计算机网络原理

1、什么是网络?

将终端设备连接起来并可以传输数据
有“线”：可能是有形的线也可能是无线链路

直接相连的网络
- 点对点连接
- 多路访问链路
  - 某些节点需要决定数据传给哪个节点，这种行为称为选择路由，简称选路
  - 途中每个节点称为路由器

unicast：单播：一对一，一个往另一个发，单个主机地址发送
multicast：多播：一对多，一个往多个发，多个主机地址发送
broadcast：广播：散发

simplex：单工：只能单向传播数据：例如：广播、电视
half-duplex：半双工：可以异步双向传播数据：例如：对讲机
full-duplex：全双工：可以同步双向传播数据：例如：电话

SAN：system area network：系统域网：例如：电脑+鼠标+USB
LAN：local area network：局域网：例如：小实验室
MAN：metropolitan area network：城域网
WAN：world area network：广域网

2、什么是因特网?

需要
- 设备终端系统：进行收发
- 路由器：将数据分为数据块（数据报）并根据路由表进行选路
- 连通并提供通信服务

通信服务类型
- 可靠服务与不可靠服务
  - 可靠服务：收发完全相同：如文件
  - 不可靠服务：收发不完全相同：如视频
- 面向连接服务与无连接服务
  - 面向连接服务：需要有连接，如电话（双方都要在）
  - 无连接服务：不需建连接，如寄信（对方可能不在），因特网属于这一种
- 无确认服务与有确认服务
  - 无确认服务：不需确认对方是否收包，因特网属于这一种
  - 有确认服务：需要确认对方是否收包
- 请求响应的服务与消息流服务
  - 请求响应的服务：有请求才有交互
  - 消息流服务：一直发消息，如电视

因特网是无确认的无线服务，也称为数据报服务

因特网的组成：
- 网络边界：主机
- 接入网络：有线或无线接入
- 网络核心部分

3、网络核心部分

两种数据传输技术

- 电路交换
  - 电话系统，呼叫建立连接后会建立一条专用电路
  - 特点
    - 固定带宽，固定线路，专用
    - 为每个呼叫预留资源（带宽，交换能力）
    - 非共享，保障性能
    - 需要呼叫建立过程
  - 如何做到预留资源?
    - 频分多路复用（FDM）：分不同频率信道传输，如：电视，广播
    - 时分多路复用（TDM）：分不同时段信道传输
  - 缺点
    - 一直占用（不管有没有数据交互）

- 包交换（也称为分组交换）
  - 互联网
  - 特点
    - 统计多路复用，按需分配，先请求先发
    - 资源共享，存在资源竞争问题，总请求远大于总资源量
    - 出口处可能大量包阻塞（阻塞严重会出现丢包）
    - 适合突发数据，如上网
  - 缺点
    - 包延迟
    - 数据拥塞
      - 严重时会丢包

4、协议

协议定义了网络实体间传输消息的规则（包括：消息格式，顺序...）

5、网络结构

网络是一个复杂系统

- 对于任意一个复杂系统怎么处理?
  - 模块化
  - 分治法

分层结构，每一层都规定对应服务

- 目的：应对网络系统的复杂性
- 劣势：效率比较低（存在调用过程），需要存储空间比较大（数据压栈）

互联网的分层

- 物理层(physical) -> 数据链路层(data link) -> 网络层(network) -> 传输层(transport) -> 应用层(application)
- 物理层
  - 通过线路（可以是有形的线也可以是无线链路）传送原始的比特(bit)流（“线上的比特”）
    - 一个物理层协议只完成一个节点到另一个节点的传送（单跳）
- 数据链路层
  - 通过物理网络传送包(package)
    - 包是通过网络层交过来的数据报(datagram)
  - 一个数据链路层的协议只完成一个节点到另一个节点的传送（单跳）(hop-by-hop, node-to-node)
  - 常用协议：PPP，Ethernet

网络层

把包里面的数据拿出来，进行路由选择(routing)，决定要往哪个方向传输

负责从源(source)通过路由选择到目的地(destination)的过程，达到从源主机传输数据到目标主机的目的(host-to-host)

常用协议：IP

传输层

网络层只把数据送到主机，但不会送到进程

传输层也称为端到端的传送(end-to-end)，为进程间传输提供服务的

负责进程与主机(host)间的传输，主机到主机(host-to-host)的传输交由网络层负责

常用协议：TCP，UDP

应用层

专门针对某些应用提供服务

常用协议：FTP（传送文件用），SMTP（发邮件用），POP3（收邮件用），HTTP（传送网页用）

6、协议栈

数据传输流程

源主机：消息源 -> 应用层封装 -> 传输层封装 -> 网络层封装 -> 数据链路层封装 -> 物理层传送

中间路由：物理层接收 -> 数据链路层解封 -> 网络层解封 -> 网络层封装 -> 数据链路层封装 -> 物理层传送

目标主机：物理层接收 -> 数据链路层解封 -> 网络层解封 -> 传输层解封 -> 应用层解封 -> 传输完毕

封装的数据都统称为协议数据单元(protocol data unit, PDU)，也称为包(package)

协议簇：不同层的协议所构成的网络

7、ISO/OSI 参考模型

ISO规定的七层结构

动因：TCP/IP当时还没出来，各自用各自的协议造成通信困难

开放系统互联模型(open system interconnection, OSI)

物理层 -> 数据链路层 -> 网络层 -> 传输层 -> 会话层(session) -> 表示层(presentation) -> 应用层

会话层

通过数据流建立会话关系

数据恢复

表示层

数据压缩、解压，加密、解密，数据类型、格式变换

8、OSI协议栈

网络层协议要完全相同

相邻两个节点的数据链路层与物理层协议要完全相同

网络层之上层协议要在源主机与目标主机处完全相同（对等实体）

9、丢失和延迟

丢失：如果传输队列里面的缓冲区没有空间了后来的包就会丢失

延迟

查路由表的处理延迟

包排队的排队延迟

把包发送出去的发送延迟

包在线路上传送的传送延迟

处理延迟、排队延迟没办法计算

发送延迟：带宽/包大小

传送延迟：传输距离/光速

应答过程：还需计算回来的时间

吞吐量：某段时间发送数据量/时间长度：单位时间实际传输数据大小

10、物理层

两种传输方式

- 模拟信号传输
  - 不包含二进制信息
- 数字信号传输
  - 包含二进制信息

物理层需要完成的工作

- 信息源 -> 调制/编码 -> 信道传输 -> 解调/解码 -> 传输目的地
  - 调制出来的结果为模拟信号，编码出来的结果为数字信号
  - 物理层介质
    - 双绞线（网线）、同轴电缆（电视信号线）、光纤（单模/多模）
    - 多模光纤：直径大、传输速率小、传输距离小、便宜

11、数据链路层

功能

- 网卡发送具有一定格式的帧，网卡接收具有一定格式的帧（帧里面有比特流）
- 差错检测：检测物理层传输的错误（加入校验码，一般不纠正，有错直接舍弃数据）
- 差错控制：协调冲突（通过多路访问链路的介质访问控制以减少冲突）
- 分帧

差错检测

- 比特错误：位错误
- 方法
  - 帧尾部加入校验码，目的地通过校验码进行差错检测
- 种类
  - 奇偶校验码：生成：看1的个数是奇数还是偶数
    - 一维奇偶校验：可发现一个bit的错误，不可纠正
    - 二维奇偶校验：可以纠正
  - 校验和：生成：将所有的比特序列加起来，如果有进位就再加1，最后再求反码
    - 检验方法：用相同算法算一遍看结果对不对
      - 通过模拟电路的方法可以更加方便地检测
  - 循环冗余校验码(CRC)：比特序列尾部加上3位0，通过除一个4位的码得到的3位余数即为校验码
    - 注意：除法过程中的减法为不进位减法，即直接进行抑或操作
- 包错误：丢包、错序、重复
- 方法
  - 1、反馈：确认是否收到包
  - 2、设定一个时间(timeout)，每隔一段时间发包，如果在这段时间内没有收包反馈则认为发生丢包
    - 太长：丢包代价大　　太短：已经收包的情况下误判断为丢包

停等协议

- 必须等到对方收到上一块数据后再发下一块数据
- 错误类型
  - 数据丢失
    - timeout时间内没有收包反馈则认为发生丢包
  - 反馈丢失
    - 确认信息丢失
    - 接收方会接收到相同的数据帧，出现重复错误
    - 接收方需要被检测出来（对包进行编号，检测编号）
  - 反馈延迟
    - timeout时间内没收到确认信息
    - 接收方会接收到相同的数据帧，出现重复错误
    - 接收方需要被检测出来（对包进行编号，检测编号）

停等协议效率不高：吞吐量小，带宽利用率（吞吐量/带宽）小

不停等传输

停等传输效率低，因此不等反馈一直发

问题：中途丢包

Go-Back-N

接收方检测包序号是否丢失

确认号发回来说明到这确认号之间的包已经收到（5代表1，2，3，4，5五个包都收到了）

如果有一帧丢失，即使剩下的帧收到了也认为没有收到

发送方在每一帧发送时都启动一个timeout计时器，如果timeout则重发这一帧及之后的帧

滑动窗口协议

每一帧都有一个超时时间(timeout)

超时后重传

直到上一帧的确认号(ACK)到来之后才会发下一帧

确认号：期待接收序号为n的帧，之前的帧已经全部收到，并依次顺序交给了上层

如果中间某帧丢失，那后面的帧收到以后不交给上层，确认号仍为丢失那帧的帧号，但也要返回确认信息

缺点：效率低

滑动窗口协议的改进1：go-back-n

原理：连发数帧，把丢失的及其之后的帧全部重传（如不这样做会导致错序或重复错误）

需要有发送窗口(SWS)和接收窗口(RWS)

发送窗口也称为发送缓冲区，它限制了在下一个确认到来之前可以发送多少未确认的帧

未确认的帧都会在里面被保留以准备将来可能发生的重传

假设SWS大小为n，那么在某一帧的确认到来前还可以再发n-1帧

发送窗太小可鞯会导致重复错误

接收窗口也称为接收缓冲区，它应与发送窗大小一样大

接收窗口会就将已经到来的还未交给上层的帧保留下来

缺点：效率还不够高（丢失帧之后的帧能否不重传）

滑动窗口协议的改进2：selective repeat（选择性重发）

实现机制：不仅要有ACK（因为ACK没告诉后面的帧是否成功接收），还要有否定性确认帧(NAK)

否定性确认帧：小于n的帧全部收到并依次交给了上层而第n帧丢失了，要求重发第n帧

当发送端收到NAK后，NAK所指示的那一帧及之后的帧的超时时间(timeout)会被延长

如果NAK丢失，NAK所指示的那一帧及之后的帧全部重传

如果没有接收窗就会出现重复错误

滑动窗口协议的改进3：selective acknowledge（选择性确认）

实现机制：第n帧收到以后会反馈发送端第n帧已经收到

延迟确认：每收到一定数量的帧再发回确认（可以少发确认号）

捎带确认：在接收方往发送方发送数据时捎带确认信息

PPP协议(point-to-point protocol)

应用：ADSL（PPPoE：point-to-point protocol over Ethernet）

VPN（PPTP：point-to-point tunnel protocol）

数据帧构成（从前到后）：

1Byte标志位(flag)：头尾标志位完全相同

1Byte地址：目标地址

1Byte控制位：没有序列号因此不可用滑动窗口协议

1~2Byte协议位：指明上层协议

小于等于1500Byte数据

2/4Byte校验码(FCS：frame check sequence)：帧校验序列（常用CRC-16或CRC-32）

1Byte标志位(flag)：头尾标志位完全相同

链路控制协议（LCP：link control protocol）
可初始化或结束链接
允许主站间对连接设置和加密方式进行协定
网络控制协议（NCP：network control protocol）
对于每个网络层协议都会有一个独立的网络控制协议用于封装或协定网络设置
IP使用IP控制协议(IPCP)来获取IP地址，子网掩码和DNS地址

12、局域网

特点
多路访问链路
只用一条线即可实现与其他主机的通信
采用共享链路及广播的方式
传输速率高，地理范围小

使用协议
介质访问控制(MAC)子层：用于解决冲突
以太网协议(Ethernet)：CSMA/CD
令牌环网络(Token Ring, FDDI)
WiFi
逻辑链路控制(LLC)子层（802.2）：用于面向连接的网络

Pure Aloha协议：
一有数据就发送不管别人
效率很低，冲突几率高
Slotted Aloha协议：
将时间分槽，只有在时间槽开始的位置才能发送（否则就需要等待）
效率较Pure Aloha翻倍
但需要有一个同步的信号，而且也会出现冲突

CSMA（载波监听多路访问）算法
发送前先监听载波，如果信道空闲立即发送，信道忙就等待
1-persistent（一监时） CSMA：持续监听直至信道空闲则立即发送
non-persistent（非监时） CSMA：不持续监听，随机等待一段时间监听一次（节约能量）
p-persistent（p监时） CSMA：信道空闲时以概率p发送，以概率1-p延迟发送；信道忙时下一时间槽再发送（适用于slotted）

CSMA/CD（带冲突检测的载波监听多路访问）
适用于以太网
边发送边监听，监听到冲突以后立即停止发送与冲突点都随机延迟一段时间，然后再使用CSMA算法
流程：
1、接收数据包分帧
2、监听信道，如果空闲就开始发送（检测96bit数据，如果这96bit都空闲才认为信道空闲）；如果忙持续监听直至信道空闲（1监时）
3、发送过程中如果没有检测到冲突则返回发送成功信号，否则立即停止发送并发送jam信号(32bits)，并随机延迟一段时间，继续监听信道（回到流程2）尝试下一次发送（注：如果发生了16次或以上的冲突将不再尝试继续发送，返回“发送失败”）
延迟算法：二进制指数退避算法(binary exponential back-off algorithm)
假设时间常数t为在10Mbps发送512位所需时间(51.2微秒)
首次冲突：在0 * t或1 * t中随机选择一个时间进行延迟
再次冲突：在0 * t，1 * t，2 * t或3 * t中随机选择一个时间进行延迟
10次及以后的冲突：在0 * t，1 * t… 1023 * t中随机选择一个时间进行延迟

接收算法

每个网卡一个专属地址

会将正确的帧收下，错误的帧丢弃

正确不仅指的是目的地址正确，还要模式正确（接收多播或广播数据需要预设模式）

帧格式

8Byte前导位(preamble)：用于同步，不是用来建帧分帧的

如何分帧?

在CSMA/CD中每一帧在发送前都会监听信道（96bits）

因此在帧与帧之间存在帧间空隙(inter-frame space/gap)

根据帧间空隙就可成帧

6Byte目的地址：说明接收主机的地址

此处说的地址为MAC地址，亦称为物理地址、硬件地址

地址全球唯一：6Byte地址中前3个Byte指示制造厂商，后3个Byte为序列号

U/L位：MAC地址的第一个Byte的第七个bit

0：全球地址（绝对地址）　　1：相对地址

I/G位：MAC地址的第一个Byte的第八个bit

0：单播　　1：多播

单播情况下目的地址即为接收主机的MAC地址

广播情况下全为1

为什么要设置目的地址?

因为多路访问链路中所有设备都会监听到链路上的数据，所以要定义给谁

网卡会根据地址进行判断是否接收下此数据

这样做比在操作系统层进行判断效率高得多

6Byte源主机地址

2Byte类型或长度：说明数据报类型或数据报长度

当此值大于1500时为类型，小于1500时为长度

类型是必需的信息

当此值表示长度时类型在哪里表示?

在pay-load（数据）前面加上

1Byte的DSAP，1Byte的SSAP，1Byte的控制位

3Byte的00-00-00

2Byte的类型

46Byte~1500Byte数据（pay-load）

为什么最少46Byte?

当A给B发送数据时，一旦发生冲突B应该要在A发送完成前反馈回信息

起初带宽为10Mbps，来回时间间隔为51.2毫秒

因此至少要发送10Mbps * 51.2us = 512bit = 64Byte的数据

因此pay-load部分至少为46Byte（注：不算8Byte前导位）

如果100Mbps带宽怎么办?

限制发送距离

如果1000Mbps带宽怎么办?

距离不能过分减少

采用“载波延伸”的办法（加入冗余数据把帧拉长）

浪费带宽多?

采用“帧突发”的方法：对于A->B的连续多帧，后面的帧不再拉长

4ByteCRC-32校验码

以太网的物理层（仅作了解）

以太网所用协议相同，速率不同主要是由于物理层线材和编码方式上有差别

10Mbps：少用光纤，采用曼彻斯特编码
1000Mbps：用双绞线或光纤
双绞线时双向同时达到一对线250Mbps，四对线双向同时达到1000Mbps
10Gbps：用光纤
中继器与交换机
中继器（集线器）：用电子线路模拟冲突信号；无存储功能，收到后立即转发（泛洪）；半双工
交换机：具有存储转发功能

13、网桥

网桥(bridge)：将多个局域网连接起来（此时每个局域网成为网段(segment)）
功能：
1、泛洪（flood）
广播设备：从一个端口进来，就立即转发到其他端口上去
2、存储转发（forward）
泛洪效率低，容易引发冲突
因此要有功能实现存储转发：将数据只转发到特定的局域网上
通过查询MAC地址表实现
有两种特殊情况
1、MAC地址表中查询不到：进行泛洪
2、A网段的数据要传回A网段：丢弃此数据
3、过滤（filter）
同个局域网上传送的数据也会被传到网桥上（因为网桥与他们局域网在链路上存在连接）
网桥应该丢弃这些数据，否则会出现重复错误
4、自学习（self-study）
网桥会根据MAC地址表进行存储转发
但是手动设置MAC地址表相当麻烦（尤其是节点多且移动频繁的网络上）
因此网桥需要自学习得到MAC地址表

自学习算法：
初始化MAC地址表为空
MAC地址表记录下每一次发送端所属的网段，并更新生存期(TTL：time-to-live)
生存期是为了应对目标节点关闭或移动的问题
MAC地址表中过了TTL以后的项会被删除
每有一次数据交互，对应的MAC地址表中项的会重新计时
证明：
证明发送端和网桥是连通的
存在TTL

14、生成树算法（802.1D）

对于存在多个网桥的情况下可能会出现回路，这会导致严重错误
尤其是广播的情况下…
提出生成树算法来解决这个问题
S1、找根节点：网桥下标最小的为根节点
每个节点都扩散消息。如果一个网桥收到比自己小的消息就不再扩散，直至只有一个扩散
S2、其他网桥找一条到根节点的最短路径（多条最短路径下往下一个网桥下标小的走）
网桥出这条路径的那个端口称为根端口(root port)
S3、每个网段也找一条到根节点的最短路径（多条最短路径下往下一个网桥下标小的走）
如果网桥也相同那就往连接端口号小的走
这样每个局域网就会对应一个网桥，此时这个网桥称为指定(designated)网桥
此时网桥上语气连接的端口称为指定端口(designated port)
S4、断开其他路径
非根端口、指定端口的端口为阻塞端口(blocked port)

15、VLAN（虚拟局域网，802.1Q）

同一局域网的不同主机间隔离问题

- 主机分组标记，同一组内可以交换信息，不同组间信息隔离
- 网桥只往相同VLAN ID端口转发数据
  - 默认一开始全部属于VLAN 1

不同局域网间的隔离交互问题

- 网桥间要存在干道(trunk)，通过网桥判断信息交互
  - trunk两端的网桥接口为trunk接口，trunk接口往来的帧要加VLAN ID
  - 除了trunk接口以外均为主机接口，主机接口往来的帧不允许加VLAN ID
- 网桥不仅往相同VLAN ID端口转发数据，还往干道上通过trunk接口发送数据和VLAN ID信息
  - 往帧上面加上VLAN ID信息，否则会被默认为VLAN 1
  - 仅仅是往trunk上发的才需要加上VLAN ID信息，往主机接口上的不能加入VLAN ID信息
- 其他网桥接收到以后根据VLAN ID信息后再往相同VLAN ID端口转发数据
  - 对于从主机接口收到的帧，没有VLAN ID信息，认为是普通的帧
    - 收到以后会指定一个VLAN ID给它
    - 如有VLAN ID信息则会被丢弃
  - 只有从trunk接口收到的帧，如果没有VLAN ID信息就认为是VLAN 1

或者根据网卡MAC地址来指定VLAN ID

- 上面讲的通过设定端口的办法为静态（static）VLAN
- 根据网卡MAC地址进行设定的办法称为动态（dynamic）VLAN

如何往帧加上VLAN ID信息

- 在protocol的2Byte之后加上2Byte信息
  - 内有3bit的PRI
  - 1bitCFI
  - 12bit VLAN ID
- 对应以太网的帧结构
  - protocol的2Byte对应以太网的类型
  - 从新加上的2Byte的信息开始到payload结束都为数据部分

生成树的改进

- 生成树可以共用一棵（CST：common spanning tree）
  - 公用可能会导致根节点的流量压力很大
- 也可以每个VLAN一棵生成树（PVST：per-VLAN spanning tree）
  - 树太多使得生成树算法的开销太大
- 也可以结合两种（MSTP：multiple spanning tree）

16、交换机（switch）

功能：从输入端口收到以后将其转到输出接口

- 使用透明网桥算法
- 相当于多端口网桥

结构：

- 总线结构，总线带宽大于每一个接口的带宽
- 共享高速存储器(high speed memory)

工作方式

- 存储转发(store and forward)：先把整个帧收下来，然后根据透明网桥算法转出去
- 直通(cut through)：不用收整个帧，只用收目的地址，然后就根据透明网桥算法转出去
- 无碎片(fragment free)：至少收64个字节再转，以免发生冲突时取消发送的情况下不会发碎片出去
- 自协商
- 全双工(full-duplex)：由于交换机具有存储转发功能，因此具有同时发送和接收的功能，无冲突
  - 因为没有冲突，因此只要两端接口协议相同，会自动关闭CSMA/CD算法
- 自动翻转(auto-mdix)：网线应该交叉接线才能连通，因为交换机和主机指定接口收发
  - 如果是直连的，在交换机端会自动翻转

17、令牌环网（token ring）

以太网不能保证帧一定能发送出去，而且可能出现冲突

令牌环网（802.5）：另一种以太网

有一个特殊的帧作为令牌，令牌绕环而行，截到令牌的主机(host)才会发送数据

每个主机拿到令牌有限定时间，之后必须释放令牌

发送的数据帧也会绕环而行，目标主机发现是自己的就将其备份，把副本收下来

最终数据帧被发送方收走

有优先权，优先权大的先拥有发送权

故障：

拿到令牌的主机故障会导致令牌丢失

需要有一个监控令牌的主机，如果发现线路上令牌丢失会重新设定令牌

如果监控令牌的主机也故障

其他主机也需要监控监控令牌的主机

18、其他网络

FDDI：类似于令牌环的工作原理，光纤连接

802.11(Wi-Fi)

源路网桥(source route bridging)：在发送帧在线路上传送时记住所有转发信息，到达目的地后将信息返回给源主机，源主机以后会把这些信息加到发到该目标主机的所有帧上

19、网络层

主要功能：

通过路由选择(routing)使发的包能通过物理网络（例如：以太网，令牌环网）到达目的地

存储转发(forwarding)

拥塞(congestion)：网络层收到大量数据导致超出网络处理能力

流量控制：在两个节点之间控制包的传输速度，如果太快会超过缓冲区空间，出现溢出错误(overflow)

网络架构

固定位速率，不拥塞，保证带宽、次序，不丢包，保证实时性

可变位速率（保证位速率），不拥塞，保证带宽、次序，不丢包，保证实时性

可用位速率（保证最小位速率），

网络类型

线路交换：FDM，TDM

包（分组）交换：

数据报：

包每到一个路由器都看一下往哪里走，每一步独立地走（不保证带宽，时间比较长（虚拟电路一步到位），但出现故障的时候可以比较灵活地进行处理）

虚拟电路：

面向连接，首先找到源节点到目的节点的通路，找到通路以后目的节点反向向源节点发信号，告知通路的情况（每个链路对于发往不同地方的包（对于不同连接路径）一个虚电路的号（VCI(virtual circuit identify)值），一个端口可能对应多个VCI，每过一个路由都记录下一条记录（进路由的端口，进路由的链路VCI，出路由的端口，出路由的链路VCI）），最终使得源节点到目的节点都沿着这一条路径走（路由会根据收到的包的VCI值和端口，根据查表结果修改包的VCI值，并往查表得到的指定端口上发出去）。（可以通过VCI控制每条线路的访问量，可以保证带宽，但如果某条线路出现故障代价很大）

种类：

交换式虚电路：发送完了以后目标向源发信号释放虚电路

永久虚电路：发送完了以后不释放虚电路（相当于专线）

20、IP协议

一、
版本号：4位
IHL：头长度4位
服务类型：8位（一般路由器不管）
    原来
        服务类型选择：3位，数值越大服务越好
        低延迟选择：1位
        高吞吐量选择：1位
        高可靠性选择：1位
        低开销选择：1位
        保留位（无实际作用）：1位
    现在都整体打包：差分服务
总长度：16位

二、
标识：16位，每发一个会加1
（空白1位）
DF位：标识是否可以拆分IP数据帧，对于不可拆分又长度超出以太网传输限制的会直接丢弃
MF位：标识分成以太网帧以后是否最后一个帧，最后一帧为0，其他的均为1
偏移量：13位
    分成以太网帧以后每个帧头离原IP数据帧头部的偏移量
    中间帧丢失可以通过这个进行估算丢了多少数据
    但最后一个帧丢失了就不行，因此要存在MF位
    offset由于只有13位，因此最终记录下的为离原IP数据帧头部的偏移量除以8以后的结果
        因此在IP数据帧拆分的时候要保证数据长度可以被8字节整除

三、
TTL：生存期8位：
    限制包的存在时间，防止包在网上兜圈，停留太久
    每经过一个路由器就减1
    全球最远只用经过32个路由器
    如果路由收到TTL为0的包会丢弃并往源主机发一个ICMP包，说明发生丢包
协议：上传协议8位
校验和：16位，只对头部进行校验，不对数据进行校验

四、
源IP地址：32位

五、
目的IP地址：32位：经过的每个路由器都会取出目的IP地址进行选路

六、
选项（可变）+ 填充：每行一共32位（可以0~10行，即0~40字节）
    代码：1B
        松散源路由：指明一系列必须经过的路由器
        记录路由：记录下每个转发路由器的IP地址
            数据时每个地址4B，最多允许40B（即最多经过10个路由）
        严格源路由：指明一系列必须且只能经过的路由器
    长度：1B
    数据：可变（和下面的数据不同）

七、
数据：实际传输的数据

以太网中IP帧：
将IP帧放到以太网帧的data部分，然后类型字段说明为IP帧（0x0800）
可能要将IP帧进行拆分，分成几个帧（以太网数据部分最多1500字节，IP帧还有头部）
因此会存在次序问题，这也就是为什么会存在偏移量、DF、MF位的原因

21、IP地址

IP地址是全局地址，点分十进制表达（一共四个数，每个数一个字节，一共32位）

第一部分为网络部分，说明是在哪个网络里面

网络前缀的确定方法：

A类网：第一位为0：后面的7位为网络部分（最多容纳128个网段，但1600万+的主机）

B类网：前两位为10：后面14位为网络部分（16384个网段，65536个主机）

C类网：前三位为110：后面21位为网络部分

D类网：前四位为1110：后面全为多播地址

E类网：前四位1111：被保留

第二部分为主机部分，说明是在这个网络里面的哪台主机

之前的分配方法比较浪费，32位本来可以容纳4G的地址

比如说一个网段里面只有10个主机，就会产生大量浪费（C类网每个网段可容纳256个主机）

把主机部分拆出一部分出来作为子网号

比如10台主机只用4位的主机号，因此可以拆出4位的子网号

但注意主机部分不能全0，也不能全1

即2台主机本来只用1位的主机号，但必须分配2位

但一开始没有这个机制?

加上子网掩码：网络部分和子网号对应子网掩码为1，主机部分对应子网掩码为0

如果每个子网的数量变化很大?

四个子网分别20、10、50、100台主机

解决办法：子网可以再划分：哈夫曼编码避免IP地址冲突

还是用子网掩码进行区分：变长子网掩码

无类域间路由选择（CIDR）

将多个网段组成一个超网

比如：把多个C类网合并在一起

网络部分要拿出几位出来给到主机部分

从而网络部分、子网部分的边界都可以自由移动

通过子网掩码进行区分边界

将主机部分全部置1，把这个编码记下来

（注：子网部分也在主机部分里面）

特殊地址：

32位全0：匿名源地址

除主机部分全0：说明源地址和目的地址都在一个网段里面

32位全1：IP分组广播地址，并不会全网广播而是发送给网关

主机部分全0：检测是否在一个网段上

主机部分全1：进行本网段的广播

网络部分为127：只会发给自己（不会从网卡发出去），用于测试

其他的都起码会经过网卡发出去

私有地址（网段与网段间可以重复，但是转出去有算法（NAT：私有地址转换技术）进行转换）：

10.0.0.0 ~ 10.255.255.255

172.16.0.0 ~ 172.31.255.255

192.168.0.0 ~ 192.168.255.255

网络地址转换法：把私有的源地址转换成目的地址（目的地址不会用私有地址），而当反馈送回来的时候又要将全局的目的地址转换为私有的目的地址

映射时不仅对IP地址进行映射，也要对端口号进行映射

端口号一共两个字节，因此一个IP地址可以给65536个用

IP地址是分层的，因此通过IP地址可以进行定位

多播地址
需要和MAC地址做映射

22、路由表

路由表中项目：目的网段号 + 子网掩码 + 下一跳的跳点 + 从哪个接口进行转发
查表规则：得到IP分组以后取出目的地址

1、目的地址和子网掩码相与
2、得到的结果与目的网络号进行匹配
3、如果存在匹配项看是否有下一跳的跳点
最长匹配原则：如果有多个匹配项看子网掩码哪个1的个数多就匹配那一项（地址更加具体）
存在默认路径一定与任何项进行匹配，保证一定发出去
4、如有下一跳的跳点就往下一跳发，如果没有下一跳的跳点就往对应主机发
没有下一个跳点说明已到该网段，就将主机部分往下传

23、ARP（地址解释协议）

在数据链路层传输数据的时候需要知道目的地的MAC地址，但现在我们只知道目的地的IP地址
因此会发一个ARP广播帧，只要对应IP地址的主机收到以后就会将MAC地址封装成帧以单播方式发回去

这里面涉及到两个帧：ARP请求帧、ARP响应帧
ARP帧中包含：源MAC地址，目的MAC地址、源协议地址、目的协议地址

知道以后会存下来：每个存放的都有一个生存期（TTL），避免换网卡、IP地址更换等情况
对于多播的情况下需要预设最后23位全1，对于广播的情况下需要预设48位全1

24、其他协议

DHCP（动态统计配置协议）：IP地址的分配与续租
ICMP：如果出错（比如TTL到期或者目的主机不可达）会发回ICMP包给源主机

ping用的是ICMP中的echo request
常见两种错误：

连接超时：到了目的主机但是目的主机的反馈回不来，错误信息发回给收包的主机因此发包的主机收不到信息，因此超时

目的主机不可达：根本到不了目的主机，错误信息发回给发包的主机

25、交换机与路由器

交换机

数据链路层上的，有扩散广播的功能
从一个端口到另一个端口转发帧
存在MAC地址表
只在一个子网里面用
交换机通过虚电路的办法可以实现路由器的功能

路由器

网络层上的，没有扩散广播的功能，唯一选择一条路走，选不到就丢包
从一个目的网络到另一个目的网络转发IP分组
存在路由表
把子网和子网相互连接起来

26、路由协议

路由表不可能全部都要自己设置，因此需要有机制自动建造路由表
路由表中最关键的是目的网段和下一跳的跳点

方法：将网络展开成图，每条路径赋予权重，相当于求解每个路由到可能的目标主机的单源最短路径问题（Dijkstra算法，距离向量-链路状态方法等路由算法），只要能求出最短路径就可以得到每个路由到目的地的唯一路径，从而得到唯一的下一跳的跳点。最后得到的路由表中项称为动态路由（自己设置的称为静态路由）

问题一、但全世界网络这么大，怎么成图?

全世界的网进行了划分，每个划分块自己管理自己的路由表，每一块有一个自动系统(autonomous system)。自动系统指由同一个机构管理的网络。自动系统内部用路由协议建路由，自动系统与自动系统之间有网关，但不能够用路由协议建路由，需要用外部网关协议（EGP协议）建立路由，发送到对应的网关后再用内部网关协议（ISP协议）在自动系统内部建路由。

内部网关协议（ISP）
- RIP协议
- OSPF协议

外部网关协议（EGP）
- BGP协议

问题二、如果网络层出现回路，虽然有TTL不会出现死循环但是会浪费带宽，证明最短路径算法不会出现回路

反证法：假设最短路径会出现回路，那么从一个路由到一个主机必有至少两条路径，那么总有一条是最短的，另一条不是最短的，出现矛盾

27、RIP（路由信息）协议

目的：通过距离向量方法在自动系统内部建立动态路由
寻路原理：询问邻居路由到目的点的距离并结合到邻居的距离选最短路
技术细节：

1、每30秒每个路由器都把自己的路由表发给所有邻居（里面包含到目的主机的距离）
   - V1：采用广播方式发送，因此与本路由直接相连的路由都可以收到
   - V2：采用多播方式发送，因此只有启用rip V2的才会收到
     - 比V1多出了网络号和子网掩码
2、所有路由器都利用邻居发来的路由表建造自己的路由表
3、初始时每个路由器只有直连网路由，也就是路由器到相邻的路由器的路由（里面要有权重）
4、任何两个直连的路由器之间的权重默认都为1，也就是按跳计数
5、规定16跳等于无穷大，相当于不可达，即任何两个节点间最多15个路由器

例子：假设两个路由器R1，R2相连
R1路由：

目的网段	距离（跳数）	下一跳的路由
N1	5	R7
N2	6	R7
N3	1	-
N4	2	R2
N6	3	R5
N7	6	R2

R2路由：

目的网段	距离（跳数）	下一跳的路由
N1	5	R6
N2	4	R4
N3	2	R1
N4	1	-
N5	6	R6
N7	9	R4

当R2送路由表给R1以后，R1更新后的结果是：

目的网段	距离（跳数）	下一跳的路由
N1	5	R7
N2	5	R2
N3	1	-
N4	2	R2
N6	3	R5
N5	7	R2
N7	10	R2

问题：

1、为什么规定30秒的转发周期?

转发周期不能太小，否则路由表就占据大部分的带宽，影响网络正常工作

2、为什么规定16跳等于无穷大?

假设一个新连的网需要ping通，假设两台PC机之间有m台路由，那么最大路由表延迟为(m - 1) * 30，平均延迟(m - 1) * 15，当m太大时会出现慢收敛问题

假设H1 -> R1 -> R2 -> H2，一开始R1的表中保存到H1的距离为1跳，R2的表中保存到H1的距离为2跳。此时H1与R1之间的连接断开，R1的表中保存到H1的距离为无穷大，但是此时R2的表送了过来造成R1的表中保存到H1的距离为3跳。但实际上都到不了，而且出现环路。16跳等于无穷大从而保证了R1和R2最后这一项都变成了无穷大

但没有根本解决这个问题，因此有触发更新机制。当连接断开时立即通知邻居然后通过邻居再传播出去。但这又引发了下一个问题，触发更新丢失怎么办？因此有水平分割机制，从某个接口学来的路由不要再回送到那个接口上面去，这才真正解决计数到无穷的问题。

3、在180秒内邻居如果没送过来路由表

到邻居的距离改成无穷大，240秒以后才会真正删除这个路由

4、RIP原始协议到每个目的地只有一条路由，因此如果出现多条路由可能会出现丢包

28、OSPF（开放最短路径优先）协议

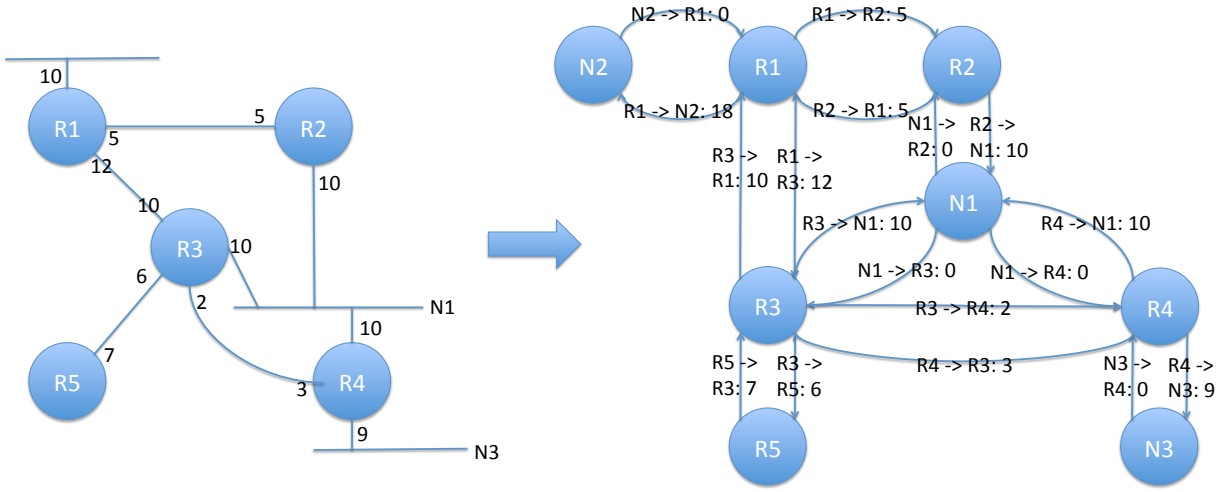
目的：通过链路状态方法在自动系统内部建立动态路由
寻路原理：先查询到整个的走法然后再走
技术细节：

1、每个路由器都保存本自动系统的整个网络图（所有路由器保存的图都相同）
2、所有路由器都求到图里面所有网络的最短路径：采用Dijkstra算法求最短路径
3、通过最短路径构造路由表
路由器A找到网络N1的最短路径以后把这条最短路径上下一个路由器作为下一跳点

问题：

1、如何构造这样的一幅图

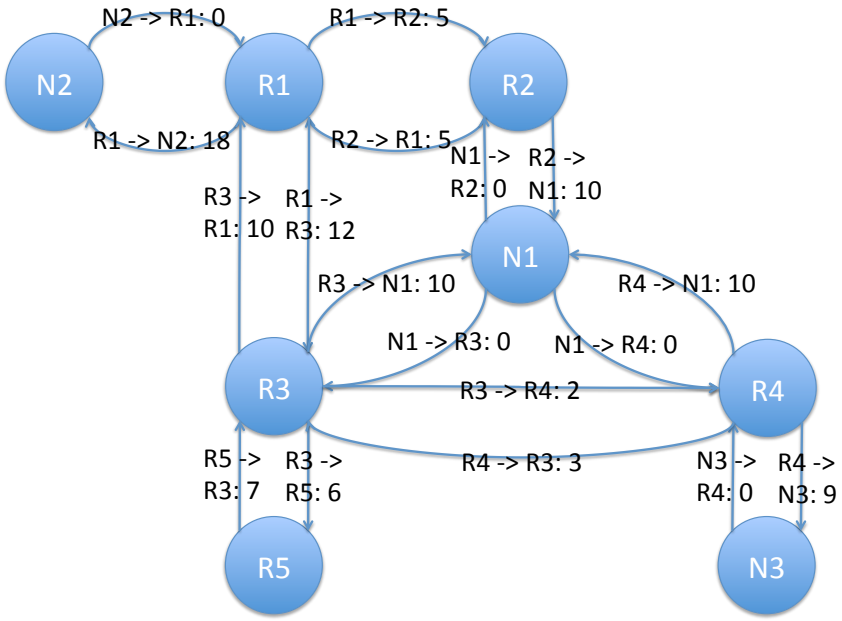
路由器链路状态通告（router LSA）：把每个路由器所有的发出边都放在里面
举例：对于R1来说
<R1, N2>
<R1, R2>
<R1, R3>
然后每个路由器把自己的链路状态通告泛洪出去从而本自动系统内部所有路由器都可以收到
但这样从网络发出去的边没有发出去?
因此存在网络链路状态通告（network LSA），里面存放每个网络发出的边
只有中转网才有网络链路状态通告
这个信息由指定路由器收集
举例：对于N1来说
<N1, R2>
<N1, R3>
<N1, R4>
但网络本身是不会发送的，因此要选一个指定路由器代发
选路由器ID最大的进行代发
网络的ID就用网络号
路由器的ID的选取见问题4
举例：
图的构造



2、如果有这样一幅图怎样实现这个协议

图的特征：有向图，两节点间不同方向可能权重不同；规定从网络到路由（末端网，其他网为中转网）为0

举例：下面有这样的一幅图



最后得到R1的路由表

目的网段	距离	下一跳的路由
N1	15	R2
N2	18	-
N3	23	R3

3、普通相邻和全相邻

普通相邻只要求A往B发，B能回复A

全相邻要求相邻两个路由器的能交换完整的路由器LSA

4、指定路由器选取问题

每40秒选取一次

先看优先权

然后再看路由器的ID，选ID最大的

会有2个，一个是指定路由器，另一个是备份指定路由器

如果指定路由器出现问题就由备份指定路由器进行转发

5、路由器ID问题

先看有没有直接配置的

如果没有看有没有loopback接口的IP地址里面最大的

因为loopback接口是软接口，会一直处于活动状态

如果还没有看活动的接口里面IP最大的

6、链路状态发生变化：存在触发更新机制

每30秒触发更新一次

当每个路由器存放的图不一样的时候会产生环路

触发更新时所有路由器都立即更新，不会像RIP一样出现慢收敛问题

7、分区问题：OSPF是分区的，每个区路由器保存自己区内部的图

如果区合区之间的路由器称为区边界路由器（ABR），从而保存了不同区的多张图

区边界路由器都需要加到所有区（即使不相邻）的网络的最短路径，作为节点加进去

从而区内所有的路由器都可以找到到区外的网络的最短路径

这样就节省了到区外所有的路由器的最短路径

但分区的方法很少用

OSPF分组

1、hello包

用于与直接相邻的路由器交换信息，建立相邻关系，并定期(默认10 秒)告知对方自己的存在

2、Database Description包

用于相邻路由器之间交换LS数据库中的LSA头部，以便确定过时或缺少的LSA

3、Link State Request包

用于向相邻路由器请求传送指定的LSA

4、Link State Update包

用于每30分钟、链路状态变化和被请求时传送完整的LSA

5、Link-StateAcknowledge包

对收到LSA进行确认。一个确认分组可以同时确认多个LSA

指定路由器除上述作用外还有两个作用：

1、当LSA同步的时候，只和指定路由器同步

2、当LSA转发的时候也只通过指定路由器进行转发

29、network命令

说明连接了哪些直连网，并需要将这些网络的信息发送出去

rip协议是有类网，不需设定子网掩码，只看网络部分

ospf协议用通配符进行匹配，需要设定子网掩码，网络部分要完全匹配，其它部分任意

30、BGP协议

自动系统之间网络要交互不宜采用前面两种协议

OSPF：图太大

RIP：跳数只有15，跳数不够，加大跳数就会出现慢收敛问题

共同问题：最短路径并不合适，因为最短路径权重可以随意由管理员进行修改

BGP协议负责将自动系统内的网信息给到自动系统外部

采用方法：扩散（扩散只往启动BGP协议的路由器进行扩散）

扩散可能形成回路

避免自动系统外部回路：记录经过的自动系统的序列号，不经过同样的序列号

避免自动系统内部回路：只能由自动系统内部收到的第一个路由器根据自己的路由表进行本网段内的发送，别的路由器没有转发的资格（无中转）

原因：要让全世界任何一个路由器都能通

31、多播

多播的路由问题：怎样到达想要达到的多个主机

一种思路：进行扩散，由上层决定是否接收：简单但浪费带宽

另一种思路：将主机和路由器连接成树状，在树状节点上进行传输，如果某个枝干连接出来的所有主机和路由都没有想收的（由协议进行判断），就不往这个枝干上进行转发。而其他枝干都会收到。

32、传输层

传输层是实现端对端的传输，应用层进程得到消息交给传输层后变成数据段，然后交予网络层。

主要有两种协议：

TCP协议：面向连接可靠有序的传输，存在反馈消息，无错序，无重复，不丢包，进行流量和拥塞控制。数据就像在管道上面传输一样，先发的一定先到。通过字节流的传输方法，因此没有边界，接收方就看成一个个字节流进来，接收端再进行封装。

UDP协议：不可靠无序传输，没有反馈消息，可能错序，丢包，没法实现控制，存在边界（超出边界会出现数据截断）

但都没有延迟和带宽的保证（因为因特网本来就没有保障）

方法：将进程绑定到指定IP的指定端口号上，对于发往指定IP指定端口号的数据发往对应的进程

UDP帧格式：

一、源端口号（16位），目的端口号（16位）
二、长度（16位），校验和（16位）
三、数据

在校验和里面会有加上伪头部信息，比如说源IP地址、目的IP地址

端口分为下面三种情况：

知名端口号
注册端口号
动态端口号

TCP通过源端口号，目的端口号，源IP地址，目的IP地址进行指定，服务器的端口号和IP是绑定的，客户端的IP地址和端口号是操作系统根据实际情况指定的。

TCP帧格式：

一、源端口号（16位），目的端口号（16位）
二、序号（32位），一定要有
三、确认号（32位），不一定要有
四、头部长度（4位），0（6位），标识（6位），窗口大小（16位，进行流量控制，由接收方告诉发送方然后由发送方进行控制）
五、校验和（16位），紧急数据指针（传送带外数据，说明数据部分中哪些部分是控制指令16位）
六、选项
七、数据

标识：

URG：是否有紧急数据
ACK：确认号是否有效
PSH：告诉接收方将缓冲区的数据尽快交予上层
RST：重置连接
SYN：同步标识
FIN：结束标识

TCP的完整过程：

1、建立连接
2、连接建立后可以传输数据（全双工，完全对称）
3、数据发送完毕后释放连接（双发都可以发起）

TCP的连接建立方式

三步握手协议：第一步客户机向服务器发送SYN同步信号x，服务器接收后第二步服务器向客户机发送SYN同步信号y和ACK确认信号x + 1（x + 1就是客户机给服务器的第一个数据的编号，同时会告诉发送方滑动窗口的大小（每次发送的数据段大小）MSS，限制发送滑动窗口的大小WIN（告诉发送端将发送窗口大小调整为WIN）），然后客户端再向服务器发送ACK确认信号y + 1

一次握手的问题：
两次握手的问题：

TCP的发送：按字节进行编号（每个字节一个号），每一个数据段都用第一个字节的编号作为自己的编号。这样做丢失时可以知道丢了多少数据。数据段的大小为MSS长

TCP的连接释放方式
发起释放的主机发送FIN（结束发送数据）结束信号x，接收方收到以后看自己还有没有数据要发，如果也没有了就发回ACK确认信号x + 1和FIN结束信号y（这两个可以合在一个包里面，也可以分开；同时也存在FIN同时发送的情况），然后发送方再发回ACK确认信号y + 1。在发送方再发回ACK确认信号y + 1后需要等待两个MSL（maximum segment lifetime）的长度，保持这个连接。因此在这段时间内发起的重连不回再连回原来的端口号，以避免各种异常情况（比如中间节点保留数据的情况）。

TCP的丢失处理：
丢失后会超时，超时后重传。在链路层每一帧都有一个超时定时器，但在TCP里面总共只有一个超时定时器（给到期待接收的帧用）。因此会比链路层慢。
选择性确认：如果接收端收到
提高速度的方法：重复ACK：如果接收方收到重复的对于某个帧的ACK（说明发生丢失）立即重传

**********以下内容未整理，先不要参考**********

超时时间（timeout）计算

sample RTT的情况：
sample RTT：突然下降或上升的情况
sample RTT：上升

alpha的取值问题：
太大：趋于1：完全等于sample RTT：如果RTT变化立即变化就会立即影响，对于突然下降的情况就会立即调整，后面的就会超时
太小：趋于0：完全等于原来的estimate RTT：无法适应实际RTT的变化，导致上升的情况无法适应

deviation

TCP的拥塞控制
如果网络拥塞谁都传不了，

加性增加，乘性减少（AIMD）算法：
有一个拥塞窗口，发送窗口大小实际为Adv WIN的大小和Cong WIN的大小取最小值
Cong WIN
加性增加：一开始为1MSS，发送成功后变成2MSS，再到3MSS
乘性增加：一开始为1MSS，发送成功后变成2MSS，再到4MSS
拥塞判断：发ICMP消息或者根据超时重传
出现拥塞以后指数级减少MSS

慢启动(slow start)算法：
从1MSS开始指数增加Cong WIN的大小
Tahoe：
增加到超时以后立即降到1MSS
从下一次开始增加到上一次超时时间的一半指数增加，到后面就线性增加
Reno：
如果没发生超时但ACK到的时间？？

长肥管道的序号问题：
如果带宽很大，使得发送窗口很大。但是序号只有32位，循环使用，因此可能出现信号回绕
因此加上时间戳TS，序号相同的根据TS进行区分

Winscale？？

死锁问题：

WIN1000丢失会出现死锁问题

发送少数据问题：

如果发送的数据很少需要频繁发送

Nagle算法：第一个包无论多少字节都发过去，第二个包开始把中间的数据再发过去。这样在往返时间少时，往返时间长时。如果超过MSS时就立即发

Clark算法：接收方取到足够大的数据量（接收窗口的一半或以上）以后再交给上层发回ACK