

△ Network Core

~~packet switching~~

Lecture 1 Part



store and forward

⇒ queueing delay, loss

Packet Switching: queueing delay, loss



Queuing and loss:

- * if arrival rate is higher than speed of a transmission pair of link for a period of time T , packets will be discarded.
- * packets will be transmitted on link.
- * packets can be discarded (lost) if memory buffer fills up.

丢包情况
retransmitted by previous
source and system
not at all

packet switching allows more users to use network.

~~优点~~

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 对带宽要求很高 带宽要求
- New methods should be developed

~~Circuit switching~~

PS

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, 在终点可根据顺序还原合成完整数据

计算举例

□ 在一个电路交换网络上, 从主机A到主机B发送一个640,000字节的文件需要多长时间?

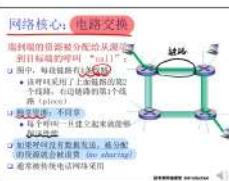
- 所有的链路速率均为1.536 Mbps
- 每条链路平均时延为240ms
- 建立唯一的电连接500 ms

每条链路的速率 (一个时间片): $1.536 \text{ Mbps} / 24 = 64 \text{ kbps}$

传输时间: $640,000 \text{ bytes} / 64 \text{ kbps} = 10 \text{ s}$

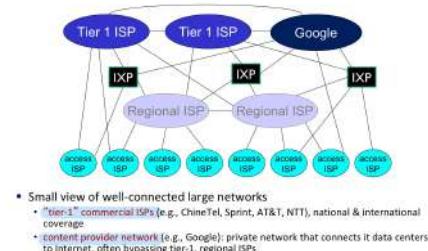
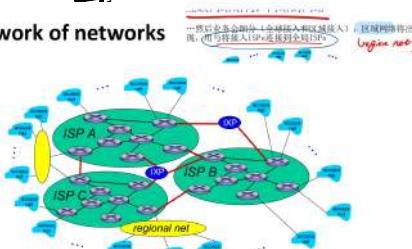
共用时间: 传输时间 + 建立连接时间 $= 10 \text{ s} + 500 \text{ ms} = 10.5 \text{ s}$

总耗时: 10.5s



端系统通过 ISP 连接 Internet \Rightarrow connect each access ISP to one global transit ISP

Network of networks



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播时间

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 减少到 100kbps。

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

$$1 \text{ Kbps} = 10^3 \text{ bps}$$

$$1 \text{ Mbps} = 10^6 \text{ bps}$$

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

可移植性差. poor scalability.

P2P architecture

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

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，通过交换消息（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

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 (私有IP地址)

● 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

一个进程可以有多少 network identifiers.

Addressing processes 对进程进行编址

○ 每台设备都有 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:f0c4:4269:7442:7ea3]:80

● Can a process have multiple network identifiers?

IP + port number = end point

○ 端口知道自己的网络地址和自己的物理地址

IP + port 确认一个 process.

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

TCP

Lecture 2 P22

SSL: provides encrypted TCP connection

UDP

Lecture 2 P24

存在的必要性 / 优先.

TCP, UDP 都不提供服务质量: delay guarantees, bandwidth guarantees.

transmitted data may be lost or received out of order.



Two socket types for two transport services:

- UDP: unreliable datagram 不可靠的报文数据
- TCP: reliable, byte stream-oriented 面向字节流

HTTP

(Application Layer)

Lecture 2 P32



CS model

use TCP

关注下 HTTP 请求, 响应报文格式

stateless (server 不存储 client 的信息)
无状态的 server 支持更多的 client)



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/1.1



Persistent HTTP

Non-persistent HTTP issues:

- Requires 2 RTTs for each object
 - OS overhead for each TCP connection
- Browsers often open separate TCP connections to fetch referenced objects

Q: Is persistent HTTP perfect? <=

Persistent HTTP issues:

- Server keeps connection open after sending response
- Subsequent HTTP messages between client/server send over open connection
- Browser must keep connection open as it encounters a referenced object.
As little as one RTT for all the referenced objects
- As little as one RTT for all the referenced objects

↓
response time
RTT + 读取时间



HTTP 请求报文

HTTP request message

Two types of HTTP message: REQUEST, RESPONSE

HTTP 1.1 REQUEST message:

ASCII (human readable format)

HTTP 1.1 RESPONSE message:

Binary (machine readable format)

HTTP/1.1 200 OK

Content-Type: text/html

Content-Length: 12345

Content: Hello World!



HTTP request, response 部分

Cookies

Lecture 2 Pg3

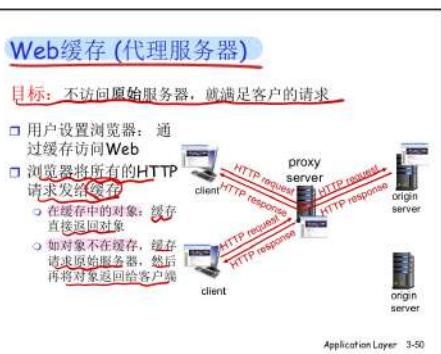
使得 HTTP: stateless → stateful.



隐私问题

Web Caches

Lecture 2 Pg7.



Web 缓存 (代理服务器)

目标: 不访问原始服务器, 就满足客户的请求

- 用户设置浏览器: 通过缓存访问 Web
- 浏览器将所有的 HTTP 请求发给缓存
- 在缓存中的对象: 缓存直接返回对象
- 如对象不在缓存, 缓存再将对象返回给客户机

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)

多款网关部署了缓存
节省带宽
节省带宽
节省带宽

减少了通过链路的带宽, Web 访问更快. (速度更快)
(increased access link speed)

应用层协议: HTTP, SMTP, FTP

DNS

c/S model

Lecture 3 p5

实现主机名 - IP地址的转换

(a distributed, hierarchical database)

不集中管理:

集中 DNS

Why not centralize DNS?

- Single point of failure
- Traffic volume
- Distant centralized database
- Maintenance

VDP

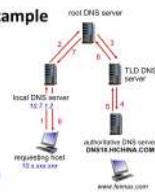
Recursive query

P11

DNS name resolution example

• Recursive query: 递归查询

- Puts burden of name resolution on contacted name server
- Heavy load at upper levels of hierarchy



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

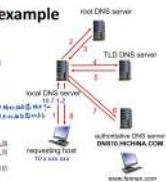
Iterated query

DNS name resolution example

• Host at JXTU wants IP address for www.feimiao.com

• Iterated query: 迭代查询

- contacted server replies with name of server to contact
- "I don't know the name, but ask this server!"



P2P

P29.

△文件分发



老师

BitTorrent Lecture 3 p36

Lecture 3 p36

Piece selection - Macro view



⇒ selection policy

Random First Piece

- Initially, a peer has nothing to trade
- Important to get a complete piece ASAP
- Select a random piece of the file and download it

First Fit

- Download the first piece that fits in the available space
- This ensures that the next consecutive selectable piece are left till the end to download

Endgame Mode

- Near the end, missing pieces are requested from every peer containing them
- This is called 'endgame mode'
- It's important to have a good endgame strategy
- Some inefficiency is wasted, but in practice, this is not too much

Choking

Choking

- Choking is a temporary refusal to upload. It is one of BT's most powerful idea to deal with free riders (those who only download but never upload).

• For avoiding free riders and avoiding network congestion.

• The **Don't-Ask strategy** is based on game-theoretic concepts.

Optimistic unchoking

- A peer needs always to have four peers currently sending her chunks
- Other peers are choked by Alter (do not receive chunks from her)

• If a peer has less than 4 unchoked peers, starts sending chunks

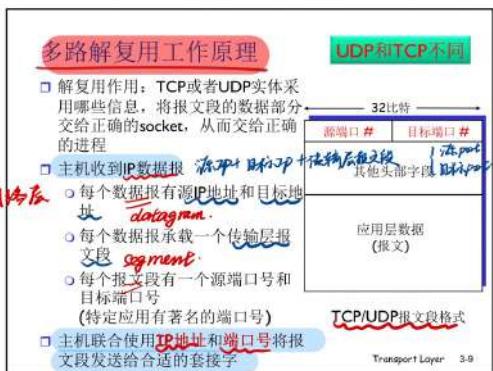
• "optimistically choke", this peer

• rarely chosen peer may join up

- Reasons:
 - To prevent currently unchocked connections that are better than the unchocking peer
 - To provide minimal service to new peers

多路复用 / 解复用

Lecture 4 P11



UDP

TCP

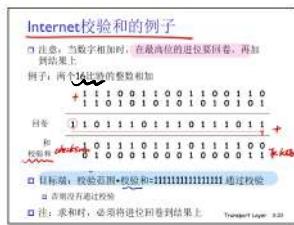
TCP报文段以直连方式传到不同的进程
不同的socket

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 (e.g. bit errors)
addition of pseudo header, UDP header, UDP data



- 目标端：校验和应和=1111111111111111通过校验
 - 若失败，则丢弃该报文
 - 注：求和时，必须将进位回卷到结果上
- Transport Layer 3.9

UDP checksum

Goal: detect "errors" in transmitted segment

Sender:

- Treat segment contents, including header fields, as sequence of 16-bit integers.
- Checksum: addition from's complement' of pseudo header, UDP header and UDP data
- Sender puts checksum value into UDP checksum field.

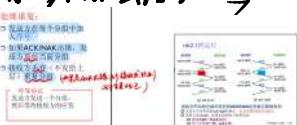
Receiver

- Compute checksum of received segment
- Check if computed checksum equals checksum field value:
 - NO - error detected
 - YES - no error detected, But maybe errors nonetheless?

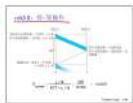
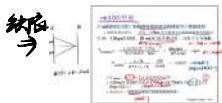
Reliable data transfer (RDT)

Lecture 4 P26

- rdt 2.0 → ACK/NAK出错？ 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 (可能会丢失分组)



Lecture 5, 6 还没有

超时重传机制

一致数据流 → 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

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

~~最长前缀匹配~~

Longest prefix matching

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

Destination Address Range	Line Interface
00000000000000000000000000000000	0
00000000000000000000000000000001	1
11001000000000000000000000000000	2
otherwise	3

Examples:
DA: 00000000000011110000000000000000 which interface?
DA: 11001000000000000000000000000000 which interface?

• switching fabrics 支持结构

P29

Switching fabrics 支持结构

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



• one output ports

P30

⇒ 优先级调度

P31

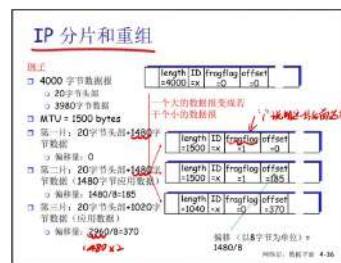
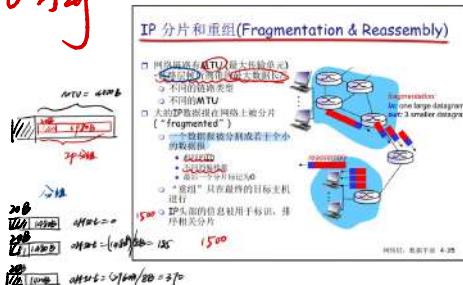
- | |
|-----------------------|
| FIFO |
| Priority |
| RR |
| Weighted Fair Queuing |

数据平面

• IP datagram.



~~IP 分片和重组 (Fragmentation & Reassembly)~~
MTU 计算



目标主机累积收到花时间内快进所有分组

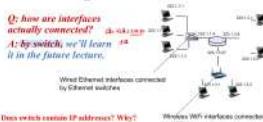
由收到的分组重新组装

• IP addressing

Lecture 7 Pg



IP addressing: introduction

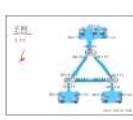
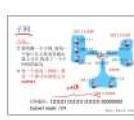


地址块与网段间不是一一对应

$\Rightarrow a.b.c.d/x$
↑
subnet mask.

• Subnets

判断子网掩码



CIDR.

IP addressing: CIDR

CIDR: Classless Inter Domain Routing

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



192.168.2.1

• 主机如何获得 IP 地址。 \Rightarrow DHCP Lecture 7 P12

(UDP)

DHCP: Dynamic Host Configuration Protocol

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] *询问*
- DHCP server responds with “DHCP offer” msg [optional] *响应*
- host requests IP address: “DHCP request” msg *广播*
- DHCP server sends address: “DHCP ack” msg

• Subnet 什么获得 IP. ?

IP addresses: how to get one?

→ 从哪里获得 IP 地址？

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

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

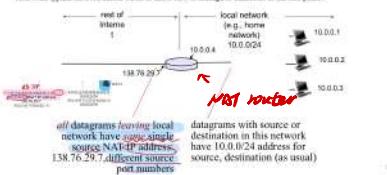
ISP's block	11001000_00010111_00010000_00000000	200.23.16.0/23
Organization 0	11001000_00010111_00010000_00000000	200.23.16.0/23
Organization 1	11001000_00010111_00010010_00000000	200.23.16.0/23
Organization 2	11001000_00010111_00010010_00000000	200.23.20.0/23
Organization 7	11001000_00010111_00011110_00000000	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?



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:

○ 由出数据包，替换源地址和目的地址为 NAT IP 地址和出的端口，且端口和端口号不要

... 远端的 C/S 将会用 NAT 地址，新端口号作为目的端址

○ 记住每个转换映射 (在 NAT 转换表中)

... 即 IP, 端口 vs NAT 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)

NAT: Network Address Translation



REF: BETH 444

• IPv6

(IPv4的32bit) 不够用

头部格式改变帮助QoS

32bit → 128bit

IPv6 数据报格式:

- 固定的数据报头长度
- 数据报传输过程中，不允许分片

分块大小→支持→发送 ICMP
请求报文

图例: 图版 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"的概念没有被严格地定义)

Next header: 标示上层协议

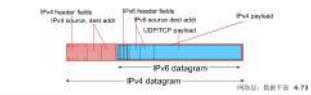
version	PL	Flow label	payload length
64	1	000000000000000000000000	128 bits
			destination address (128 bits)
			options
			data

32 bits

图例: 图版 4-71

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 day”
 - 在IPv4和IPv6路由器混合时，网络如何运行？
- 层剥: 在IPv4路由器之间传输的IPv4数据报中剥除IPv6数据报



• SDN

OpenFlow 数据平面抽象

- 由分组（桢）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、将匹配的分组发给控制器
- 优先级Priorit: 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



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)

图例: 图版 4-96

Destination-based layer 2 (switch)

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

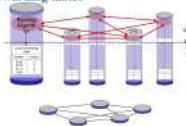
layer 4:
层4
层3:
层3
层2:
层2

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 logically centralized controller interacts with local agents (LAs) to manage the network.



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

$D_i(y) = \text{estimate of least cost from } i \text{ to } y$

\times maintains estimate of distance vector $D_i = [D_{ij}(y)] \forall y \in N$

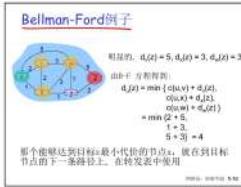
\times periodic update

\times known cost to each neighbor of i (cost)

Maintains its neighbors' distance vectors. For each neighbor y , x maintains

$D_{ij} = [D_{iy}(y) \forall y \in N]$

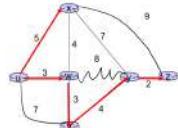
Distance vector protocol + Bellman-Ford for each neighbor $y \in N$



Dijkstra's algorithm: example 1

Step	N	D(v) v1 d1,2 d1,3 d1,4 d1,5	D(w) w1 d2,1 d2,3 d2,4 d2,5	D(x) x1 d3,1 d3,2 d3,4 d3,5	D(z) z1 d4,1 d4,2 d4,3 d4,5
0	U	7, 2, 4, 3, 5	U	U	U
1	UW	6, w, U	6, w, U	11, w	11, w
2	UWU	6, w, 6, w	6, w, 6, w	11, w	11, w
3	UWW	6, w, 6, w, 10, w	6, w, 6, w, 10, w	11, w	10, w
4	UWWY	6, w, 6, w, 10, y	6, w, 6, w, 10, y	11, w	10, y
5	UWWZY	6, w, 6, w, 10, y, 12, y	6, w, 6, w, 10, y, 12, y	11, w	10, y

- constructed shortest path tree by tracing predecessor nodes
- tree can exist (can be broken arbitrarily)



• Scalable routing

lecture 8 P.14

Internet approach to scalable routing

将路由器分为 AS

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 ASes, has link(s) to router(s) in other ASes

Inter-AS routing 对外网关协议

- Routing among ASes
- Gateways perform inter-domain routing (as well as intra-domain routing)

层次路由

- 层次路由: 将互联网分成一个个AS(路由器区域)
 - 某个区域内的路由器集合, 自治系统: "autonomous systems" (AS)
 - 一个AS用AS Number (ASN)唯一标识
 - 一个ISP可能包括1个或者多个AS
- 路由变成了: 2个层次路由
 - AS内层路由: 在同一AS内路由器运行相同时的路由协议
 - "Intra-AS" routing protocol: 内部网关协议
 - 不同的AS内路由器有不同的IP地址
 - 避免冲突解决和管理问题
 - 如: RIP, OSPF, IS-IS
 - 可以通过配置静态路由
 - AS之间"AS"路由协议
 - "Inter-AS" routing protocol
 - 外部网关协议: EGP, BGP

WAN, Internet, LAN

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



BGP 基础

- BGP 是 2 个 BGP 路由器("peers")在一个半永久的 TCP 连接上交换 BGP 路由信息
 - 通常不直接基于网段级别的“路径” (BGP 是一个“路径”协议, 而不是“网段”协议)

□ (AS1 先向 AS2 提供 32 位子网的可达性信息, AS2X)

▪ 24 位 “AS” 的信息, 识别 5 ASes (74000)

▪ 16 位上, AS-Path 属性, 它记录所有 AS 子网数信息

▪ 32 位 24 位 “AS”的下一跳 “next hop”

▪ 32 位 32 位 “AS” 属性

▪ 32 位 32 位 “metric” 属性

AS-Path

Next Hop

Metric

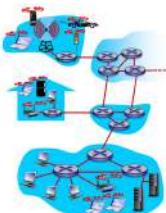
AS-Path

Next Hop

Metric

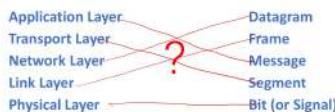
Link Layer.

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



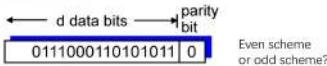
~~OSI Model~~

Detecting errors.

△ parity checking

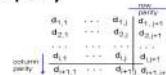
Parity checking – Single bit parity

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

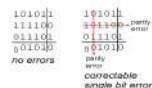


Parity checking – 2D parity

- Detect and correct single bit errors



- Two-dimensional even parity



△ checksum

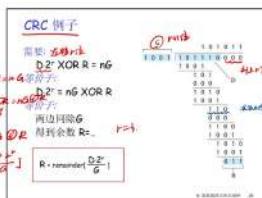
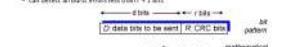
Checksum

- 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，看看不一样，使用addition (one's complement sum) of segment contents

△ CRC

Cyclic redundancy check (CRC)

- view Data Bits, D , as a binary number
- choose an $r+1$ bit pattern (**generator**, R)
- both parties must agree on that
- generate r CRC bits, from $D \oplus R^r$, such that
 - $D \oplus R^r = 0$ (or equivalently, $D \equiv 0 \pmod{R}$)
 - generator known G , divided D by R by \oplus
 - If non-zero remainder, error detected
 - can detect all burst errors less than $r+1$ bits



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), 同或得0, 异得1)
- 题目会给你用起来的多项式。假设给你的是 $X^8 + X^2 + X + 1$, 展开 $1+X^8+0+X^2+0+X^6+0+X^5+1+X^4+0+X^3+1+X^2+1+X+1$, 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
- 算完之后的那部分就是加到数据和发送方算出的原数据的校验后。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 (MAC)

Each adapter on LAN has unique LAN address



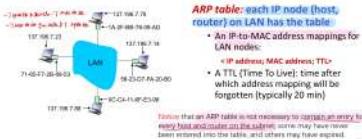
ARP实例

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



ARP: address resolution protocol

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

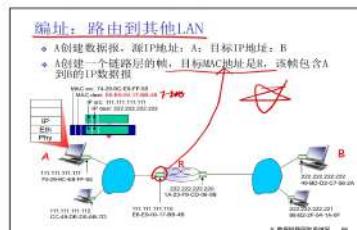


ARP协议: 在同一个LAN (网络)

- A是发送端,B是接收端
- A在自己的ARP表中查自己的IP-MAC地址映射关系
- A向自己的IP-MAC地址映射关系对应的MAC地址广播ARP请求
- B接收到ARP请求后, 根据自己的IP-MAC地址映射关系, 将自己的IP-MAC地址映射关系对应的MAC地址广播出去
- A接收到B的响应后, 将B的IP-MAC地址映射关系插入自己的ARP表中
- A将B的IP-MAC地址映射关系插入自己的ARP表中

→ ARP消息在同一个LAN上广播
请求报文广播.

△ 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

- **Address:** 6 bytes, destination MAC address
 - If adapter receives frame with matching destination address, or with broadcast (e.g. ARP packet), it passes data in frame to network layer protocol
 - Otherwise, adapter drops frame
- **Type:** indicates higher layer protocol (most 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
 - If a corrupted frame is received, the sending adapter uses higher layer (e.g., TCP), otherwise discards data lost
- Ethernet's MAC protocol: unslotted CSMA/CD with binary backoff (using with sensing channel idle)

APP表: IP层上 $\xrightarrow{\text{X}}$ 两个ARP表.

路由表: 路由器上

> forwarding table

Switch table: 支线加上

• Switch.

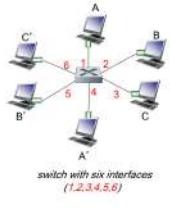
多路同时传输

Switch: multiple simultaneous transmissions

- Hosts have dedicated, direct link to switch
- 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



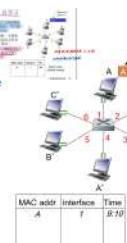
switch with six interfaces
(1,2,3,4,5,6)

Self-learning

- Switch learns which hosts can be reached through which interface

- When frame received, switch "learns" location of sender/ incoming LAN segment/link
- Records MAC/interface (sender/location) pair in switch table

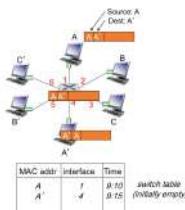
- Entries get removed if no frames with that MAC received after timeout (which can be configured)



MAC addr	Interface	Time
A	1	8:10

Self-learning

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



MAC addr	Interface	Time
A	1	8:10
A'	4	8:15

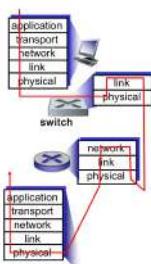
Switches vs. routers

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



Lecture 10 Pg5

请求 goole 网址的全过程.

Network Security

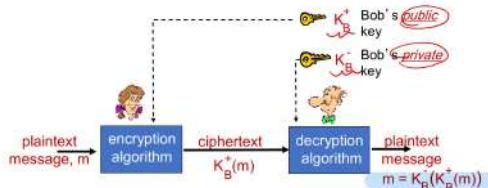
△ Symmetric key cryptography

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

有3種不能解密的
key

Lecture 10 P51

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

rsa 特點: secure

特點: } computationally intensive
} DES is faster than RSA

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

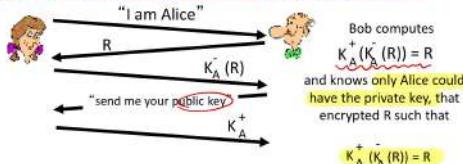
session key, K_S

- 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?

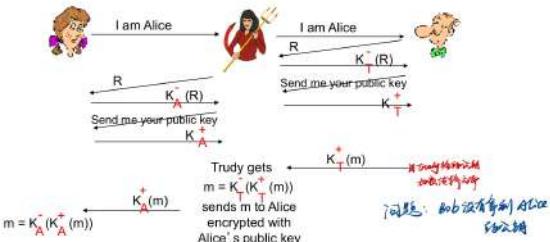
ap5.0: use nonce, public key cryptography



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

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

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



13

密文 Bob 从 Alice 得到: $K_A^+(Alice, K_A^+)$

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'

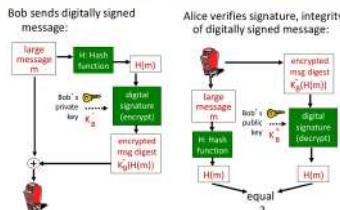
non-repudiation:

- 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

$\Rightarrow H(m)$: Hash function.

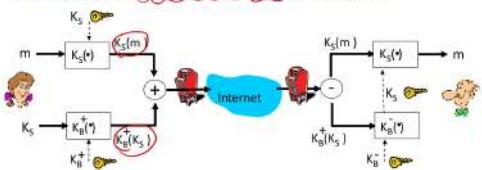
Digital signature = signed message digest



~~Confidential~~ Confidential

Secure e-mail

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



Alice:

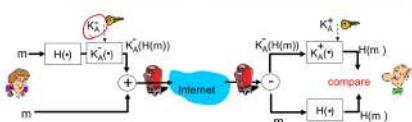
- generates random symmetric private key, K_s
- encrypts message with K_s (for efficiency)
- also encrypts K_s with Bob's public key K_b ($K_b(K_s)$)
- sends both $K_s(m)$ and $K_b(K_s)$ to Bob

28

认证 + 完整性

Secure e-mail (continued)

Alice wants to provide sender authentication message integrity



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

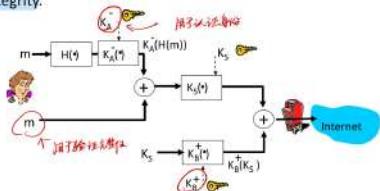
发送机密性 + 签名

30

认证 + 完整性 + 加密

Secure e-mail (continued)

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 + SSL confidentiality / integrity / authentication

IPsec : IP + SSL