计算机网络（2023-2024 秋）期末重点

xx

2024 年 11 月 14 日

1 直连网络

网络模型：从模块化到网络分层。由于模块众多，且每个模块都需要相应的接口，所以维护起来相当困难。因此，必须将网络分层。

- OSI 七层网络模型
  - 应用层：应用协议：HTTP，FTP
  - 表示层：加解密/数据格式化
  - 会话层：进程管理/双工/半双工/单工/断点续发
  - 传输层：实现端到端的数据传输
  - 网络层：分组的路由，实现主机到主机的通信
  - 数据链路层：提供点到点的数据帧传输
  - 物理层：处理链路上，原始比特的传输
- TCP/IP 体系结构
  - 应用层：数据表示、数据加密、回话控制
  - TCP/UDP：定义端到端协议，为应用程序提供可选择的逻辑信道
  - IP 层：网际协议，确定分组转发路径，使主机可以把分组发往任何网络
  - 子网层：多种网络协议，如以太网协议

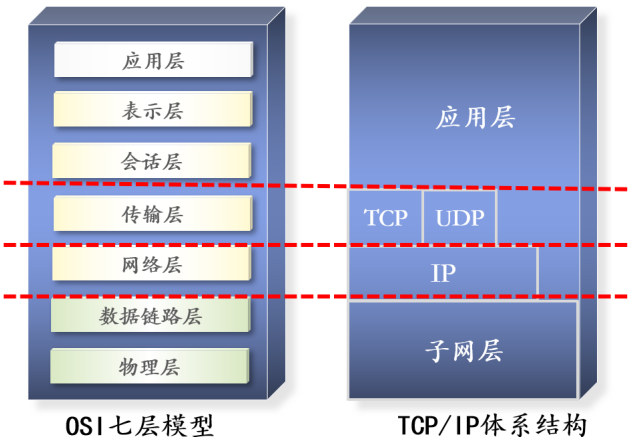


图 1: OSI 与 TCP/IP 模型

数据的封装与解封（掌握）：首先应用数据传送到应用层，加上应用层头部后，成为 应用层 Data，再传送到传输层。加上传输层头部后，成为 传输层报文（message），再传送到网络层。加上网络层头部后，成为 IP 分组 ，再传输到数据链路层。在数据链路层，加上链路层 头部和尾部 ，成为 数据链路层帧 ，再传输到物理层。在物理层，将 bit 数据编码调制后，再传输到信道中。到了接收端后，再由下而上，逐层去除数据包的头部及尾部（解封）。最后将应用数据发送给应用进程。

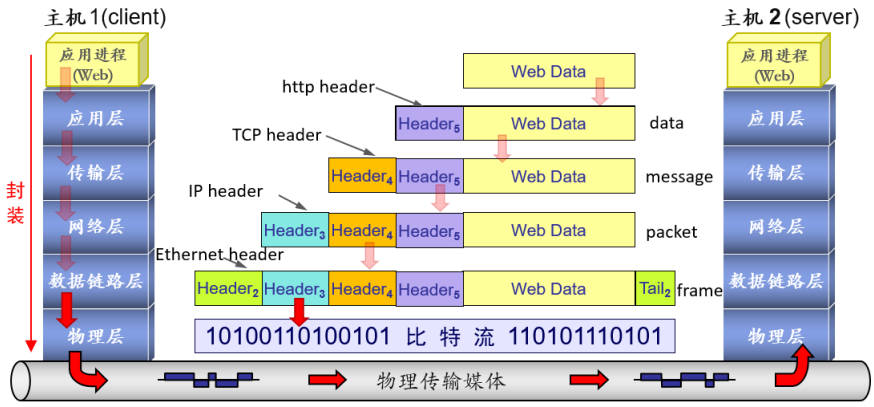


图 2: 数据封装与解封

差错检测：bit 数据在传输时，由于收到干扰，因此在接收端收到的 bit 数据，可能存在差错。通过在数据帧中，加入冗余信息的方式来实现差错检测。当接收端检测出错误数据时，可以通知发送方重传数据副本（重传机制），或通过加入的冗余信息，重新构造正确的数据（纠错码）。

组帧
- 面向字节的组帧
  - 起始标记法：使用特定字符表示帧的开始和结束
  - 字节计数法：将一个帧中的字节数放在首部的一个字段中
- 面向比特的组帧：使用特定的 bit 组合代表帧的开始和结束。使用比特填充法区分数据与特定的 bit

差错检测
- 奇偶校验
- 校验和
- 循环冗余校验（CRC）

单向奇偶校验/单个奇偶校验
- 偶校验：d bit 数据 +1 bit 校验位 = 偶数个 1
- 奇校验：d bit 数据 +1 bit 校验位 = 奇数个 1

只能检测奇数个比特错误。因此失效的概率为 50%

二维奇偶校验：将 d 位 bit 信息，划分为 i 行 j 列。因此总共拥有 i+j+1 个奇偶校验位。可以定位出错的 bit 并实现偶数比特错误的检测。

图 3: 2 维奇偶校验

校验和：发送方将传输的所有字相加，并将相加的结果作为校验和。接收方做相同的工作，并与接收到的校验和比较，以判断是否出错。校验和的冗余较少，只有 16bit。并且在上层软件中，很

容易实现。它的检测能力有限，如出错增加的值与减少的值相同时，无法检错。但是经过验证表明，这种形式的校验已经足够了。

循环冗余校验：n 次多项式可以表示 n+1 比特的信息。因此收发双方约定一个 k 次幂的除数 C(x)。发送方将比特消息 M(x) 转换为加了 k 比特冗余的消息 P(x)。接收方将 P(x) 除以 C(x)，如果不能整除，则代表传输出错。常见的多项式 C(x) 有 CRC-8，CRC-10，CRC-12 等。

- 媒体共享
  - 静态划分信道
    - 时分复用：将不同的用户分配到不同的频带
    - 频分复用：将时间划分为等长的时分复用帧（TDM），用户在 TDM 帧中占用的位置不变
    - 波分复用：将用户分配到不同的波段
    - 码分复用：使用伪随机码扩频，在接收端使用相同的伪随机码解调不同用户的伪随机码相互正交，因此在同一信道传输时，不会相互干扰
  - 动态媒体接入
    - ALOHA
    - CSMA：载波侦听多点接入
    - CSMA/CD：带碰撞检测的 CSMA
    - CSMA/CA：带碰撞避免的 CSMA

ALOHA ：发生碰撞时，节点发送完碰撞帧后，以概率 p 重传该帧，以 1-p 的概率等待一个帧传输时间。下一个时隙，同样以概率 p 重传，概率 1-p 等待。

CSMA（Carrier Sense Multiple Access）：载波侦听多点接入。多个节点以多点接入的方式连接在同一根总线上。因此所有节点位于同一冲突域内。节点在发送数据前，先检测信道，是否有其他节点也在发生数据。若有，则暂时不发送数据，以免发生碰撞。但是由于信号传播存在延迟的原因，CSMA 无法彻底解决碰撞的问题。

带碰撞检测的 CSMA（CSMA/CD）：节点边发送数据边检测信道上的信号电压大小。当几个节点同时发送数据时，由于信号相互叠加，因此总线上的信号电压摆动值会增大。利用这一特性，当总线上的电压摆动值超过一定阈值后，就认为总线上至少有两个节点在发送数据，数据发生了碰撞。此时就立即停止发送数据，避免浪费网络资源。等待一段随机时间后，再发送数据。

带碰撞避免的 CSMA(CSMA/CA) ：CSMA/CD 要求节点具有碰撞检测的能力，支持该能力的硬件代价较大。同时，由于无线信道特有的隐藏终端等问题，导致 CSMA/CD 并不起作用。碰撞避免是指，当信道空闲一个 DIFS（分布式帧间间隔）后，节点根据各自当前的状态，按照二进制指数退避算法选择随机的退避时间，退避完成后开始发送数据。若退避过程中，检测到信道忙，则冻结退避计时器。等待信道空闲一个 DIFS 后，继续退避直到完成。

2 交换网络

网桥：网桥将多个冲突域连接在一起。网桥使用转发表/转发数据库 FDB（Forwarding Database）来实现数据帧的转发。FDB 存储了目的 MAC 地址与输出（输入）端口的映射关系。当数据帧中的 MAC 地址对应的端口号与接收到该数据帧的端口相同时，丢弃该数据帧。当数据帧中的 MAC 地址对应的端口号与接收端口不同时，那么在对应端口转发将该数据帧。若 FDB 中不存在对应的条目，那么将该数据帧在所有的端口广播。

如何生成 FDB 表：当网桥接收到一个数据帧后，会记录该数据帧的源 MAC 地址以及对应的输入端口。如果 FDB 中不存在该条目，则将该映射关系写入 FDB 表。同时，FDB 表中的 MAC 地

址通过老化机制（Aging）来更新。如果 FDB 表中存在接收到的映射关系，则更新该映射关系。

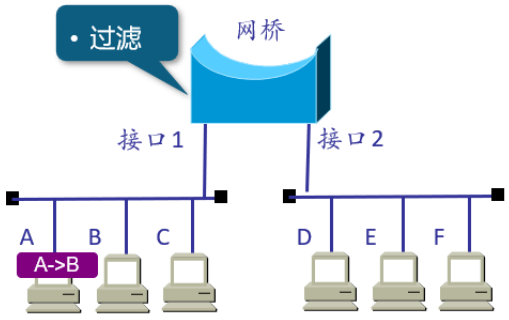


图 4: 网桥

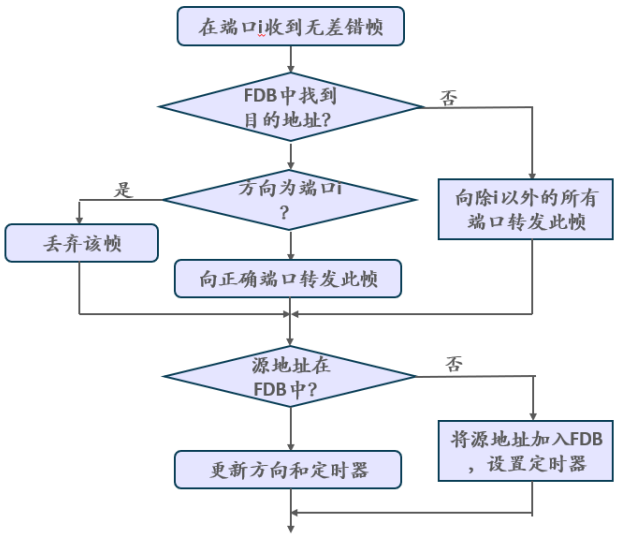


图 5: 网桥的工作流程

生成树算法：在数据转发中，如果形成了环路，就会导致广播风暴。因此，需要使用生成树算法，为网络中的每对节点（源-目的）分配唯一确定的一条路径。在生成树算法中，选取一个根节点，其他节点计算确定到根节点的最短路径。在选取根节点时，网桥间会彼此交换配置信息。配置信息包括本网桥认定的根节点的标识符（初始时，网桥都认为自己是根节点），本网桥到根节点的距离（初始时为 0），以及发布该配置消息的网桥标识。当收到邻居节点的配置信息后，如果邻居节点所选取根节点的标识符小于自己所选取的或者距离根节点的距离更小，则更新自己选择的根节点，并将跳数加 1。一直持续这个过程，最后所有节点都会选择同一个节点作为自己的根节点。然后同一网段指派离根节点更近的网桥作为数据的传输路径。如果有多个距离相同的网桥，则选取标识符更小的。如 Figure6中的例子，A 指派 B_5 为自己的网桥。

IP 的设计思路
- 尽最大努力交付，不保证 QoS
- 源结点和目标节点建立虚拟电路连接

路由器：是一个多输入端口，多输出端口的专用计算机。它使用 分布式路由算法 为 IP 分组寻找一条端到端的路径（通常是代价最小的）。并将传输路径保存到转发表（FIB）中。当路由器的某个端口收到 IP 分组后，会根据分组的目的 IP 地址，查找转发表，并将该分组从对应的端口输出到下一跳路由器。直到该分组到达目的地址。

输入端口：物理层 ⇒ 数据链路层 ⇒ 网络层

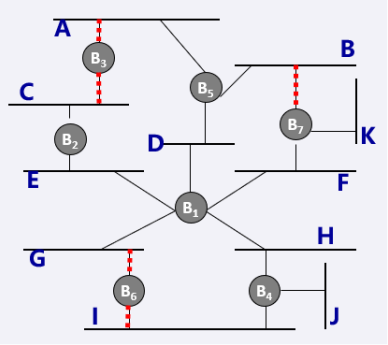


图 6: 选取网桥的实例

输出端口：网络层 ⇒ 数据链路层 ⇒ 物理层

分布式路由算法
- 域内
  - 使用内部网关协议 IGP 和统一的度量标准，在 AS(自治系统) 中路由数据包
  - 性能为目标导向，全局有统一目标
- 域间
  - 使用外部网关协议 EGP，将数据包路由到其他 AS
  - 策略和经济目标为导向，并且 AS 间没有统一的策略

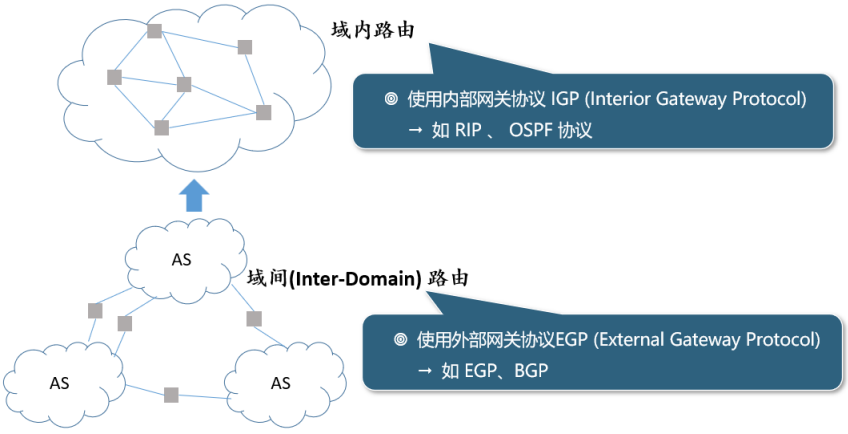


图 7: IGP 和 EGP 的区别

RIP、OSPF 与 BGP 的的对比（重点 ★）

RIP：一种基于距离路径向量算法的 IGP 协议。它要求每个路由器维护一个自己到其他每一个目的网络的最小距离记录。初始时，路由器向邻居节点请求对方的路由表（路由表中保存了它到其他已知节点的最小距离以及下一跳节点。初始时，路由表仅有邻居节点的信息），然后迭代计算自己的路由表。当路由条目发生变化时，向邻居节点通告更新后的路由表。并且每个路由器每隔 30s，会向邻居节点发送一次自己的路由表。如果经过几次更新周期，都未收到某一链路的更新信息，则认为该链路发生了故障（被动探测）。此外节点还可以持续发送控制分组检测到另一节点的链路，通过是否能接收到应答，来判断该链路是否正常（主动探测）。RIP 支持的最大距离为 16，也就是 16 跳。因此一条链路中，最多有 15 个路由器。

表 1: RIP 与 OSPF 的对比

	距离向量方法	链路状态方法
路由信息扩散范围	相邻节点	所有节点
路由信息内容	发送节点的全部信息	仅与发送节点直接相连的链路状态
空间开销	只保存邻居距离信息	需要保存全网拓扑信息
收敛速度	较慢	较快
可扩展性	较差	较好
路由环路	可能会出现环路	可能会出现短暂环路
路由协议	RIP	OSPF

RIP 的局限性：
- 可扩展性有限：每条链路最多有 15 个路由器
- 不能在丢失率高的网络中使用：RIP 需要定期更新
- 不能动态选择路由：RIP 使用固定的费用值、距离等作为选择路由的依据
- 开销较大：路由表随网络规模的扩大而增大。并且需要定期更新
- 收敛速度慢：当网络发生故障时，往往需要数分钟才能发现

OSPF ：初始时，每个节点仅有邻居节点的链路状态信息。然后通过广播扩散与本路由器相领的所有路由器的链路状态的方式，使每个节点获得了整个自治网络的拓扑信息，即建立了链路状态数据库（包含接口的 IP、掩码，链路类型和到连接路由器的开销）。然后每个节点使用单源最短路径算法独立计算本节点到其他节点的最短路径，从而得到一个以自己为根的树型结构。当网络拓扑发生变化时，路由器重新广播自己的邻接关系。

域间路由的挑战：
- 可扩展性问题：路由表中包含的合法 IP 前缀数量众多。
- 域的自治问题：每个 AS 都有自己的路径度量标准，因此域间路由不可能得到全局最优的路由路径，仅通知可达性。
- 域间的信任问题：如果一个 AS 配置错误，可能会导致其下游 AS 的传输故障

BGP(边界网关协议) ：是一种域间路由协议。在该协议中，每个 AS 的管理员都会至少选取一个路由器作为 BGP 代言人。然后不同 AS 的 BGP 代言人间建立起 TCP 连接。在此基础上，利用 BGP 会话交换路由信息。路由信息是一个保存有途径 AS 的列表。路由更新时，以 AS 列表的形式通知到达某个特定网络的完整路径。当路由器收到至某 AS 的路由通告时，它会根据本地策略，选择到该 AS 的较好路径。当 AS 收到路由更新消息后，检查其是否在对应的路径中。如果在，则丢弃该更新消息；如果不在，则将自己添加到路径中，并通告该路由器更新。

IP 地址是两层结构（即是标识也是位置）：网络号（由 IP 地址管理机构分配）+ 主机号（由网络号单位自行分配）。这样可以大幅度减少路由表的条目，并且当网络拓扑发生变化时，更新的次

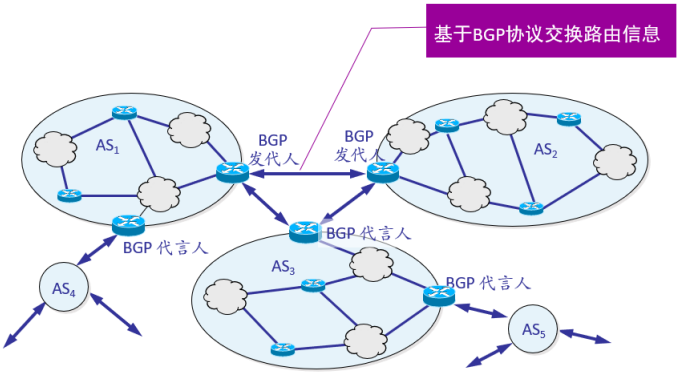


图 8: BGP

数也会大量减少。

分类 IP、子网 IP 和无类 IP 的区别（重点 ★）

分类 IP 地址：按照允许的最大网络数，将 IP 地址分为了 A，B，C，D 和 E 类。它们的网络号所占的位数不同，如 A 类 IP 地址的网络号为 8 位，最多允许 255 个 IP 地址。但是不同类 IP 地址的网络号与主机号位数固定，导致网络规模不可调，因此造成 IP 地址被大量的浪费。

划分子网：在 IP 地址中增加一个子网号，将二级的 IP 地址变为三级。子网号是从主机号中借用的若干位，它不改变二层 IP 地址中的网络号。在转发数据时，路由器仍是根据目的的网络号，先找到连接在本单位网络上的路由器。然后该路由器，根据网络号和子网号找到目的子网，最后将数据交付给目的主机。划分子网，实现了多个物理网络公用一个网络地址，从而减少了 IP 地址的浪费。但并没有完全解决该问题。

无类 IP 地址：IP 地址又变为了两级编址，即网络前缀 + 主机号。其中网络前缀的长度可变，用于替代三级 IP 地址中的网络号与子网号。然后在 IP 地址的最后加上一个斜杠“/”，并写上网络前缀所在的位数（CIDR 法）。路由器在转发数据时，使用最长前缀匹配法（前缀越长代表路由越精确），也就是在路由表的匹配结果中，选取具有最长网络前缀的路由作为下一条路由地址。由于需要匹配所有可能的前缀，所以 CIDR 会增加路由器的算法复杂度。然而这些复杂度在现代网络中已不是问题。

IP 数据报都是最终都是在物理网络中传播的，因此需要将 IP 地址映射为硬件地址，即 MAC 地址。由于 IP 地址的长度限制（32 比特），无法将 MAC 地址直接写入 IP 地址中，因此需要使用 ARP 协议（查表）完成 IP 地址与 MAC 地址间的映射。

ARP 协议：每个三层（物理层，数据链路层和网络层）节点都有一个 ARP 高速缓存。该缓存用于存储节点所在局域网内各节点的 IP 地址到其硬件地址的映射表。当一个节点 A 向该局域网中的某一节点 B 发送数据时，该节点会读取出 IP 报中的目的地址，并查找其 ARP 缓存中是否拥有该目的 IP 地址的条目。如果有，则将其硬件地址写入 MAC 帧，并通过局域网将 MAC 帧发往其硬件地址。如果没有，则在局域网中广播 ARP 请求。B 收到请求后，单播自己的硬件地址。然后 A 和 B 都会将对方的映射关系写入自己的 ARP。

IPv6：相比于 IPv4，IPv6 考虑了 QoS、移动性以及安全性等。IPv6 相比于 IPv4 没有本质上的增强，这也是即使部署了大规模 IPv6 网络，IPv6 的流量也很少的原因。

3 网络传输

传输层：实现应用进程间的端到端通信

传输层协议
- UDP：尽力而为
- TCP：可靠的、面向连接的字节流服务
  - 可靠传输
  - 拥塞控制
  - 流量控制
  - ...

UDP 与 TCP 的区别：UDP 只是在 IP 数据报服务的基础上增加了端口功能以及差错检测的功能。因此 UDP 也是尽最大努力交付，没有拥塞控制，不保证 QoS。并且为了减少开销和启动延迟，UDP 无需通信双方建立连接。而 TCP 则需要通信双方建立逻辑连接，然后在此基础上实现可靠传输、流量控制以及拥塞控制等。

滑动窗口（TCP 的核心）：发送方和接收方各自维护一个缓冲区。发送缓冲区用于存储已被应用进程写入，但还未发送的数据以及已发送未确认的数据。那么发送窗口大小就等于最后被确认的字节与最后发送字节之间的距离。接收缓冲区用于存储未被应用进程读取的数据（包含按序到达和乱序到达的数据）。然后接收方根据自己的接收能力调整接收端口的大小，并通知发送方，从而可以实现流量控制。

流量控制与拥塞控制的区别（重点 ★）

流量控制与拥塞控制的区别：流量控制是面向发送方和接收方的。由于双方的资源与服务能力不同，因此不可能以同样的速率向对端发送数据。所以需要根据接收方的实际情况调节数据的发送速率，这就是流量控制。而拥塞控制是面向发送方的。由于 TCP 不清楚数据是否会经过低速网络。当低速网络发生拥塞时，IP 可能会丢包，并触发重传。重传又会使拥塞的情况变得更加严重。因此当网络可能发生拥塞时，需要限制发送方的发送速率，这就是拥塞控制。

流量控制的做法是，接收方需要根据自己的接收能力，确定接收窗口的大小，并通过 TCP 报文段中的“AdvertisedWindow”字段，通知发送方。然后发送方根据 AdvertisedWindow 值确定有效窗口的大小，从而限制发送速率。

拥塞控制的做法是，当报文段超时或收到一个重复的 ACK（数据乱序到达接收方，说明其前面的分组可能丢失）后，就认为网络出现了拥塞，并需要减慢发送速率。当收到新的 ACK 后，将拥塞窗口（cwnd）的大小增加 $\frac{1}{cwnd}$（拥塞避免）。此外，当 TCP 传输启动时，拥塞窗口从很小的初值开始，发送成功后则快速增大，以探测网络的负载能力（慢启动）。此外，还有快重传和快启动机制，以减少由于等待超时而引起的连接无效时间。

TCP 自适应重传：
- 原始算法
- Karn/Partridge 算法
- Jacobson/Karels 算法

原始算法：将发送报文的时间和接收到其 ACK 的时间差作为样本值 SampleRTT。然后将新的样本值加权后，作用于 RTT 的平均运行值 EstimatedRTT 上。α 称为平滑系数（推荐值为 1/8），它的取值反应了对 RTT 瞬时波动的敏感程度。然后在 EstimatedRTT 的基础上，计算超时值 RTO。然而这种方法，可能会将重传报文的 ACK 误判为初始报文的 ACK，从而导致 SampleRTT 过大。

$$EstimatedRTT(n) = (1-\alpha) \times EstimatedRTT(n-1) + \alpha \times SampleRTT(n)$$

$$RTO(n) = 2 \times EstimatedRTT(n)$$

Karn/Partridge 算法：为了避免报文重传对 SampleRTT 的影响。在该方法中，每次超时重传一个报文段时，便会停止计算 SampleRTT，从而使 EstimatedRTT 和 RTO 的值更加准确。同时，当发生重传时，会将下一次 RTO 的值，设为上次的 γ（一般取 2）倍，而不以 EstimatedRTT 为基础。当不再发生报文重传时，再以 EstimatedRTT 为基础，计算 RTO。

Jacobson/Karels 算法:在该方法中,RTO 与 EstimatedRTT 的并不是呈简单的线性关系,而是根据 SampleRTT 的波动程度 RTTDviation 来决定。当 SampleRTT 波动较小时,认为 EstimatedRTT,较为准确，因此无需通过乘以 2 的方式来计算 RTO；当 SampleRTT 波动较大时，认为 RTO 应远不止 EstimatedRTT 的两倍。δ 的取值一般为 1/4。

$$RTO(n) = EstimatedRTT(n-1) + 4 \times RTTDviation(n)$$

$$RTTDviation(n) = (1-\delta) \times RTTDviation(n-1) + \delta \times [EstimatedRTT(n) - SampleRTT(n)]$$

4 网络应用

DNS(Domain Name System，域名系统) ：能够将主机名（域名）解析为 IP 地址。当前互联网中的域名采用了 层次化的命名方式 。即域名可以划分为顶级域名、二级域名、三级域名以及四级域名。其中顶级域名由国家顶级域名（如 cn，us 等）、通用顶级域名（如公司，教育机构等）以及基础结构域名（arpa：反向域名解析）组成。二级域名又可以按照行政区域等划分。

DNS 服务器的层次化结构
- 根域名服务器
- 顶级域名服务器
- 权威域名服务器
- 本地域名服务器

根域名服务器：最高层次的域名服务器，保存了所有顶级域名服务器的域名和 IP 地址。当 DNS 服务器遇到无法解析的域名时，都会请求根域名服务器。

顶级域名服务器：维护一些顶级域名，如国家地区域名。一般由专业机构来维护和管理顶级域名服务器。

权威域名服务器：一个 DNS 服务器管辖的范围被称为区。在每个区内，都有一个权威域名服务器，用于保存该区中所有主机的域名到 IP 地址的映射。

本地域名服务器：通常离终端用户比较近。当主机发出 DNS 查询请求时，这个查询请求报文会首先发送给本地域名服务器。

递归查询和迭代查询 ：当主机向本地域名服务器发出 DNS 查询请求后，通常会采用 递归查询 。如果本地域名服务器没有保存该主机名到 IP 的映射关系时，并会以 DNS 客户的身份，向根域名服务器继续发送查询请求报文，此时通常采用 迭代查询 。即当根域名服务器收到本地域名服务器发出的查询请求报文后，要么返回所查询的 IP 地址，要么返回下一步需要查询的域名服务器地址。

5 未来互联网体系结构

网络的细腰结构：在当前的 IP 网络体系结构中，认为细腰结构存在于 IP 层。然而由于网络中存在着大量的重复流量，因此在未来的互联网体系结构中，认为细腰结构应该上移，从而更多地面向网络服务。

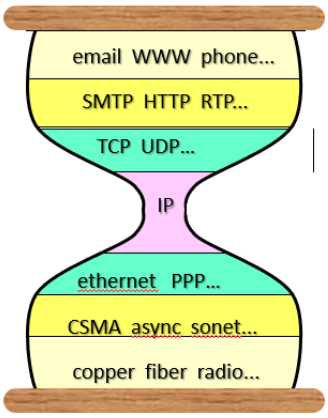


图 9: 细腰结构

网络的移动问题 ：IP 具有二义性，即是标识也表位置。当终端快速移动时，不得不切换网络，其 IP 地址会发生变化。终端被迫断开之前建立的传输连接，并需要再次连接。因此，为了解决网络的移动问题，就需要剥离 IP 的二义性。当前的解决方法有 SOFIA，MobileFirst，NDN 等。

SOFIA：

6 网络安全

网络安全的含义：网络安全是指网络信息系统中的硬件、软件及其系统的数据收到保护，不会因为偶然的或恶意的破坏、更改、泄露，系统能够连续、可靠、正常地运行，服务不中断。

保护的对象
- 硬件
- 软件
- 系统中的数据

面对的行为
- 破坏
- 更改
- 泄露

网络安全的五大要素
- 保密性：保证数据避免非授权的泄露，保障数据来源的可靠性
- 完整性：阻止对数据进行非授权篡改
- 可用性：数据可访问，无延迟
- 可控性：可以控制数据的传播范围和传播方式
- 可审查性：对出现的网络安全问题，可以提供调查的依据和手段

协议层中的网络安全
- 物理层：传输线路封装在包含高压氩气的密封管内，漏气时会报警
- 数据链路层：点到点的线路加解密，不经过中间路由器，如 VPN
- 网络层：防火墙、IP 报文头的安全域
- 传输层：端到端连接的加解密，如 TLS、SSL
- 应用层：安全机制大多数机制在这里，包括 HTTP、FTP、SMTP、DNS、Telnet 等

加密技术：使用某种手段将数据变为乱码（加密）传送，在接收端使用相同或者不同的手段还原数据（解密）。

- 加密类型
  - 哈希：无密钥，单向加密
  - 单密钥加密（对称加密）：1 个密钥
  - 公开密钥加密：1 个公钥和 1 个私钥

- 单密钥加密（对称加密）
  - 如：DES，3DES，AES，IDEA 等
  - 优点：加解密速度快
  - 缺点：
    1. 两两使用一对密钥，因此密钥管理量大
    2. 收发双方需要传输密钥，因此对密钥传输信道的安全性要求更高
    3. 对称加密无法实现数字签名（加密和解密使用了相同的密钥）

DES(Data Encription Standard)是一种对称加密技术。输入的参数有 56bits 的 key 和 64bits 的数据以及 Mode（加密 or 解密）。DES 的加密和解密过程一致，只是解密时取子密钥的顺序与加密时的相反。然而当前特殊的并行处理硬件可以使用穷举法破解 DES 加密。

- DES 的工作流程：
  1. 64bit 的明文数据经过初始置换 IP 分为左右两部分（L0 和 R0），各为 32bits
  2. 左右两部分数据在子密钥 $K_1 \sim K_{16}$ 的控制下进行 16 轮相同的乘积变换

     其中 $K_1 \sim K_{16}$ 由初始密钥通过 16 次移位交换产生

     $$L_n = R_{n-1}$$

     $$R_n = L_{n-1} \oplus f(R_{n-1}, K_n)$$

     - $f(\cdot)$
       1. 秘钥置换
       2. 扩展置换
       3. S-盒代替
       4. P-盒置换
  3. 最后将 R_{16} 和 L_{16} 拼接起来，并通过逆置换 IP，得到输出密文

公开密钥加密：发信息者公开自己的公钥，然后使用公钥对明文加密，受信者收到密文后，使用私钥解密。

图 10: 非对称加密解密流程

非对称加密的优缺点
- 优点
  1. 加解密能力分开，私钥不能由公钥推到出来
  2. 多个用户加密的消息，只能由一个用户解读（在公网中实现保密通信）
  3. 一个用户加密的消息，可以由多个用户解读（数字签名）
  4. 无需事先分配密钥
  5. 密钥持有量相比对称加密，减少了很多
- 缺点：加解密速度较慢

RSA 算法
1. p, q 分别为两个素数
2. $n = p \times q \quad z = (p-1) \times (q-1)$
3. 选取与 z 互质的数 d
4. 寻找 e 使得 $e \times d \ \% \ z = 1$
5. 加密 M：$C = M^e \ \% \ n$
6. 解密：$M = C^d \ \% \ n$

公钥（e，n），私钥（d，n）。如果要通过 e 破解 d 的话，需要知道 z，并将 z 分解为两个很大的素数的乘积。因此没有很好的破解方法。

$eg. p = 7, q = 11, M = 42$

$1. n = 7 \times 11 = 77, z = (7-1) \times (11-1) = 60$

$2. select \ d = 7, e = 43$

$3. 7 \times 43 \ \% \ 60 = 301 \% 60 = 1$

$4. Encrption : C = 42^{43} \ \% \ 77 = 14$

$5. Decrption : M = 14^{7} \ \% \ 77 = 42$

数字签名
- 接收方能够验证发送方宣称的身份
- 接收方不能伪造发送方对报文的签名
- 发送方不能否认报文签名的有效性

为什么对称加密不能做数字签名，而非对称加密却可以？

数字签名的作用是任何人都可以验证数字签名的有效性，但只有持有正确私钥的人才能创建该签名。在对称加密中，发信方和收信方使用了相同的密钥。这意味着，知道密钥的任何人，都可以创建签名，因此无法保证身份验证和不可否认性。在非对称加密中，私钥用于创建签名，而公钥用于验证签名。这意味着只有持有私钥的个人（即消息发送者）可以生成签名，任何人都可以使用公钥来验证签名，从而确保了身份验证、不可否认性和消息完整性。

单向校验和：在某些应用中，可能无需加密，但身份校验是必须的。假设有一明文报文 p，某一函数 CK(p)，很容易计算出，而几乎不可能从 CK(p) 中计算出 p，这种函数具有单向性质。

基于单向校验和的身份验证流程：首先发信者对明文报文 m 签名，并计算 CK(m)，然后使用私钥将其加密，得到 Da(CK(m))，最后将 [m,Da(CK(m))] 对偶传送给接收方。接收方使用公钥对 Da(CK(m)) 解密，得到 CK(m)。然后接收方对 m 使用 CK 函数，看结果是否与接收到的 CK(m) 一致。如果一致，则说明此报文未经篡改。

对称与非对称加密的混合使用：使用公开密钥加密对称密钥，从而减小了对称密钥被窃取并破解的可能。

7 网络攻击

网络攻击
- 主动攻击
  - 篡改消息
  - 伪造消息
  - 拒绝服务（DoS）
- 被动攻击
  - 窃听
  - 流量分析
  - 破解弱加密的数据流

主动攻击：导致某些数据流的篡改和虚假数据流的产生。

被动攻击：不修改数据，而是截取/窃听未经用户许可的信息。

8 QoS

QoS 服务质量并不创造带宽，而是对现有的网络资源进行有效的管理

指标
- 带宽
- 延迟
- 丢包率

目标
- 避免并管理 IP 网络拥塞
- 减少 IP 报文的丢失率
- 调控 IP 网络的流量
- 为特定用户或特定业务提供专用带宽
- 支撑 IP 网络上的实时业务

端到端时延 = 传播时延（固定）+ 排队时延（可变）+ 处理时延（可变）

传播时延：数据包在线路上传播花费的时间

处理时延：入端口 ⇒ 出端口队列

排队时延：出端口队列 ⇒ 发送

数据丢包的原因：
- 有线网络：网络拥塞导致路由器的输出队列满，从而不得不丢弃数据包
  （也有可能发生在路由器故障，错误或输入队列丢包）
- 无线网络：由于信道质量差导致

QoS 的三种服务模型：
- Best-Effort Service：尽力而为
- Integrated Service(Int-Ser)：综合服务
- Differentiated Service(Diff-Ser): 区分服务

Int-Ser 的原理:在发送数据前，使用 RSVP 信令来向网络申请特定的服务（流量参数和特定的服务质量，包括带宽、延迟等）。

RSVP 的特点
- 工作在传输层
- 不处理数据，只是一个网络控制协议
- 可在单播、组播网路中，进行资源预留
- 单向的资源预留协议
- 面向接收端的资源预留协议
- 对不支持 RSVP 的路由器透明

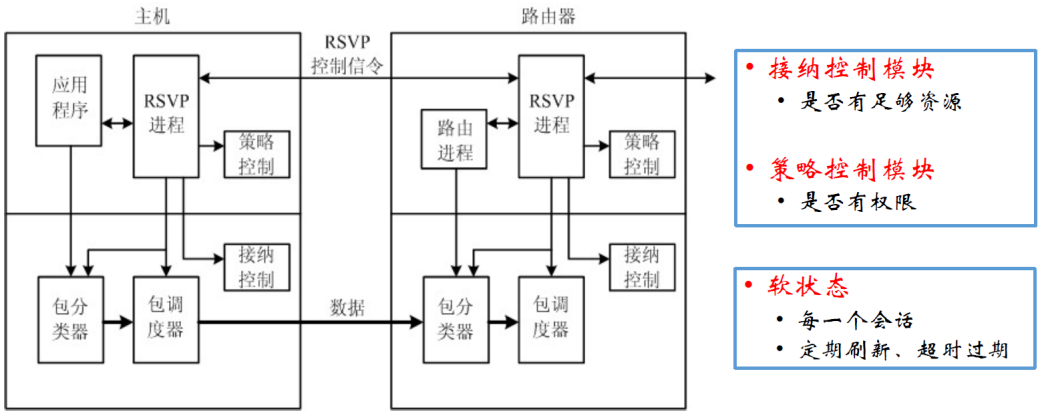


图 11: RSVP 协议节点模型

RSVP 工作原理：发送端定期发送 Path 消息，Path 消息的 IP 头中的目的地址为接收端的地址。每个中间路由器接收到 Path 消息后，都会记录上一跳节点的地址信息。当接收端接收到 Path 消息后，回复 Resv 消息，并沿着 Path 消息的路径反向逐跳传递，即每到一个中间路由器，都将路由器记录的上一条节点的地址信息填在 Resv 消息的 IP 头的目的地址中。这样做的原因是，让 Resv 消息的路径不一定与 Path 相同，以致于每一条路径都可以预留相应的资源。

- Int-Serv
  - 优点：能够提供绝对有保证的 QoS
  - 缺点：
    - 可扩展性差：RSVP 需要为每一个会话预留软状态
    - 路径上的所有路由器都必须支持 RSVP
    - 不适合短生存周期的数据流：包预留的开销可能大于数据流中所有包的开销

Diff-Serv 工作原理：用户首先和网络服务商（ISP）签订服务等级协议（Service Level Agreement, SLA）。数据包到达 ISP 的边缘路由器时，会按照数据包 IP 报文头中的 DS 字段，将其分类、计量、标记，也有可能被整形（粗颗粒度化）。然后在 Diff-Serv 的核心路由器中，对经过粗颗粒化的数据流进行调度分配路由。

- Diff-Serv
  - 优点
    - 扩展性好，Diff-Serv 的 DS 标记只规定了有限数量的业务级别，状态信息正比于业务级别
    - 具有层次化结构，不同的区域有不同的服务提供策略
    - 不影响路由，仅限于队列调度与缓冲管理
  - 缺点
    - 仅实现了粗细腻度的等级服务
    - 是一种相对优先级策略，不能完全保证端到端 QoS 的性能
    - 组件分散，需要协同一致、统一的策略和管理

Diff-Serv 与 E-LSP 的映射:流量分类和标记

- Diff-Serv DS 位与 MPLS EXP 位结合的方法
  - E-LSP
  - L-LSP

MPLS 的 EXP 位是用来表示 0~7 的报文优先级的字段。

E-LSP：在 LER（标记边缘路由器）上将 Diff-Serve 中的 DSCP 字段（位于数据包 IP 头部的的服务类别 TOS 标记字节中）映射到 MPLS 标签的 EXP 位。然后通过 EXP 位向 LSR（标签交换路由器）表示 QoS 分组的要求。这个的 LSP（标签交换路径）最多支持 8 个服务等级。LSR 根据 LABEL 和 EXP 位对分组进行队列调度，根据 EXP 进行报文丢弃处理。因此同一 LSP 中的分组可能被分到不同的队列，从而实现不同的 QoS 控制。

L-LSP：是将 DSCP 字段映射为一个 LSP。因此一个 LSP 中只支持一个 QoS 等级

队列调度：不同等级的分组放入不同的队伍中，路由器按照一定的队列调度算法，决定从哪个队列中取出数据分组进行服务。

- 队列机制的组成
  - 分类器：对于相应的流分类，可将数据分入不同的子队列
  - 排队策略：决定一个数据包能否被排队以及如何排队
  - 服务策略：决定如何从队列中选取数据包，并将其放置在发送队列中
- 队列
  - FIFO
  - PQ(Priority Queue)：优先级队列, 按照优先级转发数据包，算法简单，但可能会导致低优先级的数据包饿死
  - CQ(Custom Queue)：定制队列，为 16 个子队列分配一定的带宽，每次调度每个队列只能发送一定量的数据。这样保证某些数据包不被饿死。缺点是高优先级的队列也需要等待轮询，因此无法严格保证高优先级业务的 QoS。
  - 带抢占的 CQ：在 CQ 的基础上，指定一个最高优先级的队列。当这个队列有数据时，便立即转发。
  - WFQ(Weighted Fair Queue)：加权公平队列
  - CBWFQ(Class Based WFQ)：基于类的加权公平队列

WFQ中有两个阈值，HQO 和 CDT，分别表示 WFQ 系统允许的最大队列长度和 WFQ 开始丢包的预警值。当第 N 个数据包到达时，如果 N 大于 HQO，则根据该数据包是否为当前队列的最后一个数据包来判断是否丢弃。如果是最后一个数据包，那么丢弃；如果不是，队列中的最后一个数据包，并将第 N 个数据包排队。如果 N 小于 HQO，则再比较 N 与 CDT 的大小。如果 N 小于 CDT，则将第 N 个数据包排队；否则再次根据该数据包是否为队列中的最后一个数据包，判断其是否丢弃。

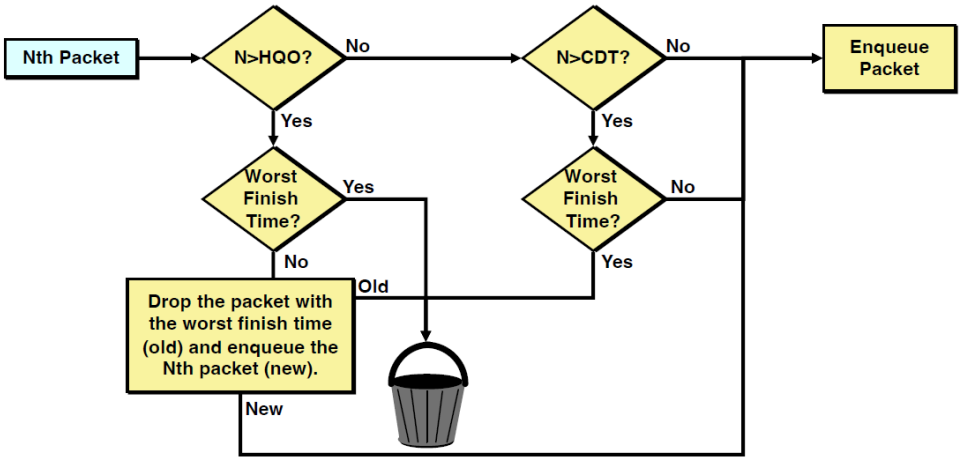
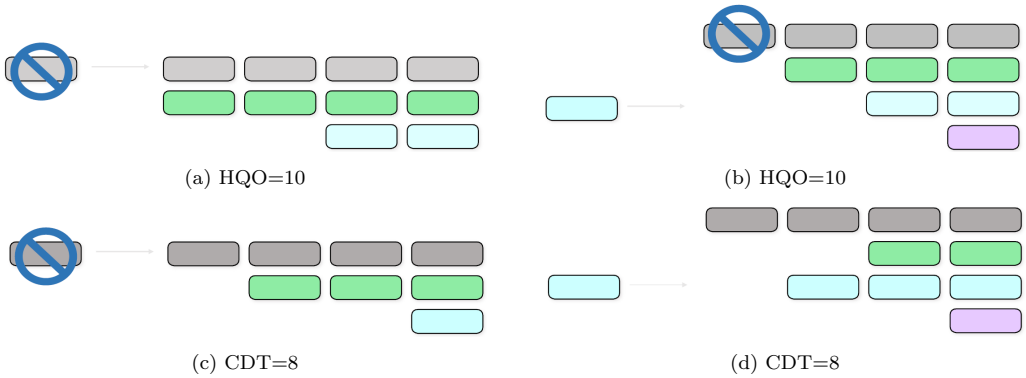


图 12: WFQ 排队模型



(a) HQO=10 (b) HQO=10

(c) CDT=8 (d) CDT=8

图 13: WFQ 排队案例

排队案例 13a中，N≥HQO，且为最后一个数据包，丢弃。
排队案例 13b中，N≥HQO，不是最后一个数据包，保留。并将最后一个数据吧丢弃。
排队案例 13c中，N≥CDT，且为最后一个数据包，丢弃。
排队案例 13d中，N≥CDT，但不为最后一个数据包，保留。

- WFQ 的优缺点
  - 优点：
    - 配置简单
    - 兼顾所有流的带宽
    - 优先保障高优先级的 QoS
  - 缺点：
    - 不能对流进行客户化定制
    - 不能提供固定的带宽保证

- 限速：简单丢弃超过带宽限制的突发数据包或将其将级转发
- 整形：缓存超过带宽限制的突发数据包，等流量下降后再发送。因此流量曲线更平滑

如何判断当前流量是否超过了带宽限制？用令牌桶, 当 1bit 或 1byte 数据通过接口后，会移除令牌桶中的一个令牌。只有当令牌桶不为空时，才能够转发数据。当令牌桶为空时，任何流量都会被被视为超过了额定带宽。同时，接口还会按一定速度往令牌桶中添加令牌。令牌的添加速度就控制了用户流量的带宽。

9 网络测量

网络测量的重大发现
- TCP 协议占据了大部分流量
- 网络是双向的，但不对称
- 大象流与老鼠流：9% 的流占据了 90% 的流量。50% 的数据包达到了最大长度，而 40% 的数据包小于 40
- 蜻蜓流：45% 的流持续时间不超过 2s
- 乌龟流：不到 2% 的流持续时间超过 15min，但承载了 50% 以上的流量

网络测量的三个基本指标
- 带宽
- 时延
- 丢包率

(与 QoS 的指标相同)

网络测量的内容
- 拓扑
- 性能
- 流量

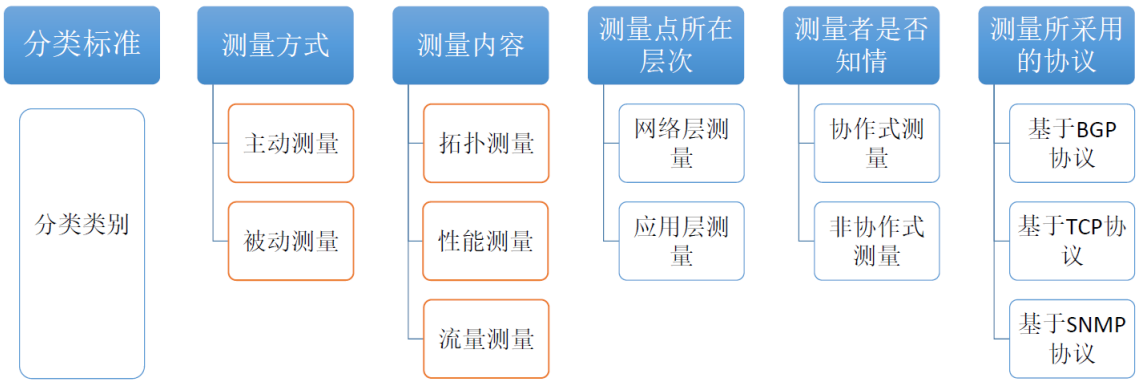


图 14: 网络测量的分类

主动测量
- 定义：由用户主动发起，将探针分组注入网络，根据测量数据流的传送情况来分析网络
- 优点：
  1. 使用方便：只需要在本地发送测试包，观察网络的相应即可，适合端到端的网络测量
  2. 安全：不涉及用户的网络信息
- 缺点：
  1. 潜在地增加了网络的负载
  2. 需要消耗较多的计算资源

被动测量
- 定义：通过捕获网络中已有的数据包，来记录和分析网络流量，以获得网络的性能
- 优点：
  1. 测量的是网络中真正的流量
  2. 能够达到对观察点网络的详尽理解
- 缺点：
  1. 容易捕获到网络中的敏感信息，给用户隐私带来一定威胁
  2. 只能获得网络的局部数据，因此测量范围所限

- 带宽
  - 链路带宽：链路上数据包的最大传输速率
  - 瓶颈带宽
    - 含义：路径中处理能力最低的链路所能达到的最大传输速率（路有多宽）
    - 测量方法：源端向目的端发送一系列的相同大小的测量包，统计并计算得到改组数据包的最小 RTT 值。重复这个过程，每次选取大小不同的测量包。最后可以得出 RTT 值与测量包大小的直线，该直线斜率的倒数即为链路带宽，通过逐跳的链路带宽，可以间接获得瓶颈带宽
  - 可用带宽
    - 含义：当前应用程序与其他数据流共享链路时，该应用程序能够得到的带宽
    - 测量方法：当测试报文发送速率小于链路可用带宽时，传输时延相对固定。当测试报文的发送速率大于可用带宽时，网络出现排队现象，网络时延增大。因此导致时延增大的发送速率转折点，即为可用带宽的大小

- 拓扑测量
  - 分级
    - IP 级
    - 路由器级
    - PoP 级
    - AS 级
  - IP 级路由测量方法：TraceRoute
    - 原理：探测节点发送 UDP 消息，中间节点收到后回复 ICMP 消息
    - 优点：不受网络管辖范围限制
    - 缺点：
      - 测量效率低：需要发送一系列的探测包
      - 测量存在误差：部分中间节点不回复 ICMP 消息

- 别名解析：
  - 定义：路由器有多个端口，每个端口都对应一个 IP 地址。别名解析用于监测哪些 IP 地属于同个路由器，从而把同个路由器的 IP 地址聚合起来。
  - 不使用的后果：
    - 虚拟节点
    - 虚拟链路
    - 虚拟拓扑
  - 方法
    - 使用 DNS 域名反向查询：假定同一路由器的多个接口拥有相同的域名，因此对可能的 IP 地址做反向域名查询，即可实现别名解析
    - 使用 IP 数据包的 Identification 字段：Identification 字段唯一，且同一路由器发出的 IP 数据包的 Identification 字段是连续的（并且所有路由器都满足）
    - 基于 ICMP 消息：
      - 基于 TTL 超时：向目标 IP 地址发送 TTL 逐渐增加的消息，可用触发 TTL 超 ICMP 消息。如果目标 IP 在同个 TTL 值时，触发了 ICMP 消息且 ICMP 消息的源地址相同时，可认为目标 IP 属于同一路由器
      - 基于端口不可达：向目标 IP 地址的一个已知不可达端口发送数据包，如果目标 IP 都返回端口不可达的 ICMP 消息，且 ICMP 消息的源地址相同则认为目标 IP 属于同一路由器

10 CDN(内容分发网络)

zipf 分布：互联网上 20% 的内容吸引了 80% 的流量。在 log-log 坐标中，内容的访问次数 y，与其排名 i 成反比。$y = 1/i^s$。

- zipf 带来的启示
  - 在靠近用户的地方，缓存流行内容
  - 将内容拷贝到不同地域的多台服务器中，减轻服务器负载，提升用户感知的质量

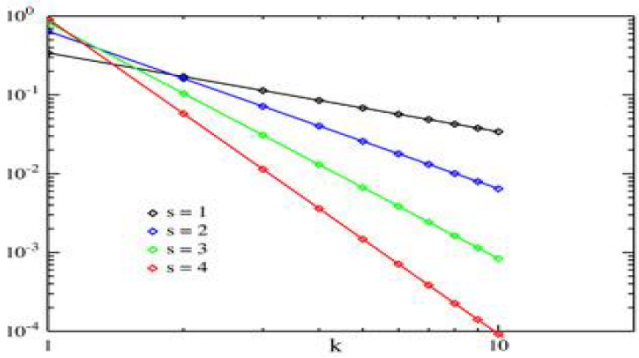


图 15: zipf 分布

web 缓存的问题
- 版权问题
- 很多内容无法缓存/缓存后无效，如动态数据，依赖于参数或 cookie 的数据以及加密的数据

CDN：在用户附近的服务器上缓存内容，来提高内容的加载速度，减轻原始服务器的负载。

CDN 的工作原理：原始服务器向 CND 服务商付费后，原始服务器便会发布内容至 CDN 分发节点。然后 CDN 服务商分布在全球各地的服务器便会主动复值 CDN 分发节点中的内容。同时 CDN 服务器定期检查缓存的内容是否与原始服务器中的内容一致，以确保内容的更新。当用户请求原始服务器的内容时，用户的请求会被重定向到 CDN 服务器中。如果 CDN 服务器中有相应的内容，则直接响应用户的请求；若没有，则 CDN 分发节点会向原始服务器请求相应的内容。

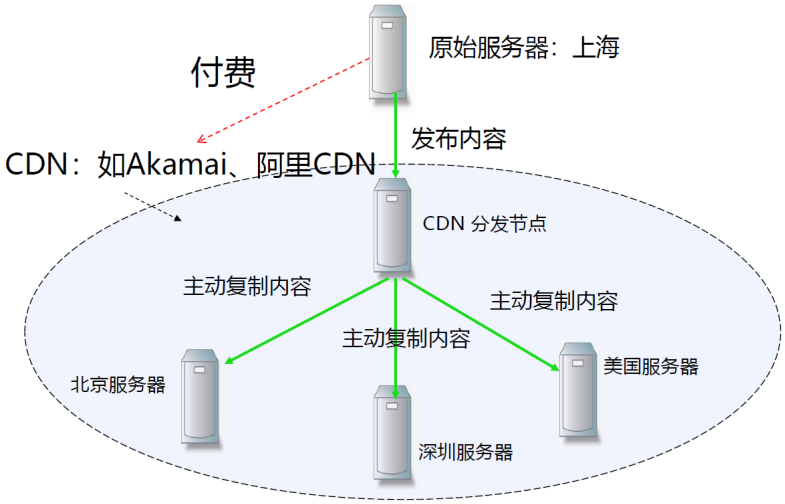


图 16: CDN 原理

CDN 的核心问题
- 选取哪个 CDN 服务器
  - 负载轻的服务器
  - 性能高的服务器（处理能力，带宽等）
  - 距离客户近的服务器（物理距离、RTT）
  - 成本（跨网收费）
- CDN 服务器缓存什么内容：预测视频的流行度

用 RTT 计算 CDN 服务器与客户间的距离：选取一些源点，所有节点都 ping 这些源点，通过 RTT 可以得出，这些节点到不同源站之间的距离，从而建立多维的坐标系。

如果 CDN 服务器存不下全部视频了
- 缓存视频的一部分：如视频的开头部分
- 缓存替换：缓存请求大于两次的视频
  - LRU：淘汰最近最少使用的数据
  - LFU：淘汰最不经常使用的数据

DASH（自适应码率调整）：根据吞吐量、播放缓冲区等信息，自动选择视频快的码率。

视频点播 QoE 关注的参数
- 高码率
- 少卡顿：相比于卡顿时间，用户对卡顿次数更敏感
- 快启动，少切换

QoS 与 QoE 的区别：QoS 用于保证网络层的服务质量，是一种客观的技术。而 QoE 反应的是用户主观的体验，如用户打分等。

直播时延
- 服务器发出到用户接收的时延
- 用户同时接收间的时延

P2P：对等网路，是一种分布式网络。P2P 与传统 CS（Client-Servicer）系统不同的是，其中每个参与节点既可以作为客户端也可以作为服务器，从而与网络中的其他节点直接交换数据。

非结构化 P2P（没有固定的结构）
- Flooding（泛洪搜索）：源节点向直接相连的邻居节点发出数据请求，邻居节点收到请求后，会将其复制转发给自己的邻居节点。反复迭代下去，直到找到所需的资源
- Bit Torrent：文件被分为多个小块，用户在下载文件的同时，也会向其他用户上传已下载的文件，这样可以显著减少单个资源提供者的带宽需求，并使文件传输速度不受单个下载源速度的限制

Bit Torrent
- 整体算法：首先 web 服务器上存放了一个.torrent 文件。.torrent 文件包含了要共享的文件信息，如文件名、大小、文件的散列信息以及一个指向 Tracker 的 url。Tracker 帮助下载者获取其他 peer 的信息，然后下载者利用这些信息与其他 peer 建立连接。如 Figure17所示。
- 如何选择 peers
  - choking：与之相连的前 4 个 peer 拥有最高的上传速率，每个 10s 计算一次哪些 peers 需要被阻塞
  - unchoking：第 5 个 peer 被称为的”optimistic unchoking”，拥有最高的下载速率，每隔 30s 重新计算一下”optimistic unchoking”

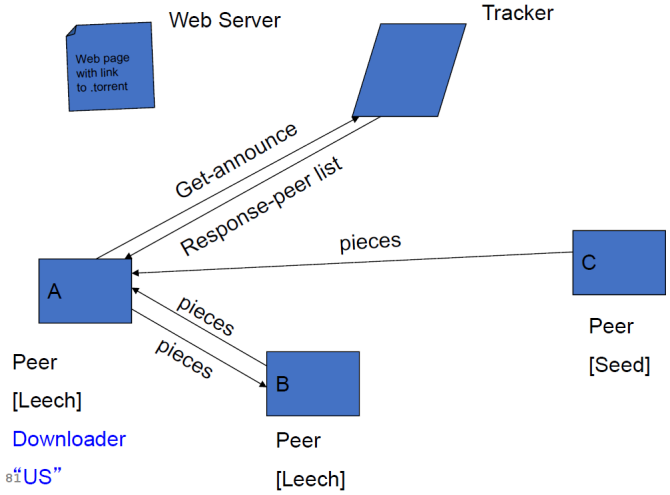
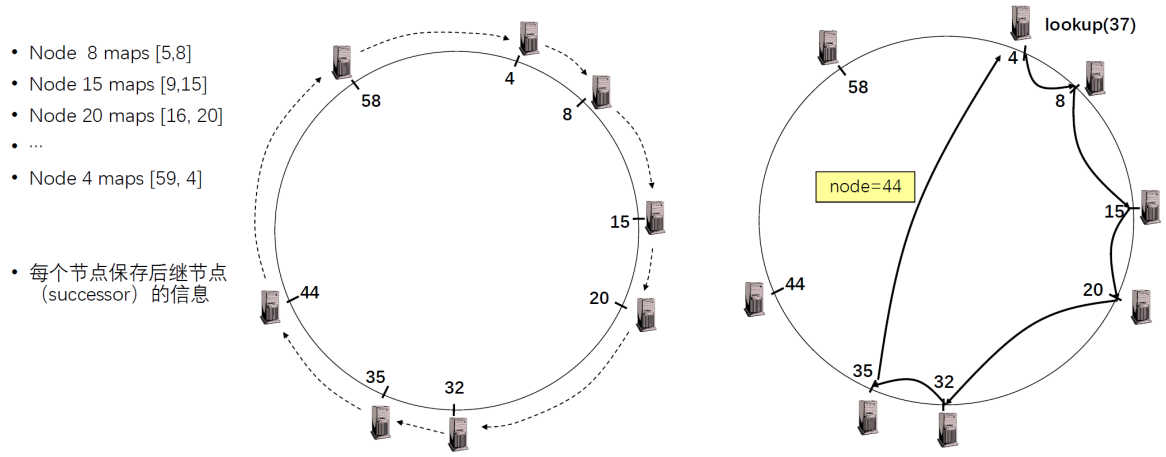


图 17: BT 整体结构

结构化 P2P：Chord 搜索，根据资源 ID，高效搜索服务节点的方法。



(a) Chord 搜索—节点映射　　(b) Chord 搜索—O(n) 搜索

图 18: Chord 搜索 1

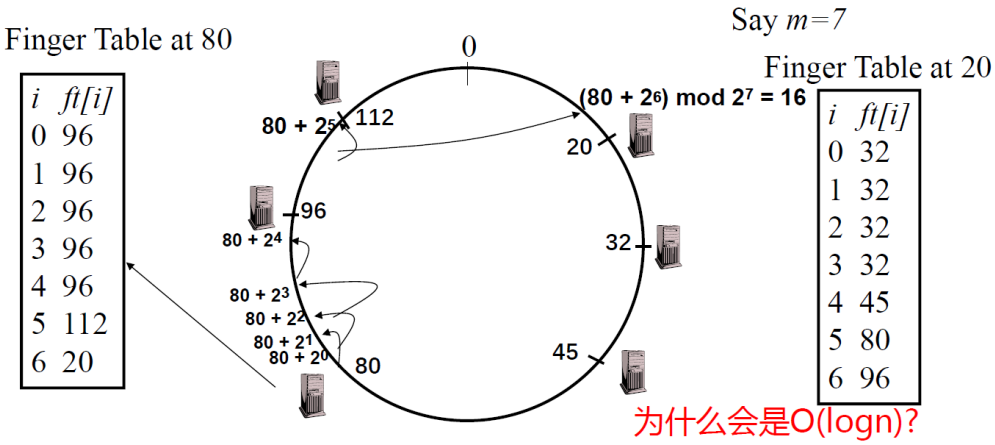


图 19: Chord 搜索—O[log(n)] 搜索

Chord 中的基本要素有节点 NID(节点 ID) 和 KID（资源 ID）。其 NID 为节点 IP 地址的 hash 值，KID 为 (key，value) 对的 hash 值。在 Chord 圆环中，一个键的责任节点是在圆环上顺时针方向找到的第一个标识符大于或等于键标识符的节点。这个节点称为键的“后继节点”。每个节点都保存了后续节点的信息。如 Figure18a的例子所示，总共有 8 个节点，通过 hash 计算得出它们的 NID 分别为 4，8...58，因此按照映射规则，[59,4] 的 KID 被映射到 Node 4 中，[5,8] 的 KID 被映射到 Node 8 中。

使用复杂度为 O(n) 的查找方法时，系统会沿着 Chord 环顺时针依次查找每一个节点。因此在最坏的情况下，需要查找 n 个节点。如 Figure18b所示，NID 为 4 的节点正在查找 KID 为 37 的资源。因此它顺时针依次访问其他节点，由于每个几点都保存了后续节点的信息。因此，当查询到 NID 为 35 的节点时，便已获得了 KID 为 37 的资源所在的位置了。

使用 指纹表 搜索，能够将复杂度将为 O[log(n)]。每个节点都会保存一个指纹表，每个指纹表包含最多 m（指圆环的大小）个条目。其中第 n 个节点的指纹表的第 i 个条目指向满足 $NID \geq n + 2^i (mod\ 2^m)$ 的第一个节点。如 Figure19所示，KID 为 80 的节点，其指纹表的第一个条目为 96（$80 + 2^0\ mod\ 2^7 = 80$）。当某节点查找所需资源时，可以根据指纹表，进行指数级的跳跃查找，而无需按顺时针逐个节点查找。因此，通过指纹表，可以将 Chord 查找的复杂度将到 O[log(n)]。

P2P 的主要挑战
- 节点的动态性：需要定期更新
- Overlay 与 IP 网络层的不匹配：上层网络的邻居不一定是物理网络的邻居

11 区块链

比特币和以太坊的区别：比特币只是将区块应用在了虚拟的货币交易上。而以太坊引入智能合约并开放编程接口，拓宽了区块链的应用。

授权链（联盟链和私有链）：比特币和以太坊对不限制接入的设备。而授权链只允许经授权后的设备才能接入。并且接入授权链后，必须公开自己的身份。授权链目前可用将规模做得更大并且拥有更高的性能。并且由于公开了接入身份，因此授权链能够被更好地监管。但同时带来了隐私保护的问题。

区块链的交易流程：由 A 发起并签名一次交易，并在网络中广播该交易。该交易经过共识节点的验证，在确认无误后，对交易进行记录并将账本区块广播到网络中 (共识机制：通过某种方式，如挖矿，来产生一个记账节点)。链中的所有节点收到账本区块后，再次验证后，便会将其记录在本地的区块链账本的末端。当网络中，绝大多数节点都完成了验证并记录后，这笔交易便完成了。

- 区块链
  - 宏观上：分布式系统（分布式部署相同的系统，相同的规则，共享相同的数据）
  - 微观上：
    - 区块：记录所有经数字签名后的共享信息 (账本中的纸)
    - 哈希链：将上一个区块的 hash 值，作为下一个区块内容的一部分。因此哈希链将区块按顺序串联。由于后续区块记录了上一区块的 hash 值，如果想要篡改某一区块，那么需要篡改该区块的所有后续区块。因此区块链的数据不可删除，不可篡改，不可抵赖。同时由于使用数字签名，因此区块链的数据也不可伪造

- 区块链和分布式数据库的区别：
  1. 分布式数据库中的每个节点存储一部分数据，而区块链一般要求节点存储的相同的数据
  2. 分布式数据库有唯一的拥有者，而区块链是共享的
  3. 分布式数据库的操作有增删改查，而区块链的操作只有增查
  4. 分布式数据库的节点间相互信任，而区块链的节点相互不信任，相互制约
  5. 区块链的容错性比分布式数据库更高