);

// shutdown the connection since no more data will be sent
iResult = shutdown(ConnectSocket, SD_SEND);
if (iResult == SOCKET_ERROR) {
    wprintf(L"shutdown failed with error: %d\n", WSAGetLastError());
    closesocket(ConnectSocket);
    WSACleanup();
    return 1;
}
```

函数 recv

- 函数 recv 从连接套接字或绑定的无连接套接字接收数据。

```
int recv(  
    _In_ SOCKET s,  
    _In_ char *buf,  
    _In_ int len,  
    _In_ int flags  
) ;
```

s [in]
buf [in]
len [in]
flag [in]

Differs from flag parameter in send.



函数 recv

- 参数 flag

Value	Meaning
0	Default.
MSG_PEEK	Peeks at the incoming data. The data is copied into the buffer, but is not removed from the input queue. The function subsequently returns the amount of data that can be read in a single call to the recv (or recvfrom) function, which may not be the same as the total amount of data queued on the socket.
MSG_OOB	Processes Out Of Band (OOB) data.
MSG_WAITALL	The receive request will complete only when one of the following events occurs: <ul style="list-style-type: none">• The buffer supplied by the caller is completely full.• The connection has been closed.• The request has been canceled or an error occurred. This flag is not supported on datagram sockets or message-oriented sockets.

函数 recv

- 返回值

- If no error occurs, `recv` returns the number of bytes received and the buffer pointed to by the `buf` parameter will contain this data received. If the connection has been gracefully closed, the return value is zero.

```
// Receive until the peer closes the connection
do {

    iResult = recv(ConnectSocket, recvbuf, recvbuflen, 0);
    if ( iResult > 0 )
        wprintf(L"Bytes received: %d\n", iResult);
    else if ( iResult == 0 )
        wprintf(L"Connection closed\n");
    else
        wprintf(L"recv failed with error: %d\n", WSAGetLastError());

} while( iResult > 0 );
```

函数 sendto/recvfrom

- UDP 的 send/recv

```
int sendto(
    _In_   SOCKET s,
    _In_   const char *buf,
    _In_   int len,
    _In_   int flags,
    _In_   const struct sockaddr *to,
    _In_   int tolen
);

int recvfrom(
    _In_           SOCKET s,
    _Out_          char *buf,
    _In_           int len,
    _In_           int flags,
    _Out_          struct sockaddr *from,
    _Inout_opt_    int *fromlen
);
```

函数 sendto

```
// Initialize Winsock
iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);
if (iResult != NO_ERROR) { ... }
// Create a socket for sending data
SendSocket = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
if (SendSocket == INVALID_SOCKET) { ... }
// Set up the RecvAddr structure with the IP address of
// the receiver (in this example case "192.168.1.1")
// and the specified port number.
RecvAddr.sin_family = AF_INET;
RecvAddr.sin_port = htons(Port);
RecvAddr.sin_addr.s_addr = inet_addr("192.168.1.1");
// Send a datagram to the receiver
wprintf(L"Sending a datagram to the receiver...\n");
iResult = sendto(SendSocket, SendBuf, BufLen, 0, (SOCKADDR *) &RecvAddr, sizeof
(RecvAddr));
if (iResult == SOCKET_ERROR) { ... }
// When the application is finished sending, close the socket.
wprintf(L"Finished sending. Closing socket.\n");
iResult = closesocket(SendSocket);
if (iResult == SOCKET_ERROR) { ... }
```

函数 recvfrom

```
// Initialize Winsock
iResult = WSAStartup(MAKEWORD(2, 2), &wsaData);
if (iResult != NO_ERROR) { ... }

// Create a receiver socket to receive datagrams
RecvSocket = socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
if (RecvSocket == INVALID_SOCKET) { ... }

// Bind the socket to any address and the specified port.
RecvAddr.sin_family = AF_INET;
RecvAddr.sin_port = htons(Port);
RecvAddr.sin_addr.s_addr = htonl(INADDR_ANY);
iResult = bind(RecvSocket, (SOCKADDR *) &RecvAddr, sizeof(RecvAddr));
if (iResult != 0) { ... }

// Call the recvfrom function to receive datagrams on the bound socket.
wprintf(L"Receiving datagrams...\n");
iResult = recvfrom(RecvSocket, RecvBuf, BufLen, 0, (SOCKADDR *) &SenderAddr,
&SenderAddrSize);
if (iResult == SOCKET_ERROR) { ... }

// Close the socket when finished receiving datagrams
wprintf(L"Finished receiving. Closing socket.\n");
iResult = closesocket(RecvSocket);
if (iResult == SOCKET_ERROR) { ... }
```

函数 WSAStartup

- 函数 WSAStartup 启动使用 Winsock DLL 的过程。
 - 当前版本的 Windows 套接字规范是版本 2.2。

```
int WSAStartup(  
    _In_    WORD wVersionRequested,  
    _Out_   LPWSADATA lpWSAData  
);
```

wVersionRequested [in]

The highest version of Windows Sockets specification that the caller can use. The high-order byte specifies the minor version number; the low-order byte specifies the major version number.

lpWSAData [out]

A pointer to the WSADATA data structure that is to receive details of the Windows Sockets implementation.

函数 WSASStartup

- 假如一个程序要使用2.1版本的Socket，程序代码如下

```
#define MAKEWORD(a, b) (((WORD)((((BYTE)((((DWORD_PTR)(a)) & 0xff)) | \
    ((WORD)((BYTE)((((DWORD_PTR)(b)) & 0xff)))) << 8))

wVersionRequested = MAKEWORD( 2, 1 );
err = WSASStartup( wVersionRequested, &wsaData );
```

函数 closesocket

- 函数 closesocket 关闭已经存在的插口。

```
int closesocket(  
    _In_  SOCKET s  
)
```

s [in]

A descriptor identifying the socket to close.



内容纲要

1

客户端和服务器端

2

网间进程通信

3

Socket API

4

客户服务器端程序实例

5

小结



服务器端编程

```
/* server.c - code for example server program that uses TCP */
#ifndef unix
#define WIN32
#include <windows.h>
#include <winsock.h>
#else
#define closesocket close
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#endif

#include <stdio.h>
#include <string.h>

#define PROTOPORT      5193          /* default protocol port number */
#define QLEN            6              /* size of request queue */

int    visits        = 0;           /* counts client connections */
```

平台间差异



服务器端编程

```
/*
 * Program:    server
 *
 * Purpose:   allocate a socket and then repeatedly execute the following:
 *             (1) wait for the next connection from a client
 *             (2) send a short message to the client
 *             (3) close the connection
 *             (4) go back to step (1)
 *
 * Syntax:    server [ port ]
 *
 *           port - protocol port number to use
 *
 * Note:      The port argument is optional. If no port is specified,
 *           the server uses the default given by PROTOPORT.
 *
 */
main(argc, argv)
int    argc;
char   *argv[];
```

服务器端编程

{

```
    struct hostent *ptrh; /* pointer to a host table entry */
    struct protoent *ptrp; /* pointer to a protocol table entry */
    struct sockaddr_in sad; /* structure to hold server's address */
    struct sockaddr_in cad; /* structure to hold client's address */
    int sd, sd2; /* socket descriptors */
    int port; /* protocol port number */
    int alen; /* length of address */
    char buf[1000]; /* buffer for string the server sends */
```

```
#ifdef WIN32
```

```
    WSADATA wsaData;
    WSAStartup(0x0101, &wsaData);
```

```
#endif
```

```
    memset((char *)&sad, 0, sizeof(sad)); /* clear sockaddr structure */
    sad.sin_family = AF_INET; /* set family to Internet */
    sad.sin_addr.s_addr = INADDR_ANY; /* set the local IP address */
    /* Check command-line argument for protocol port and extract */
    /* port number if one is specified. Otherwise, use the default */
    /* port value given by constant PROTOPORT */
```

WSAStartup应为第一个调用的Windows Socket函数，
用于定义版本号。

服务器端编程

```
if (argc>1 && argv[1][0]=='?')
{
    printf("Program:    server\n");
    printf("Purpose:   allocate a socket and then repeatedly execute the
following:\n");
    printf("              (1) wait for the next connection from a client\n");
    printf("              (2) send a short message to the client\n");
    printf("              (3) close the connection\n");
    printf("              (4) go back to step (1)\n");
    printf("Syntax:      server [ port ]\n");
    printf("              port - protocol port number to use\n");
    printf("Note:        The port argument is optional. If no port is
specified,\n");
    printf("              the server uses the default given by PROTOPORT.\n");
    exit(1);
}
```

服务器端编程

```
if (argc > 1) { /* if argument specified */  
    port = atoi(argv[1]); /* convert argument to binary */  
} else {  
    port = PROTOPORT; /* use default port number */  
}  
if (port > 0) /* test for illegal value */  
    sad.sin_port = htons((u_short)port);  
else { /* print error message and exit */  
    fprintf(stderr, "bad port number %s\n", argv[1]);  
    exit(1);  
}  
/* Map TCP transport protocol name to protocol number */  
  
if ( ((int)(ptrp = getprotobynumber("tcp"))) == 0) {  
    fprintf(stderr, "cannot map \"tcp\" to protocol number");  
    exit(1);  
}
```

htons将u_short从主机顺序转换到TCP/IP网络字节顺序（大端）。

Getprotobynumber 检索协议名称对应的的协议信息。

服务器端编程

协议族
IPv4

```
/* Create a socket */  
sd = socket(PF_INET, SOCK_STREAM, pptrp->p_proto);  
if (sd < 0) {  
    fprintf(stderr, "socket creation failed\n");  
    exit(1);  
}  
  
/* Bind a local address to the socket */
```

新套接字的类型：供了序列、可靠、双向，
基于连接的带OOB数据传输机制的字节流。

协议类型

socket函数创建一个绑定到一个
特定运输服务提供商的套接字。

对套接字赋值的
本地地址指针

```
if (bind(sd, (struct sockaddr *)&sad, sizeof(sad)) < 0) {  
    fprintf(stderr, "bind failed\n");  
    exit(1);  
}
```

识别未绑定的套
接字的描述符。

```
/* Specify size of request queue */
```

bind函数将本地地址与套接
字相关联。

```
if (listen(sd, QLEN) < 0) {  
    fprintf(stderr, "listen failed\n");  
    exit(1);  
}
```

队列长度

listen将套接字置于侦听传入
连接的状态之下



服务器端编程

一个死循环

关闭套接字。
注意是哪个套接字

```
/* Main server loop - accept and handle requests */  
  
while (1) {  
    alen = sizeof(cad);  
    if ((sd2=accept(sd, (struct sockaddr *)&cad, &alen)) < 0) {  
        fprintf(stderr, "accept failed\n");  
        exit(1);  
    }  
    visits++;  
    sprintf(buf, "This server has been contacted %d time%s\n",  
            visits, visits==1?"." :"s.");  
    send(sd2,buf,strlen(buf),0);  
    closesocket(sd2);  
}  
}
```

Accept函数允许在套接字上传入的连接尝试。

可选的指针，指向到一个缓冲区，它接收连接实体的地址，如已知的通信层。

Send函数向一个已连接的套接字发送数据。



客户端编程

```
/* client.c - code for example client program that uses TCP */

#ifndef unix
#define WIN32
#include <windows.h>
#include <winsock.h>
#else
#define closesocket close
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#endif

#include <stdio.h>
#include <string.h>

#define PROTOPORT      5193          /* default protocol port number */
extern int           errno;
char    localhost[] = "localhost"; /* default host name */
```

客户端编程

```
/*
 * Program:    client
 *
 * Purpose:   allocate a socket, connect to a server, and print all output
 *
 * Syntax:    client [ host [port] ]
 *
 *           host - name of a computer on which server is executing
 *           port - protocol port number server is using
 *
 * Note:      Both arguments are optional. If no host name is specified,
 *            the client uses "localhost"; if no protocol port is
 *            specified, the client uses the default given by PROTOPORT.
 *
 */
main(argc, argv)
int    argc;
char   *argv[];
```

客户端编程

```
{  
    struct hostent *ptrh; /* pointer to a host table entry */  
    struct protoent *ptrp; /* pointer to a protocol table entry */  
    struct sockaddr_in sad; /* structure to hold an IP address */  
    int sd; /* socket descriptor */  
    int port; /* protocol port number */  
    char *host; /* pointer to host name */  
    int n; /* number of characters read */  
    char buf[1000]; /* buffer for data from the server */  
  
#ifdef WIN32  
    WSADATA wsaData;  
    WSAStartup(0x0101, &wsaData);  
#endif  
    memset((char *)&sad, 0, sizeof(sad)); /* clear sockaddr structure */  
    sad.sin_family = AF_INET; /* set family to Internet */  
  
    /* Check command-line argument for protocol port and extract */  
    /* port number if one is specified. Otherwise, use the default */  
    /* port value given by constant PROTOPORT */  
}
```

客户端编程

```
if (argc>1 && argv[1][0]=='?')
{
    printf("Program:    client\n");
    printf("Purpose:   allocate a socket, connect to a server, and print all
output\n");
    printf("Syntax:    client [ host [port] ]\n");
    printf("          host - name of a computer on which server is
executing\n");
    printf("          port - protocol port number server is using\n");
    printf("Note:      Both arguments are optional. If no host name is
specified,\n");
    printf("          the client uses \"localhost\"; if no protocol port is\n");
    printf("          specified, the client uses the default given by
PROTOPORT.\n");
    exit(1);
}
```



客户端编程

```
if (argc > 2) {                                /* if protocol port specified */
    port = atoi(argv[2]);                         /* convert to binary */
} else {
    port = PROTOPORT;                            /* use default port number */
}
if (port > 0)                                    /* test for legal value */
    sad.sin_port = htons((u_short)port);
else {
    /* print error message and exit */
    fprintf(stderr,"bad port number %s\n",argv[2]);
    exit(1);
}

/* Check host argument and assign host name. */

if (argc > 1) {
    host = argv[1];                             /* if host argument specified */
} else {
    host = localhost;
}
```



客户端编程

```
/* Convert host name to equivalent IP address and copy to sad. */
ptrh = gethostbyname(host);
if ( ((char *)ptrh) == NULL ) {
    fprintf(stderr, "invalid host: %s\n", host);
    exit(1);
}
memcpy(&sad.sin_addr, ptrh->h_addr, ptrh->h_length);

/* Map TCP transport protocol name to protocol number. */
if ( ((int)(ptrp = getprotobynumber("tcp"))) == 0 ) {
    fprintf(stderr, "cannot map \"tcp\" to protocol number");
    exit(1);
}

/* Create a socket. */

sd = socket(PF_INET, SOCK_STREAM, ptrp->p_proto);
if (sd < 0) {
    fprintf(stderr, "socket creation failed\n");
    exit(1);
}
```

从主机名获取指向
主机的指针



客户端编程

```
/* Connect the socket to the specified server. */

if (connect(sd, (struct sockaddr *)&sad, sizeof(sad)) < 0) {
    fprintf(stderr, "connect failed\n");
    exit(1);
}

/* Repeatedly read data from socket and write to user's screen. */

n = recv(sd, buf, sizeof(buf), 0);
while (n > 0) {
    write(1,buf,n);
    n = recv(sd, buf, sizeof(buf), 0);
}

/* Close the socket. */
closesocket(sd);

/* Terminate the client program gracefully. */
exit(0);
}
```

recv函数从连接的套接字或无连接的套接字绑定接收数据。



Microsoft .Net Framework

- System.Net.Sockets 命名空间
- 详见UDP课的PPT

C++ 实例

```
#using <System.dll>

using namespace System;
using namespace System::Text;
using namespace System::IO;
using namespace System::Net;
using namespace System::Net::Sockets;
String^ DoSocketGet( String^ server )
{
    //Set up variables and String to write to the server.
    Encoding^ ASCII = Encoding::ASCII;
    String^ Get = "GET / HTTP/1.1\r\nHost: ";
    Get->Concat( server, "\r\nConnection: Close\r\n\r\n" );
    array<Byte>^ByteGet = ASCII->GetBytes( Get );
    array<Byte>^RecvBytes = gcnew array<Byte>(256);
    String^ strRetPage = nullptr;

    // IPAddress and IPEndPoint represent the endpoint that will
    // receive the request.
    // Get first IPAddress in list return by DNS.
```

C++ 实例

```
try
{
    // Define those variables to be evaluated in the next for loop and
    // then used to connect to the server. These variables are defined
    // outside the for loop to make them accessible there after.
    Socket^ s = nullptr;
    IPEndPoint^ hostEndPoint;
    IPAddress^ hostAddress = nullptr;
    int conPort = 80;

    // Get DNS host information.
    IPHostEntry^ hostInfo = Dns::Resolve( server );

    // Get the DNS IP addresses associated with the host.
    array<IPAddress^>^IPAddresses = hostInfo->AddressList;

    // Evaluate the socket and receiving host IPAddress and IPEndPoint.
    for ( int index = 0; index < IPAddresses->Length; index++ )
    {
        hostAddress = IPAddresses[ index ];
```

C++ 实例

```
hostEndPoint = gcnew IPEndPoint( hostAddress,conPort );  
  
// Creates the Socket to send data over a TCP connection.  
s = gcnew Socket( AddressFamily::InterNetwork, SocketType::Stream,  
ProtocolType::Tcp );  
  
// Connect to the host using its IPEndPoint.  
s->Connect( hostEndPoint );  
if ( !s->Connected )  
{  
    // Connection failed, try next IPaddress.  
    strRetPage = "Unable to connect to host";  
    s = nullptr;  
    continue;  
}  
  
// Sent the GET request to the host.  
s->Send( ByteGet, ByteGet->Length, SocketFlags::None );  
}
```

C++ 实例

```
// Receive the host home page content and loop until all the data is received.  
Int32 bytes = s->Receive( RecvBytes, RecvBytes->Length, SocketFlags::None );  
strRetPage = "Default HTML page on ";  
strRetPage->Concat( server, ":\r\n", ASCII->GetString( RecvBytes, 0, bytes ) );  
while ( bytes > 0 )  
{  
    bytes = s->Receive( RecvBytes, RecvBytes->Length, SocketFlags::None );  
    strRetPage->Concat( ASCII->GetString( RecvBytes, 0, bytes ) );  
}  
}  
}  
catch ( SocketException^ e )  
{  
    Console::WriteLine( "SocketException caught!!!" );  
    Console::Write( "Source : " );  
    Console::WriteLine( e->Source );  
    Console::Write( "Message : " );  
    Console::WriteLine( e->Message );  
}
```

C++ 实例

```
catch ( ArgumentNullException^ e )
{
    Console::WriteLine( "ArgumentNullException caught!!!" );
    Console::Write( "Source : " );
    Console::WriteLine( e->Source );
    Console::Write( "Message : " );
    Console::WriteLine( e->Message );
}

catch ( NullReferenceException^ e )
{
    Console::WriteLine( "NULLReferenceException caught!!!" );
    Console::Write( "Source : " );
    Console::WriteLine( e->Source );
    Console::Write( "Message : " );
    Console::WriteLine( e->Message );
}
```



C++ 实例

```
catch ( Exception^ e )
{
    Console::WriteLine( "Exception caught!!!" );
    Console::Write( "Source : " );
    Console::WriteLine( e->Source );
    Console::Write( "Message : " );
    Console::WriteLine( e->Message );
}

return strRetPage;
}

int main()
{
    Console::WriteLine( DoSocketGet( "localhost" ) );
}
```

C++ 实例：输出

Default HTML page on 127.0.0.1:

HTTP/1.1 200 OK

Server: ASP.NET Development Server/11.0.0.0

Date: Mon, 29 Apr 2013 09:54:33 GMT

X-AspNet-Version: 4.0.30319

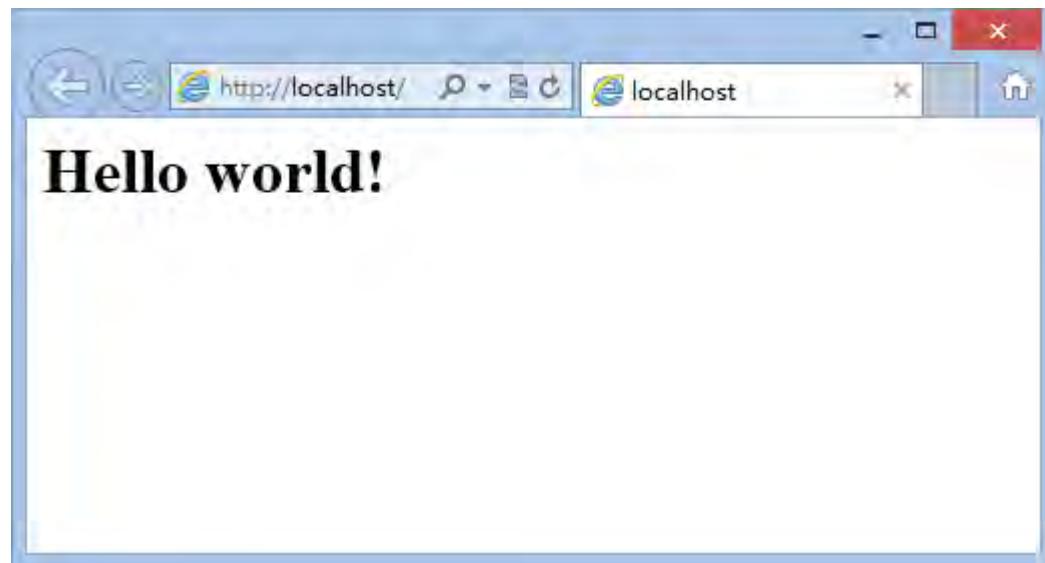
Cache-Control: private

Content-Type: text/html; charset=utf-8

Content-Length: 25

Connection: Close

```
<h1>Hello world!</h1>
```



计算机网络

Computer Network

14

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

15

域名系统

理论课程

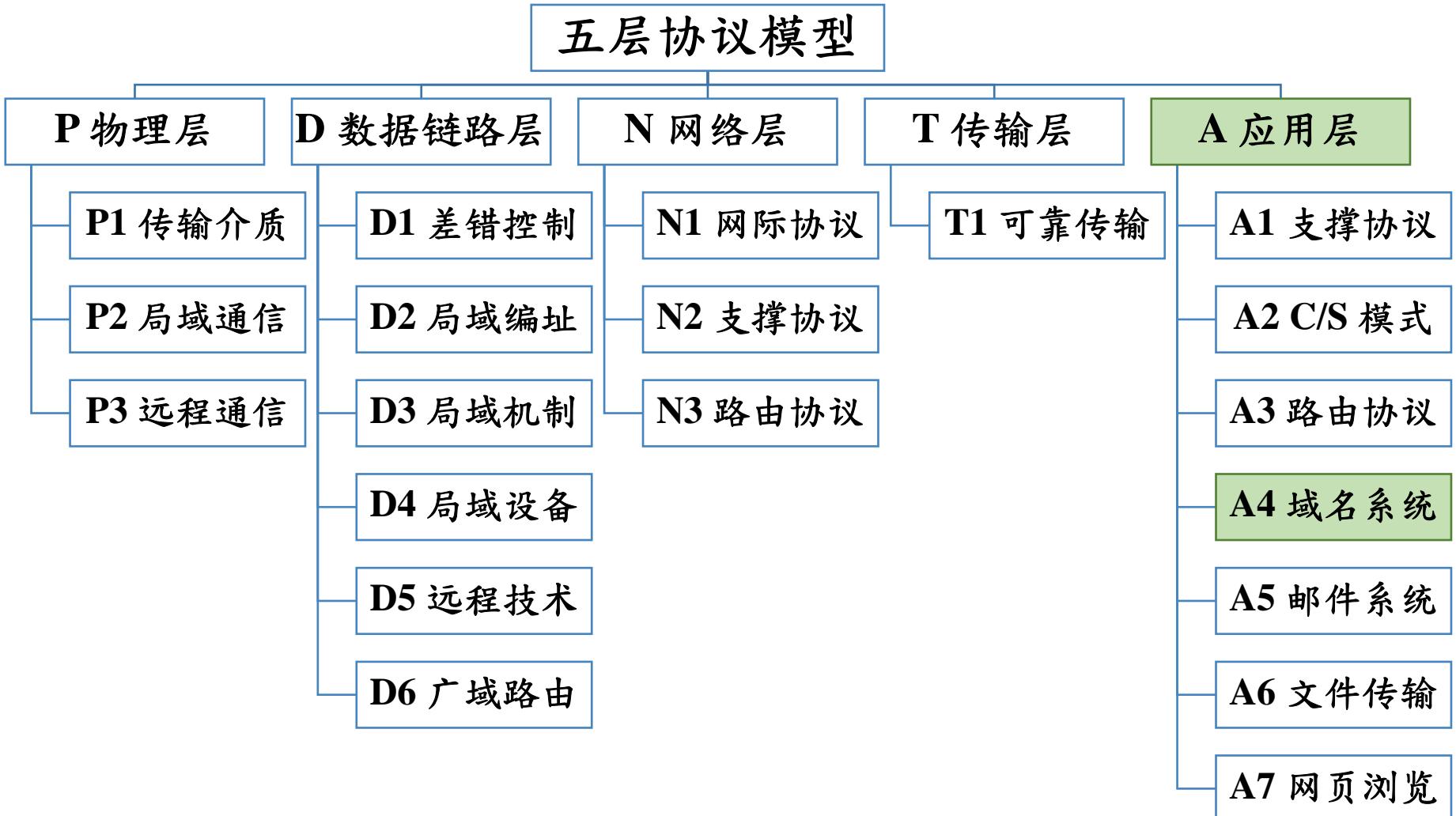


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 域名、域名分级
- 域名服务器分级
- 域名服务（DNS）
 - 递归、迭代的工作原理



对应课本章节

- PART I Introduction And Internet Applications
 - Chapter 4 Traditional Internet Applications
 - 4.17~4.26 Domain Name System (DNS); Domain Names That Begin With www; The DNS Hierarchy And Server Model; Name Resolution; Caching In DNS Servers; Types Of DNS Entries; Aliases And CNAME Resource Records; Abbreviations And The DNS; Internationalized Domain Names



内容纲要

1

域名结构

2

域名服务器的模型

3

域名解析过程

4

选作作业

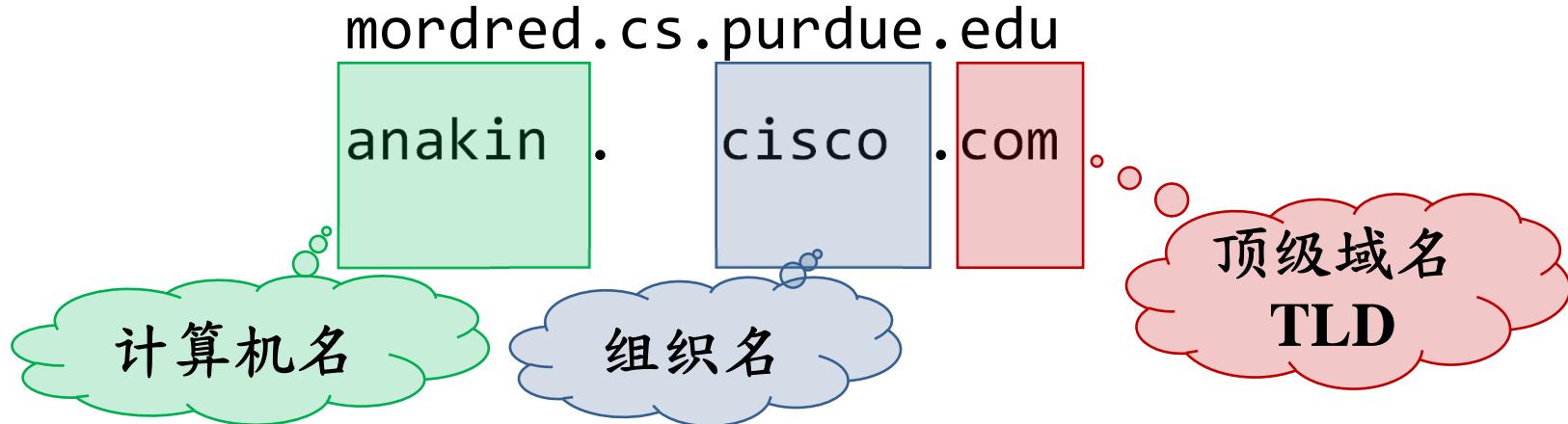


域名系统

- 域名系统（ Domain Name System ， DNS ）
 - 提供了将人类可读符号域名映射到计算机地址的服务
 - 注意：计算机地址不只是IP地址
- 分布式
 - 名字到 IP 地址的解析由若干个域名服务器程序完成。
 - 域名服务器程序在专设的结点上运行，运行该程序的机器称为域名服务器。

域名结构

- 域名
 - 因特网上的主机或路由器所具有的唯一的层次结构的名字。
- 层次树状结构
 - 域名的结构由标号序列组成，各标号之间用点隔开：
 - 与IP地址的点完全不同



域名与IP地址的区别

- 域名只是个逻辑概念，并不代表计算机所在物理地点。
- 域名和IP地址的区别
 - 变长的域名和使用有助记忆的字符串，是为了便于人使用。
 - IP地址是定长的32位数字则非常便于机器进行处理。



顶级域名 (Top Level Domain, TLD)

- 国家或地区顶级域名
- 通用顶级域名
- 基础结构域名 : arpa

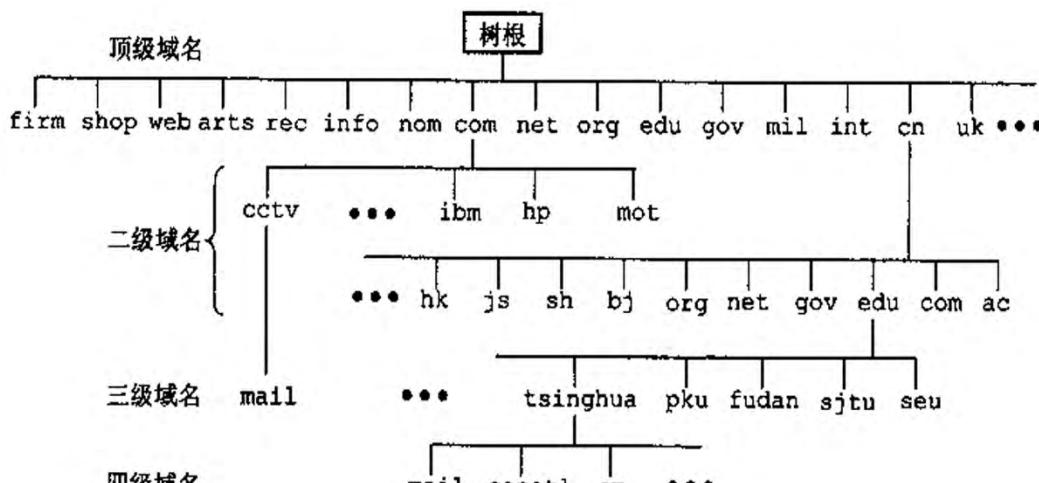


图 12-1 因特网的名字空间

Domain Name	Assigned To
aero	Air transport industry
arpa	Infrastructure domain
asia	For or about Asia
biz	Businesses
com	Commercial organizations
coop	Cooperative associations
edu	Educational institutions
gov	United States Government
info	Information
int	International treaty organizations
jobs	Human resource managers
mil	United States military
mobi	Mobile content providers
museum	Museums
name	Individuals
net	Major network support centers
org	Non-commercial organizations
pro	Credentialed professionals
travel	Travel and tourism
country code	A sovereign nation



域名区

- 区（ zone ）

- 一个服务器所负责管辖的（或有权限的）范围。
- 各单位根据具体情况来划分自己管辖范围的区。但在一个区中的所有节点必须是能够连通的。
- 每一个区设置相应的权限域名服务器，用来保存该区中的所有主机的域名到IP地址的映射。

- DNS 服务器的管辖范围

- 不是以“域”（ domain ）为单位，而是以“区”为单位。

域名别名

- 许多组织指定反映计算机提供服务的域名

ftp.foobar.com

www.foobar.com

- 助记符，但不需要

- 使用www来命名运行Web服务器的计算机是一个惯例

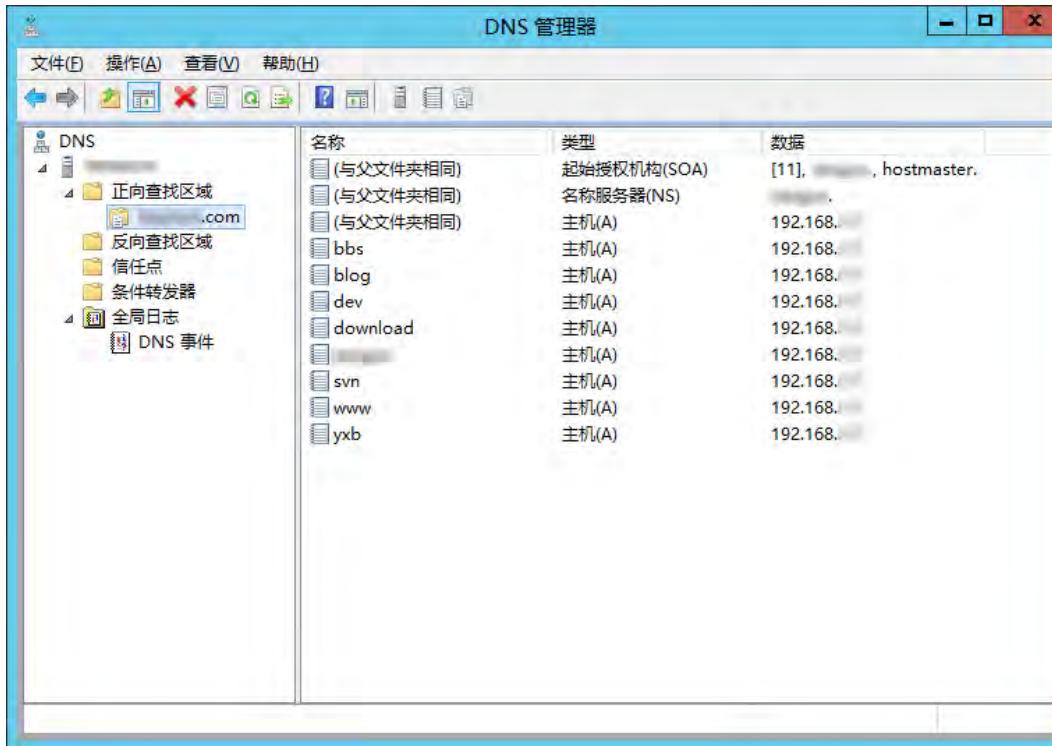
- 任意计算机可以运行Web服务器，即使域名不包含www

- 一个具有www域名的计算机不需要运行Web服务器



DNS层次结构和服务器模型

- 每个组织可以自由选择其服务器的详细信息
 - 将组织所有域名放置在单个物理服务器或多台服务器之间



内容纲要

1

域名结构

2

域名服务器的模型

3

域名解析过程

4

选作作业



域名服务器的四种类型

- 根域名服务器
 - 存储所有顶级域名服务器的地址信息。
- 顶级域名服务器
 - 保存顶级域名下一级区的权限域名服务器。
- 权限域名服务器
 - 保存该区中的所有主机的域名到IP地址的映射。
- 本地域名服务器
 - 进行具体主机的域名解析服务。



根域名服务器

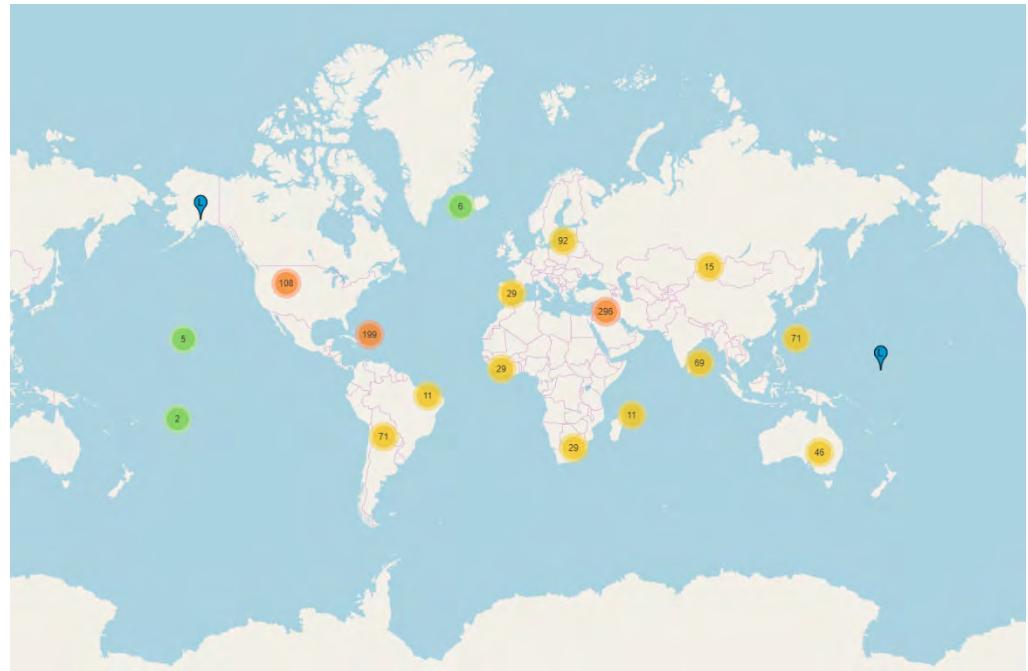
- 根域名服务器是最重要的域名服务器。
 - 根域名服务器知道所有顶级域名服务器的域名和 IP 地址。
 - 本地域名服务器，只要自己无法解析某个域名，就首先求助于根域名服务器。



根域名服务器

- 因特网逻辑上有 13 个不同 IP 地址的根域名服务器
 - 名字用一个英文字母命名，从 a 一直到 m（前 13 个字母）。
 - 世界已有 900 多个根域名服务器

a.root-servers.net
b.root-servers.net
...
m.root-servers.net



顶级域名服务器

- 顶级域名服务器
 - 负责管理在该顶级域名服务器注册的所有二级域名。
- 当收到 DNS 查询请求时，就给出相应的回答
 - 可能是最后的结果
 - 也可能是下一步应当找的域名服务器的 IP 地址

权限域名服务器

- 权限域名服务器
 - 负责一个区的域名服务器。
- 作用
 - 当一个权限域名服务器还不能给出最后的查询回答时，就会告诉发出查询请求的 DNS 客户，下一步应当找哪一个权限域名服务器。

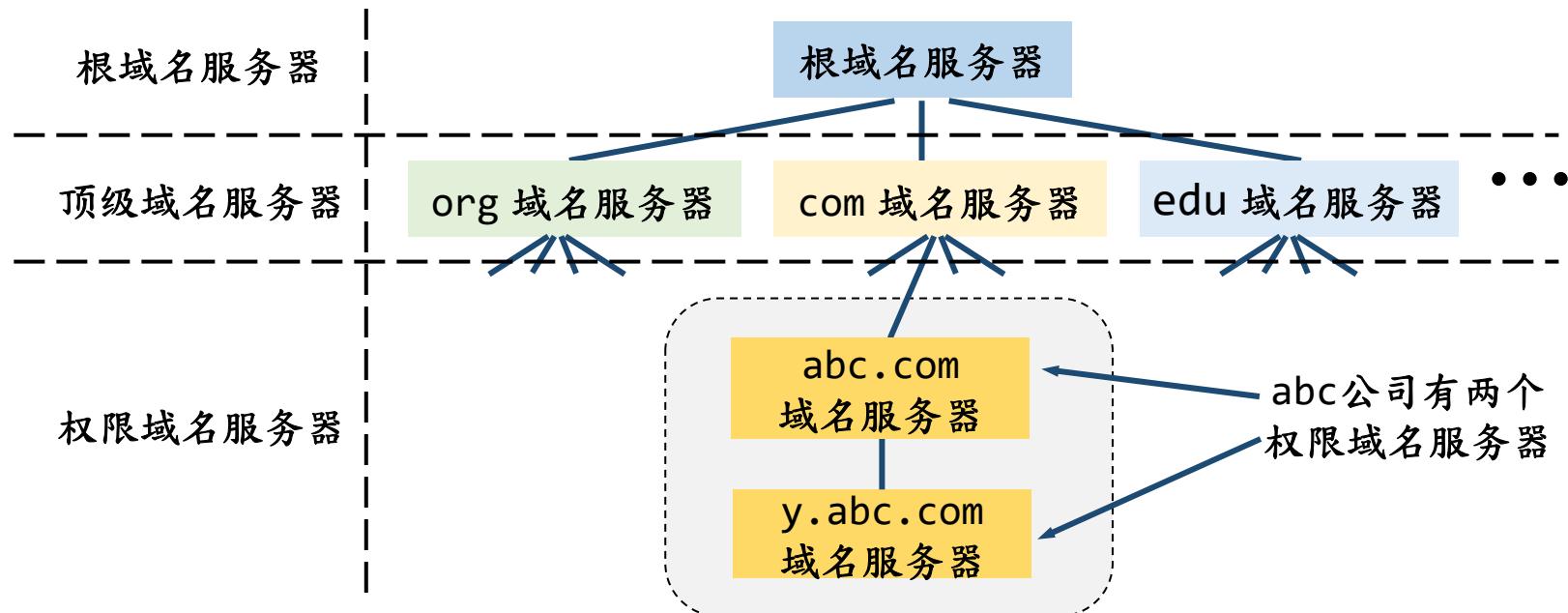
本地域名服务器

- 本地域名服务器（默认域名服务器）
 - 当一个主机发出 DNS 查询请求时，这个查询请求报文就发送给本地域名服务器。
- 每一个因特网服务提供者 ISP，或一个大学，甚至一个大学里的系，都可以拥有一个本地域名服务器，
- 本地服务器对域名系统非常重要。



本地域名服务器和根域名服务器

- 每台主机应知道本地域名服务器（ local name server ）
 - 一般是通过网卡属性获得信息并手工配置
- 每台LNS应知道根域名服务器（ root name servers ）



提高域名服务器的可靠性

- DNS 域名服务器都把数据复制到几个域名服务器来保存，其中的一个是主域名服务器，其他的是辅助域名服务器。
- 当主域名服务器出故障时，辅助域名服务器可以保证 DNS 的查询工作不会中断。
- 主域名服务器定期把数据复制到辅助域名服务器中，而更改数据只能在主域名服务器中进行。这样就保证了数据的一致性。



内容纲要

1

域名结构

2

域名服务器的模型

3

域名解析过程

4

选作作业



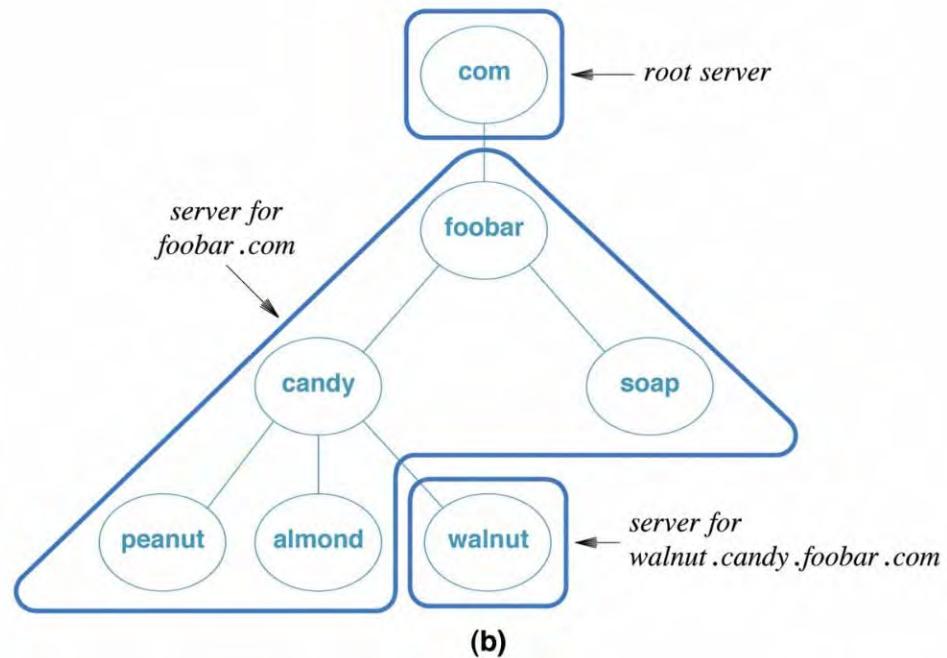
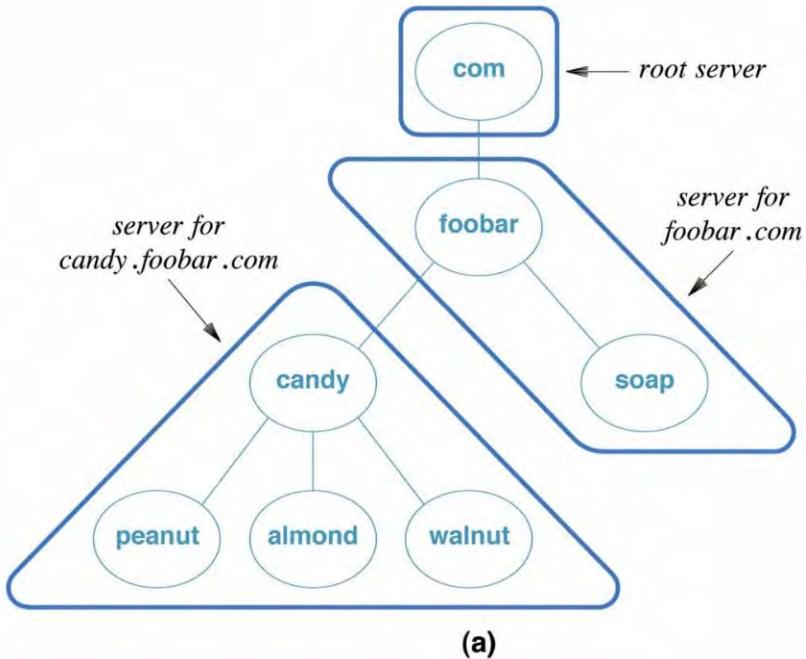
域名的解析过程

- 递归查询：主机向本地域名服务器的查询。
 - 如果主机所询问的本地域名服务器不知道被查询域名的 IP 地址，那么本地域名服务器就以 DNS 客户的身份，向其他根域名服务器继续发出查询请求报文。
- 迭代查询：本地域名服务器向根域名服务器的查询。
 - 当根域名服务器收到本地域名服务器的迭代查询请求报文，要么给出所查询 IP 地址，要么告诉本地域名服务器下一步应向服务器查询，然后让本地域名服务器进行后续查询。

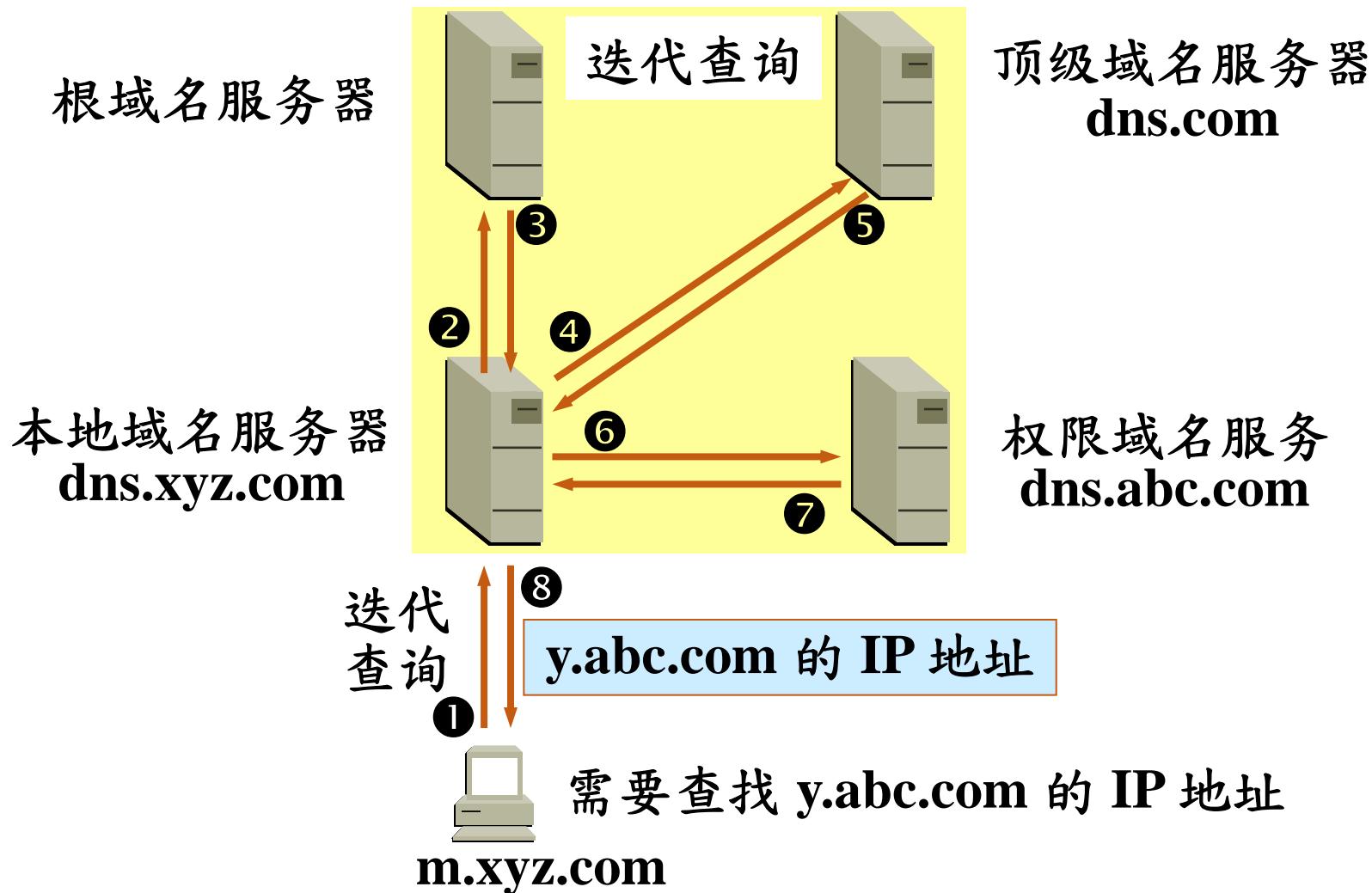


层次结构和服务器模型

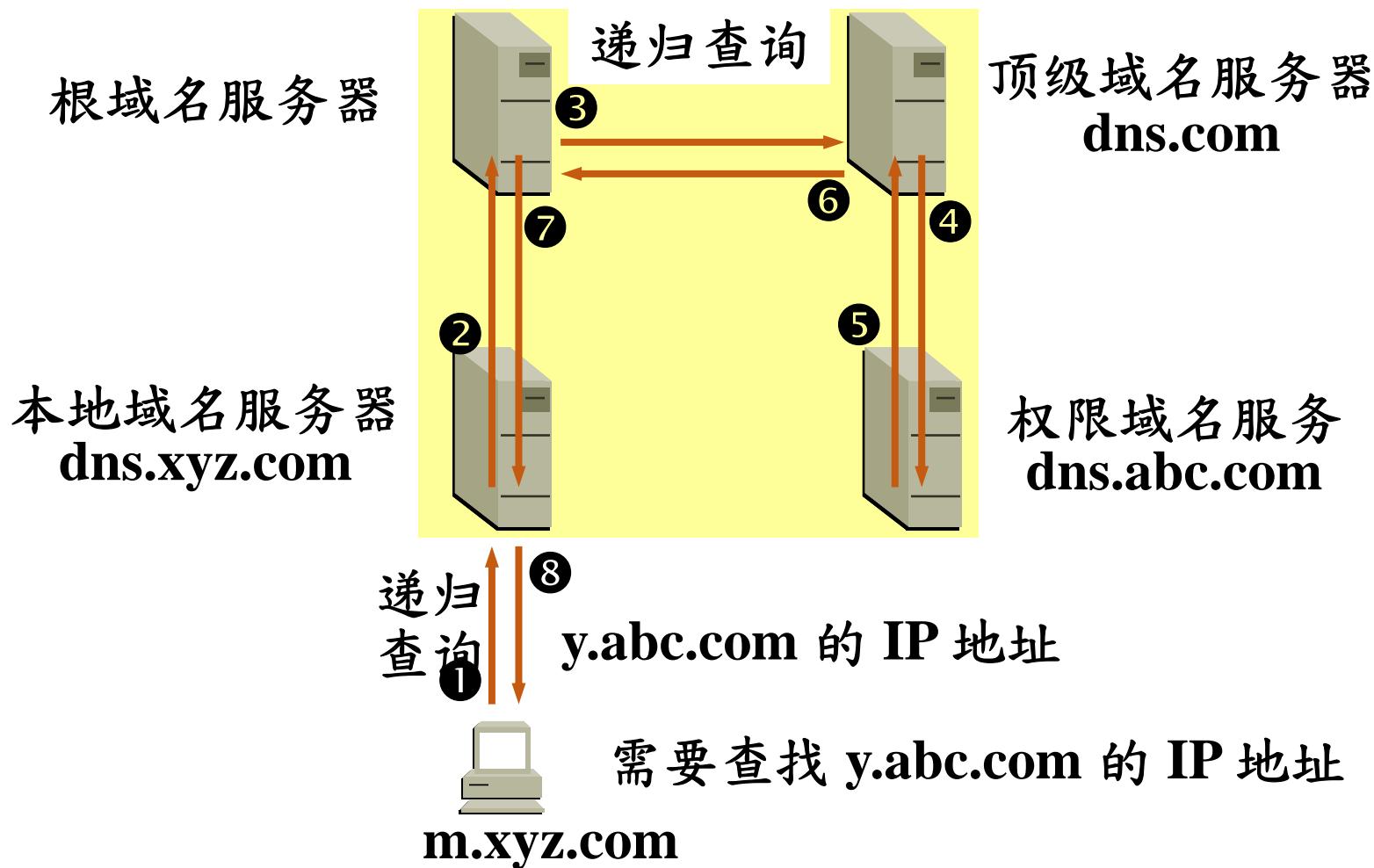
- 允许每个组织将域名分配到计算机或更改这些域名而
不通知中央管理局



本地域名服务器迭代查询



本地域名服务器递归查询（较少用）



域名高速缓存

- 每个域名服务器都维护一个高速缓存，存放最近用过的名字以及从何处获得名字映射信息的记录。
- 可大大减轻根域名服务器的负荷，使因特网上的 DNS 查询请求和回答报文的数量大为减少。
- 为保持高速缓存中的内容正确，域名服务器应为每项内容设置计时器，并处理超时的项。
 - 当权限域名服务器回答一个查询请求时，在响应中都指明绑定有效存在的时间值。增加此时间值可减少网络开销，而减少此时间值可提高域名转换的准确性。



域名解析

- 域名解析（ name resolution ）
 - 域名翻译成一个地址被称为域名解析
 - 执行的软件称为域名解析器（或解析器， resolver ）
 - Socket API : gethostbyname
 - 解析器通过连接DNS服务器成为客户端
 - DNS服务器返回对调用者的回答
- 使用流（ stream ）模式或消息（ message ）模式



DNS条目类型

- DNS数据库中的每个条目由三个项目组成
 - 一个域名，一个记录类型（ record type ）和一个值
- 发送到DNS服务器的查询指定域的域名和类型，服务器只返回与查询类型相匹配的绑定
- 主要的类型将域名映射到IP地址
 - DNS将此绑定分类为类型A：用于应用程序
 - DNS支持几个其他类型，包括类型MX，指定邮件交换器
 - 返回的地址取决于类型



资源记录

- 每个服务器用资源记录（ Resource Record ）的集合实现区域信息。本质上，资源记录是名字到值的绑定
 - <名字Name, 值Value, 类型Type, 分类Class, 生存期TTL>
- 名字name/值value：主机名字到IP地址
- 分类Class
 - 允许其他实体（除InterNIC外）定义有用的记录类型，目前广泛使用的分类是因特网使用的分类，记IN。
- 生存期TTL：指明资源记录的有效期限



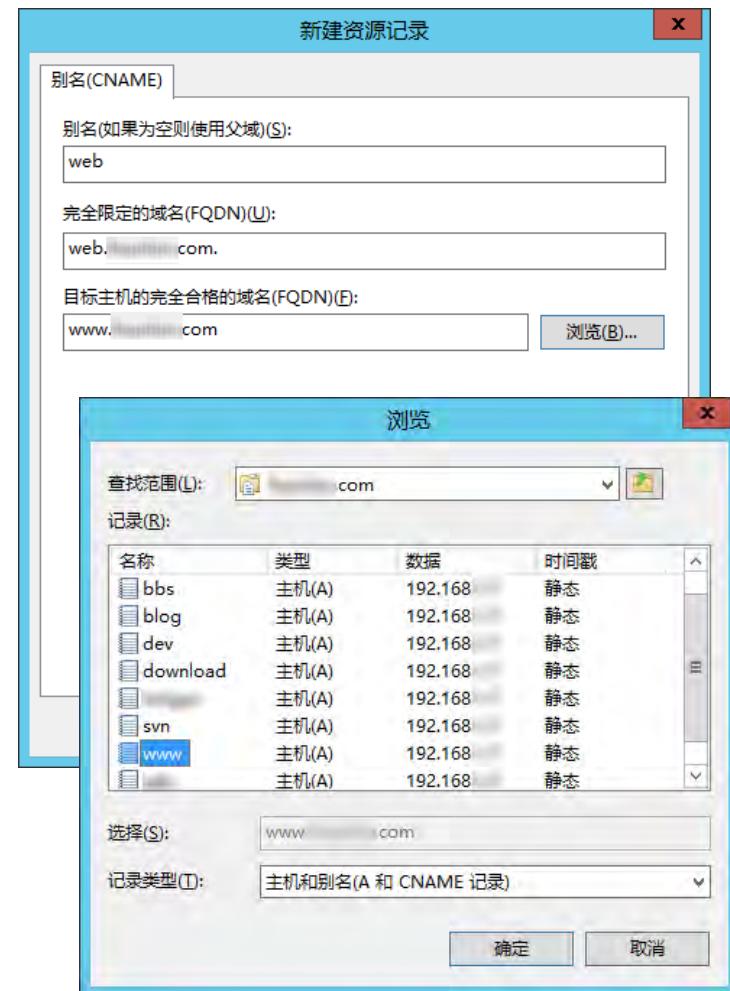
资源记录中各个字段的含义

- 类型type

- NS：值字段给出了运行名字服务器的主机域名，而该名字服务器知道如何解析特定的域名
- CNAME：值字段给出了特定主机的规范名字，主要用于定义主机别名
- MX：值字段给出了运行邮件服务器的主机域名，而该邮件服务器知道如何接收解析指定域的值

别名和CNAME资源记录

- DNS提供了一个CNAME
 - 为另一个DNS条目提供别名很有用
- 假设foobar.com下有名为abc的计算机，运行一段时间后，希望将其部署为Web服务器。
 - 最好以www.foobar.com为域名
 - CNAME可不必重复部署或改名



国际化域名

- DNS 使用 ASCII 字符集
- 国际化域名应用 (IDNA)
 - 因特网工程工作组 (<http://www.ietf.org>) 在 RFC 3490 下定义的一个协议
- 中文域名



内容纲要

1

域名结构

2

域名服务器的模型

3

域名解析过程

4

选作作业



选作作业

- 用Wireshark监听收发DNS的数据流
 - 访问厦门大学信息学院软件工程系主页
 - 浏览器对DNS的访问是基于TCP的还是UDP的
- 请你的同学配合，在不同地方ping一些门户网站的主机，查看DNS是否指向同一个IP地址，这样做有何好处？（是不是意味着访问不同的内容？）



计算机网络

Computer Network

15

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

计算机网络

Computer Network

16

电子邮件

理论课程

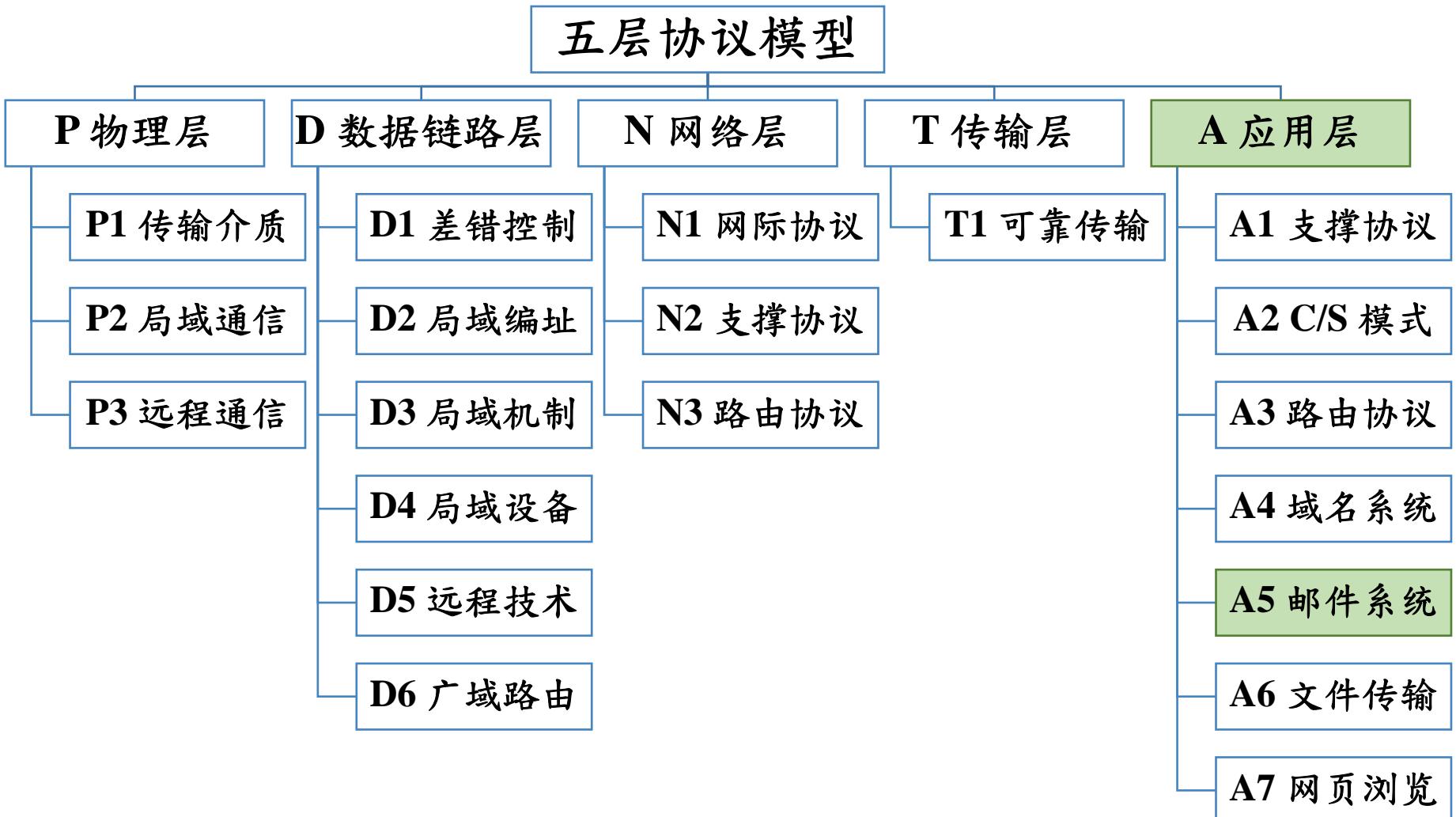


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- 电子邮件的格式
- 主要构成：MTA、MUA、MDA
- 主要协议（作用、原理、端口号）
 - 电子邮件的传输：SMTP
 - 电子邮件的传输扩展：MIME
 - 电子邮件的访问：POP3，IMAP

对应课本章节

- PART I Introduction And Internet Applications
 - Chapter 4 Traditional Internet Applications
 - 4.12 Electronic Mail
 - 4.13 The Simple Mail Transfer Protocol (SMTP)
 - 4.14 ISPs, Mail Servers, And Mail Access
 - 4.15 Mail Access Protocols (POP, IMAP)
 - 4.16 Email Representation Standards (RFC2822, MIME)

内容纲要

1

电子邮件概述

2

电子邮件格式

3

邮件代理和协议

4

SMTP协议

5

MIME标准



电子邮件概述

- 电子邮件 (e-mail)
 - 1972年BBN的Ray Tomlinson发明，并采用 @ 符号。
 - 把邮件发送到收件人使用的邮件服务器，并放在其中的收件人邮箱中，收件人可随时上网到自己使用的邮件服务器进行读取。
- 特点
 - 使用方便，传递迅速和费用低廉。
 - 现在电子邮件可传送文字信息，还可附上声音和图像。



常用的邮件服务器和客户端

- 常用邮件服务器（包括但不限于）
 - 国际：GMAIL；163，126，TOM
 - 单位：XMU，IEEE
- 常用邮件客户端（包括但不限于，非广告）
 - 国外：Outlook，Thunderbird
 - 国内：Foxmail，QQMail，尚邮
 - Android手机：K9Mail



内容纲要

1

电子邮件概述

2

电子邮件格式

3

邮件代理和协议

4

SMTP协议

5

MIME标准



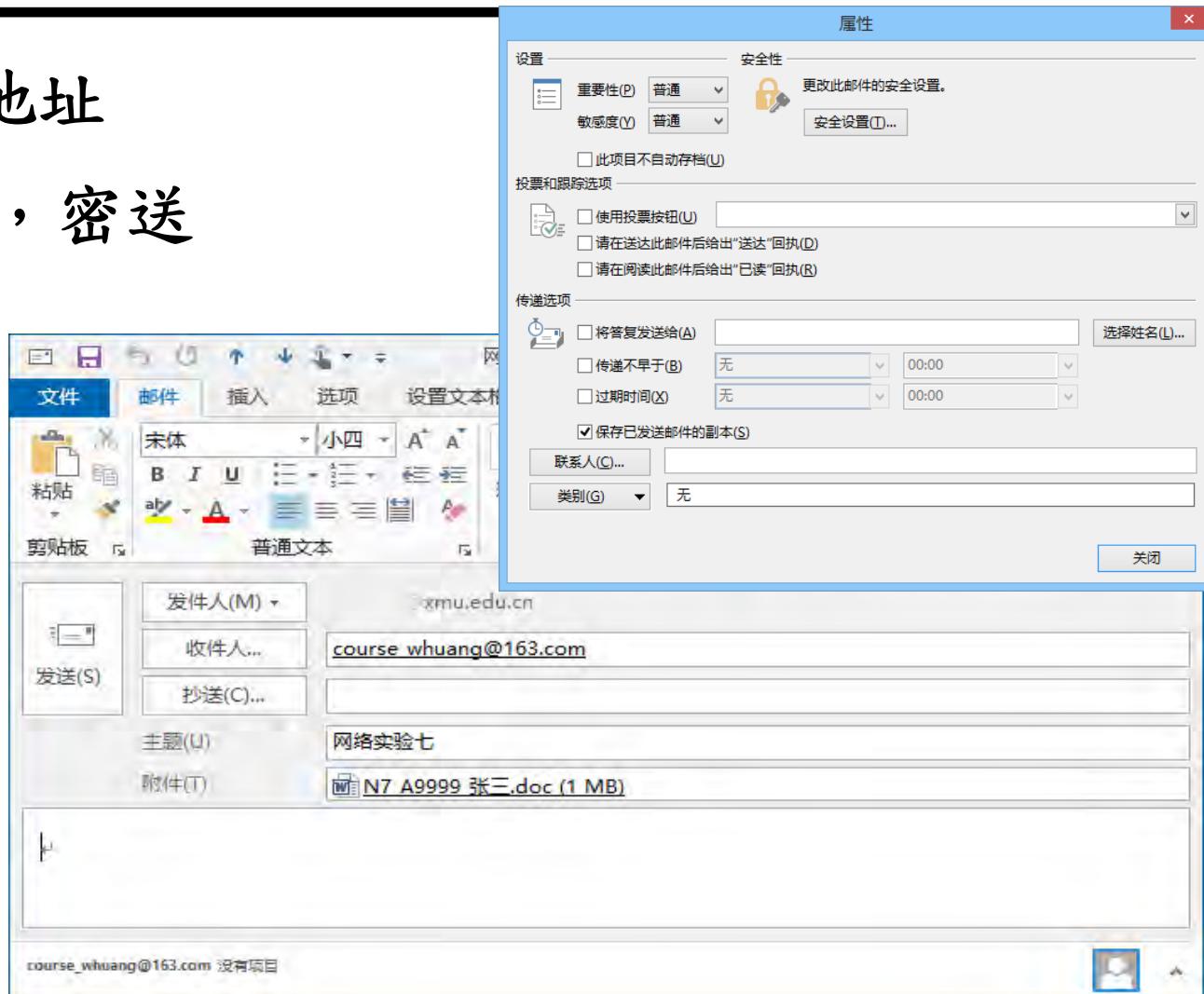
电子邮件地址

- 电子邮件系统要求每个用户有一个地址
 - 形式是：用户名@主机域名
 - 这里@念作“at”，意思为“在”。
 - 左侧为用户名，在该域名的范围内是唯一的。
 - 右侧为邮件服务器所在主机的域名，在全世界必须是唯一的。
 - 邮箱所在主机通常称为邮件服务器。

whuang@xmu.edu.cn

电子邮件的组成

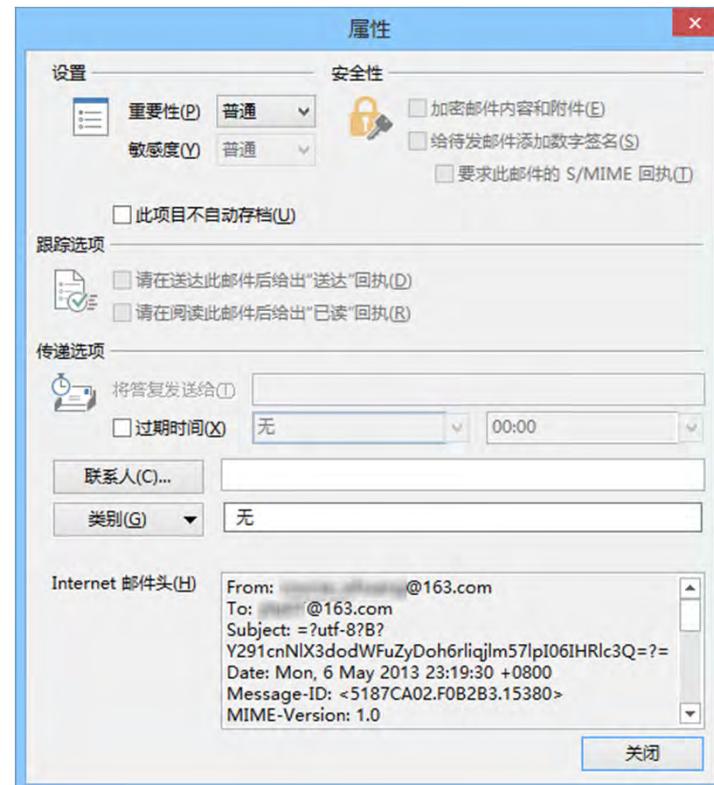
- 发件人，回复地址
- 收件人，抄送，密送
- 时间
- 主题，内容
- 附件
- 扩展属性
 - 是否要求回执
 - 重要性



查看Internet邮件头

- 如图，邮件头说明了什么？

```
From: *****@163.com
To: *****@163.com
Subject: =?utf-
8?B?Y291cnNlX3dodWFuZyDoh6rlqjlm57lpI06IHRlc3Q=?=
Date: Mon, 6 May 2013 23:19:30 +0800
Message-ID: <5187CA02.F0B2B3.15380>
MIME-Version: 1.0
Content-Type: Text/HTML;
charset="utf-8"
X-CM-TRANSID: EMCowEB5p3z_yYdRYrLXAg--.25871S2.re
X-CM-SenderInfo: xfrx22xbzx3xdqjqiywtou0bp/
Delivered-To: *****@163.com
Errors-to: *****@163.com
Return-path: *****@163.com
References: <000001ce4a6d$1c1482c0$543d****$@163.com>
In-Reply-To:
<000001ce4a6d$1c1482c0$543d****$@163.com>
Content-Transfer-Encoding: base64
```



内容纲要

1

电子邮件概述

2

电子邮件格式

3

邮件代理和协议

4

SMTP协议

5

MIME标准



三种代理

- 邮件用户代理（ Mail User Agent , MUA ）
 - 帮助用户读取、编写和回复邮件，再将这些信息转给MTA发送；有Outlook、Foxmail等。
- 邮件传输代理（ Mail Transport Agent , MTA ）
 - 把邮件由一个服务器传到另一个服务器或邮件投递代理。
- 邮件投递代理（ Mail Delivery Agent , MDA ）
 - 将MTA接收的邮件，根据收件人地址投放到用户的邮箱里。在投放过程中，还可以进行邮件过滤、自动回复等功能。

邮件协议

- 常见的电子邮件协议
 - SMTP (Simple Mail Transfer Protocol , 简单邮件传输协议)
 - POP3 (邮局协议)
 - IMAP4 (Internet邮件访问协议)
- 表示邮件的协议
 - MIME : 扩展了电子邮件标准，使其能支持非ASCII字符、二进制附件等多种格式的邮件消息。
- 上网访问邮箱则通过HTTP或HTTPS协议



电子邮件最主要的组成构件

- 用户代理
 - MUA, MTA, MDA

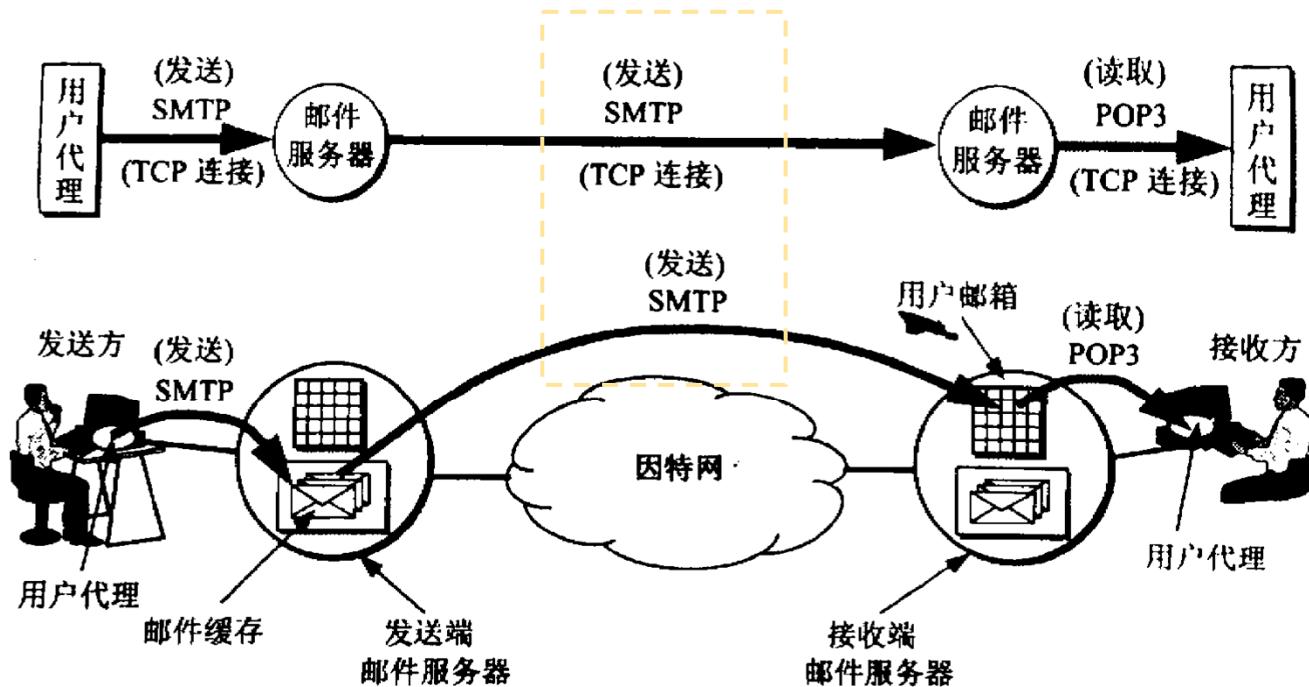


图 12-8 电子邮件的最主要的组成构件

电子邮件软件

- 电子邮件软件分为两个概念模块：
 - 电子邮件接口应用
 - 一种机制，为用户撰写和编辑传出的消息，以及读取和处理传入的电子邮件
 - 邮件传输程序
 - 作为客户端向目标计算机上的邮件服务器发送邮件；
 - 邮件服务器接受传入消息并将每个邮件存放在相应的用户邮箱中。
- 电子邮件的规格可分为三大类：传输、访问和表达

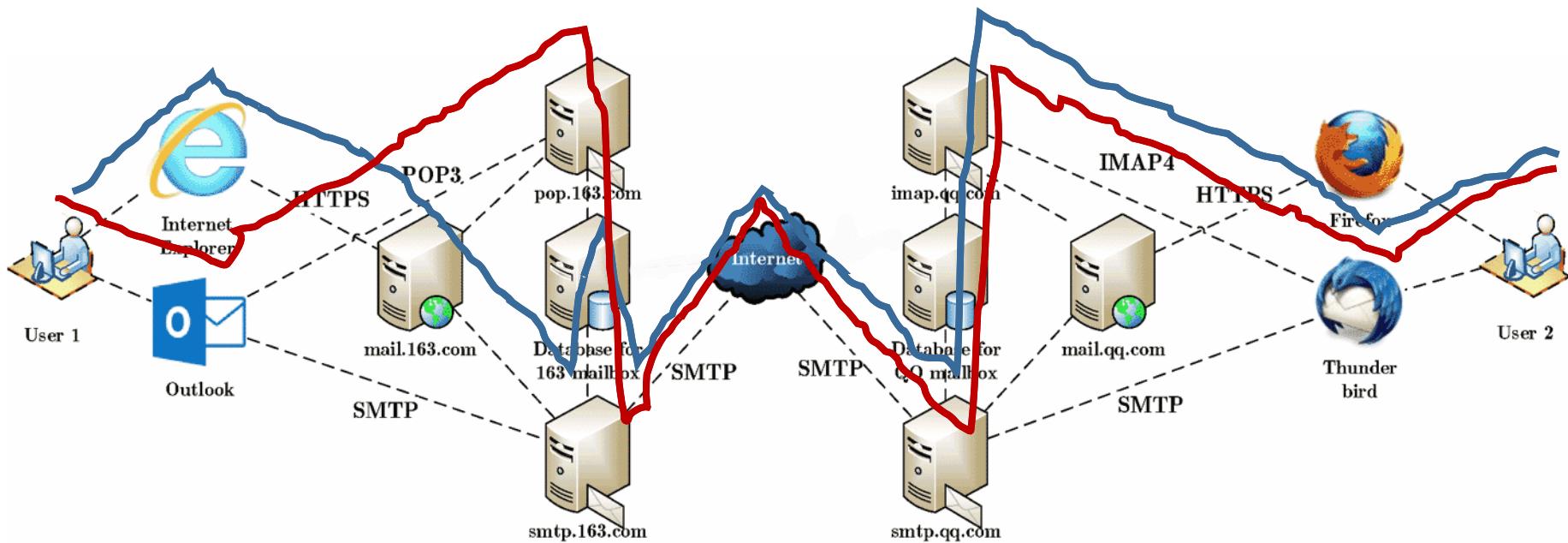


邮件服务器

- 邮件服务器需要使用发送和读取两个不同的协议
 - 功能：发送和接收邮件，同时还要向发信人报告邮件传送的情况（已交付、被拒绝、丢失等）。
- 一个邮件服务器既可以作为客户，也可以作为服务器。
 - 当邮件服务器 A 向另一个邮件服务器 B 发送邮件时，邮件服务器 A 就作为 SMTP 客户，而 B 是 SMTP 服务器。
 - 当邮件服务器 A 从另一个邮件服务器 B 接收邮件时，邮件服务器 A 就作为 SMTP 服务器，而 B 是 SMTP 客户。



电子邮件如何传输



发送和接收电子邮件的重要步骤

- 发件人调用用户代理撰写和编辑要发送的邮件。
- 发件人的用户代理把邮件用 SMTP 协议发给发送方邮件服务器，
- SMTP 服务器把邮件临时存放在邮件缓存队列中，等待发送。
- 发送方邮件服务器的 SMTP 客户与接收方邮件服务器的 SMTP 服务器建立 TCP 连接，然后就把邮件缓存队列中的邮件依次发送出去。



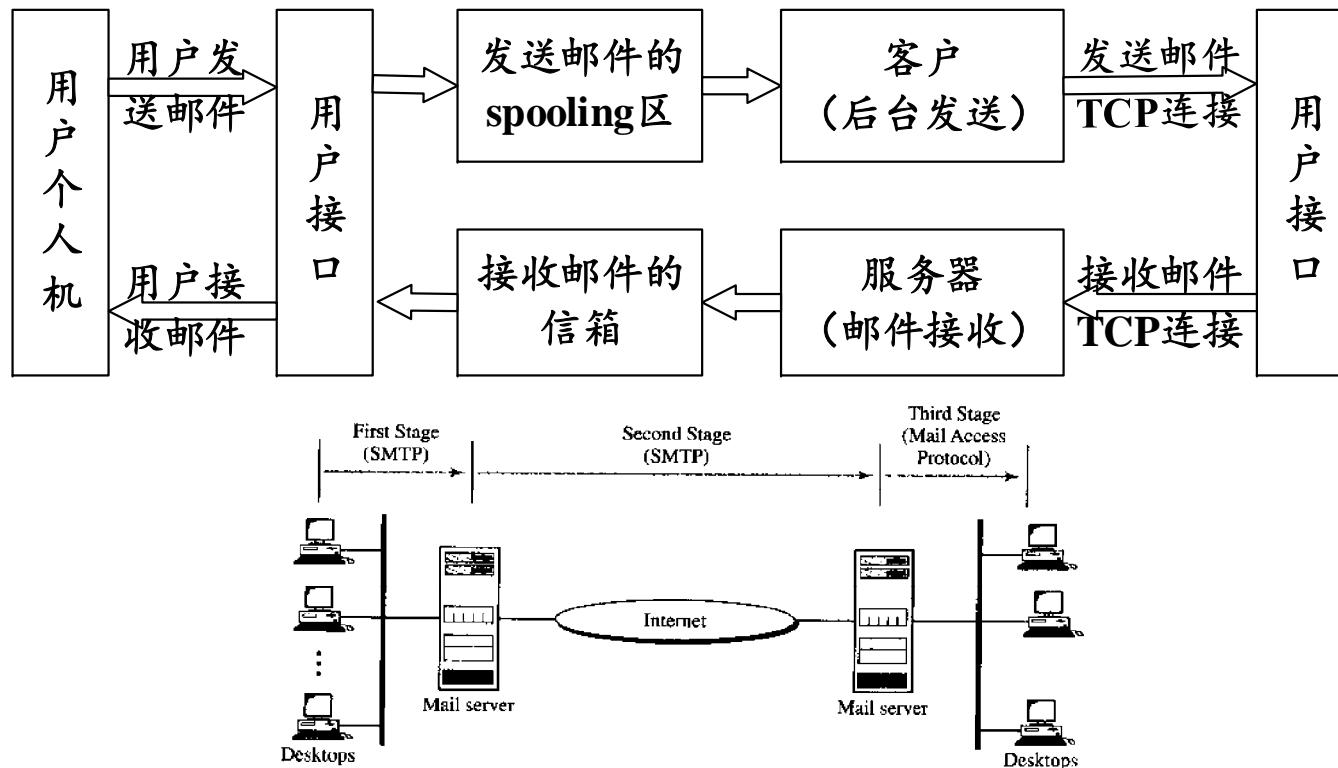
发送和接收电子邮件的重要步骤

- 运行在接收方邮件服务器中的SMTP服务器进程收到邮件后，把邮件放入收件人的用户邮箱中，等待收件人进行读取。
- 收件人在打算收信时，就运行PC机中的用户代理，使用POP3（或IMAP）协议读取发送给自己的邮件。
- 请注意，POP3服务器和POP3客户之间的通信是由POP3客户发起的。



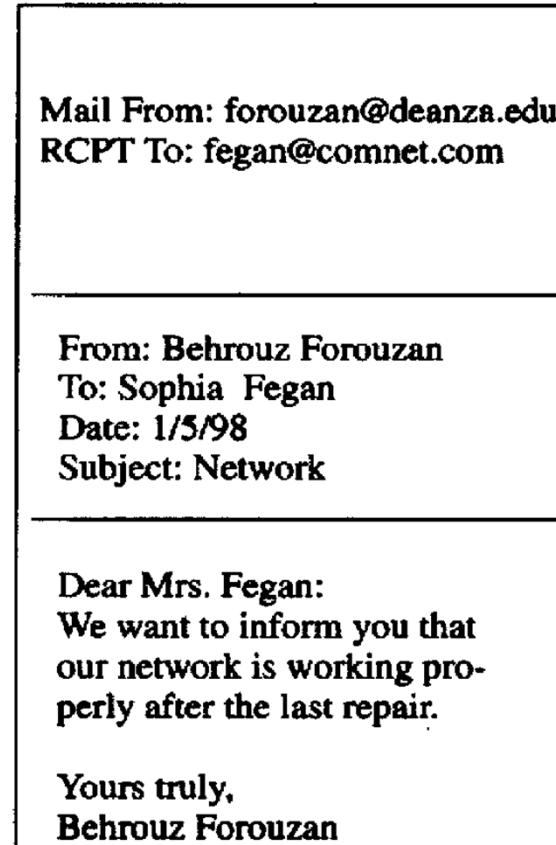
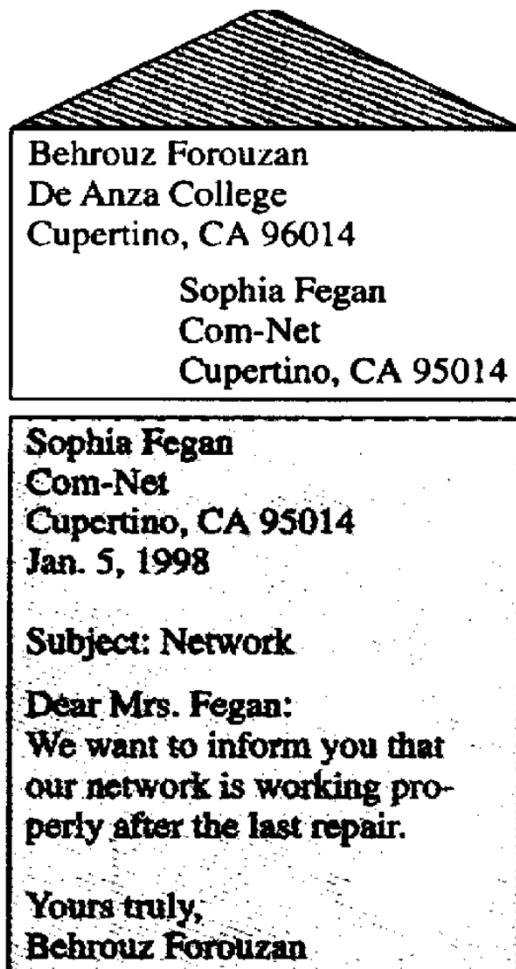
TCP/IP电子邮件系统的模型

- TCP/IP电子邮件系统采用端到端传输方式
 - 发送方的MTA负责将邮件传送到接收方的MTA



邮件格式

- 信封
- 消息
 - 邮件头
 - 邮件主体



内容纲要

2

电子邮件格式

3

邮件代理和协议

4

SMTP协议

5

MIME标准

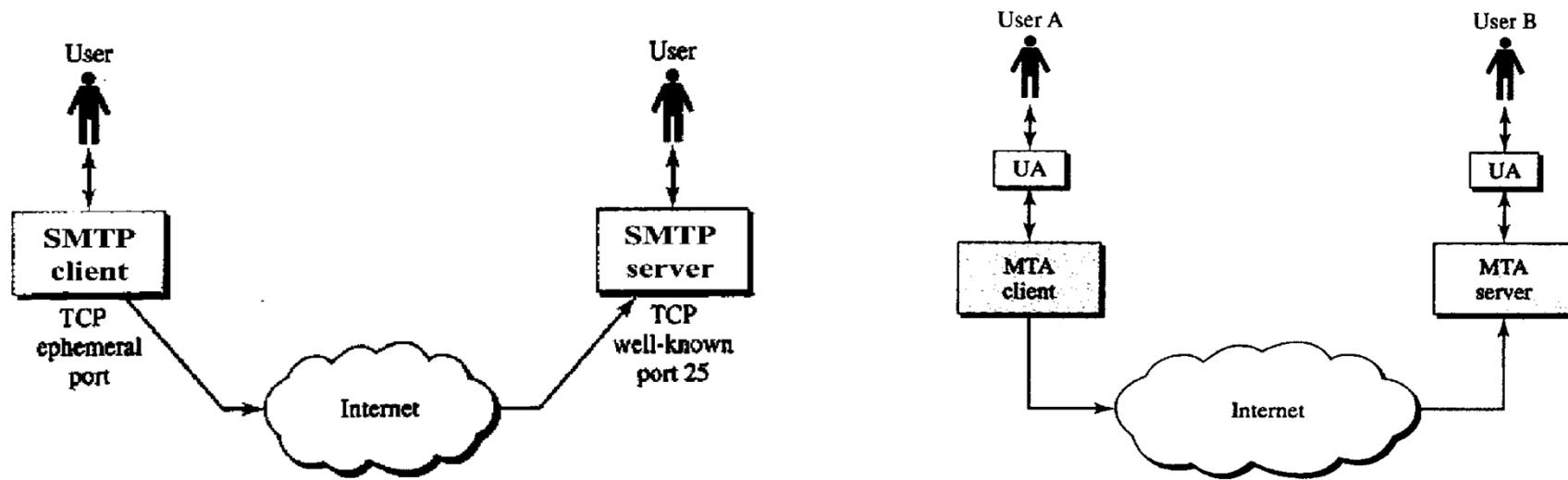
6

POP3协议



简单邮件传输协议（ SMTP ）

- 简单邮件传输协议（ Simple Mail Transfer Protocol ）
 - 邮件传输程序使用的标准协议，规定在两个相互通信的 SMTP 进程之间应如何交换信息。
 - 端口号： 25 (明文), 465 (SSL 加密)



SMTP

- SMTP可以被描述为

- 遵循流范式
- 使用文本控制消息
- 只传送文本消息
- 发送一个给定消息的副本
- 允许客户端列出用户，然后向列表中的所有用户发送消息的单个副本

SMTP通信的三个阶段

- **连接建立**

- 连接是在发送主机的 SMTP 客户和接收主机的 SMTP 服务
器之间建立的。
 - SMTP 不使用中间的邮件服务器。

- **邮件传送**

- 邮件接收后，收件人并不需要马上阅读

- **连接释放**

- 邮件发送完毕后，SMTP 应释放 TCP 连接。

SMTP

- 使用客户服务器方式

- 负责发送邮件的
SMTP 进程是客户
- 负责接收邮件的
**SMTP 进程是服
务器**

Server: 220 somewhere.com Simple Mail Transfer Service Ready
Client: HELO example.edu
Server: 250 OK

Client: MAIL FROM:<John_Q_Smith@example.edu>
Server: 250 OK

Client: RCPT TO:<Mathew_Doe@somewhere.com>
Server: 550 No such user here

Client: RCPT TO:<Paul_Jones@somewhere.com>
Server: 250 OK

Client: DATA
Server: 354 Start mail input; end with <CR><LF>.<CR><LF>
Client: ...sends body of mail message, which can contain
Client: ...arbitrarily many lines of text
Client: <CR><LF>.<CR><LF>
Server: 250 OK

Client: QUIT
Server: 221 somewhere.com closing transmission channel

SMTP命令列表

命令	描述
DATA	开始信息写作
EXPN<string>	验证给定的邮箱列表是否存在，扩充邮箱列表，也常被禁用
HELO<domain>	向服务器标识用户身份，返回邮件服务器身份
HELP<command>	查询服务器支持什么命令，返回命令中的信息
MAIL FROM<host>	在主机上初始化一个邮件会话
NOOP	无操作，服务器应响应OK
QUIT	终止邮件会话
RCPT TO<user>	标识单个的邮件接收人；常在MAIL命令后面可有多个rcpt to：
RSET	重置会话，当前传输被取消
SAML FROM<host>	发送邮件到用户终端和邮箱
SEND FROM<host>	发送邮件到用户终端
SOML FROM<host>	发送邮件到用户终端或邮箱
TURN	接收端和发送端交换角色
VRFY<user>	用于验证指定的用户/邮箱是否存在；由于安全方面的原因，服务器常禁止此命令

别名扩展

- 邮件转发器允许报文副本再发到多个目的地
 - 通常，服务器会查询一个小型的邮件别名数据库，把传入的收件方地址映射为一组地址 A，然后再向 A 中的每个地址转发一份副本。
 - 别名映射可以是“多对一”或“一对多”



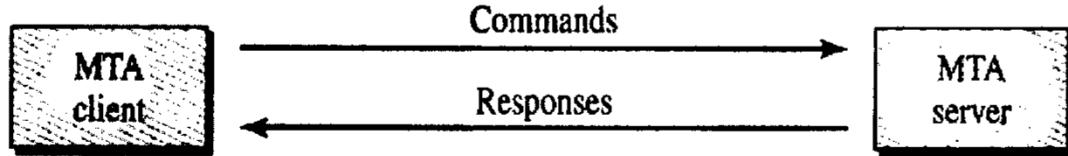
命令和响应

- 用命令和响应在MTA客户端和MTA服务器传输消息
- 命令格式：KEYWORD: INFO，命令或答复由回车结束
- 示例：

HELO: xyz.xmu.edu.cn

MAIL FROM: abc@xyz.xmu.edu.cn

RCPT TO: def@ghijk.com



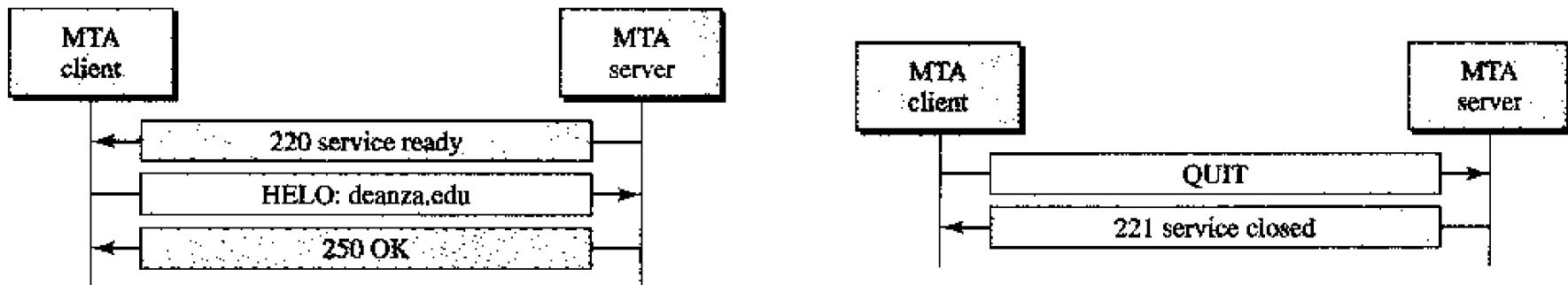
- 响应从服务器发送到客户端。

— 响应是三位数的代码，后面可能有额外的文本信息。

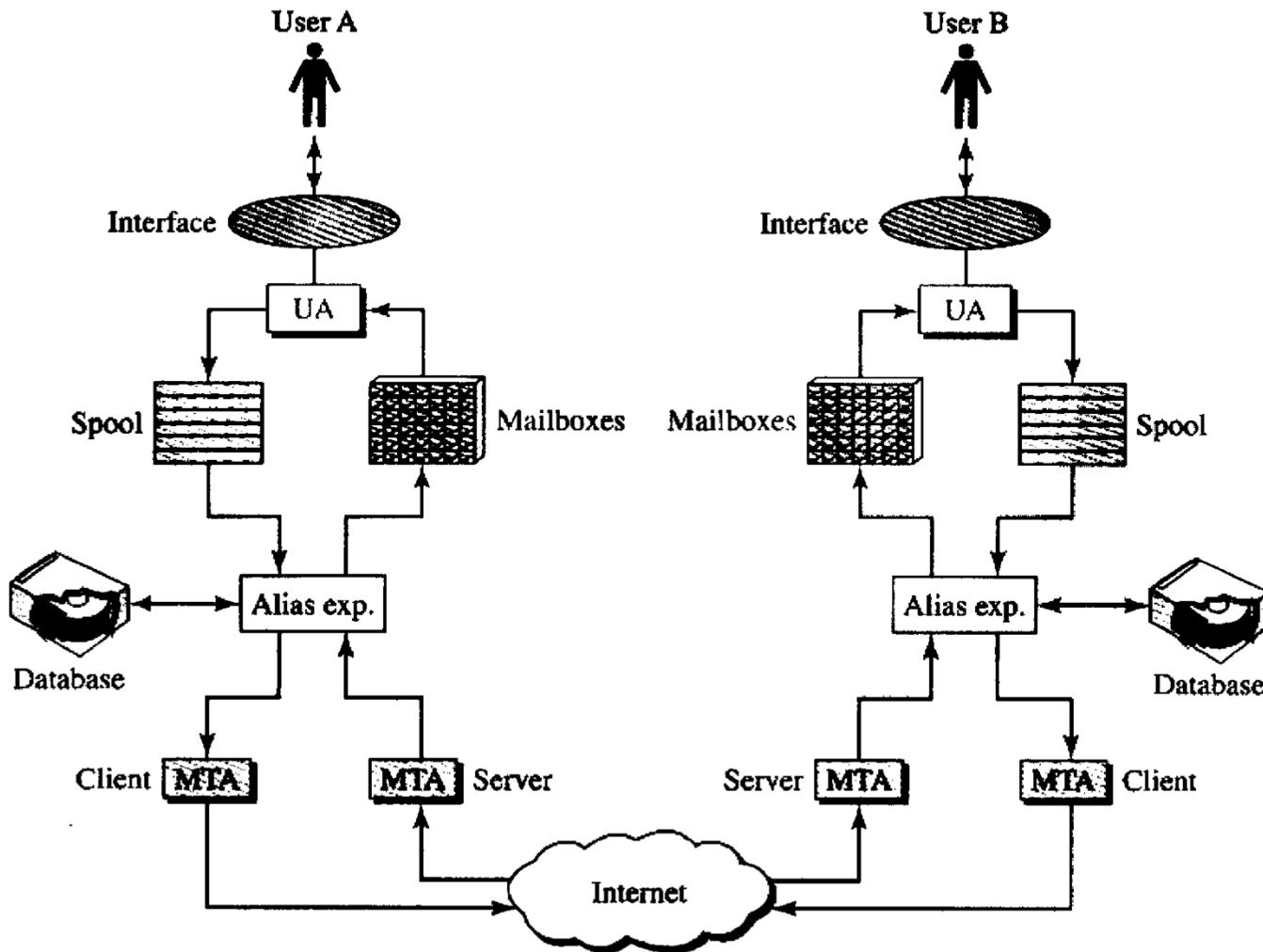
- 三个阶段：连接建立，邮件传输和连接终止。

邮件传输阶段

- 建立连接
 - 在客户端与已知端口25进行TCP连接后，SMTP服务器启动连接阶段。
- 连接终止
 - 消息成功传输后，客户端终止连接。



完整的邮件系统



内容纲要

3 邮件代理和协议

4 SMTP协议

5 MIME标准

6 POP3协议

7 IMAP4协议



多用途互联网邮件扩展（MIME）

- Multipurpose Internet Mail Extensions

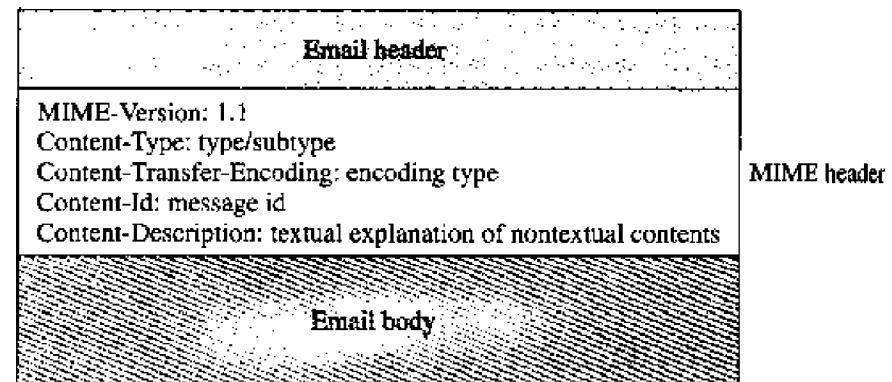
- MIME是允许SMTP发送的非ASCII数据的补充协议。
- 不是邮件协议，不能替代SMTP。
- 仅对SMTP扩展。在接收端服务器的SMTP接收NVT（网络虚拟终端）ASCII数据并将其传递到MIME被转换回原始数据。



MIME头

- 在原始SMTP头部部分添加五个标头以定义传输参数
 - **MIME-Version** 标识MIME的版本
 - **Content-Type** 说明邮件的性质
 - **Content-Transfer-Encoding** 主体是如何编码的
 - **Content-Id** 邮件的唯一标识符
 - **Content-Description**

内容描述



MIME头的参数值

- **MIME-Version:** 1.0
- **Content-Type:** [type]/[subtype]; parameter
 - **Text**：用于标准化地表示的文本信息，可以多种字符集；
 - **Multipart**：用于连接消息体的多个部分构成一个消息；
 - **Application**：用于传输应用程序数据或者二进制数据；
 - **Message**：用于包装一个E-mail消息；
 - **Image**：用于传输静态图片数据；
 - **Audio**：用于传输音频或者音声数据；
 - **Video**：用于传输动态影像数据，音视频。



MIME头的参数值

- Content-Type: [type]/[subtype]; parameter
 - multipart/mixed : 存在附件
 - multipart/related : 存在内嵌资源
 - multipart/alternative : 只有纯文本与超文本正文
 - application/xhtml+xml : XHTML文档
 - image/jpeg
- Content-Transfer-Encoding: [mechanism]
 - 7bit, 8bit, binary, quoted-printable, base64

MIME的编码

- MIME编码格式

Type	Description
7bit	NVT ASCII characters and short lines
8bit	Non-ASCII characters and short lines
binary	Non-ASCII characters with unlimited-length lines
Base64	6-bit blocks of data are encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters are encoded as an equal sign followed by an ASCII code

- 支持的文件类型

Table 22.3 Data types and subtypes in MIME

Type	Subtype	Description
Text	Plain	Unformatted text
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the same message
Message	RFC822	Body is an encapsulated message
	Partial	Body is a fragment of a bigger message
	External-Body	Body is a reference to another message
Image	JPEG	Image is in JPEG format
	GIF	Image is in GIF format
Video	MPEG	Video is in MPEG format
Audio	Basic	Single channel encoding of voice at 8 KHz
Application	PostScript	Adobe PostScript
	Octet-stream	General binary data (eight-bit bytes)



Base64 编码

- Base64

- 当最高比特不一定为零时发送字节数据的一种解决方案。
- 将数据转换为可打印字符。
- 过程将二进制数据分成24位块。
- 每个块分为4个部分，每一个由6位。
- 每一个6位部分解释为一个可打印字符。

Base64 编码过程

- 传递24bit : 01001001 00110001 00111001

- 先划分为4个6bit组，即：

010010 010011 000100 111001

(18:S) (19:T) (4:E) (57:5)

- 对应的base64编码为：STE5。其ASCII码为：

01010011 01010100 01000101 00110101

- 三个字符用四个字符传，适用于ASCII码不多的情况。

• 码表

- 0-25 : A-Z ; 26-51 : a-z ; 52-61 : 0-9 ;
 - 62 : + , 63 : /



MIME示例

From: “=?gb18030?****=?” <*****@***.com>
To: “=?gb18030?B?****3dodWFuZw==?=” <*****@***.com>
Subject: =?gb18030?B?u9i****do=?=
=?gb18030?B?****k=?=
Mime-Version: 1.0
Content-Type: multipart/mixed;
 boundary="-----_NextPart_*****" (**Mime_separator**)
Content-Transfer-Encoding: 8Bit
Date: Sat, 11 May 2013 13:16:33 +0800
X-Priority: 3
Message-ID: <*****@***.com>

This is a multi-part message in MIME format.
-----=_NextPart_*****
Content-Type: multipart/alternative;
 boundary="-----_NextPart_*****";
-----=_NextPart_*****
Content-Type: text/plain;
 charset="gb18030"
Content-Transfer-Encoding: base64

内容纲要

3 邮件代理和协议

4 SMTP协议

5 MIME标准

6 POP3协议

7 IMAP4协议



Email 访问

- 互联网服务供应商开始提供电子邮件服务
- 电子邮件访问如下两种形式之一
 - 专用电子邮件接口应用访问电子邮件
 - 网页的网络浏览器
- 访问协议与传输协议不同
 - 访问只涉及单个用户与单个邮箱交互
 - 传输协议允许用户向其他用户发送邮件



Email 访问

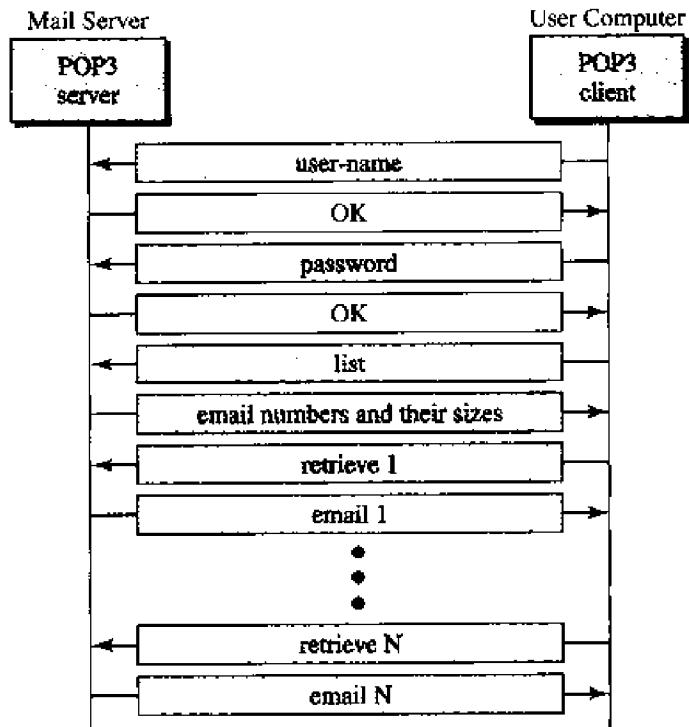
- 接入协议的特点
 - 提供对用户邮箱的访问
 - 允许用户查看标头、下载、删除或发送消息
 - 客户端运行在用户的个人计算机上
 - 服务器运行在存储用户邮箱的计算机上
- 查看没有下载邮件内容的邮件列表
 - 特别是在双方连接缓慢的情况下
 - 例：手机用户可查看头部和删除垃圾邮件而不必下载内容



邮局协议3（POP3）

- Post Office Protocol (POP)

- POP3允许用户通过PC机动态地检索邮件服务器上的邮件
- 下载邮件，删除邮件
- 端口号：110（明文），
995（POP3S，with SSL）



POP3收邮件过程

- POP3服务器向客户发送一行欢迎词，进入授权状态。
- 授权状态
 - 客户发送USER命令给出用户在邮件服务器上的邮箱名，若是合法用户，服务器回答“+OK”。
 - 客户再发送PASS命令给出口令。POP3服务器确定用户是否有权访问该邮箱，若有权访问，服务器再次回答“+OK”，若非法用户，服务器回答“-ERR”。
 - USER和PASS命令用口令方式对用户进行授权验证。



POP3收邮件过程

- 事务状态

- 若对用户的授权验证成功，则服务器申请资源与用户的邮箱关联，进入事务(transaction)状态
 - 在事务状态，服务器将存储的邮件，从1开始编号；客户可发送命令检索(RETRO)、删除 (DELE)(作删除标记)项目等。

- 更新状态

- 客户发QUIT命令，进更新(update)状态，删去标记的邮件
 - 关闭TCP连接，服务器释放资源，POP3会话结束



POP3 会话实例

```
+OK Welcome to coremail Mail Pop3 Server (163coms[...  
AUTH  
-ERR Not support ntlm auth method  
CAPA  
+OK Capability list follows TOP USER PIPELINING...  
USER y****7  
+OK core mail  
PASS *****  
+OK 7 message(s) [231904 byte(s)]  
STAT  
+OK 7 231904  
LIST  
+OK 7 231904 1 9854 2 140937 3 10224 4 17139 ...  
UIDL  
+OK 7 231904 1 1tbiMAkaCFEAER4pHgAAsC 2 1tbiNQUc...  
RETR 7  
+OK 1698 octets  
Received: from [192.168.7.128] (unknown [119.233.1...  
f0uMueDze6FWdWBsy3zL/K1nBCoAwr8FFrAhNS4gN8AhsIy++3...  
QUIT  
+OK core mail
```

POP3 会话实例

Received: from [192.168.7.128] (unknown [119.233.192.167])
by smtp13 (Coremail) with SMTP id *****fVdw9pFRM59yBA--.***S2;
Tue, 14 May 2013 16:31:45 +0800 (CST)
X-Coremail-DSSMTP: 119.233.192.167
Message-ID: <5***F670.5*****@163.com>
Date: Tue, 14 May 2013 16:31:44 +0800
From: W** H**** <*****@163.com>
User-Agent: Mozilla/5.0 (Windows NT 5.1; rv:17.0) Gecko/20130328 Thunderbird/17.0.5
MIME-Version: 1.0
To: *****@163.com
Subject: Test a rar attach
Content-Type: multipart/mixed;
boundary="-----*****20706"
X-CM-TRANSID:*****fVdw9pFRM59yBA--.***S2
X-Coremail-Antispam: *****9EdanIXcx71UUUUU7v73
***** YCTnIWIEvJa73UjIFyTuYvjxU4mFCUUUUU
X-CM-SenderInfo: ***** biJRIIdCE9o8Tv-YAAAsQ

This is a multi-part message in MIME format.

-----*****20706

Content-Type: text/plain; charset=GB2312



厦门大学
XIAMEN UNIVERSITY



信息学院 黄 烽
(国家示范性软件学院) 博士/副教授
School of Informatics Dr. Wei Huang

POP3 会话实例

Content-Transfer-Encoding: 7bit

-----*****20706

Content-Type: *****;

name="result.rar"

Content-Transfer-Encoding: base64

Content-Disposition: attachment;

filename="result.rar"

*****AACOZRtLPytq0IdNQoAIAAAAHJ1
c3VsdC50eHQAs0hgB*****dNHhnB4bx0x86eD5
I+CoS7d*****V4H8e3v5a/lubX52V88wwBSxn5eKz
0sy50mimnT26*****0ab8/VX1YLQJruTj/5jjHmqqKkguvW5kUICb20jP3
VdHj1Ij*****++3MGE3QrE90wHD
R6Ay*****1qVqWp5y9cTE5eQATF*****UhPkYvygn
S5gHFg*****A==

-----*****20706--



内容纲要

3 邮件代理和协议

4 SMTP协议

5 MIME标准

6 POP3协议

7 IMAP4协议



因特网邮件访问协议4（IMAP4）

- Internet Mail Access Protocol (IMAP)
 - 三种工作模式：离线、在线和断连方式。
 - 端口号：143（明文），993（IMAPS, with SSL）
 - 先身份验证，鉴定登录名和口令，后获得访问权
- IMAP协议适合使用多台计算机的用户
 - 让邮件服务器维护一个中心数据库，能够被多台机器访问
 - 不允许用户将邮件下载到自己的计算机上，只能在线访问邮箱，但可以只读邮件的某一部分



IMAP与POP3 的区别

- IMAP4只下载邮件的主题，并不下载内容
- 邮件客户端软件阅读邮件时才下载邮件内容
- 支持维护自己在服务器上的邮件目录
- 支持直接抓取邮件的特定部分（如文本）
- POP3是“脱机”协议，IMAP是联机协议



垃圾邮件（SPAM）

- 不请自来、强行塞入信箱的垃圾邮件
- SPAM的主要特性包括：
 - 未经消费者的同意，与消费者需求不相关
 - 以诈欺的方式骗取邮件地址
 - 攻击性的广告：夸张不实、不健康、钓鱼网站
 - 散布的数量庞大
- 阻拦垃圾邮件的方法
 - 发信来源，关键词、特征匹配



选作作业

- 下载邮件服务器软件，以localhost为域名，新建 admin@localhost 邮箱，尝试搭建邮件服务器，并用邮件客户端下载
 - Ipswitch IMail Server，WinMail
- 探究邮件服务器和客户端软件各有什么功能
- 用OmniPeek监听收发邮件的数据流



计算机网络

Computer Network

16

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

17

文件传输

理论课程

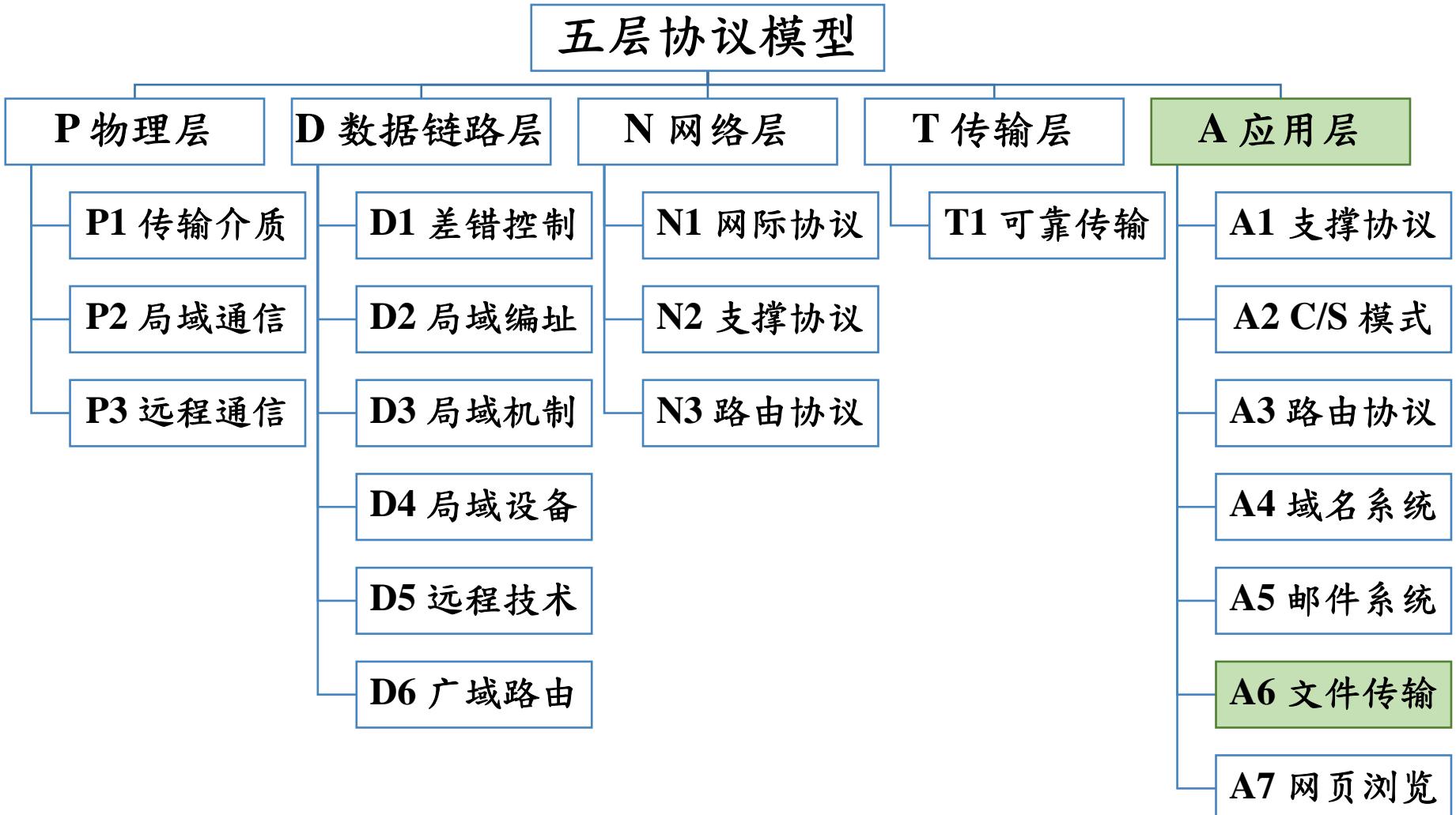


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- FTP

- 工作原理与通信模式
- 主动和被动工作模式

对应课本章节

- PART I Introduction And Internet Applications
 - Chapter 4 Traditional Internet Applications
 - 4.10~4.11 File Transfer Protocol (FTP); FTP Communication Paradigm

内容纲要

1

FTP协议

2

TFTP协议

3

NFS协议

4

SMB协议

5

Telnet协议



FTP (File Transfer Protocol)

- 文件传送协议 FTP (File Transfer Protocol)
 - 提供交互式的访问，允许客户指明文件的类型与格式，并允许文件具有存取权限。
 - 屏蔽了各计算机系统的细节，因而适合于在异构网络中任意计算机之间传送文件。
 - RFC 959 很早就成为了因特网的正式标准。
- 文件传送并非很简单的问题
 - 众多的计算机厂商研制出的文件系统数百种，且差别很大。

FTP 特点

- FTP 使用基于流的客户服务器方式。
 - 一个 FTP 服务器进程可同时为多个客户进程提供服务。
 - FTP 的服务器进程由两大部分组成：
 - 一个主进程，负责接受新的请求；
 - 另外有若干个从属进程，负责处理单个请求。



FTP 两个不同的端口号

- FTP 使用2个端口号，数据连接与控制连接不会混乱。
 - 传输文件还可利用控制连接（如：客户发送请求终止传输）
 - 端口**21**：控制链接
 - 当客户软件向服务器软件发出建立连接请求时，寻找连接服务器软件的**21号**端口
 - 同时还告诉服务器软件自己的另一端口号，用于建立数据传送连接。
 - 端口**20**：数据连接
 - 接着，服务器软件用自己传送数据的**20号**端口与客户软件所提供的端口号码建立数据传送连接。



FTP两个连接

- 控制连接

- 在整个会话期间一直保持打开，客户发出的传送请求通过控制连接发送给服务器端的控制进程，但不用来传送文件。

- 数据连接

- 实际用于传输文件。服务器端的控制进程在接收到 FTP 客户发送来的文件传输请求后创建“数据传送进程”和“数据连接”，用来连接客户端和服务器端的数据传送进程。
 - 数据传送进程实际完成文件的传送，在传送完毕后关闭“数据传送连接”并结束运行。



FTP文件类型

- ASCII 文件
 - 这是用于传输文本文件的默认格式。NVT ASCII 编码。
- EBCDIC文件（扩充的二—十进制交换码）
 - Extended Binary-Coded Decimal Interchange Code.
- Image文件
 - 这是用于传输二进制文件的默认格式。文件被发送为连续的比特流，没有任何解释或编码。
 - 这主要是用来传输二进制文件，如编译过的程序。



数据结构

- 文件结构（默认）

- 它是一个连续的字节流。

- 记录结构

- 文件划分为记录。只用于文本文件。

- 页面结构

- 文件分为两页，每一页有一个页和一个页头。

- 该页可以随机或顺序存储或访问。

传输模式

- 流模式（默认）
 - 数据传送从FTP的TCP作为连续的字节流。
- 分块模式
 - 数据可以按分块从FTP到TCP分发。每块前有3字节的头。
- 压缩模式
 - 如果文件大，可以压缩数据。所使用的压缩方法通常是游程长度编码。在二进制文件中，空字符通常被压缩。



命令行处理

- FTP使用控制连接在客户机控制进程和服务器控制进程之间建立通信。



FTP 命令

- 访问命令：这些命令让用户访问远程系统。

- 文件管理命令

Table 20.1 Access commands

Command	Argument(s)	Description
USER	User id	User information
PASS	User password	Password
ACCT	Account to be charged	Account information
REIN		Reinitialize
QUIT		Log out of the system
ABOR		Abort the previous command

Table 20.2 File management commands

Command	Argument(s)	Description
CWD	Directory name	Change to another directory
CDUP		Change to the parent directory
DELE	File name	Delete a file
LIST	Directory name	List subdirectories or files
NLIST	Directory name	List the names of subdirectories or files without other attributes
MKD	Directory name	Create a new directory

Table 20.2 File management commands (continued)

Command	Argument(s)	Description
PWD		Display name of current directory
RMD	Directory name	Delete a directory
RNFR	File name (old file name)	Identify a file to be renamed
RNTO	File name (new file name)	Rename the file
SMNT	File system name	Mount a file system



FTP 命令

- 数据格式命令

Table 20.3 *Data formatting commands*

Command	Argument(s)	Description
TYPE	A (ASCII), E (EBCDIC), I (Image), N (Nonprint), or T (TELNET)	Define the file type and if necessary the print format
STRU	F (File), R (Record), or P (Page)	Define the organization of the data
MODE	S (Stream), B (Block), or C (Compressed)	Define the transmission mode

- 端口定义命令

- 用户定义数据连接的客户端的端口号

Table 20.4 *Port defining commands*

Command	Argument(s)	Description
PORT	6-digit identifier	Client chooses a port
PASV		Server chooses a port

FTP 命令

• 文件传输命令

Table 20.5 *File transfer commands*

Command	Argument(s)	Description
RETR	File name(s)	Retrieve files; file(s) are transferred from server to the client
STOR	File name(s)	Store files; file(s) are transferred from the client to the server
APPE	File name(s)	Similar to STOR except if the file exists, data must be appended to it
STOU	File name(s)	Same as STOR except that the file name will be unique in the directory; however, the existing file should not be overwritten
ALLO	File name(s)	Allocate storage space for the files at the server
REST	File name(s)	Position the file marker at a specified data point
STAT	File name(s)	Return the status of files



FTP 响应

• 响应代码

Table 20.7 Responses

Code	Description
Positive Preliminary Reply	
120	Service will be ready shortly
125	Data connection open; data transfer will start shortly
150	File status is OK; data connection will be open shortly
Positive Completion Reply	
200	Command OK
211	System status or help reply
212	Directory status
213	File status
214	Help message
215	Naming the system type (operating system)
220	Service ready
221	Service closing
225	Data connection open
226	Closing data connection
227	Entering passive mode; server sends its IP address and port number
230	User login OK
250	Request file action OK

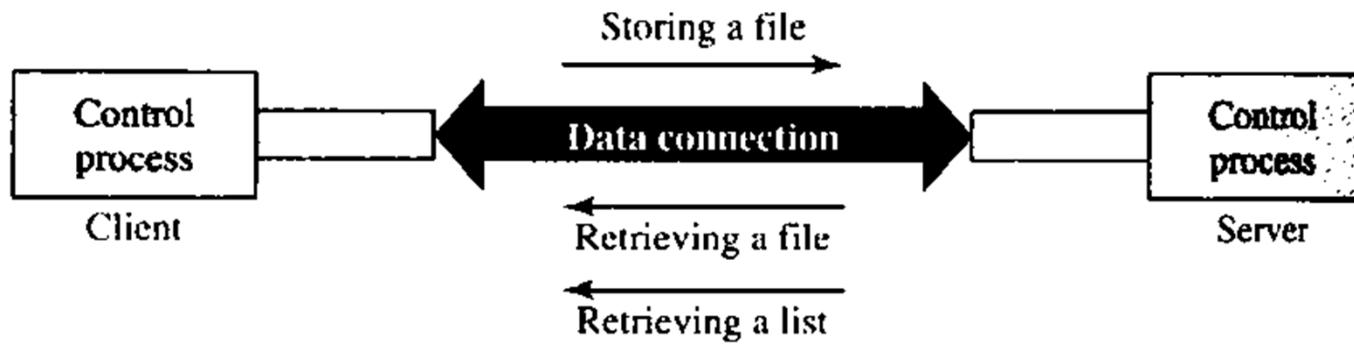
Table 20.7 Responses (continued)

Code	Description
Positive Intermediate Reply	
331	User name OK; password is needed
332	Need account for logging
350	The file action is pending; more information needed
Transient Negative Completion Reply	
425	Cannot open data connection
426	Connection closed; transfer aborted
450	File action not taken; file not available
451	Action aborted; local error
452	Action aborted; insufficient storage
Permanent Negative Completion Reply	
500	Syntax error; unrecognized command
501	Syntax error in parameters or arguments
502	Command not implemented
503	Bad sequence of commands
504	Command parameter not implemented
530	User not logged in
532	Need account for storing file
550	Action is not done; file unavailable
552	Requested action aborted; exceeded storage allocation
553	Requested action not taken; file name not allowed



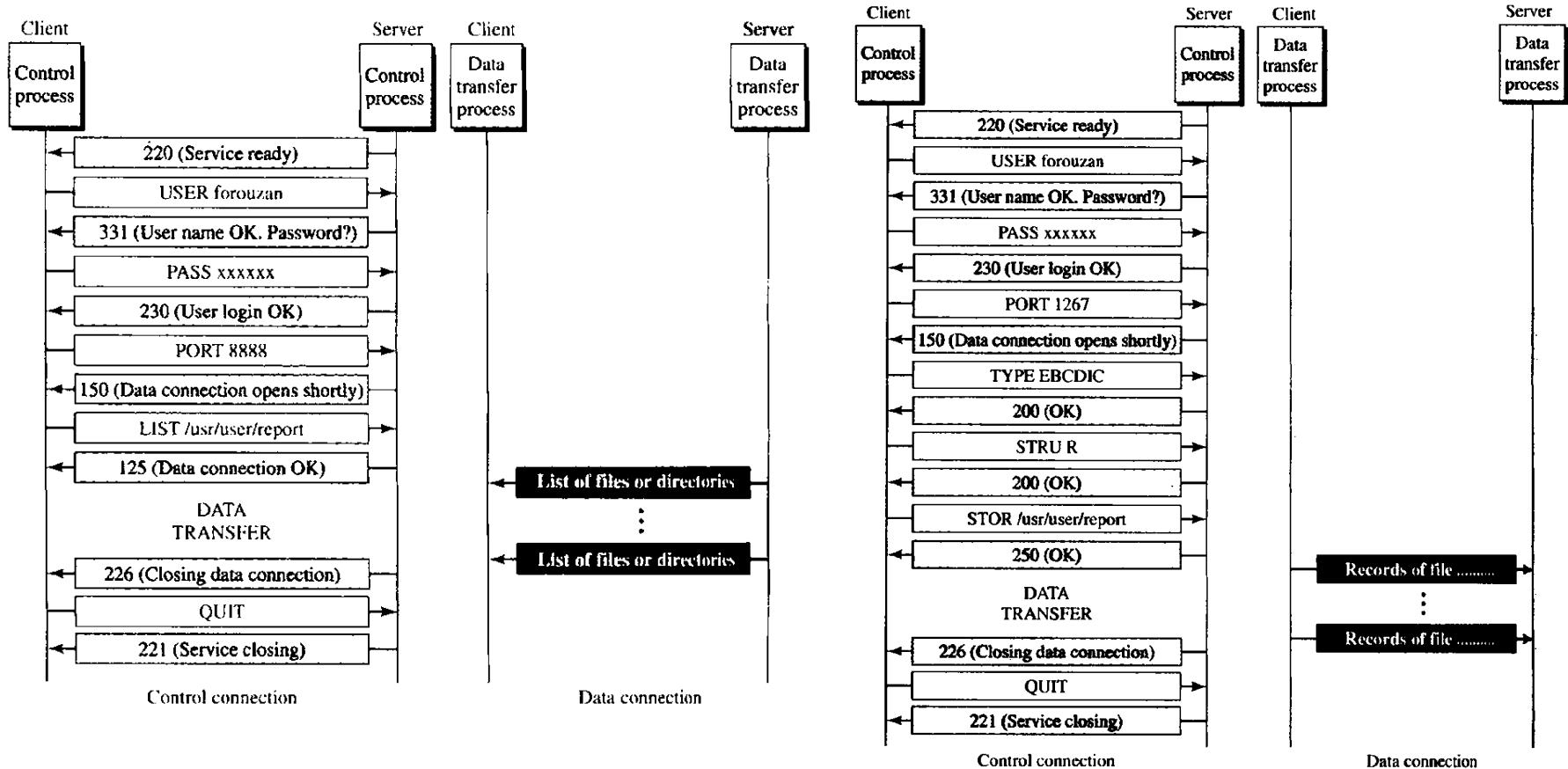
文件传输

- 在控制连接上设置的命令的控制下，在数据连接上发生文件传输。



范例

• CS交互范例



内容纲要

1

FTP协议

2

TFTP协议

3

NFS协议

4

SMB协议

5

Telnet协议



简单文件传输协议（TFTP）

- Trivial File Transfer Protocol（TFTP）
 - TFTP 使用 UDP 服务的端口 69.
- TFTP 是一种简化的 TCP/IP 文件传输协议。
- TFTP 只限于简单文件传输操作，它不提供权限控制，也不支持客户与服务器之间复杂的交互过程，没有庞大的命令集，没有列目录的功能，不能鉴别用户身份。因此 TFTP 软件比 FTP 软件小的多。



TFTP

- TFTP 是一个很小且易于实现的文件传送协议。
- TFTP 使用客户服务器方式和使用 UDP 数据报，因此 TFTP 需要有自己的差错改正措施。
- TFTP 只支持文件传输而不支持交互。
- TFTP 没有一个庞大的命令集，没有列目录的功能，也不能对用户进行身份鉴别。



TFTP 的主要特点

- 每次传送的数据 PDU 中有 512 字节的数据，但最后一次可不足 512 字节。
- 数据 PDU 也称为文件块(block)，每个块按序编号，从 1 开始。
- 支持 ASCII 码或二进制传送。
- 可对文件进行读或写。
- 使用很简单的首部。



TFTP 的工作类似停止等待协议

- 发送完一个文件块后就等待对方的确认，确认时应指明所确认的块编号。
- 发完数据后在规定时间内收不到确认就要重发数据 PDU。
- 发送确认 PDU 的一方若在规定时间内收不到下一个文件块，也要重发确认 PDU。这样就可保证文件的传送不致因某一个数据报的丢失而告失败。



TFTP 的工作类似停止等待协议

- 在一开始工作时。TFTP 客户进程发送一个读请求 PDU 或写请求 PDU 给 TFTP 服务器进程，其熟知端口号号码为 69。
- TFTP 服务器进程要选择一个新的端口和 TFTP 客户进程进行通信。
- 若文件长度恰为 512 字节的整数倍，则文件传送完毕后还必须最后发送一个只含首部而无数据的数据 PDU。
- 若文件长度不是 512 字节的整数倍，则最后传送数据 PDU 的数据字段一定不满 512 字节，这正好可作为文件结束的标志。



内容纲要

1 **FTP协议**

2 **TFTP协议**

3 **NFS协议**

4 **SMB协议**

5 **Telnet协议**



网络文件系统（ NFS ）

- 网络文件系统（ Network File System ， NFS ）
- NFS只是一种文件系统，本身没有传输功能，是基于 RPC协议实现的，才能达到两个Linux系统之间的文件目录共享；



Network File System

- NFS 允许应用进程打开一个远程文件，并能在该文件的某一个特定的位置上开始读写数据。
- NFS 可使用户只复制一个大文件中的一个很小的片段，而不需要复制整个大文件。
- 例：计算机 A 的 NFS 客户软件，把要添加的数据和在文件后面写数据的请求一起发送到计算机 B 的 NFS 服务器。NFS 服务器更新文件后返回应答信息。
- 在网络上传送的只是少量的修改数据。



内容纲要

1 **FTP协议**

2 **TFTP协议**

3 **NFS协议**

4 **SMB协议**

5 **Telnet协议**



简单文件共享：SMB协议

- Server Message Block
- 基于TCP-NETBIOS下的，一般端口为139、445。
 - NetBIOS（网络基本输入/输出系统协议）协议是由IBM公司开发，主要用于数十台计算机的小型局域网。
 - NetBIOS协议是一种在局域网上的程序可以使用的API，为程序提供了请求低级服务的统一的命令集
 - 几乎所有的局域网都是在NetBIOS协议的基础上工作的。

远程终端协议 Telnet

- Telnet 是一个简单的远程终端协议。
- 用户用 Telnet 就可在其所在地通过 TCP 连接注册（即登录）到远程的另一个主机上（使用主机名或 IP 地址）。
- Telnet 能将用户的击键传到远程主机，同时也能将远程主机的输出通过 TCP 连接返回到用户屏幕。这种服务是透明的，因为用户感觉到好像键盘和显示器是直接连在远程主机上。



内容纲要

1

FTP协议

2

TFTP协议

3

NFS协议

4

SMB协议

5

Telnet协议



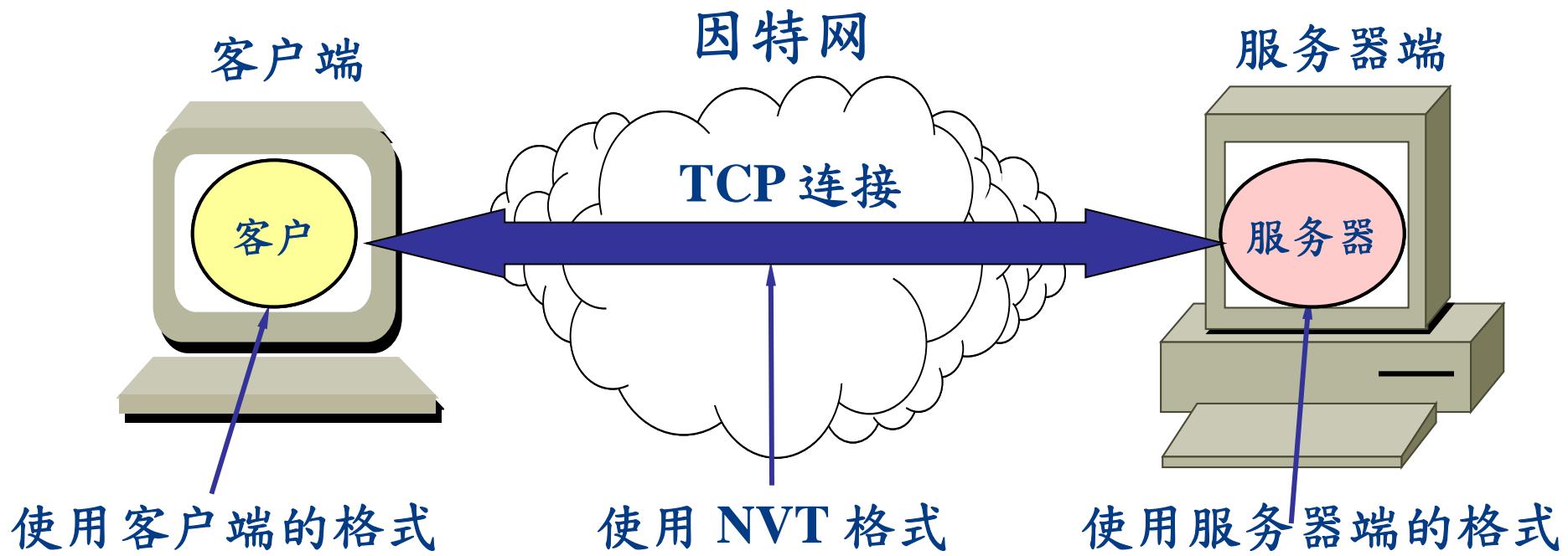
客户-服务器方式

- 现在由于 PC 的功能越来越强，用户已较少使用 Telnet 了。（但在CUI界面系统仍使用）
- Telnet 也使用客户-服务器方式。在本地系统运行 Telnet 客户进程，而在远程主机则运行 Telnet 服务器进程。
- 和 FTP 的情况相似，服务器中的主进程等待新的请求，并产生从属进程来处理每一个连接。



Telnet 使用 NVT 格式

- 网络虚拟终端 NVT



网络虚拟终端 NVT 格式

- 客户软件把用户的击键和命令转换成 NVT 格式，并送交服务器。
- 服务器软件把收到的数据和命令，从 NVT 格式转换成远程系统所需的格式。
- 向用户返回数据时，服务器把远程系统的格式转换为 NVT 格式，本地客户再从 NVT 格式转换到本地系统所需的格式。



计算机网络

Computer Network

17

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

18

万维网 和高级专题

理论课程

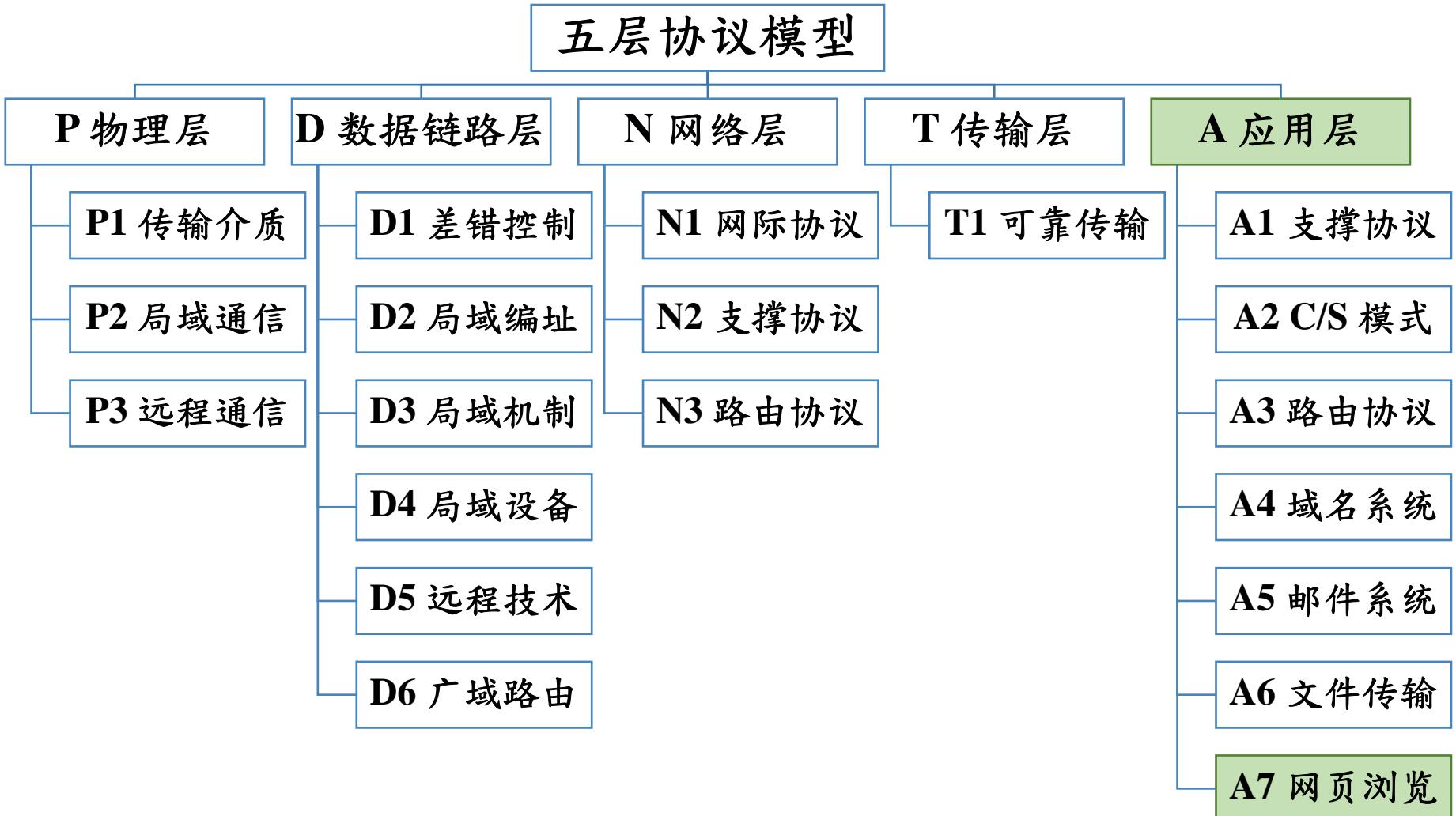


厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架



主要内容

- **HTTP**

- 工作原理与过程
- 错误代码
- URL
- HTML文档



对应课本章节

- PART I Introduction And Internet Applications
 - Chapter 4 Traditional Internet Applications
 - 4.3~4.9 Representation And Transfer; Web Protocols; Document Representation With HTML; Uniform Resource Locators And Hyperlinks; Web Document Transfer With HTTP; Caching In Browsers; Browser Architecture



内容纲要

1 万维网

2 HTTP协议

3 网络结构设计

4 网络技术展望

5 网络技术和应用趋势

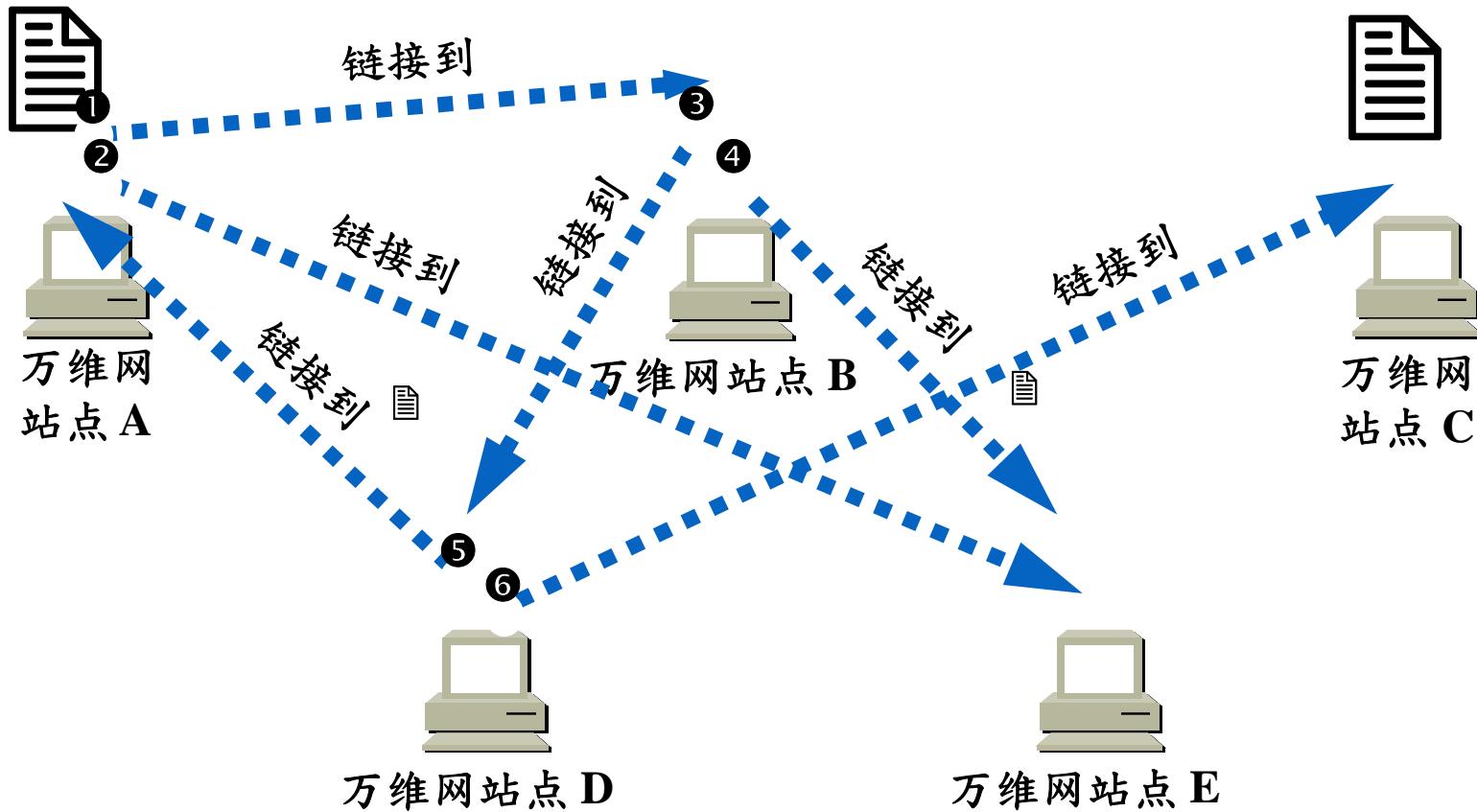


万维网概述

- 万维网 WWW (World Wide Web) 并非某种特殊的计算机网络。
- 万维网是一个大规模的、联机式的信息储藏所。
- 万维网用链接的方法能非常方便地从因特网上的一个站点访问另一个站点，从而主动地按需获取丰富的信息。
- 这种访问方式称为“链接”。



万维网提供分布式服务



超媒体与超文本

- 万维网是分布式超媒体(hypermedia)系统，它是超文本(hypertext)系统的扩充。
 - 一个超文本由多个信息源链接成。利用一个链接可使用户找到另一个文档。这些文档可以位于世界上任何一个接在因特网上的超文本系统中。超文本是万维网的基础。
 - 超媒体与超文本的区别是文档内容不同。超文本文档仅包含文本信息，而超媒体文档还包含其他表示方式的信息，如图形、图像、声音、动画，甚至活动视频图像。



万维网的工作方式

- 万维网以客户-服务器方式工作。
- 浏览器就是在用户计算机上的万维网客户程序。万维网文档所驻留的计算机则运行服务器程序，因此这个计算机也称为万维网服务器。
- 客户程序向服务器程序发出请求，服务器程序向客户程序送回客户所要的万维网文档。
- 在一个客户程序主窗口上显示出的万维网文档称为页面(page)。



万维网必须解决的问题

- 怎样标识分布在整个因特网上的万维网文档？
 - 使用统一资源定位符 **URL (Uniform Resource Locator)** 来标识万维网上的各种文档。
 - 使每个文档在整个因特网的范围内具有唯一的标识符 **URL**。
- 用何协议实现万维网上各种超链的链接？
 - 在万维网客户程序与万维网服务器程序之间进行交互所使用的协议，是超文本传送协议 **HTTP**。
 - **HTTP** 是一个应用层协议，使用 **TCP** 连接进行可靠的传送。



万维网必须解决的问题

- 怎样使各种文档在各种计算机显示并找到超链位置？
 - 超文本标记语言 **HTML (Hyper-Text Markup Language)** 使得万维网页面的设计者可以很方便地用一个超链从本页面的某处链接到因特网上的任何一个万维网页面，并且能够在自己的计算机屏幕上将这些页面显示出来。
- 怎样使用户能够很方便地找到所需的信息？
 - 为了在万维网上方便地查找信息，用户可使用各种的搜索工具（即搜索引擎）。



统一资源定位符 URL

- URL的格式 <协议>://<主机>:<端口>/<路径>

- 对可从因特网上得到的资源的位置和访问方法的简洁表示。
- URL 给资源的位置提供一种抽象的识别方法，并用这种方法给资源定位。
- 只要能够对资源定位，系统就可以对资源进行各种操作，如存取、更新、替换和查找其属性。
- URL 相当于一个文件名在网络范围的扩展。因此 URL 是与因特网相连的机器上的任何可访问对象的一个指针。



内容纲要

1 万维网

2 HTTP协议

3 网络结构设计

4 网络技术展望

5 网络技术和应用趋势



超文本传送协议 HTTP

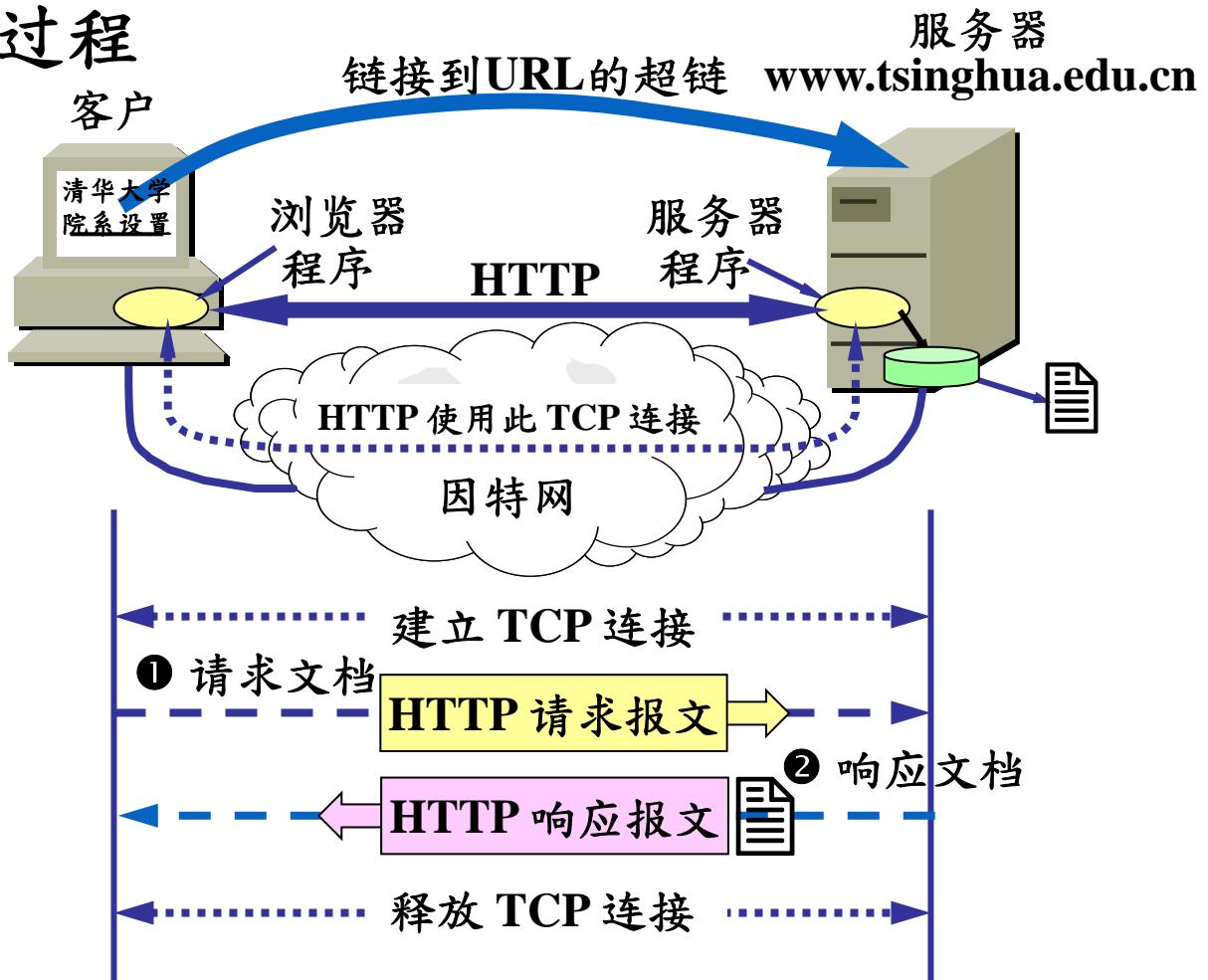
- HTTP 的操作过程

- 为了使超文本的链接能够高效率地完成，需要用 HTTP 协议来传送一切必须的信息。
- 从层次的角度看，HTTP 是面向事务的(transaction-oriented)应用层协议，它是万维网上能够可靠地交换文件（包括文本、声音、图像等各种多媒体文件）的重要基础。



超文本传送协议 HTTP

- HTTP 的操作过程



用户点击鼠标后所发生的事件

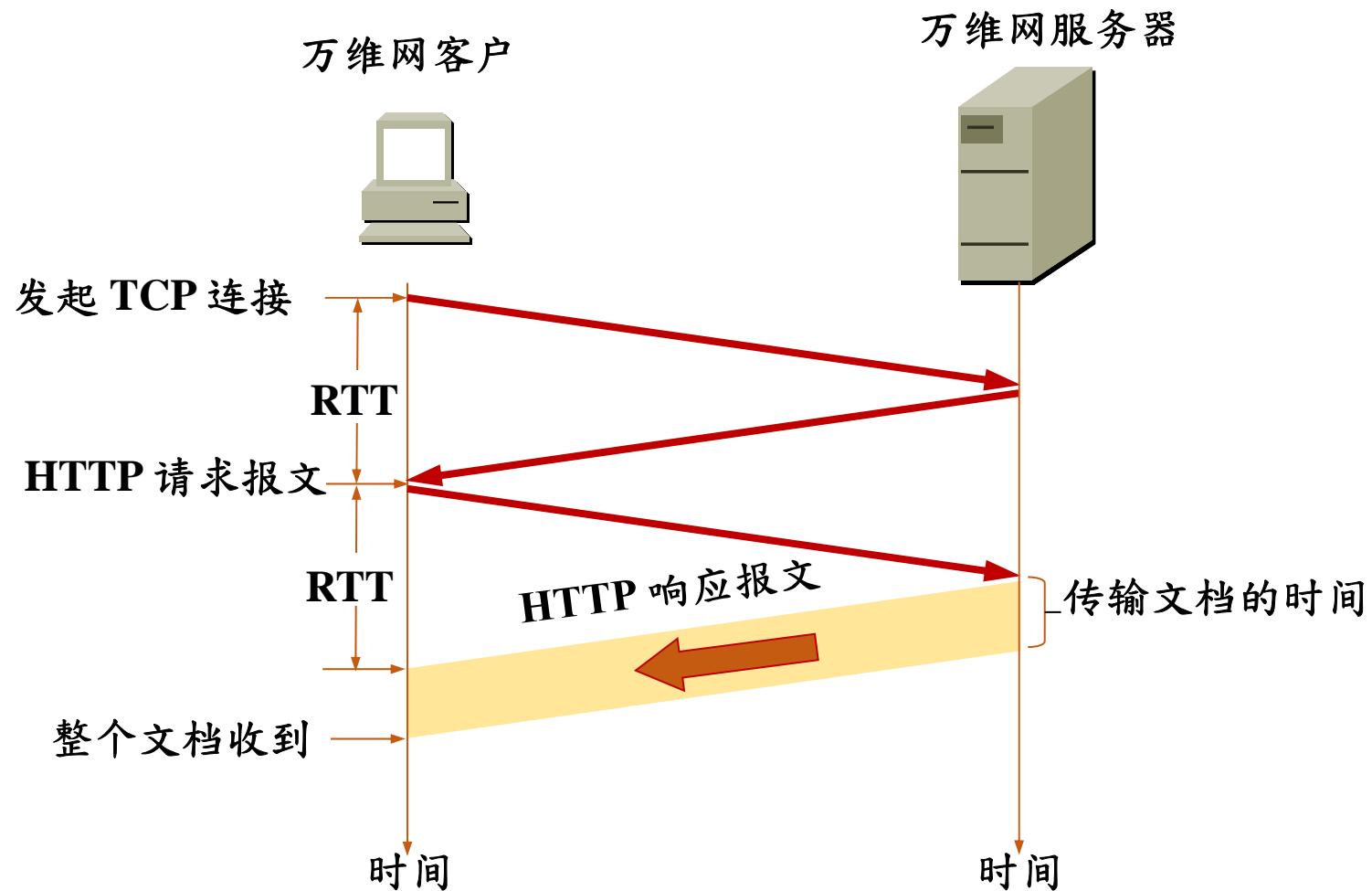
- 浏览器分析超链指向页面的 URL。
- 浏览器向DNS请求解析 `www.tsinghua.edu.cn` 的 IP 地址。
- 域名系统DNS解析出清华大学服务器的 IP 地址。
- 浏览器与服务器建立 TCP 连接
- 浏览器发出取文件命令：GET /chn/yxsz/index.htm。
- 服务器给出响应，把文件 `index.htm` 发给浏览器。
- TCP 连接释放。
- 浏览器显示院系设置文件 `index.htm` 中的所有文本。



HTTP 的主要特点

- HTTP 是面向事务的客户服务器协议。
- HTTP 1.0 协议是无状态的(stateless)。
- HTTP 协议本身也是无连接的，虽然它使用了面向连接的 TCP 向上提供的服务。

请求一个万维网文档所需的时间



持续连接 (persistent connection)

- HTTP/1.1 协议使用持续连接。
- 万维网服务器在发送响应后仍然在一段时间内保持这条连接，使同一个客户（浏览器）和该服务器可以继续在这条连接上传送后续 HTTP 请求报文和响应报文。
- 这并不局限于传送同一个页面上链接的文档，而是只要这些文档都在同一个服务器上就行。
- 目前一些流行的浏览器（例如，IE 6.0）的默认设置就是使用 HTTP/1.1。



持续连接的两种工作方式

- 非流水线方式
 - 客户在收到前一个响应后才能发出下一个请求。这比非持续连接的两倍 RTT 的开销节省了建立 TCP 连接所需的一个 RTT 时间。但服务器在发送完一个对象后，其 TCP 连接就处于空闲状态，浪费了服务器资源。
- 流水线方式
 - 客户在收到 HTTP 的响应报文之前就能够接着发送新的请求报文。一个接一个的请求报文到达服务器后，服务器就可连续发回响应报文。使用流水线方式时，客户访问所有的对象只需花费一个 RTT 时间，使 TCP 连接中的空闲时间减少，提高了下载文档效率。



代理服务器 (proxy server)

- 代理服务器(proxy server)又称为万维网高速缓存 (Web cache)，它代表浏览器发出 HTTP 请求。
- 万维网高速缓存把最近的一些请求和响应暂存在本地磁盘中。
- 当与暂时存放的请求相同的新请求到达时，万维网高速缓存就把暂存的响应发送出去，而不需要按 URL 的地址再去因特网访问该资源。



用高速缓存减少访问服务器的时延

- (1) 浏览器访问因特网的服务器时，要先与校园网的高速缓存建立 TCP 连接，并向高速缓存发出 HTTP 请求报文
- (2) 若高速缓存已经存放了所请求的对象，则将此对象放入 HTTP 响应报文中返回给浏览器。
- (3) 否则，高速缓存就代表发出请求的用户浏览器，与因特网上的源点服务器建立 TCP 连接，并发送 HTTP 请求报文

用高速缓存减少访问服务器的时延

- (4) 源点服务器将所请求的对象放在 HTTP 响应报文中返回给校园网的高速缓存。
- (5) 高速缓存收到此对象后，先复制在其本地存储器中（为今后使用），然后再将该对象放在 HTTP 响应报文中，通过已建立的 TCP 连接，返回给请求该对象的浏览器。



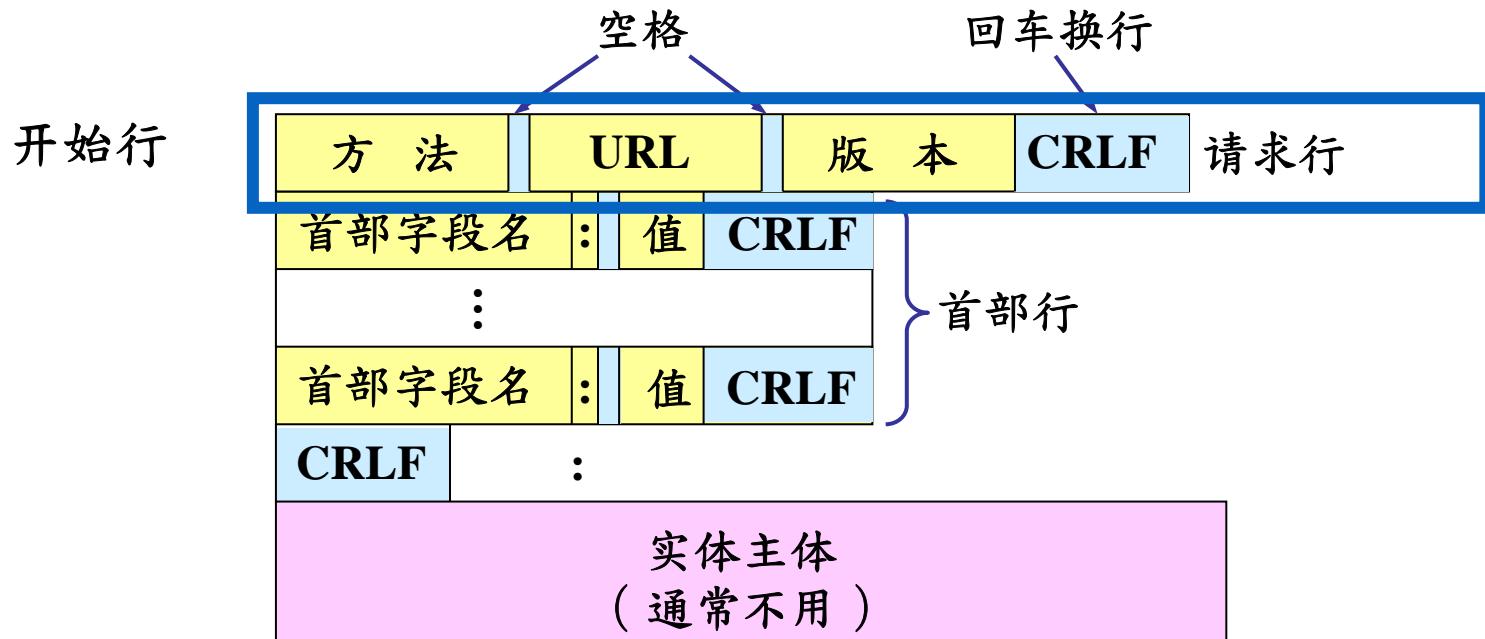
HTTP 的报文结构

- HTTP 有两类报文：
 - 请求报文——从客户向服务器发送请求。
 - 响应报文——从服务器到客户的回答。
- 由于 HTTP 是面向文本的(**text-oriented**)，因此在报文中的每一个字段都是一些 ASCII 码串，因而每个字段的长度都是不确定的。



HTTP 的报文结构

- 报文由三个部分组成，即开始行、首部行和实体主体。
- 在请求报文中，开始行就是请求行。



HTTP 请求报文的一些方法

• 方法（操作）	意义
– OPTION	请求一些选项的信息
– GET	请求读取由 URL所标志的信息
– HEAD	请求读取由 URL所标志的信息的首部
– POST	给服务器添加信息（例如，注释）
– PUT	在指明的 URL下存储一个文档
– DELETE	删除指明的 URL所标志的资源
– TRACE	用来进行环回测试的请求报文
– CONNECT	用于代理服务器

实例

- 运行于www.google.com，端口80

- 客户端请求

GET / HTTP/1.1

Host:www.google.com

(紧跟着一个换行，通过敲入回车实现)

- 服务器应答

HTTP/1.1 200 OK

Content-Length: 3059

Server: GWS/2.0

Date: Sat, 11 Jan 2003 02:44:04 GMT

Content-Type: text/html

Cache-control: private

Set-Cookie: PREF=ID=73d4aef52e57bae9:TM=1042253044:LM=1042253044:S=SMCc_HRPCQiqa

X9j; expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/; domain=.google.com

Connection: keep-alive

(紧跟一个空行，并由HTML文本组成Google的主页)



HTTP状态码

- 1xx 表示通知信息的，如请求收到了或正在进行处理。
- 2xx 表示成功，如接受或知道了。
- 3xx 表示重定向，表示要完成请求还必须采取进一步的行动。
- 4xx 表示客户的差错，如请求中有错误的语法或不能完成。
- 5xx 表示服务器的差错，如服务器失效无法完成请求。

状态码

- 同样的Page Not Found，不一样的代码
 - 左图：服务器处理的重定向（HTTP 302）
 - 右图：服务器美化的错误页面（HTTP 404）



在服务器上存放用户的信息

- 万维网站点使用 **Cookie** 来跟踪用户。
- **Cookie** 表示在 HTTP 服务器和客户之间传递的状态信息。
- 使用 **Cookie** 的网站服务器为用户产生一个唯一的识别码。利用此识别码，网站就能够跟踪该用户在该网站的活动。



超文本标记语言 HTML

- 超文本标记语言 HTML 中的 Markup 的意思就是“设置标记”。
- HTML 定义了许多用于排版的命令（即标签）。
- HTML 把各种标签嵌入到万维网的页面中。这样就构成了所谓的 HTML 文档。HTML 文档是一种可以用任何文本编辑器创建的 ASCII 码文件。
- 远程链接：超链的终点是其他网点上的页面。
- 本地链接：超链指向本计算机中的某个文件。



HTML 程序

```
<HTML>
```

```
  <HEAD>
```

```
    <TITLE>
```

```
        text that forms the document title
```

```
    </TITLE>
```

```
  </HEAD>
```

```
  <BODY>
```

```
        body of the document appears here
```

```
  </BODY>
```

```
</HTML>
```



动态万维网文档

- 静态文档是指该文档创作完毕后就存放在万维网服务器中，在被用户浏览的过程中，内容不会改变。
- 动态文档是指文档的内容是在浏览器访问万维网服务器时才由应用程序动态创建。
- 动态文档和静态文档之间的主要差别体现在服务器一端。这主要是文档内容的生成方法不同。而从浏览器的角度看，这两种文档并没有区别。



万维网服务器功能的扩充

- 应增加另一个应用程序，用来处理浏览器发来的数据，并创建动态文档。
- 应增加一个机制，用来使万维网服务器把浏览器发来的数据传送给这个应用程序，然后万维网服务器能够解释这个应用程序的输出，并向浏览器返回 HTML 文档。



内容纲要

1 万维网

2 HTTP协议

3 网络结构设计

4 网络技术展望

5 网络技术和应用趋势



设计理念

- PDIOO网络生命周期
 - 计划-设计-实现-运行-优化

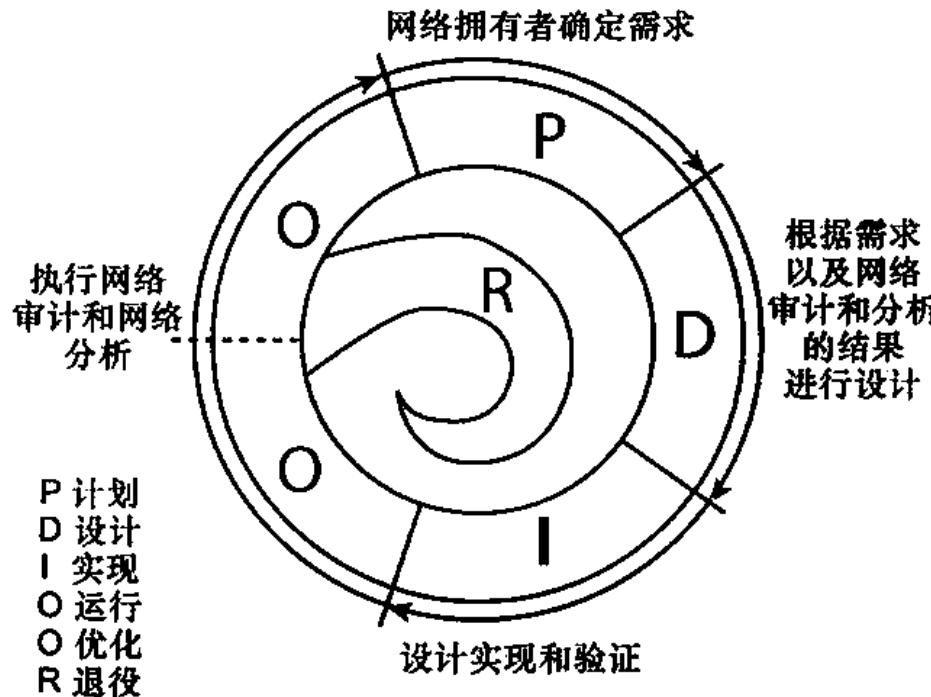


图 1-2 在 PDIOO 网络生命周期中包括设计工作的许多方面¹

PDIOO 网络生命周期

- 计划阶段：确定详细的网络需求，并检查现有的网络。
- 设计阶段：根据初始的需求和对现有网络分析中所收集到的额外信息设计网络。
- 实现阶段：根据得到认可的设计方案构建网络。
- 运行阶段：网络开始运转，并受到监测，该阶段是对设计方案的最终测试。
- 优化阶段：发现和纠正一些问题。
- 退役阶段：网络过时了或不用了。



网络设计任务

- 确定需求
 - 如果网络已经存在了，需对现有网络进行分析
- 准备概要设计
 - 完成最终的设计开发；部署网络
- 监测网络
 - 如有必要，重新设计网络
- 维护文档

网络设计任务

- 网络设计任务

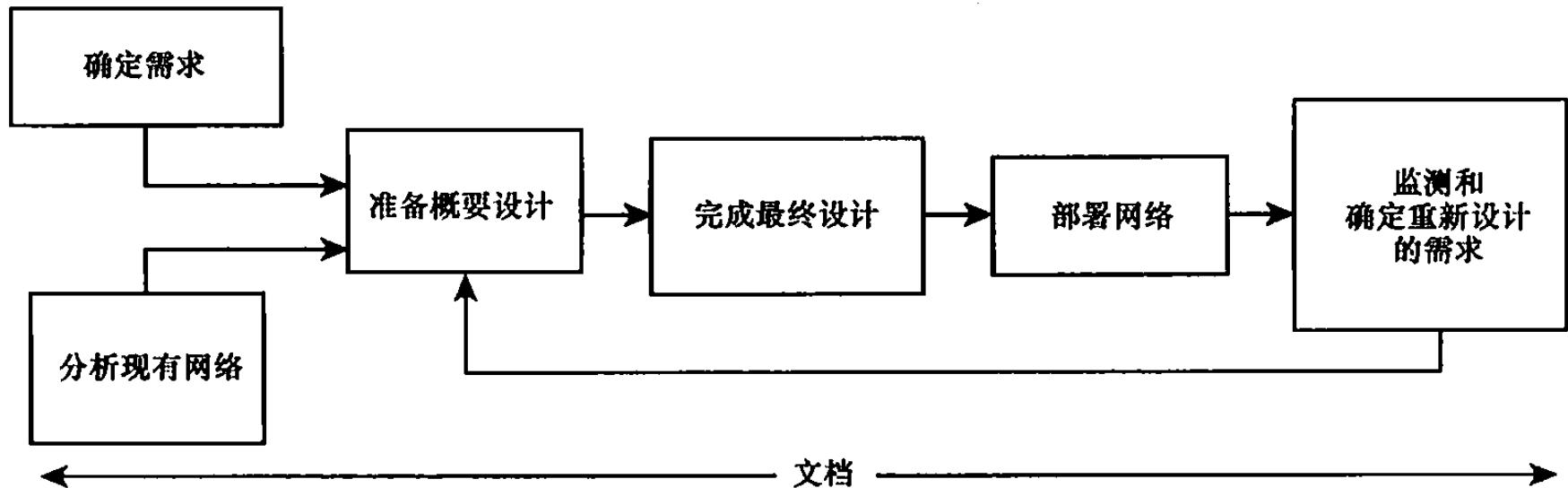


图 1-3 网络设计是一个持续性的过程

网络设计任务：确定需求

- 技术需求和限制因素：

- 网络上运行的应用程序；网络上运行的其他协议（如路由协议）
- 互联网连通性要求；
- IPV4地址的限制；支持IPv6地址；
- 布线的要求；备份的要求；私有设备和协议要求；
- 服务质量要求；
- 安全性要求；

网络设计任务：确定需求

- 技术需求和限制因素：
 - 要求的网络解决方案（语音服务、内容网络和存储网络）；
 - 网络管理；
 - 可用带宽。
- 涉及的商业问题的需求和限制
 - 预算；进度表；人力；法律法规；历史记录；策略
- 每个需求应该评估出其重要性，分配一个权重系数，以便在需求发生冲突时尽量满足最重要的需求



网络设计任务：对现有网络的分析

- 对现有网络的分析

- 现有网络可能会带来一些限制，如：现有的布线需要保留；
- 正在运行的协议；
- 现有网络的拓扑结构；
- 已安装的设备及其配置、设备的利用率以及WAN上关键链路的带宽；

网络设计任务：概要设计和方案

- 概要设计的准备
 - 考虑所有网络需求和约束条件；确定可选方案；选择设备、功能特性、布线等
- 最终设计方案开发
 - 产生详细设计图纸；配置规范；成本计算、地址规划等
- 可以实现一个原型网络来验证设计，或在现有网络中实现一个实验性的网络来验证。



网络设计任务：部署和监测

- 网络部署

- 根据计划和进度实施；
- 确定工作内容和实施细节；
- 规划好有关人员的培训等

- 监测和重新设计

- 收集运行统计信息；
- 了解网络运行的各项指标；
- 发现故障和异常情况等



模块化网络设计

- 模块化设计

- 模块是复合结构的一个组件。
- 首先建立模块，然后把模块进行组合。
- 常用的技术有：层次化设计模型；复合网络模型

层次化网络设计

- 层次化模型由下面三个功能组成：
 - 接入层：使用户和工作组可以访问网络资源；
 - 分布层：提供工作组之间以及工作组到核心层之间的连接；
 - 核心层：提供到核心层资源和分布层设备之间的高速传输服务。



层次化网络设计

- 把一个简单的网络映射到层次化模型

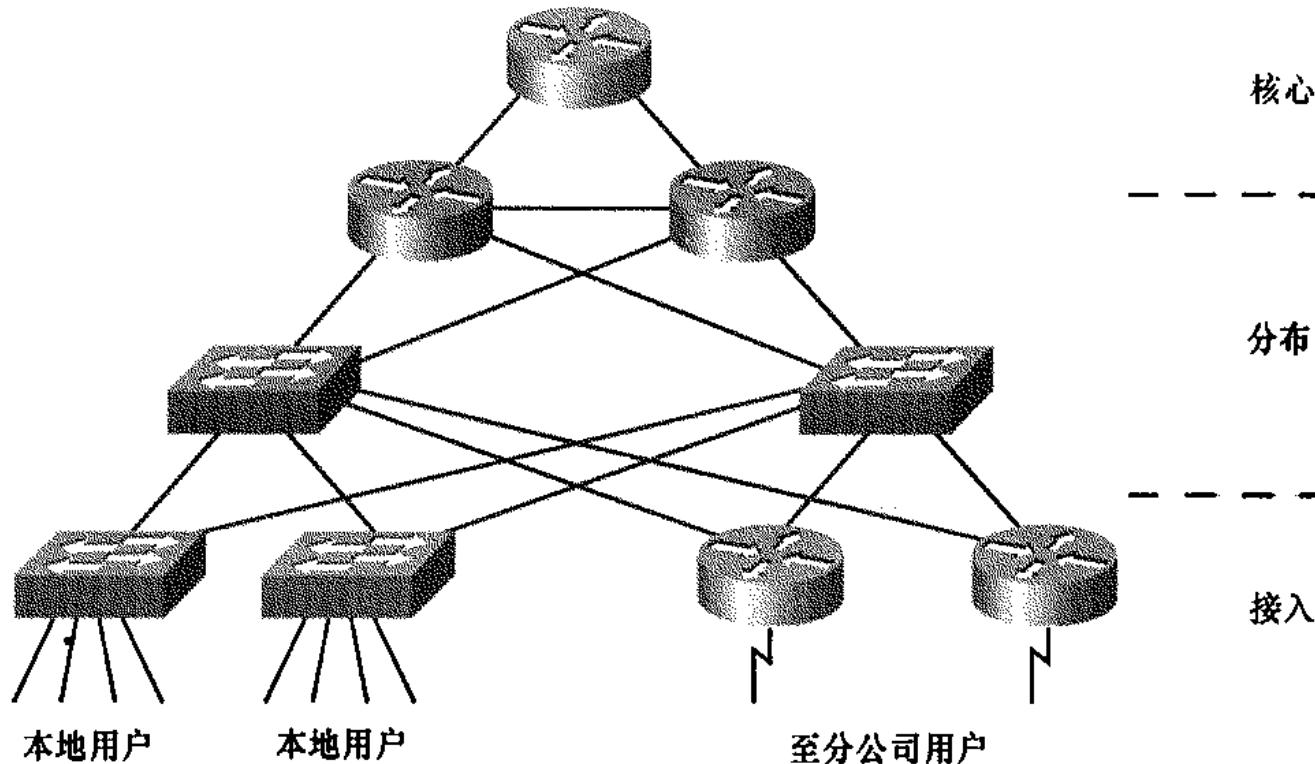


图 1-5 简单网络到层次化模型的映射

层次化网络设计：接入层

- 接入层

- 接入层是用户接入网络的地方，可以是本地，也可以是远程。
- 本地用户通过集线器和交换机接入网络。
- 远程用户可以使用**VPN**连接经互联网接入到内部网络。比如：可以通过**PSTN**、**DSL**等方式。其他接入方式可以采用**WAN**技术，比如：**FR**、**DDN**、**ISDN**



层次化网络设计：分布层

- 分布层

- 分布层是核心层和接入层之间以及接入层工作组之间的接口。
- 在接入层与核心层之间进行路由选择、路由协议处理。
- 执行路由汇总。
- 提供到接入设备和核心设备之间的冗余连接。
- 把多个低速接入的连接汇集到较高速度的核心连接上。

层次化网络设计：核心层

- 核心层：核心层提供高速的网络主干。
- 其功能和属性如下：
 - 提供高速、低时延的数据链路和设备。
 - 提供冗余设备和链路使得网络不存在单点故障，实现高可靠、高可用的骨干。
 - 使用快速收敛路由协议可以迅速适应网络变化。



路由、交换与远程访问技术

- 一个较为完整的园区网

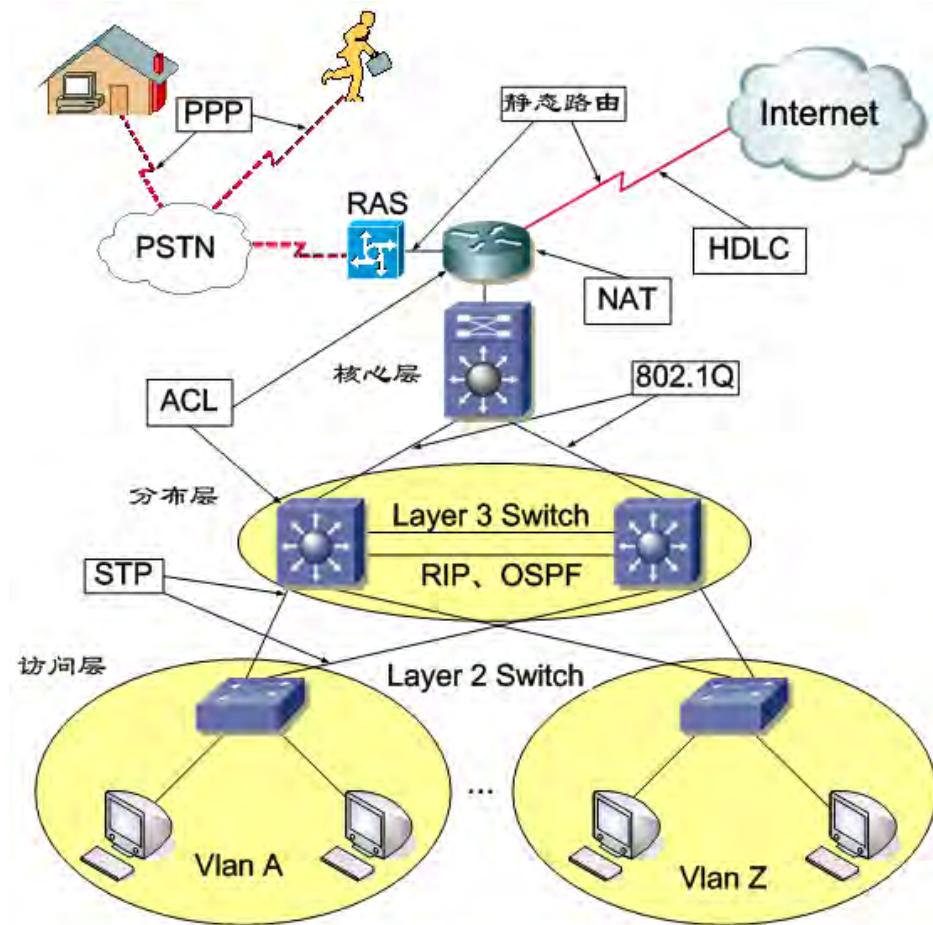
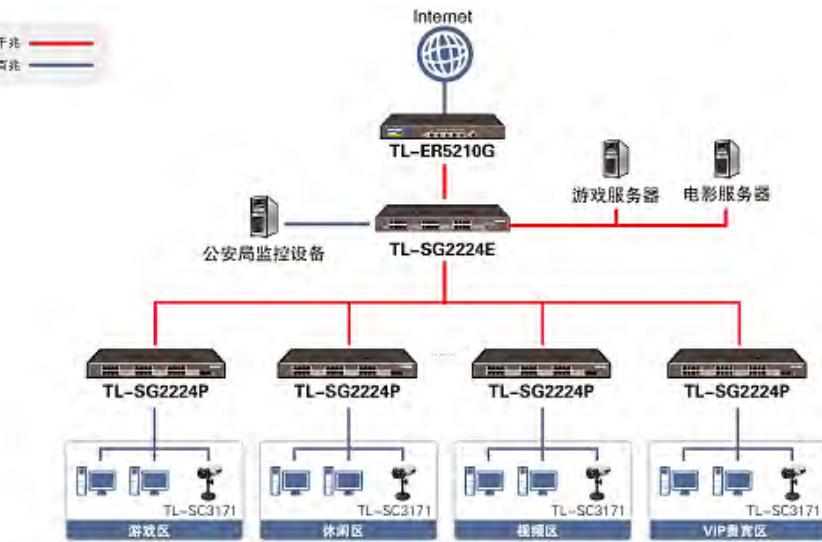
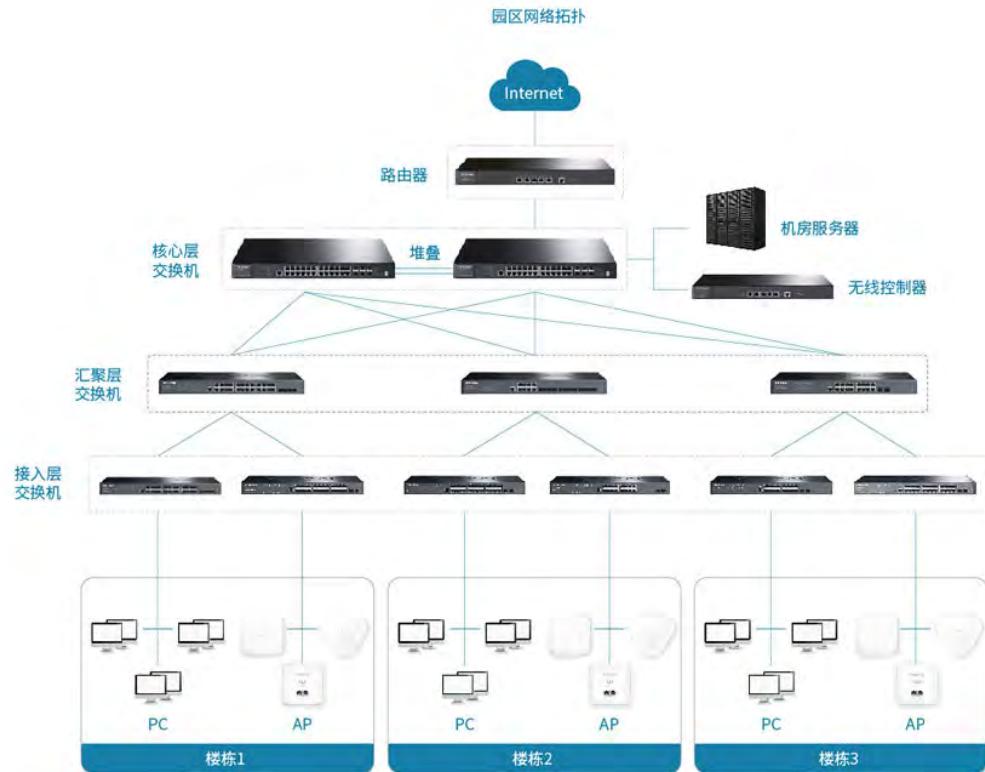
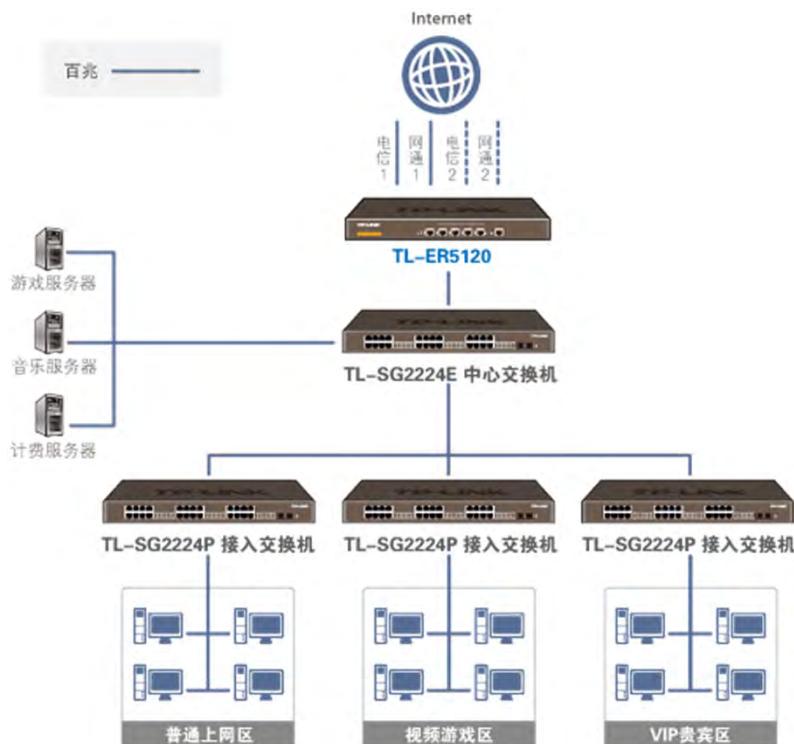


图 1-28 一个较为完整的园区网实现

网络实例

- 园区网络实例
- 网吧网络实例



内容纲要

1 万维网

2 HTTP协议

3 网络结构设计

4 网络技术展望

5 网络技术和应用趋势



IP语音

- **Real-time Transport Protocol (RTP)**

- 实时传输协议（ RTP ）提供了用于在Internet上传输实时数据的机制
- 基于UDP协议

- **Voice over IP (VoIP)**

- 世界各地的电话公司正用IP路由器取代传统的电话交换机。
- 路由器比传统电话交换机成本低得多。

网络安全技术

- 原因

- 每当出现新技术时，罪犯就会考虑他们如何利用这项技术来犯罪。

- 互联网上普遍存在的主要安全问题

- 网络钓鱼；虚假陈述；诈骗
 - 拒绝服务；失控；数据丢失



网络安全技术

- 安全攻击中使用的技术

- 窃听；重播；缓冲区溢出；地址欺骗；域名欺骗；
 - 拒绝服务（DOS）与分布式拒绝服务（DDoS）
 - SYN洪流；密码破解；端口扫描；分组拦截

- 信息安全五大要素

- 完整性、保密性、可用性、可控性和不可否认性



网络安全技术

- 用于执行安全策略的主要技术。
 - 散列；加密；数字签名；数字证书；防火墙
 - 入侵检测系统；内容扫描和深度包检查；虚拟专用网VPN



网络管理（SNMP）

- 网络管理员（网管）是负责计划、安装、操作、监视和控制构成计算机网络或内部网的硬件和软件系统的人。
- 虽然网络硬件和协议软件包含自动绕过故障或重发丢失分组的机制，但网络管理者需要检测和纠正潜在的问题。
- SNMP协议确切地定义了管理器如何与代理通信。



软件定义网络 (SDN)

- 一种新型网络创新架构，是网络虚拟化的一种实现方式
- OpenFlow协议用于控制器与网络元素之间通信
- SDN有潜力引入网络产业的重大变革。
- 一旦网络管理设施被移除，交换硬件将成为商品，网络管理者将不会有动机从单个供应商购买所有设备。



物联网（IoT）

- 物联网：使用机器对机器通信的连接的嵌入式系统。
 - 与使用传统计算机的早期应用不同，物联网应用专注于嵌入式系统。
 - 应用包括家庭自动化、智能电网、安全和零售系统。
- 许多技术需要低功耗无线通信。
 - 为了最小化功率利用率，无线网络可以使用网格方法：
 - 低功率节点同意彼此转发数据包，而不是使用能够到达长距离的强大无线电发射器。

内容纲要

1 万维网

2 HTTP协议

3 网络结构设计

4 网络技术展望

5 网络技术和应用趋势



网络技术和应用趋势

- 对可扩展因特网服务的需求
- 内容缓存（Akamai）
- 网络负载平衡器
- 服务器虚拟化
- P2P：对等通信
- 分布式数据中心和复制
- 社交网络



网络技术和应用趋势

- 移动性和无线网络
- 数字视频
- 高速接入和交换
- 云计算
- 覆盖网络
- 中间件
- IPv6的广泛部署



计算机网络

Computer Network

18

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

19

期末复习

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang

知识框架

- 核心内容
 - 协议模型
 - 七层、五层、四层
 - 各层次的协议、机制、作用
- 应用内容
 - 传输过程的数据处理细节

内容纲要

1

本书重点概述

2

各协议层重点

3

期末考试题型

4

期末考试大纲

5

总结

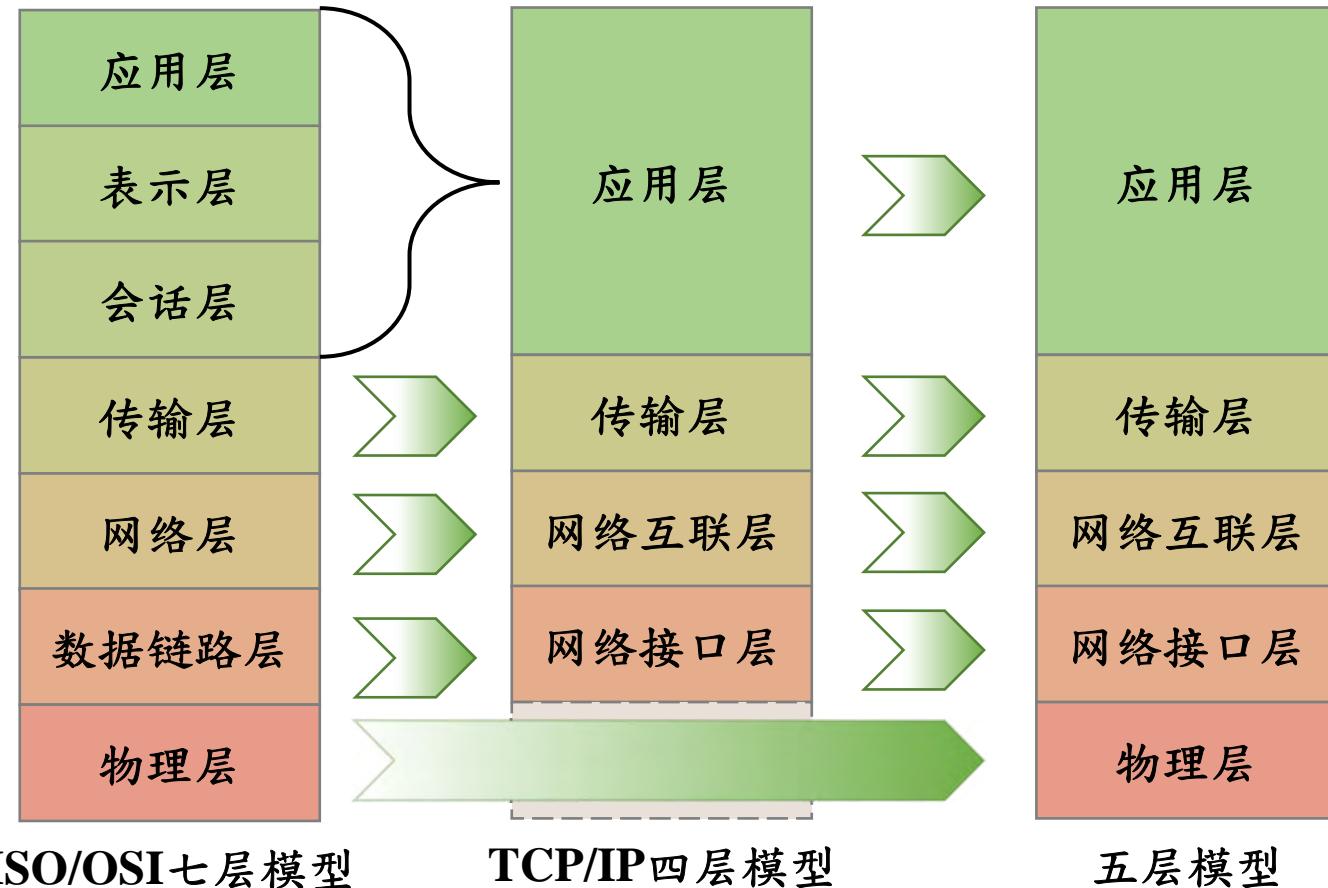


什么是重点

- 贴近生活，经常使用的是重点
 - 以太网
 - IP协议
 - TCP协议

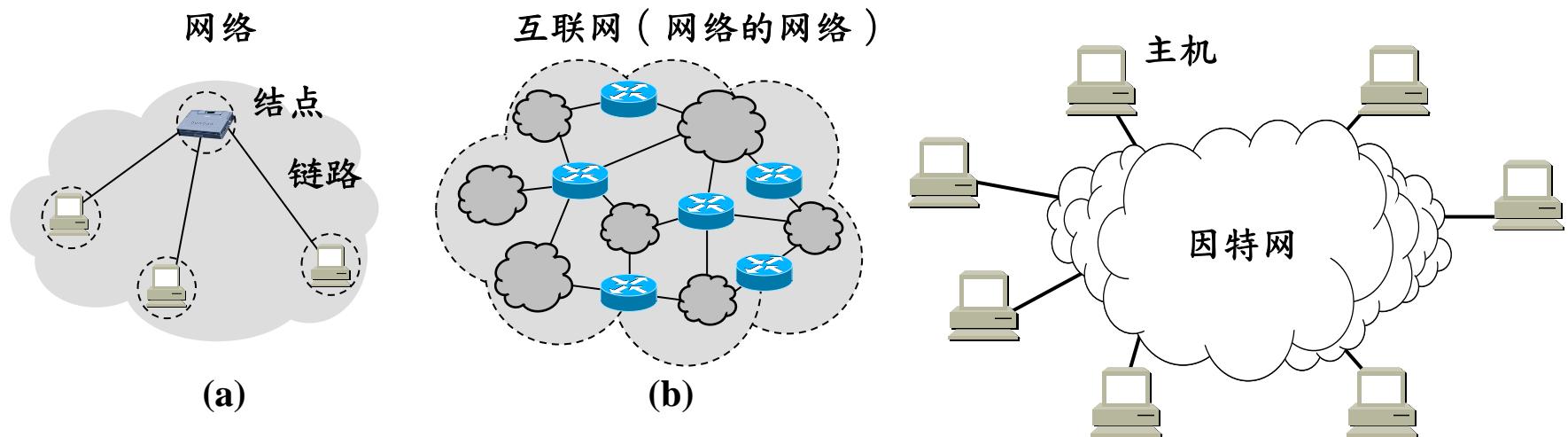
总纲

• 计算机网络的分层架构

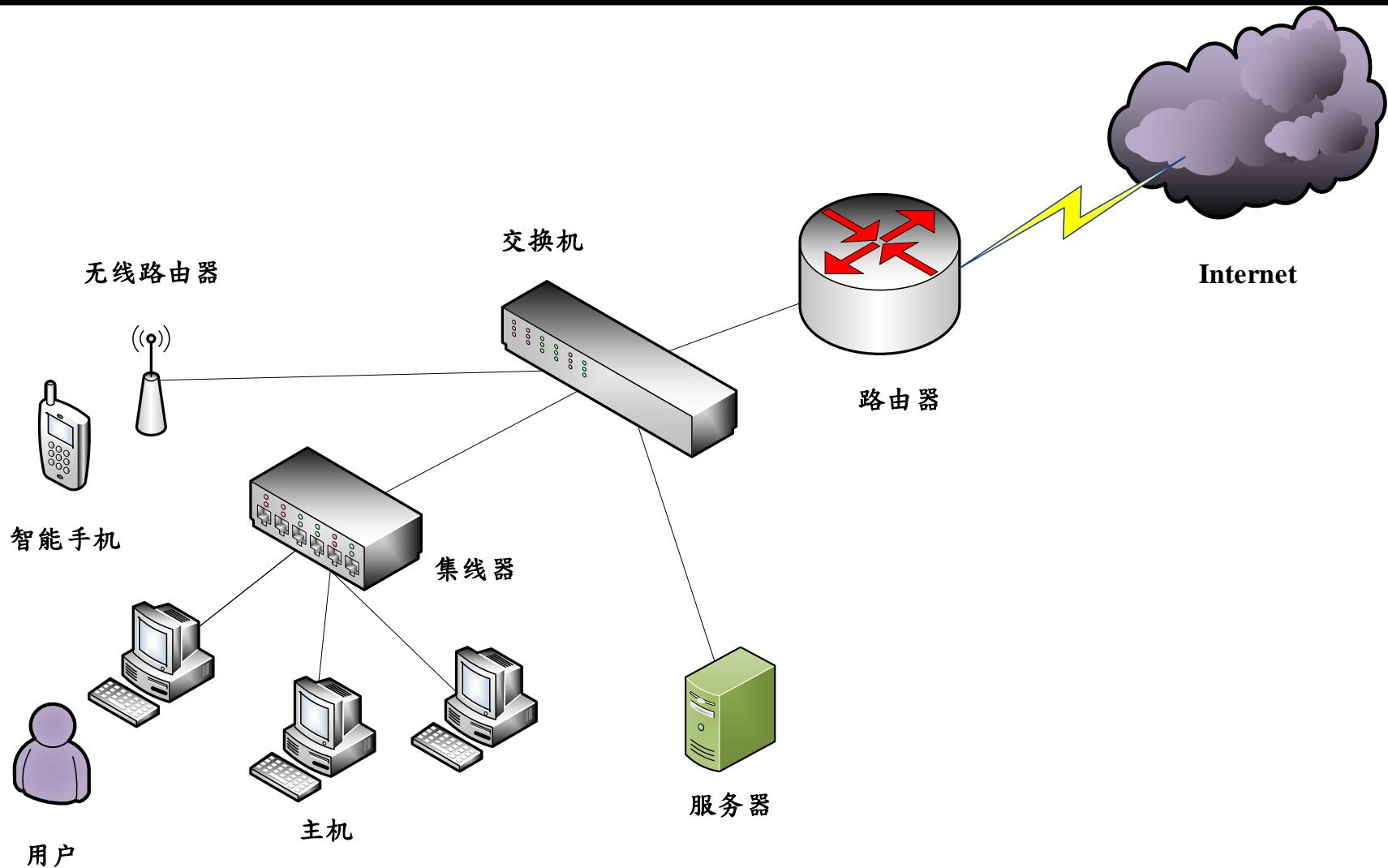


什么是因特网

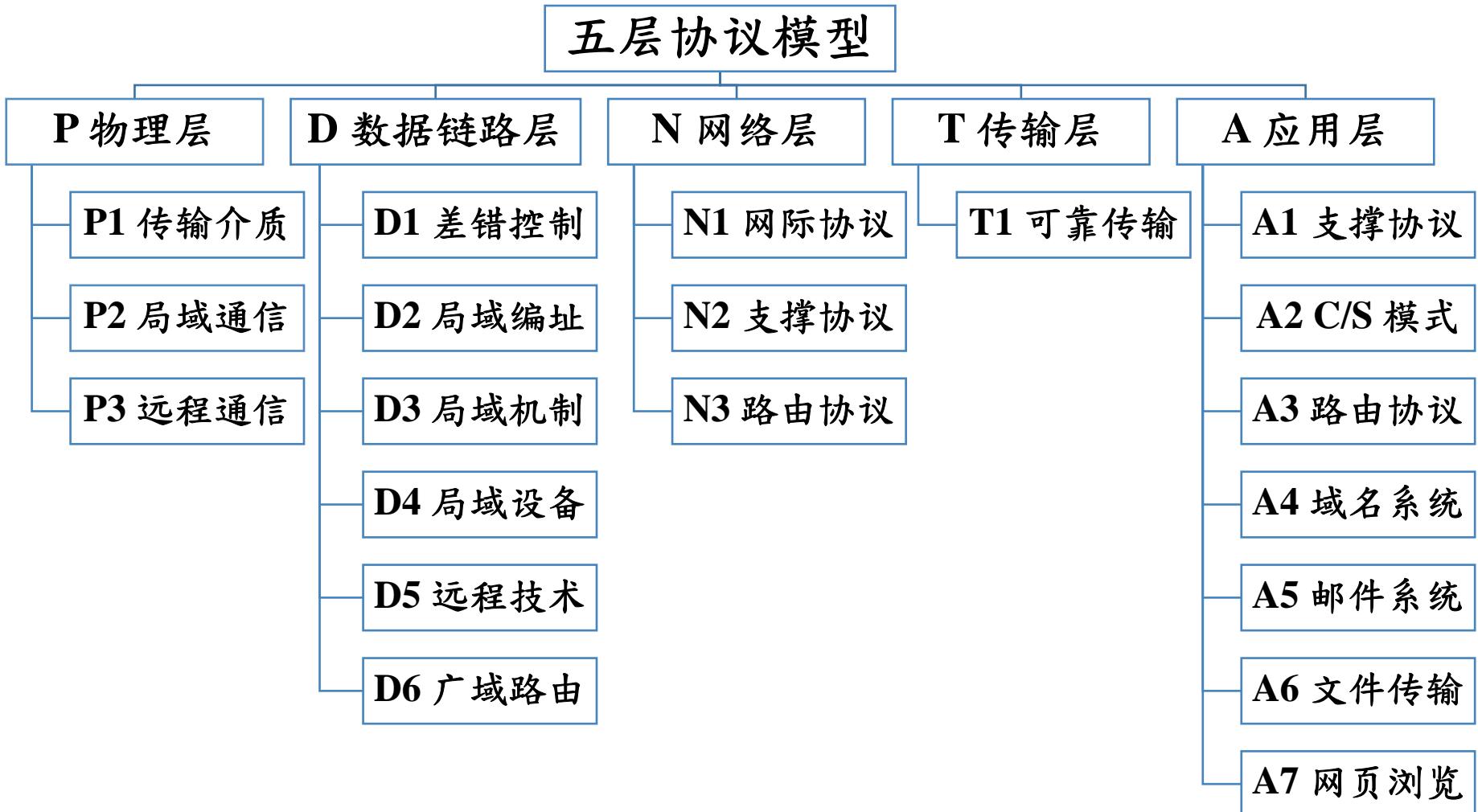
- 互联网：网络的网络（ network of networks ）
 - 第一个网络指的是主机通过共享设备和介质连接为局域网
 - 第二个网络指的是通过路由器将局域网连成广域网



什么是因特网



课程框架



重要的表格

分层名称	传输最小单位	主要设备	主要协议（或标准）		该层同类协议	层主要作用
			编址名称、方案	协议机制		



内容纲要

1

本书重点概述

2

各协议层重点

2.1

物理层

2.2

数据链路层

2.3

网络层



物理层

- 传输最小单位：位（ bit，比特 ）
- 主要扩展设备：集线器、中继器
- 主要硬件：传输介质（ 光、电、电磁波 ）
- 主要协议（ 标准 ）：**RS232C**
 - 编址：无需编址
 - 帧格式
 - 机制：编码原理
- 该层同类协议（ 标准 ）：**RJ45**



物理层

- 该层主要作用
 - 完成对比特和能量之间的转换
 - 处理与物理传输介质相关的接口



内容纲要

2

各协议层重点

2.1

物理层

2.2

数据链路层

2.3

网络层

2.4

传输层



数据链路层

- 传输最小单位：帧（ frame ）
- 主要扩展设备：网桥、交换机
- 主要协议（标准）：Ethernet
 - 编址：MAC地址（ OUI+NIC标识 ）
 - 帧格式：

前同步码	SFD	目的地址	源地址	类型	数据	CRC
------	-----	------	-----	----	----	-----

7字节 1字节 6字节 6字节 2字节 46~1500 字节 4字节

– 机制：CSMA/CD

- 该层同类协议（标准）：令牌环等

数据链路层

- 该层主要作用

- 成帧（包括查错控制）
- 介质访问控制子层(MAC)
- 逻辑链路控制子层(LLC)



内容纲要

2

各协议层重点

2.2

数据链路层

2.3

网络层

2.4

传输层

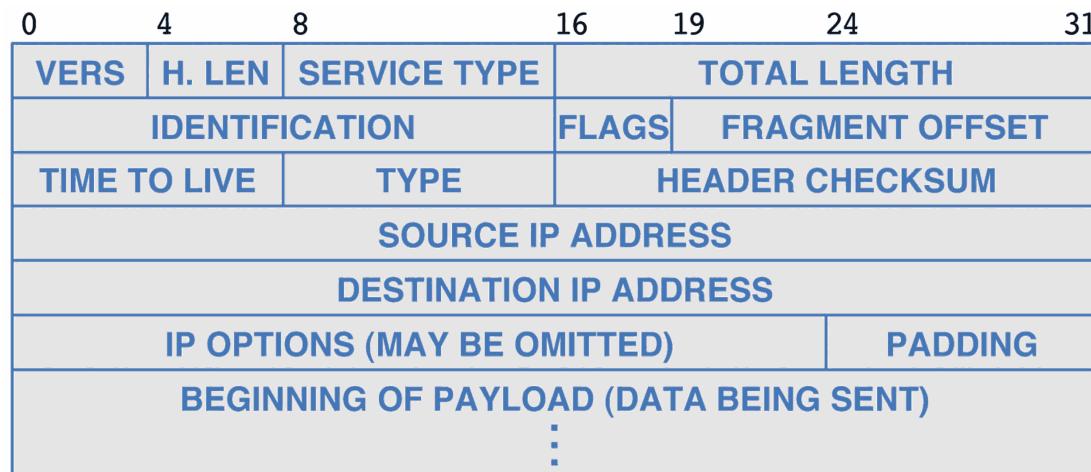
2.5

应用层



网络层

- 传输最小单位：数据报（ datagram ）
- 主要扩展设备：路由器（ 网关 ）
- 主要协议（ 标准 ）： IPv4
 - 编址：IP地址（ 网络号+子网号+主机号 ）
 - 有类、无类
 - 报文格式：



网络层

- 主要协议（标准）：IPv4

- 机制：

- 子网划分
 - 路由表的构建
 - 路由转发
 - IP报文在帧的封装
 - MTU和分片、重组



网络层

- 该层同类协议（标准）：IPv6
- 该层主要作用
 - 主机到主机间尽力而为的通信
 - 路由寻径：维护路由表和根据路由表查询转发
 - 通过询问和差错报告，确保网络连接



网络层的支撑协议

- ICMP
 - PING、TraceRoute原理
- ARP
- 路由协议
 - 内部网关协议：RIP、OSPF
 - 外部网关协议：BGP4
- DHCP、NAT

内容纲要

2

各协议层重点

2.2

数据链路层

2.3

网络层

2.4

传输层

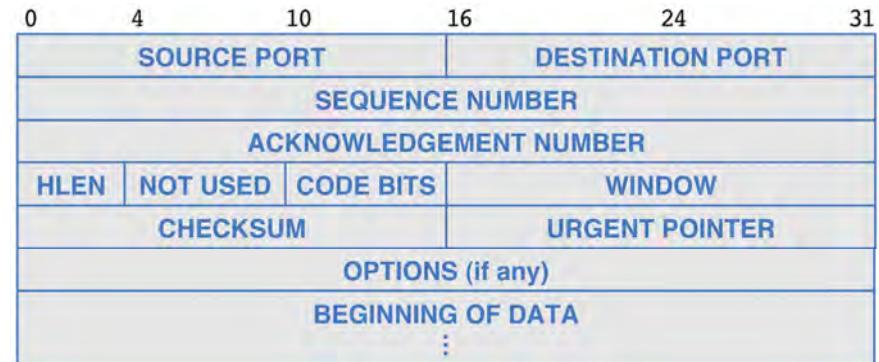
2.5

应用层



传输层

- 传输最小单位
 - TCP：数据段（ Segment ）
 - UDP：数据报（ datagram ）
- 主要扩展设备：四层交换机
- 主要协议（标准）：TCP
 - 编址：端口号（熟知端口号，登记端口号、客户端口号）
 - 报文格式：



传输层

- 主要协议（标准）：TCP
 - 基本机制
 - 流接口
 - 虚连接
 - 停止-等待协议（有差错、无差错）
 - 窗口机制
 - 超时重传
 - 流量控制
 - 拥塞控制
 - 连接管理：三次握手、四次挥手



传输层

- 该层同类协议（标准）：UDP
- 该层主要作用
 - 进程间端到端的通信
 - 提供传输的可靠性



内容纲要

2

各协议层重点

2.2

数据链路层

2.3

网络层

2.4

传输层

2.5

应用层



应用层

- 传输最小单位：数据（ Data ）
- 主要扩展设备：防火墙
- 主要协议（标准）
 - 有代表性的协议：DNS、E-mail、FTP、HTTP
 - 编址：用户自定义
 - 报文格式：用户自定义
- 该层主要作用
 - 提供最通用的应用程序
 - 完成用户信息或者软件转换信息的交互



内容纲要

1

本书重点概述

2

各协议层重点

3

期末考试题型

4

期末考试大纲

5

总结



综合题

- 给出一个组织的内部网络架构图或类似的网络示意图
- 按计算机网络分层模型，结合实例多角度提出问题
- 要求学生结合实例分析问题解决问题
- 两种类型的题目
 - 考察单个网络架构层次的单个协议或标准知识
 - 综合运用五层协议模型解决问题



内容纲要

1

本书重点概述

2

各协议层重点

3

期末考试题型

4

期末考试大纲

5

总结



考试大纲

- 第1课 传输介质

- 通信基本模型
- 引导型传输媒体
 - 金属：屏蔽双绞线，非屏蔽双绞线，同轴电缆
 - 光纤：单模和多模
- 非引导型传输媒体
 - 红外线，激光，无线电波（镭射）、卫星
- 介质间的权衡

考试大纲

• 第2课 局域异步通信

– 传输模式的类别

- 串行，平行
- 同步，异步，等时
- 单工、半双工、全双工

– 多比特下的端序：大端序，小端序

– 异步通信标准：RS-232

- 电气特性，帧、帧格式
- 参数：波特率，波特，标准化

– 两个重要通信理论：奈奎斯特定理和香农定理



考试大纲

- 第3课 远程通信

- 载波
- 调制和解调
 - 调频、调幅、调相
- 复用和解复用
 - 频分、波分、时分（同步时分、统计时分）、码分
- 基带和宽带

考试大纲

- 第4课 差错控制

- 奇偶校验的简单计算
- Internet Checksum (16 位校验和) 的简单计算
- 循环冗余校验码 (CRC , 不要求计算)



考试大纲

• 第5课 局域网分组与编址

- 交换技术：线路交换、报文交换、分组交换
- 网络接口卡（NIC）的作用
- MAC地址的构成
- 单播、广播、组（多）播
- 帧结构（头部+载荷）、成帧
- 以太网帧结构



考试大纲

- 第6课 以太网、拓扑与无线技术
 - 局域网拓扑：总线、星形、环形、网状
 - 以太网介质访问控制策略（CSMA/CD）
 - 其它网络类型的特点：LocalTalk、Token Ring、FDDI、ATM
 - 网络技术的分类：个域网、局域网、城域网、广域网
 - WLAN基本概念：蓝牙、蜂窝网络、1G~4G、GPS，及速率大致量级



考试大纲

- 第7课 局域网的布线、拓扑、接口硬件
 - 以太网的粗缆、细缆、双绞线布线
 - 物理和逻辑拓扑
 - 冲突域与广播域的概念
 - 中继器、集线器、网桥
 - 交换机、广播风暴与分布生成树



考试大纲

- 第8课 远程数字连接技术、网络性能
 - Internet 接入技术：上行和下行
 - 接入技术：宽带与窄带、ISDN、ADSL、电缆调制解调器、无线、光纤
 - 标准：数字电话标准（T、E）、干线标准（STC、OC、同步光网络）
 - 各种网络接入技术与标准大致的速率量级
 - 广域网技术的类型：虚电路、数据报，及各自的特点
 - 不同类型的网络技术：APANET、PSTN、X.25、帧中继的特点



考试大纲

- 第8课 远程数字连接技术、网络性能（续）

- 网络所有权：私有网络、公有网络的定义
- 网络的性能度量：时延、吞吐率、抖动

考试大纲

- 第9课 广域网技术与路由、协议系列
 - 分组交换机的原理、存储与转发
 - 广域网的概念和分层编址
 - 路由工作原理
 - 路由器转发表、默认路径、下一站
 - 网络协议分层的思想：网络互联、虚拟网络的概念
 - ISO/OSI网络协议的分层模型（7层）
 - TCP/IP 协议栈（5层）
 - ISO/OSI和TCP/IP分层之间对应关系、数据基本单位、各层的分工作用



考试大纲

• 第10课 网际协议

– IPV4编址

- 有类地址（A~E类）
- 无分类和CIDR表示法

– 子网划分和子网掩码

- 有分类的子网划分、无分类的子网划分

– 特殊IP地址

- 本机地址、网络地址、环回地址、直接广播地址、有限广播地址
- 网络层的广播与多播

– 多穴主机



考试大纲

- 第10课 网际协议（续）

- IPv4数据报格式中的各部分组成（不要求顺序）
- MTU与分片、分片重装和收集
- IP封装、虚拟分组
- IP数据报转发原理、转发过程中的帧头、报文头的情况



考试大纲

- 第11课 支撑协议与相关技术、IPv6

- ARP协议

- 地址解析作用，地址解析的方法，概念地址边界

- ICMP协议工作原理

- ICMP的报文种类、主要功能

- IP与ICMP的关系

- ping 命令测试可达性的原理

- tracert 命令追踪路由的原理

- 使用ICMP发现MTU

- IPv6编址方案、冒分十六进制表示法



考试大纲

- 第12课 传输控制协议

- 传输层

- 作用，端口号，端口号的分类

- UDP

- UDP的无连接、尽力交付、面向报文、允许广播

- TCP

- 特点：面向连接、点对点、可靠、全双工、字节流

- TCP段格式中的各部分组成

考试大纲

- 第12课 传输控制协议（续）

- TCP的机制

- 应答机制、超时机制、重传机制、窗口机制
 - 流量控制机制：滑动窗口
 - 拥塞控制：慢开始、拥塞避免、快重传、快恢复、随机早期检测
 - TCP的连接建立和解除（三次握手、四次挥手）

- 传输层解决网络层的主要问题：丢包、重复、乱序

考试大纲

- 第13课 因特网路由与路由协议

- 静态路由与动态路由
- 自治系统（AS）的概念
- 内部网关协议（IGP）
 - RIP协议的工作原理和特点
 - OSPF协议的工作原理和特点
- 外部网关协议（EGP）
 - BGP协议



考试大纲

- 第14课 网络编程与Socket API
 - 客户端—服务器端（C/S）交互模式工作原理
 - 并发的概念
 - Socket结构、半相关与全相关
 - 服务器与用户、服务器端与客户端，二者区别
 - Socket API主要函数（C++）
 - 流模式的客户端、服务器端Socket API调用流程
 - 报文模式的客户端、服务器端Socket API调用流程



考试大纲

- 第15课 域名服务（DNS）

- 域名、域名分级
- 域名服务器分级
- 域名服务（DNS）
 - 递归、迭代的工作原理



考试大纲

- 第16课 电子邮件

- 电子邮件的格式
- 主要构成：MTA、MUA、MDA
- 主要协议（作用、原理、端口号）
 - 电子邮件的传输：SMTP
 - 电子邮件的传输扩展：MIME
 - 电子邮件的访问：POP3，IMAP



考试大纲

- 第17课 文件传输
 - FTP工作原理与通信模式
 - FTP主动和被动工作模式



考试大纲

- 第17课 文件传输
 - FTP工作原理与通信模式
 - FTP主动和被动工作模式



考试大纲

- 第18课 万维网

- HTTP工作原理与过程
- 浏览器的结构
- HTTP错误代码
- URL
- HTML文档

考试大纲

- 第19课 高级专题（网络安全、网络发展趋势）
 - 网络防火墙的基本常识
 - 网络安全技术（加密、签名、访问控制、HTTPS、TLS等）的基本常识
 - 虚拟专用网（VPN）、代理服务器的基本常识
 - 对等计算（P2P）模式工作原理
 - 内容缓存、Web均衡负载、网络架构的基本常识



考试大纲

- 第20课 实验课

- RJ-45网线的制作与接入；
- RS232串行通信编程：打开、读、写、关闭；
- OmniPeek或Wireshark进行网络监听，用PCAP库编程，并分析以太网帧、IP报文、TCP段和FTP协议的格式。
- 观察TCP的三次握手、四次挥手。
- 路由器主要的配置：IP分配、路由表等。
- 掌握Socket API编程的基本过程：面向连接的和无连接的。
- 应用层服务器基本配置项。



内容纲要

1

本书重点概述

2

各协议层重点

3

期末考试题型

4

期末考试大纲

5

总结



复习要点

- 记住术语
 - 简称、全称
- 理解原理
 - 为什么、怎么来
- 善加思考
 - 传输机制细节处理
 - 概念之间的差异



计算机网络

Computer Network

19

谢谢观看

理论课程



厦门大学
XIAMEN UNIVERSITY



信息学院 黄煌
(国家示范性软件学院)
School of Informatics Dr. Wei Huang