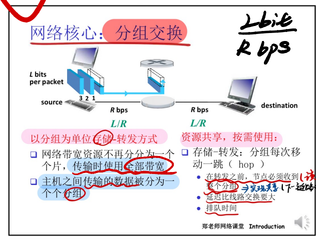
3 Network Core

packet switching

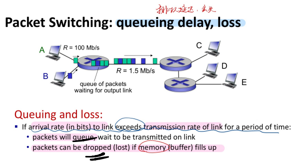
Lecture 1 P47



L bit
R bps

Store and forward

⇒ queueing delay, loss



丢失的分组 { retransmitted by previous node / source end system / not at all

优缺点 ☆

packet switching allows more users to use network.

Packet Switching特点

优点：

- resource sharing共享网络资源
- simpler, no call setup想连就连

缺点：

- excessive congestion possible: delay and loss
- protocols needed for reliable data transfer, congestion control

还需下层提供可靠数据传输和拥塞控制等服务

How to provide circuit-like behavior PS?

- Bandwidth guarantees 对钞能力要求很高 带宽要求
- New methods should be developed

Circuit switching

P55

Circuit switching

线路交换的优点：

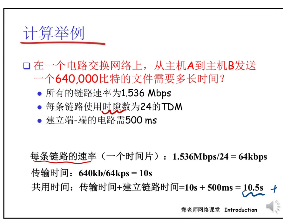
1. Dedicated resources: no sharing
2. circuit-like (guaranteed) performance

- Circuit segment is idle if not used by call (no sharing)
- Commonly used in traditional telephone networks

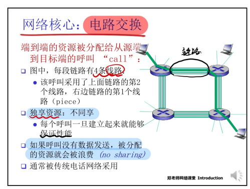
应用方式：

FDM: 按频率划分给多条线路中每一条各占一个频率区间

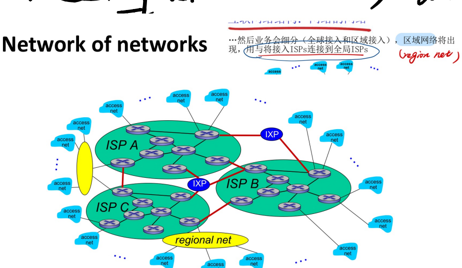
TDM: 按时间块划分给多条线路中的一条使用, 在终点可根据顺序复原合成完整数据

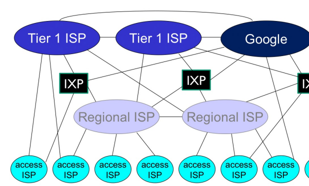


+ 电磁波传送时间所消耗的时间代价



端系统通过ISP接入Internet ⇒ connect each access ISP to one global transit ISP





- Small view of well-connected large networks
  - "tier-1" commercial ISPs (e.g., ChineTel, Sprint, AT&T, NTT), national & international coverage
  - content provider network (e.g., Google): private network that connects it data centers to Internet, often bypassing tier-1, regional ISPs

四种延迟.

Lecture 1 P68

Delay in Packet-Switched Networks.

延迟通常由四部分组成：

1. Transmission delay传输延迟，指将整个分组packet完整的从路由器打出的时间
2. Propagation delay传播延迟，指在链路上的客观物理传输时间
3. Queuing delay排队延迟，存在于过多用户占用带宽超过链路带宽上限的情况，涉及到流量强度的问题
4. Nodal processing delay路由处理延迟，通常不会问可以当没有

吞吐量.

Lecture 1 P72

Throughput 吞吐量

- Throughput: rate (bits/time unit) at which bits transferred between sender/receiver.**由多段链路中最小的带宽来决定**

R19. 假定主机 A 要向主机 B 发送一个大文件。从主机 A 到主机 B 的路径上有 3 段链路，其速率分别为 $R_1 = 500\text{kbps}$，$R_2 = 2\text{Mbps}$，$R_3 = 1\text{Mbps}$。

a. 假定该网络中没有其他流量，该文件传送的吞吐量是多少？

b. 假定该文件为 4MB。用吞吐量除以文件长度，将该文件传输到主机 B 大致需要多长时间？

c. 重复（a）和（b），只是这时 R_2 减小到 100kbps。

1 kbps = 10^3 bps

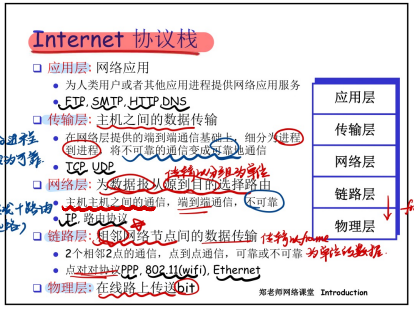
1 Mbps = 10^6 bps.

- 由此实例可知，吞吐量由最小的R1决定，故文件传输时只能以最小的带宽来传输，所以是4MB/500kbps。
- 如果R2带宽减小到100kbps比R1带宽还小，那就只能按R2的带宽来传了

分层的好处

Why layering? 分层的好处.

- **Divide complex systems to simple components**
- **Easy for maintenance**
- **Flexible for updating**



△ Application Layer

client - server architectures.

Lecture 2 P1

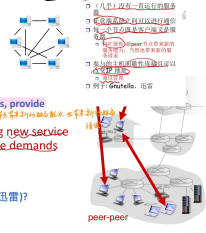
可扩展性差. poor scalability.

P2P architecture

self- scalability. 难以管理. Lecture 2 P10.

P2P architecture

- No always-on server is needed
- End systems directly exchange data
- Peers request service from other peers, provide service in return to other peers
  - *Self scalability* – new peers bring new service capacity, as well as new service demands
- Peers are intermittently connected
- Dynamic IP addresses
- Q: Did you use Thunder Downloader(迅雷)? Why does it download so fast?



进程通信.

进程通信

进程：在主机上运行的应用程序

- 在同一个主机内，使用进程间通信机制通信（操作系统定义）
- 不同主机，通过交换报文（Message）来通信
  - 使用OS提供的通信服务
  - 按照应用协议交换报文
    - 借助传输层提供的服务

clients, servers

客户端进程：发起通信的进程

服务器进程：等待连接的进程

- 注意：P2P架构的应用也有客户端进程和服务器进程之分

Application Layer 3-11

问题1：对进程进行编址（addressing）

- 进程为了接收报文，必须有一个标识
  即：SAP（发送也需要标示）
  - 主机：唯一的32位IP地址 (不对!)
    - 仅仅有IP地址不能够唯一标示一个进程；在一台端系统上有很多应用进程在运行
  - 所采用的传输层协议：TCP or UDP
  - 端口号（Port Numbers）
- 一些知名端口号的例子：
  - HTTP: TCP 80 Mail: TCP25 ftp:TCP 2
- 一个进程：用IP+port标示端节点 (end point)
- 本质上，一对主机进程之间的通信由2个端节点构成

Application Layer 3-13

一个进程有没有多个 network identifiers.

Addressing processes 对进程进行编址.

- Host device has unique 32-bit IPv4 and/or 128-bit IPv6
- Process network identifier:
  - IPv4:port 192.168.1.100:80
  - [IPv6]:port [240e:3a1:4cb1:69d0:f40c:4269:74a2:7ea3]:80
- Can a process have multiple network identifiers?

IP

IP + port 标识一个 process.

IP + port numbers = end point.

一对主机进程之间的通信由2个 end point 构成

传输层提供的服务: data integrity, throughput, timing, security.

TCP — Lecture 2 P22

SSL: provides encrypted TCP connection

UDP — Lecture 2 P24

存在的必要性 / 优点.

tcp, UDP 都不提供的服务: delay guarantees, bandwidth guarantees.

UDP存在的必要性

- 能够区分不同的进程，而IP服务不能
  - 在IP提供的主机到主机端到端功能的基础上，区分了主机的应用进程
- 无需建立连接，省去了建立连接时间，适合事务性的应用
- 不做可靠性的工作，例如检错重发，适合那些对实时性要求比较高而对正确性要求不高的应用
  - 因为为了实现可靠性（准确性、保序等），必须付出时间代价（检错重发）
- 没有拥塞控制和流量控制，应用能够按照设定的速度发送数据
  - 而在TCP上面的应用，应用发送数据的速度和主机向网络发送的实际速度是不一致的，因为有流量控制和拥塞控制

Application Layer 3-28

transmitted data may be lost or received out of order.

- Two socket types for two transport services:
  - UDP: unreliable datagram　不可靠的数据报
  - TCP: reliable, byte stream-oriented　面向字节流

HTTP (Application Layer)

Lecture 2 P32

应用　支持Web应用的协议

Web 与 HTTP

一些术语

- Web页：由一些对象组成　任何对象都可以通过URL唯一标识.
- 对象可以是HTML文件、JPEG图像、Java小程序、声音剪辑文件等
- Web页含有一个基本的HTML文件，该基本HTML文件又包含若干对象的引用（链接）（不包括对象本身，包括链接）
- 通过URL对每个对象进行引用
  - 访问协议，用户名，口令字，端口等；
- URL格式：

Prot://user:psw@www.someSchool.edu/someDept/pic.gif:port

协议名 用户:口令　主机名　路径名　端口

Application Layer 3-32

CS model

use TCP

stateless (server 不维护 client 的信息. 无状态的server 能支持更多的 client)

关注下 Http 请求, 响应报文格式

URL

https://www.google.com/images/branding/googlelogo/1x/googlelogo_color_272x92dp.png

protocol　hostname[:port]　path to a file

协议名　主机名　路径名

非持久 & 持久.

HTTP connections

Non-persistent HTTP

- At most one object sent over TCP connection
  - connection then closed
- Downloading multiple objects required multiple connections

Persistent HTTP

- Multiple objects can be sent over single TCP connection between client, server

HTTP连接

非持久HTTP	持久HTTP
最多只有一个对象在TCP连接上发送	多个对象可以在一个（在客户端和服务器之间的）TCP连接上传输
下载多个对象需要多个TCP连接	HTTP/1.1 默认使用持久连接（连接不关）
HTTP/1.0使用非持久连接	

Persistent HTTP

Non-persistent HTTP issues:

- Requires 2 RTTs per object
- OS overhead (开销) for each TCP connection　为每了TCP连接都要分配资源
- Browsers often open parallel TCP connections to fetch referenced objects

Q: Is persistent HTTP perfect? <=

Persistent HTTP issues:

- Server leaves connection open after sending response
- Subsequent HTTP messages between same client/server sent over open connection
- Client sends requests as soon as it encounters a referenced object　客户端能尽快发送服务请求
- As little as one RTT for all the referenced objects

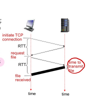
对于所有被引用的对象，少了RTT.

response time

2RTT + 传输时间

响应时间模型

- 往返时间RTT（round-trip time）：一个小的分组从客户端到服务器，在回到客户端的时间（传输时间忽略）
- 响应时间：
  - 一个RTT用来发起TCP连接
  - 一个RTT用来HTTP请求并等待HTTP响应
  - 文件传输时间
  - 共：2RTT+传输时间

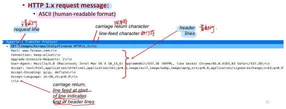


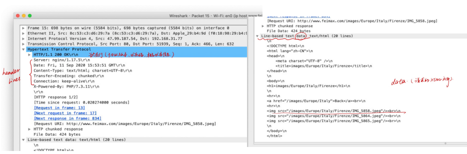
Application Layer 3-38

HTTP 请求报文

HTTP request message

- Two types of HTTP messages: *request*, *response*
- HTTP 1.x request message:
  - ASCII (human-readable format)



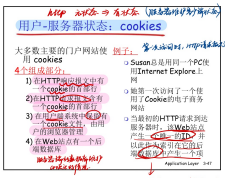


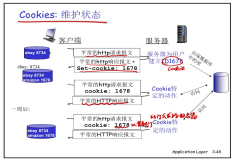
Http request, response 报文

Cookies

Lecture 2 P63.

使得 HTTP: stateless → stateful.

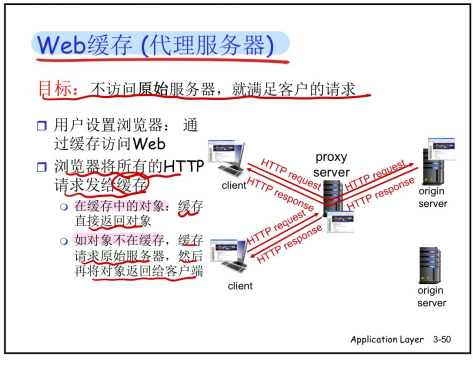




隐私问题

Web Caches

Lecture 2 P67.



Web 缓存的好处（个别）

Why Web caching?

- reduce response time for client request
- reduce traffic on an institution's access link
- Internet dense with caches: enables "poor" content providers to effectively deliver content (so too does P2P file sharing)

相对于增加接入链路的带宽，Web缓存更廉价。（延时更小）
(increased access link speed)

应用层协议：HTTP, SMTP, FTP

DNS

cS model　Lecture3 P5

实现主机名—IP地址的转换.　(a distributed, hierarchical database)

分布式，层次化.

不集中管理:

集中 DNS.

Why not centralize DNS?
- Single point of failure 单点故障
- Traffic volume 交通量(通信容量)
- Distant centralized database 远程集中式数据库
- Maintenance 维护

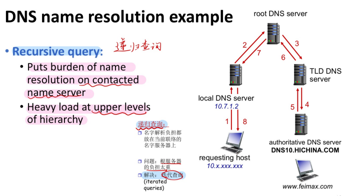
UDP

P9. Authoritative DNS servers：组织机构的DNS服务器

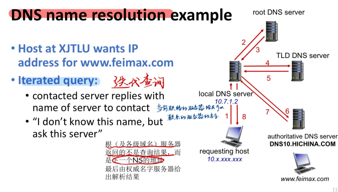
TLD name server

local name server → root name server → TLD → authoritative DNS server → local

Recursive query　P11　Iterated query.



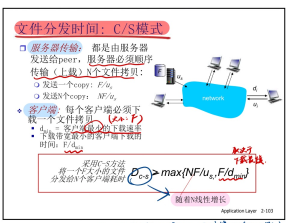
⇒

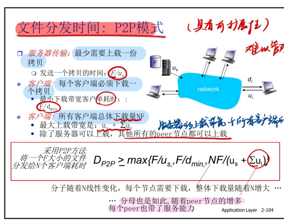


P2P

P29.

△ 文件分发

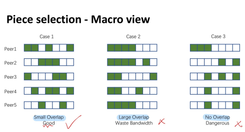


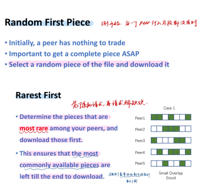


{ N较小：取决于客户端下载能力 (服务器服务能力强)
{ N较大：服务器上载带宽.

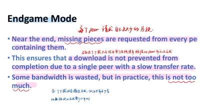
Lecture 3 P60

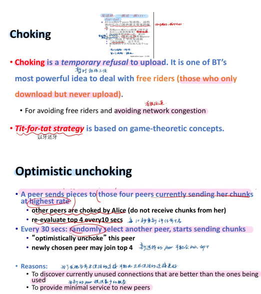
Bit Torrent.　Lecture3 P36　△ Socket programming





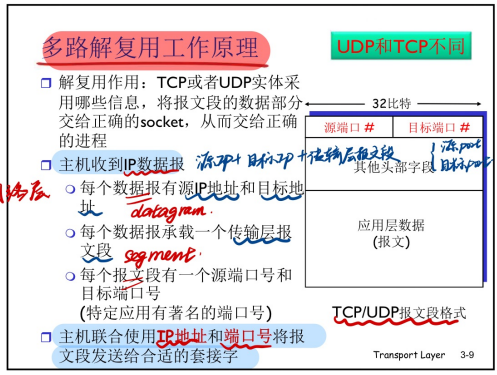
⇒ selection policy





多路复用/解复用

Lecture 4 P11



UDP

具备相同目标IP地址和目标端口号，即使是源IP地址或/且源端口号的IP数据报，将会被传到相同的目标UDP套接字上

TCP

仍无法确定是否作为一个不同的通信就是指不同的socket

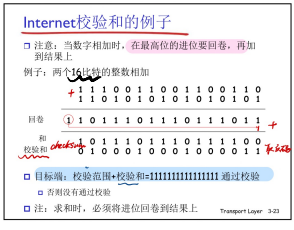
源IP地址
源端口号
目的IP地址
目的端口号

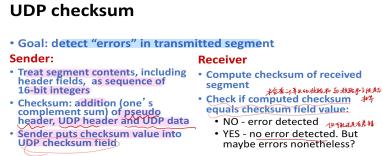
UDP checksum

Lecture 4 P25

→ detect "errors" in transmitted segment (eg: bit errors)

addition of pseudo header, UDP header, UDP data

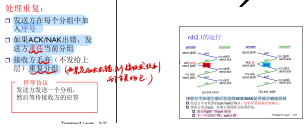




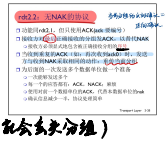
Reliable data transfer (RDT)

Lecture 4 P26

- rdt 2.0 → 若ACK/NAK出错？ → 解决 → 每个分组中加入序号.
- rdt 2.1 →

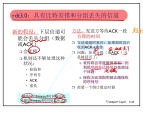


- rdt 2.2 → 无NAK.

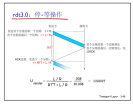


- rdt 3 → 超时重传机制 (可能会丢分组)

Lecture 5, 6 还没有



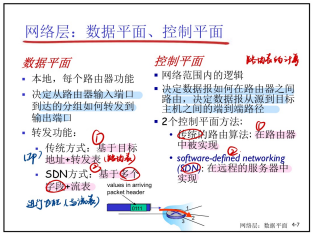
软硬



链路容量多数时可能差

一次只能发一个 → Pipelined protocols.

Network layer



Lecture 6 P18

Two key network-layer functions

•转发和路由

Network-layer functions:

- **Forwarding**: move packets from router's input to appropriate router output
- **Routing**: determine route taken by packets from source to destination
  - Routing algorithms

Analogy: taking a trip

- forwarding: process of getting through single interchange
- routing: process of planning trip from source to destination

路由器结构

Lecture6 P23.

~~最长前缀匹配~~ P27

Longest prefix matching

- When looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010*** **********	0
11001000 00010111 00011000 **********	1
11001000 00010111 00011*** **********	2
otherwise	3

Examples:

DA: 11001000 00010111 00010110 10100001 which interface?

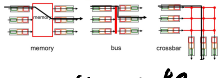
DA: 11001000 00010111 00011000 10101010 which interface?

input → switch fabric → output

- switching fabrics 交换结构 P29.

Switching fabrics 交换结构

- Transfer packet from input buffer to appropriate output buffer
- Switching rate: rate at which packets can be transfer from inputs to outputs
  - Often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- Three types of switching fabrics:



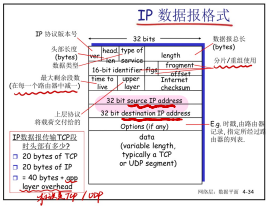
- output ports P34.

⇒ 优先级调度. P37.

{ FIFO
priority
RR
Weighted Fair Queuing

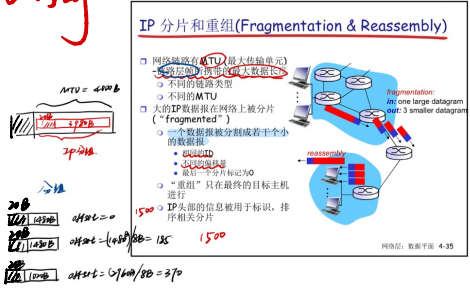
数据平面

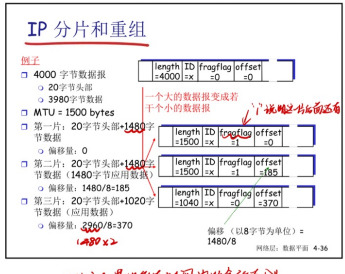
- IP datagram.



- IP分片和重组 (Fragmentation & Reassembly)

MTU 问题

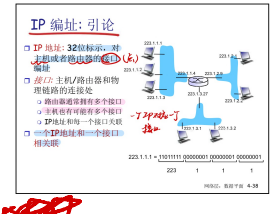


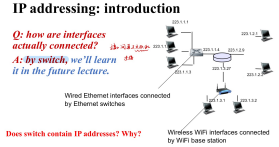


目标主机如果没能在时间内收齐所有分片,
⇒ 将收到的所有分片也丢弃

- IP addressing

Lecture 7 P6

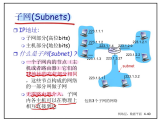


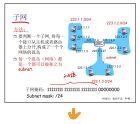


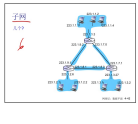
Lecture 7 P7

- Subnets

判断子网个数.







CIDR.

依据物理网络可以是任意长度.

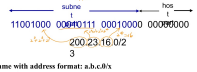
⇒ a.b.c.d /x

↑ subnet mask.

IP addressing: CIDR

CIDR: Classless Inter Domain Routing

- Subnet portion of address can have arbitrary length
- Address format: a.b.c.d/x, where x is # bits in subnet portion of address



192.168.2.|

Network name with address format: a.b.c.0/x
Subnet mask: /x
Network prefix: x

- 主机如何获得IP addresses. ⇒ DHCP Lecture 1 P12

DHCP: Dynamic Host Configuration Protocol

(UDP)

Goal: allow host to *dynamically* obtain its IP address from network server when it joins network

- Can renew its lease on address in use
- Allows reuse of addresses (only hold address while connected/ "on")
- Support for mobile users who want to join network (more shortly)

DHCP overview:

- host broadcasts "DHCP discover" msg [optional] 多状DHCP
- DHCP server responds with "DHCP offer" msg [optional] DHCP > broadcast
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

12

DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- Address of first-hop router for client 客户端的第一跳路由器的IP地址
- Name and IP address of DNS sever
- Network mask (indicating network versus host portion of address)

- Subnet 如何获得 IP.?

IP addresses: how to get one?

一个子网如何获得IP地址？

Q: how does *network* get subnet part of IP addr?

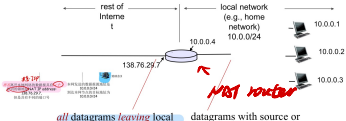
A: gets allocated portion of its provider ISP's address space

ISP's block	11001000 00010111 00010000 00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000 00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010 00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100 00000000	200.23.20.0/23
...	....	....
Organization 7	11001000 00010111 00011110 00000000	200.23.30.0/23

- NAT.

NAT: network address translation

Why NAT? If the subnet grows bigger, what if the ISP had already allocated the contiguous portions of address range? And what typical network admin wants to know how to manage IP addresses in the first place?



all datagrams *leaving* local network have *same* single source NAT IP address: 138.76.29.7, *different* source port numbers

datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

22

NAT: network address translation

Motivation: local network uses just one IP address as far as outside world is concerned:

- Range of addresses not needed from ISP: just one IP address for all devices
- Can change addresses of devices in local network without notifying outside world
- Can change ISP without changing addresses of devices in local network
- Devices inside local net not explicitly addressable, visible by outside world (a security plus)

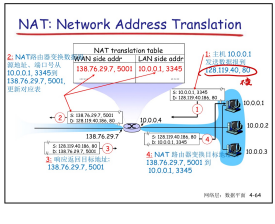
NAT: Network Address Translation

实现: NAT 路由器必须:

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

网络层：数据平面 4-63

NAT: Network Address Translation



网络层：数据平面 4-64

• IPv6

(IPv4 的 32bit) 不够用

- 头部格式改变帮助QoS

32bit → 128bit

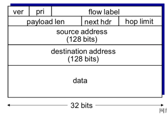
IPv6 数据报格式:
- 固定的40 字节头部
- 数据报传输过程中，不允许分片

分组太大 → 丢掉 → 发送 Icmp 错误报告

网络层：数据平面 4-70

IPv6 头部 (Cont)

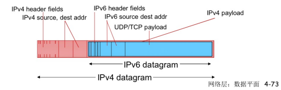
Priority: 标示流中数据报的优先级
Flow Label: 标示数据报在一个"flow."
("flow"的概念没有被严格的定义)
Next header: 标示上层协议



网络层：数据平面

从IPv4到IPv6的平移

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



网络层：数据平面 4-73

• SDN

OpenFlow 数据平面抽象

- 流: 由分组（帧）头部字段所定义
- 通用转发: 简单的分组处理规则
  - *模式:* 将分组头部字段和流表进行匹配
  - *行动:* 对于匹配上的分组，可以是丢弃、转发、修改、将匹配的分组发送给控制器
  - *优先权Priority:* 几个模式匹配了，优先采用哪个，消除歧义
  - *计数器Counters:* #bytes 以及 #packets

通用流表

路由器中的流表定义了路由器的匹配+行动规则
（流表由控制器计算并下发）

网络层：数据平面 4-96

Destination-based layer 2 (switch)

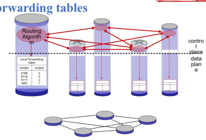
Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	22:A7:23:11:E1:02	*	*	*	*	*	*	*	*	port3

layer 4: 传输层
layer 3: 网络层
layer 2: 链路层

layer 2 frames from MAC address 22:A7:23:11:E1:02 should be forwarded to output port 3

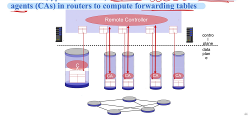
Per-router control plane

- Individual routing algorithm components ***in each and every router*** interact with each other in control plane to compute forwarding tables



Logically centralized control plane

- A distinct (typically remote) controller interacts with local control agents (CAs) in routers to compute forwarding tables



Two approaches to structuring network control plane:
- per-router control (traditional)
- logically centralized control (software defined networking) SDN

• Routing protocols

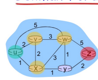
第5章
- Dijkstra algorithm. lecture 7 P53.
- Distance Vector algorithm. lecture 8 P3.

Distance vector algorithm

- $D_x(y)$ = estimate of least cost from x to y
  - x maintains estimate of distance vector $D_x = [D_x(y): y \in N]$
- Node x:
  - Knows cost to each neighbor v: $c(x,v)$
  - Maintains its neighbors' distance vectors. For each neighbor v, x maintains $D_v = [D_v(y): y \in N]$

$D_x(y) \leftarrow \min_v\{c(x,v) + D_v(y)\}$ for each node $y \in N$

Bellman-Ford例子



明显的，$d_v(z) = 5$, $d_x(z) = 3$, $d_w(z) = 3$

由B-F 方程得到：

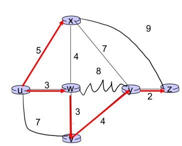
$$d_u(z) = \min\{ c(u,v) + d_v(z),\ c(u,x) + d_x(z),\ c(u,w) + d_w(z) \}$$
$$= \min\{2 + 5,\ 1 + 3,\ 5 + 3\} = 4$$

那个能够达到目标最小代价的节点x，就在到目标节点的下一条路径上，在转发表中使用

网络层：控制平面 5-52

Dijkstra's algorithm: example 1

Step	N	D(v) p(v)	D(w) p(w)	D(x) p(x)	D(y) p(y)	D(z) p(z)
0	u	7,u	3,u	5,u	∞	∞
1	uw	6,w		5,u	11,w	∞
2	uwx	6,w			11,w	14,x
3	uwxv				10,v	14,x
4	uwxvy					12,y
5	uwxvyz					



notes:
- construct shortest path tree by tracing predecessor nodes
- ties can exist (can be broken arbitrarily)

• Scalable routing lecture 8 P14

Internet approach to scalable routing

将路由器聚合到 "AS" 中

Aggregate routers into regions known as "autonomous systems" (AS) (a.k.a. "domains")

Intra-AS routing 内部网关协议

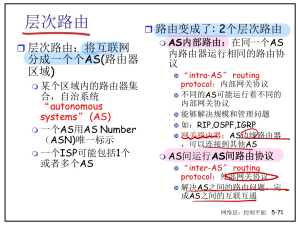
- Routing among hosts, routers in same AS ("network")
- All routers in AS must run *same* intra-domain protocol
- Routers in *different* AS can run *different* intra-domain routing protocol
- Gateway router: at "edge" of its own AS, has link(s) to router(s) in other AS'es

网关路由器：主要处理AS间的路由

Inter-AS routing 外部网关协议

- Routing among AS'es
- Gateways perform inter-domain routing (as well as intra-domain routing)

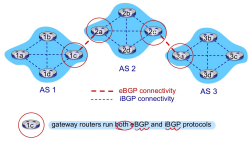
13

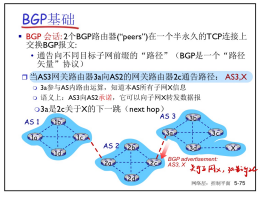


Internet inter-AS routing: BGP 外部网关协议

- **BGP (Border Gateway Protocol):** the *de facto* inter-domain routing protocol
  - "glue that holds the Internet together"
- BGP provides each AS router a means to:
  - eBGP: obtain subnet prefix reachability information from neighboring ASes. Allows subnet to advertise its existence to rest of Internet: *"I am here."*
  - iBGP: propagate reachability information to all AS-internal routers, and determine "best" routes to other networks based on reachability information and *policy*

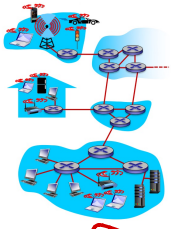
eBGP, iBGP connections





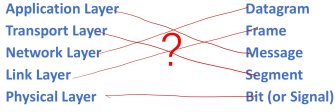
Link Layer.

What's Link Layer



- Data-Link Layer has responsibility of
  - transferring datagram
  - from one node to physically adjacent node
  - over a link
- Node:
  - Hosts and routers
- Link:
  - communication channel
  - connection adjacent nodes
- Layer-2 packet: frame
  - Encapsulates datagram

Data Unit

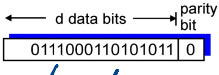


Detecting errors.

parity checking

Parity checking – Single bit parity

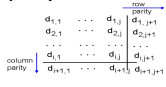
- Even/Odd parity scheme:
  - Add an additional bit
  - Total number of 1 (D+1) is even/odd
- Detect single bit errors



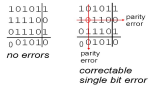
Even scheme or odd scheme?

Parity checking – 2D parity

- Detect and correct single bit errors



- Two-dimensional even parity



checksum

Checksum

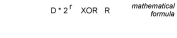
- In the **TCP and UDP** protocols, the Internet checksum is computed over **all fields (header and data fields included).**
- In **IP**, the checksum is computed over the **IP header** (since the UDP or TCP segment has its own checksum).
- 发送方首先根据数据算出来一个校验和，然后接收方再自行计算，看看一不一样，使用addition (one's complement sum) of segment contents

Cyclic redundancy check (CRC)

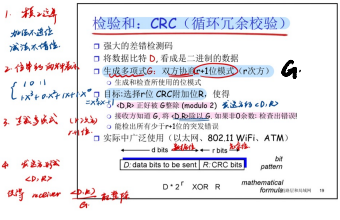
- view Data bits, D, as a binary number
- choose an r+1 bit pattern (generator), G
  - both sender and receiver agree on that
- goal: find r CRC bits, (remainder), R, such that
  - a d+r bit pattern <D, R> exactly divisible by G (modulo-2)
- receiver knows G, divides <D, R> by G.
  - if non-zero remainder: error detected!
  - can detect all burst errors less than r + 1 bits

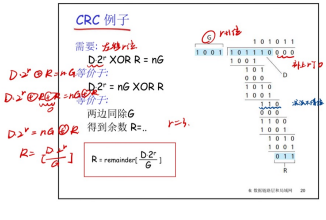
G is given for both sender and receiver. The most significant (leftmost) bit of G be a 1

R needs to be computed by the sender.



CRC





Cyclic redundancy check (CRC)

- all CRC calculations are done in modulo-2 arithmetic without carries in addition or borrows in subtraction, including Addition, Subtraction, Bitwise exclusive-or (XOR) (异或运算, 同得0,异得1)
- 题目会给你用来除的多项式, 假设给你的是X8+ X2+X1+1, 展开1*X8+0*X7+0*X6+0*X5+0*X4+0*X3+1*X2+1*X1+1，所以除数是100000111
- 在开始算CRC之前还要在原数据之后加上多项式最高阶数的0
- 算完以后的那部分就是加在原数据后面
- 校验的点在于接收方在收到数据和发送方算出的原数据的校验和后，还需自行计算出一个收到数据的校验和来与收到的进行比较，从而达到检验的效果

- 多点访问 Lecture 9 P21-33

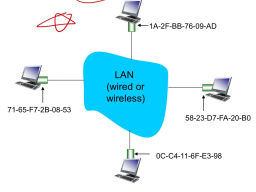
- MAC address & ARP.

MAC addresses and ARP

- **IP address**
  - IPv4 (32 bits) and IPv6 (128 bits)
  - Network-layer address
  - Layer-3 forwarding
- **MAC (LAN) address**
  - 48 bits (e.g., 12-34-56-78-90-AB)
    hexadecimal (base 16) notation (each "numeral" represents 4 bits)
  - Burned in NIC ROM
  - Used "locally" get frame from one interface to another physically-connected interface (same network, in IP-addressing sense)

MAC addresses and ARP

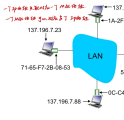
Each adapter on LAN has unique *LAN* address (MAC)



同一 LAN

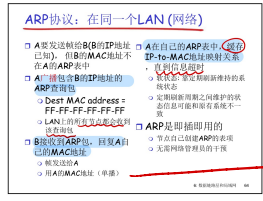
ARP: address resolution protocol

- How to determine interface's MAC address, if knowing its IP address?



ARP table: each IP node (host, router) on LAN has the table
- An IP-to-MAC address mappings for LAN nodes:
  < IP address; MAC address; TTL>
- A TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)

Notice that an ARP table is not necessary to contain an entry for every host and router on the subnet; some may have never been entered into the table, and others may have expired.

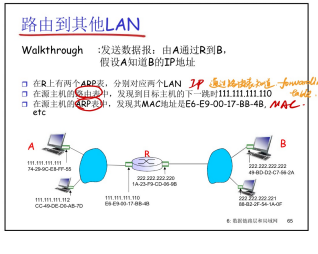


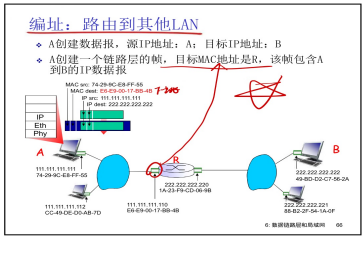
A给MAC地址写从以太网协议请求报文中得出.

ARP实例

只有跨过子网的传输才需要将帧交给路由器，帧中包含目的地IP和路由器与出发地所处子网接口的MAC地址

不同 LAN.





- Ethernet Frame. (protocols)

Ethernet frame structure

- Sending adapter encapsulates IP datagram (or other network layer protocol packet) in Ethernet frame

preamble: 8-byte
- 7 bytes with pattern 10101010 followed by one byte with pattern 10101011
- used to synchronize receiver, sender clock rates

Ethernet frame structure

- ***Addresses:*** 6 byte source, destination MAC addresses
  - If adapter receives frame with matching destination address, or with broadcast address (e.g. ARP packet), it passes data in frame to network layer protocol
  - Otherwise, adapter discards frame
- ***Type:*** indicates higher layer protocol (mostly IP but others possible, e.g., Novell IPX, AppleTalk)
- ***CRC:*** cyclic redundancy check at receiver
  - Error detected: frame is dropped

Ethernet: unreliable, connectionless

- ***Connectionless:*** no handshaking between sending and receiving NICs
- ***Unreliable:*** receiving NIC doesn't send acks or nacks to sending NIC
  - Data in dropped frames recovered only if initial sender uses higher layer rdt (e.g., TCP), otherwise dropped data lost
- Ethernet's MAC protocol: unslotted ***CSMA/CD with binary backoff (using with sensing channel idle)***

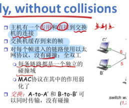
ARP表：IP节点上 —⊗— 两了ARP表.

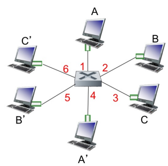
路由表：路由器上
switch table：交换机上.
} forwarding table

• switch.

Switch: multiple simultaneous transmissions
多路同时传输

- Hosts have dedicated, direct link to switch (专用, 直接)
- Switches buffer packets
- Ethernet protocol used on *each* incoming link, but no collisions; full duplex
  - Each link is its own collision domain
- ***Switching, e.g.,:*** A-to-A' and B-to-B' can transmit simultaneously, without collisions



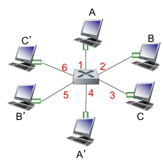


switch with six interfaces (1,2,3,4,5,6)

3

Switch (forwarding) table

- ***Q:*** how does switch know A' reachable via interface 4, B' reachable via interface 5?
- *A:* each switch has a switch table, which contains entries, and each entry has:
  - (1) A MAC address, (2) the switch interface toward that MAC address, (3) the time stamp that the entry was placed.
  - looks like a routing table!
- Q: how are entries created, maintained in switch table?
  - something like a routing protocol?

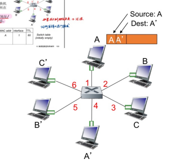


switch with six interfaces (1,2,3,4,5,6)

4

Self-learning

- Switch *learns* which hosts can be reached through which interfaces
  - When frame received, switch "learns" location of sender: incoming LAN segment/link
  - Records MAC/interface (sender/location) pair in switch table
  - Entries get removed if no frames with that MAC received after timeout (which can be configured)

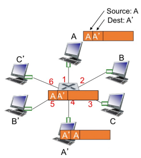


MAC addr	interface	Time
A	1	9:10

Switch table (initially empty)

Self-learning

- frame destination, A', location unknown: flood
- destination A location known: selectively send on just one link



MAC addr	interface	Time
A	1	9:10
A'	4	9:15

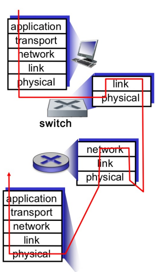
switch table (initially empty)

6

Switches vs. routers

Both are store-and-forward:
- ***Routers:*** network-layer devices (examine network-layer headers)
- ***Switches:*** link-layer devices (examine link-layer headers)

Both have forwarding tables:
- ***Routers:*** compute tables using routing algorithms, IP addresses
- ***Switches:*** learn forwarding table using flooding, learning, MAC addresses



10

Lecture 10 P25.

请求google网址的全过程.

Network Security

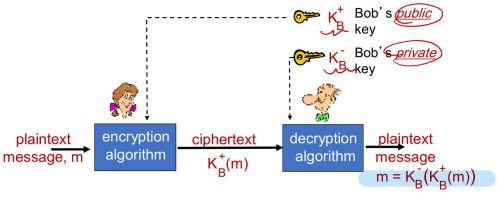
△ Symmetric key cryptography

DES, AES

symmetric key crypto

- requires sender, receiver know shared secret key
- **Q**: how to agree on key in first place (particularly if never "met")?

△ public key cryptography.



RSA算法 有了 K_B^+ 不能推出 K_B^- lecture 10 P51

$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$

RSA 快全: secure

缺点: computationally intensive
DES is faster than RSA

应用: use public key crypto to establish secure connection
then establish second key – symmetric session key – for encrypting data.
faster.

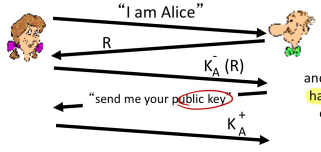
session key, K_S

- Bob and Alice use RSA to exchange a symmetric key K_S
- once both have K_S, they use symmetric key cryptography

Authentication

- ap4.0 requires shared symmetric key
- can we authenticate using public key techniques?

ap5.0: use nonce, public key cryptography

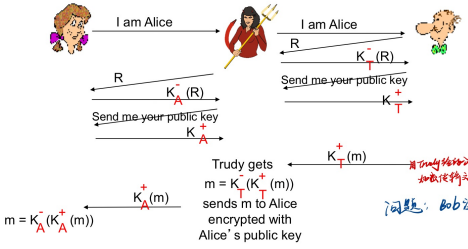


Bob computes $K_A^+(K_A^-(R)) = R$

and knows only Alice could have the private key, that encrypted R such that

$K_A^+(K_A^-(R)) = R$

man (or woman) in the middle attack: Trudy poses as Alice (to Bob) and as Bob (to Alice)



13

Message integrity

Digital signatures

- suppose Alice receives msg m, with signature: m, $K_B^-(m)$
- Alice verifies m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$.
- If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key.

Alice thus verifies that:
- Bob signed m
- no one else signed m
- Bob signed m and not m'

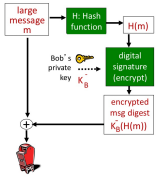
non-repudiation:
- ✓ Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m

computationally expensive to public key-encrypt long messages

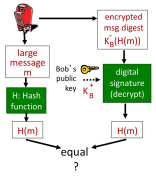
⇒ H(m) : Hash function.

Digital signature = signed message digest

Bob sends digitally signed message:



Alice verifies signature, integrity of digitally signed message:

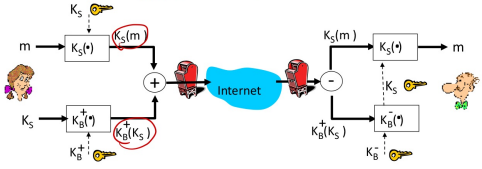


confidential

Secure e-mail

发送加密信息

Alice wants to send confidential e-mail, m, to Bob.



Alice:

- generates random symmetric private key, K_S
- encrypts message with K_S (for efficiency) $K_S(m)$
- also encrypts K_S with Bob's public key $K_B^+(K_S)$
- sends both $K_S(m)$ and $K_B^+(K_S)$ to Bob

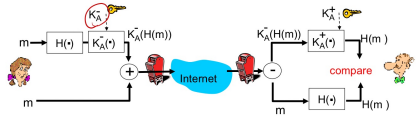
28

认证 + 完整性.

Secure e-mail (continued)

实现认证.

Alice wants to provide sender authentication message integrity



- Alice digitally signs message
- sends both message (in the clear) and digital signature

发送信息(明文) + 电子签名

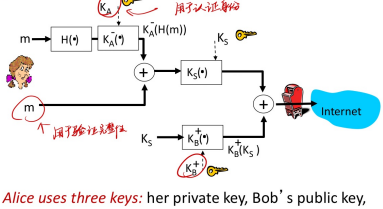
30

认证 + 完整性 + 加密

Secure e-mail (continued)

发送加密信息 + 实现认证.

Alice wants to provide secrecy, sender authentication, message integrity.



Alice uses three keys: her private key, Bob's public key, newly created symmetric key

SSL: TCP上的 confidentiality / integrity / authentication

Ipsec: IP上的