# 第一章

- 网络边缘
- 网络核心
- 接入网

## 1.2 网络边缘

- 端系统（主机）
- C/S（client-server）模式
- 对等（peer-peer）模式

面向连接：TCP

- 可靠
- 流量控制（发送、接收方）
- 拥塞控制（链路）

无连接：UDP

- 不可靠
- 无流量控制
- 无拥塞控制
- 适合实时多媒体应用

## 1.3 网络核心

- 通信网络
    - 电路交换网络
        - FDM
        - TDM
    - 分组交换网络
        - 虚电路网络
        - 数据报网络
- 电路交换（独享资源-性能保障、传统电话网络采用）
- 分组交换（package switch，分组、转发、存储、再转发，存储转发避免整条线路被独占，共享性、存储延迟、排队、延迟，分组丢弃）

带宽分片：

- 频分（FDM、按照频率分）
- 时分（TDM、有点像计算机时间片的概念）
- 波分（光通信、按照波段分）

- 码分

计算机间通信具有突发性，如果使用线路交换，则浪费的片较多

#### 网络核心的关键功能

- 路由（全局、分组由源到目标的路径、路由算法、路由表）
- 转发（局部、将分组从路由器的输入链路转移到输出链路）

分组交换：统计多路复用（复杂的TDM）、适合突发性网络应用

#### 分组交换网络 存储-转发

按照由于网络层的连接，分成：

1. 数据报网络
    - 无需建立连接（无连接，不维护主机间通信状态，分组到来就转走）
    - 每个分组独立路由
    - 路由器根据分组的目标地址进行路由
2. 虚电路网络（有连接，有连接体现在中间所有交换节点之上，维护链路通信状态）

有连接：中间路由器维护他们通信状态 面向连接：仅仅体现在端系统和tcp实体上，中间路由器不维护通信状态

## 1.4 接入网和物理媒体

- 住宅接入：modem（调制解调器）、
    - 电话调制解调器
    - DSL
    - 电网
- 企业接入：
    - 交换机级联
- 无线
    - 无线 LANs
    - 广域无线接入（基站）

#### 物理媒体

物理链路 双绞线 同轴电缆 光缆 无线链路：地面微波、LAN（e.g.,wifi）、wide-area（e.g.,蜂窝）、卫星

## 1.5 Internet结构、ISP

网络的网络 ISP ICP IXP(Internet Exchange Point)

## 1.6 分组延迟、丢失、吞吐量

产生原因

- 排队延迟：分组到达路由器速率 > 链路输出能力
- 分组丢失：分组等待排到队头（超过路由器缓存容量）

延迟

- 分组延时
  - 处理延迟：检查bit级差错、检查分组首部和决定将分组导向何处
  - 排队延时：队列长度/处理速度
  - 传输延时：分组长度/链路带宽
  - 传播延时：物理链路长度/在媒体上传播速度

术语 RTT：Round Trip Time 往返延时 TTL：生存时间 ICMP

吞吐量（瞬时、平均）：源端和目标端之间的传输速率

瓶颈链路：端到端路径上，限制端到端吞吐的链路

## 1.7 协议层次、服务模型

#### 建模方法

- 模块化(也是一种对功能封装)
- 分层（按照流程考虑、串行封装、协议）

#### 术语

SAP（Service Access Point）：例如 套接字 Socket 区分不同服务用户 处于层间

PDU（Protocol Data Unit）的不同叫法

- 报文
- 报文段
- 分组（IP数据报）
- 帧
- 位

#### Internet 协议栈

应用层：提供网络应用服务 传输层：将端到端细分为进程到进程，不可靠通信变为可靠通信 网络层：为数据报从源到目的选择路由（转发、路由） 链路层：相邻网络节点间的数据传输 物理层：线路上传bit

## 1.8 历史

线路交换 ---> 分组交换

# 第二章 应用层

## 2.1 应用层协议原理

？？？查一下前面章节讲到的一些术语 SDU，主要是封装PDU的那部分

网络应用体系架构

- C/S

    - 服务端扩展性差
- P2P
- 混合

分布式进程通信需要解决的问题：

1. 进程标识和寻址：IP（32位地址） port TCP/UDP
2. 传输层 --- 应用层可利用的服务：源、目的、货物本身（本层SDU）、TCP/UDP
3. 应用层协议

一个端口只运行一个进程？？？

TCP Socket：OS 在应用层、传输层之间建立的 IO句柄（一个整数、会话关系）

UDP Socket：本地 IP、port的二元组

## 2.2 Web、HTTP

http1.0（持久） 1.1（非持久） 流水线方式的持久http

报文格式：

- 请求报文
- 响应报文

## 2.3 FTP

## 2.4 SMTP

## 2.5 DNS

- 命名
- 解析
- 维护

## 2 Transport

**flow control 防止压倒接收方**

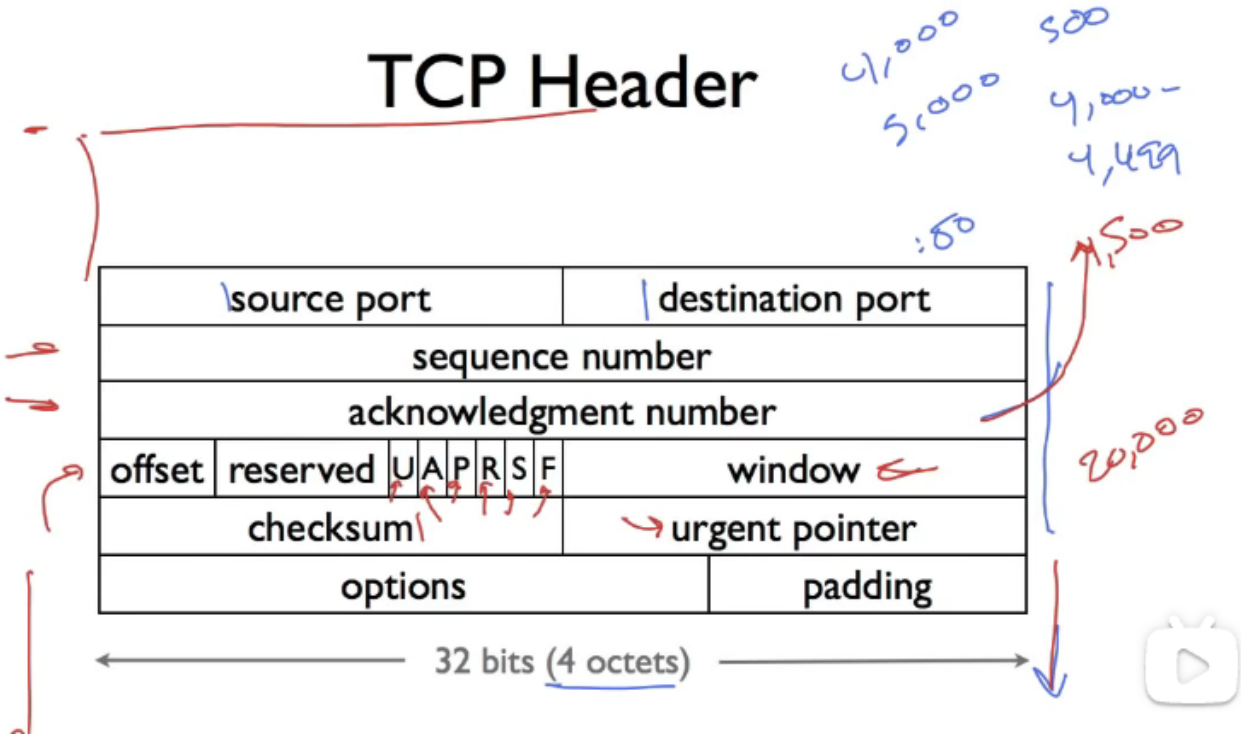
**two basic approaches**

- stop and wait
    - 发送方仅发送一个包，发送之后同步等待
- sliding window 滑动窗口协议
    - sequence number space size 不太明白

**Retransmission Strategies**

- `go back n`
- `selective repeat`

**TCP Header**




**TCP Setup and Teardown**

- 三次握手
- 四次挥手

`end to end principle`

**What i learned**

- Transport Layer： TCP、UDP、ICMP
- How TCP works：Connections and Retransmissions
- How UDP works
- How ICMP works
- The End to End Principle

## 3 packet switching

why?

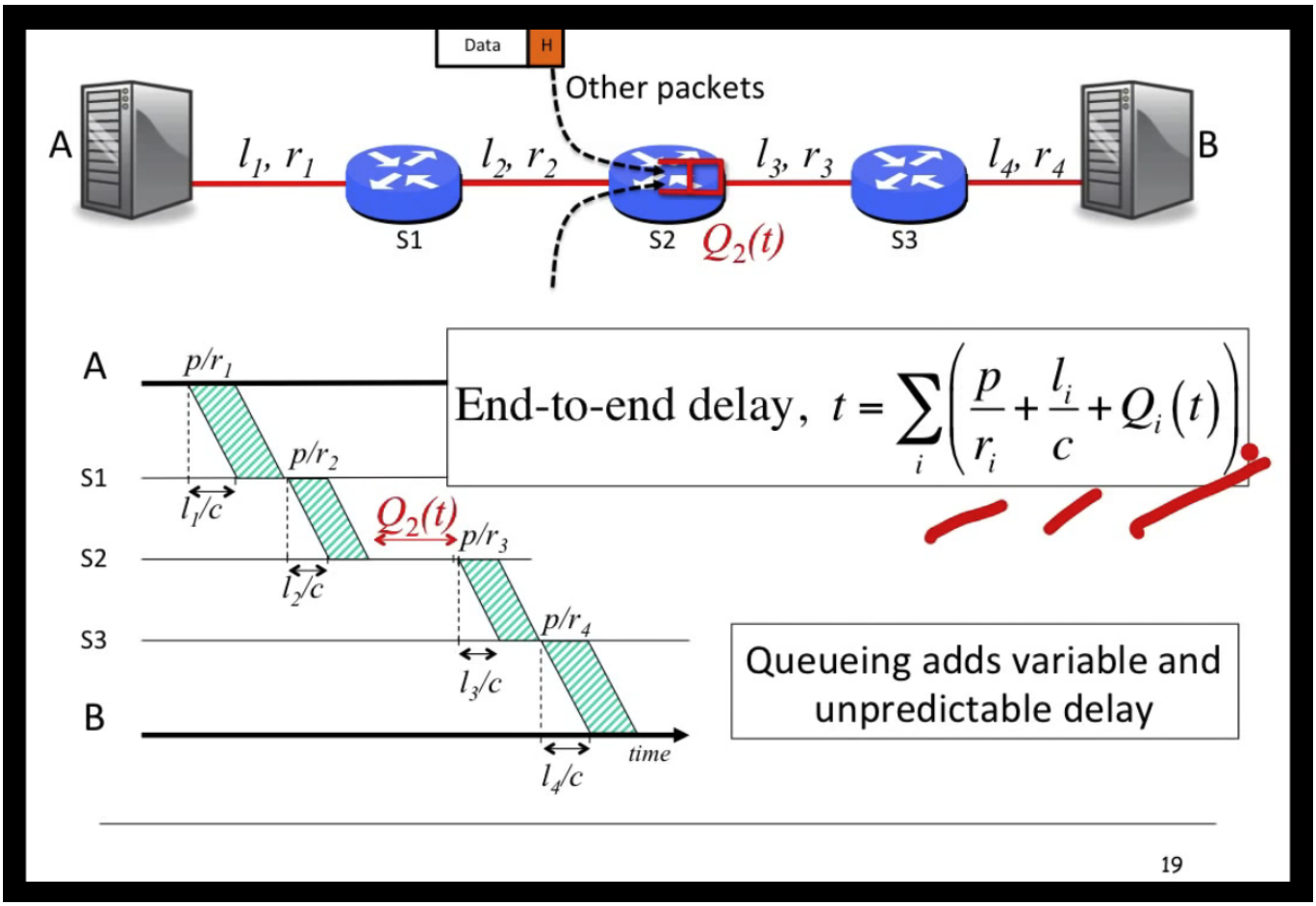
- efficient use of expensive links
- resilience to failure of links and routers

**end to end delay 延迟**

- propagation delay 传播延迟 取决于路程（fixed）
- packetization delay 打包延迟（传输延迟）（fixed）
- Queueing delay 数据包等待延迟（排队延迟）（variable）

路由器存储、转发，等到所有数据包到了，再转发，而且还要等待已经进入缓冲区的其他数据包，有了排队延迟。

一些实时应用使用`playback buffers`来避免可变的排队延迟 比如：b站、youtube视频播放缓冲




统计复用

排队延迟、占用率的举例不太看懂？？？？

**queue properties**

- burstiness increases delay
- Little's result: 占用 = 到达率 * 延迟

**how a packet switch works**

ethernet switch

# Ethernet Switch

1. Examine the header of each arriving frame.
2. If the Ethernet DA is in the forwarding table, forward the frame to the correct output port(s).
3. If the Ethernet DA is not in the table, broadcast the frame to all ports (except the one through which the frame arrived).
4. Entries in the table are learned by examining the Ethernet SA of arriving packets.

internet router

# Internet Router

1. If the Ethernet DA of the arriving frame belongs to the router, accept the frame. Else drop it.
2. Examine the IP version number and length of the datagram.
3. Decrement the TTL, update the IP header checksum.
4. Check to see if TTL == 0.
5. If the IP DA is in the forwarding table, forward to the correct egress port(s) for the next hop.
6. Find the Ethernet DA for the next hop router.
7. Create a new Ethernet frame and send it.

- lookup address
  - 最长前缀匹配（TCAM）
- switching
  - output queueing (minimizes packet delay)（FIFO，最小化延迟）
  - input queueing (with output queues to maximize throughput)（最大化吞吐量）

存储 转发，最小粒度？能够组成IP数据报？

**分组转发：延迟、交换**

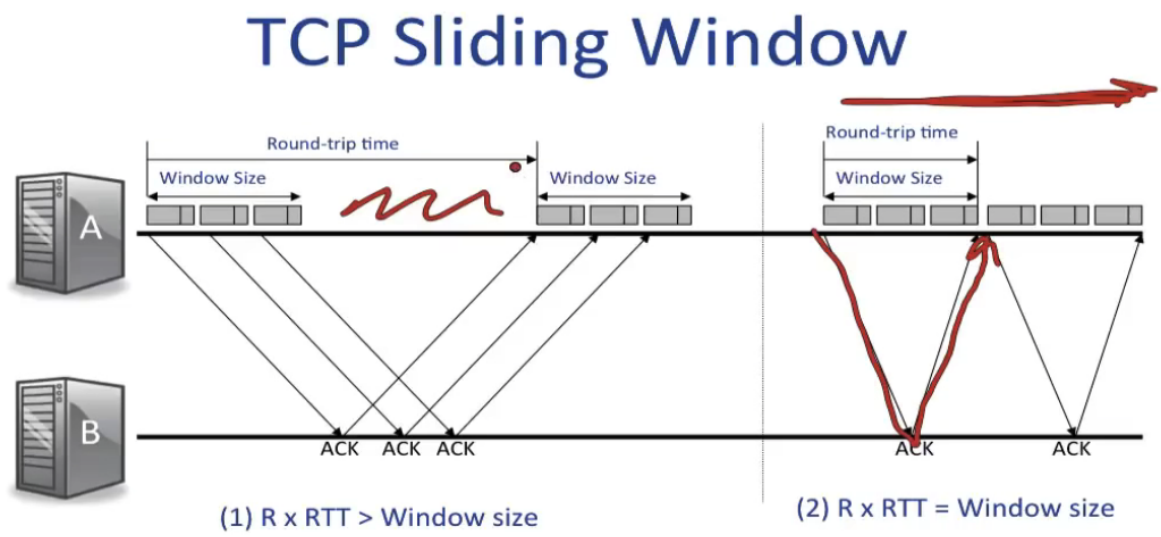
what i learned

- queueing delay and end to end delay
- why streaming applications use a playback buffer
- a simple deterministic queue model
- rate guarantees ???
- delay guarantees ???
- how packets are switched and forwarded

## 4 congestion control 防止压倒网络

`Max-min fairness`

- 实现
  - end host（Where TCP realizes）
  - int the internet
- silding window
  - R * RTT = Window size 发送速度和RTT的关系？？？



# TCP Congestion Control

TCP varies the number of outstanding packets in the network by varying the window size:

Window size = min{Advertised window, Congestion Window}

Advertised window: Receiver

Congestion Window: Transmitter ("cwnd")

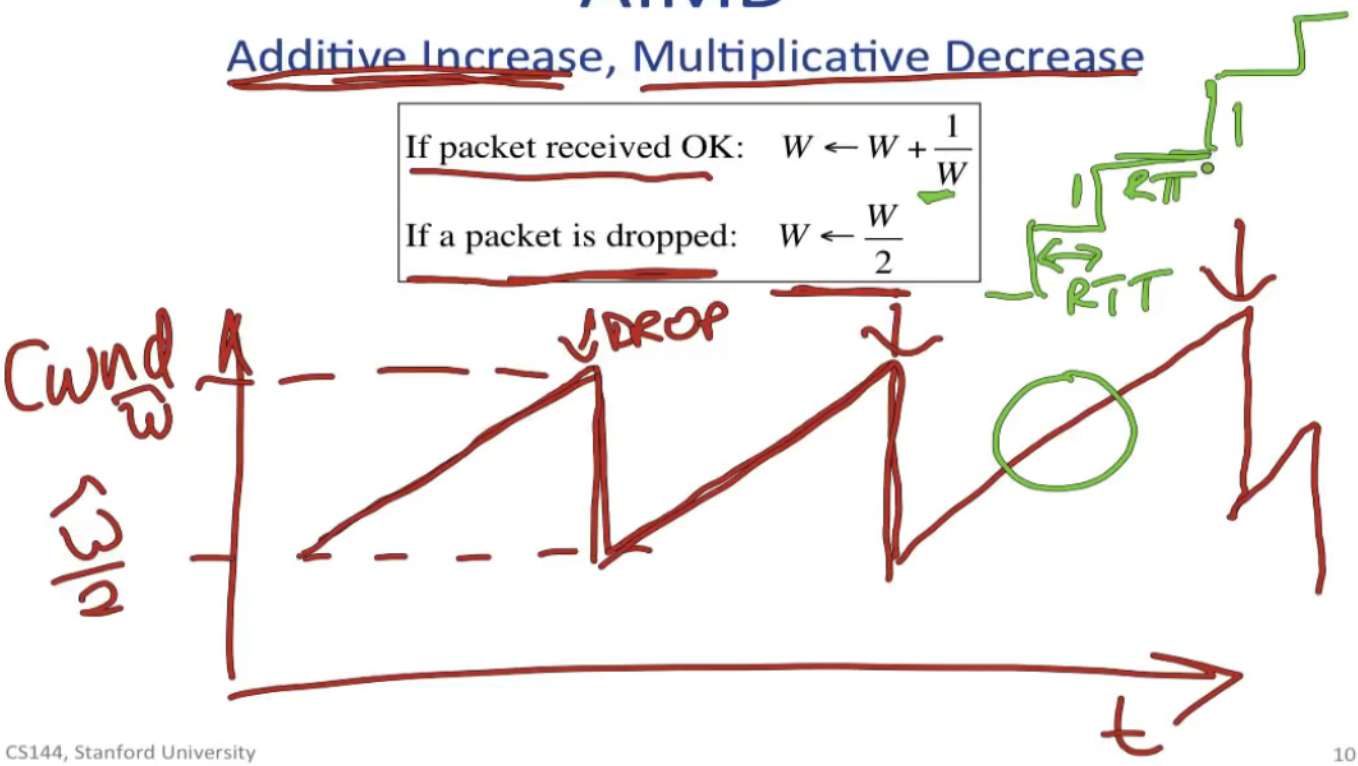
How do we decide the value for cwnd?

TCP Strategy: **AIMD**

## AIMD

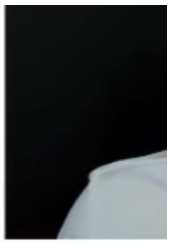
Additive Increase, Multiplicative Decrease

| | |
|---|---|
| If packet received OK: | $W \leftarrow W + \frac{1}{W}$ |
| If a packet is dropped: | $W \leftarrow \frac{W}{2}$ |




CS144, Stanford University 10

# Summary



Choice: In the network, or at the end host?

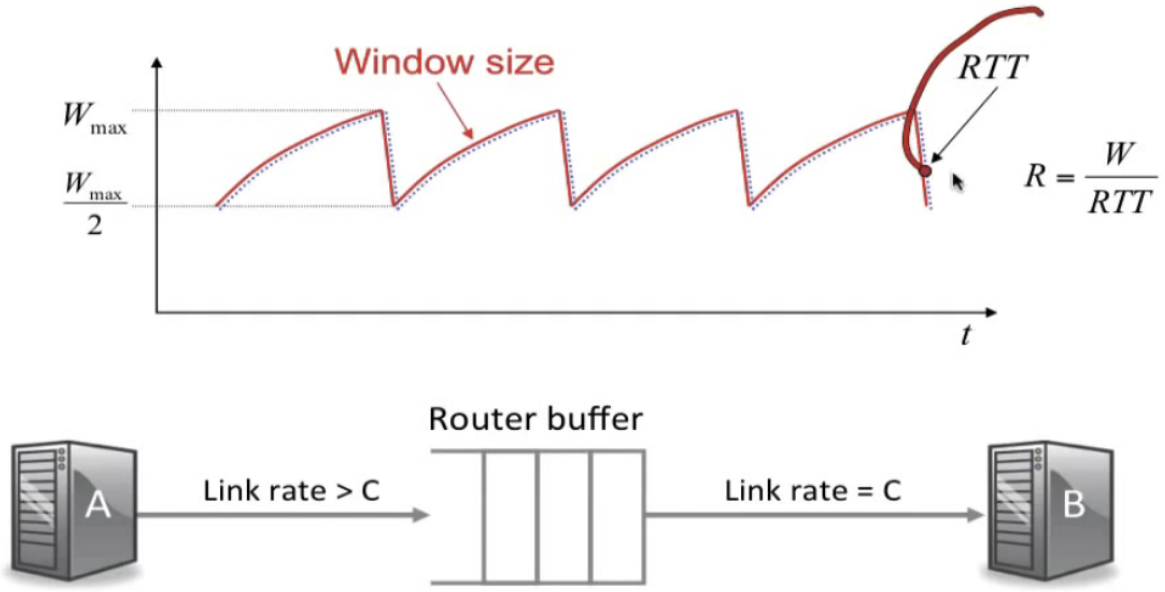
TCP controls congestion from the end-host.

- Reacts to events observable at the end host (e.g. packet loss).
- Exploits TCP's sliding window used for flow control.
- Tries to figure out how many packets it can safely have outstanding in the network at a time.
- Varies window size according to AIMD.

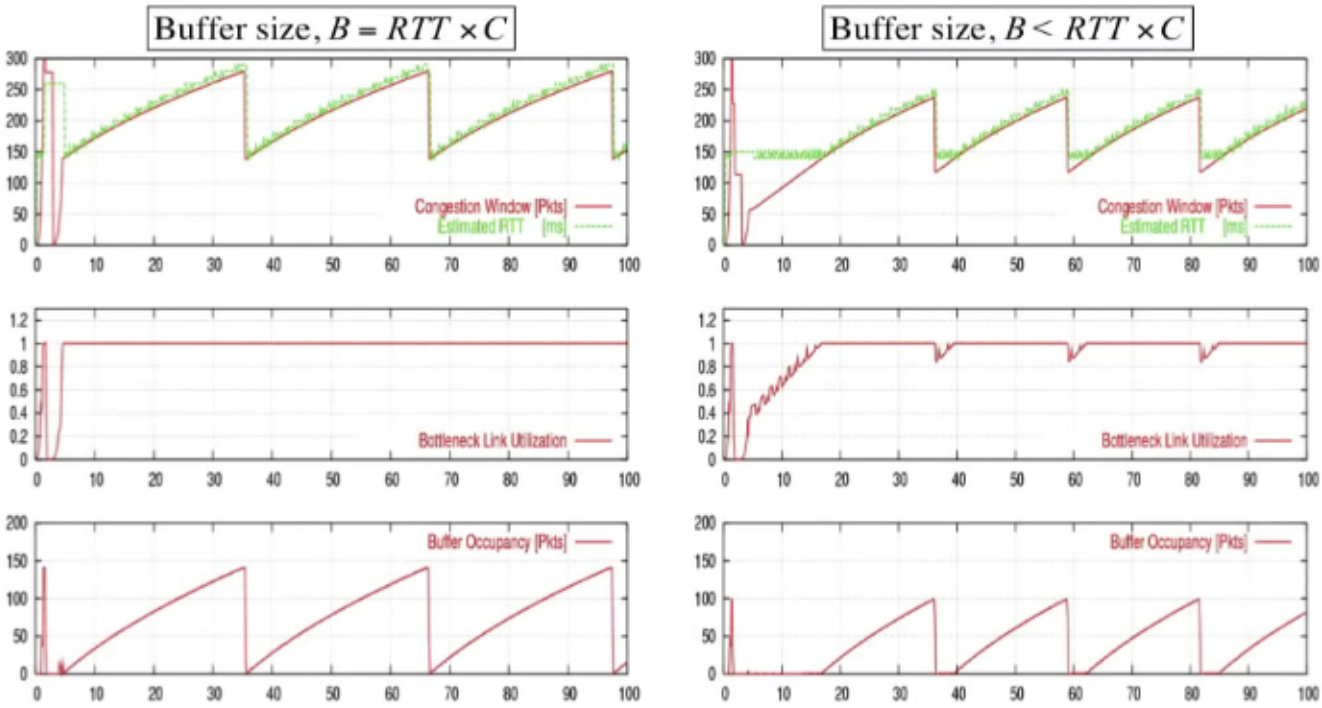
**路由器缓冲区大小**

= RTT * C

# Sending rate for single flow

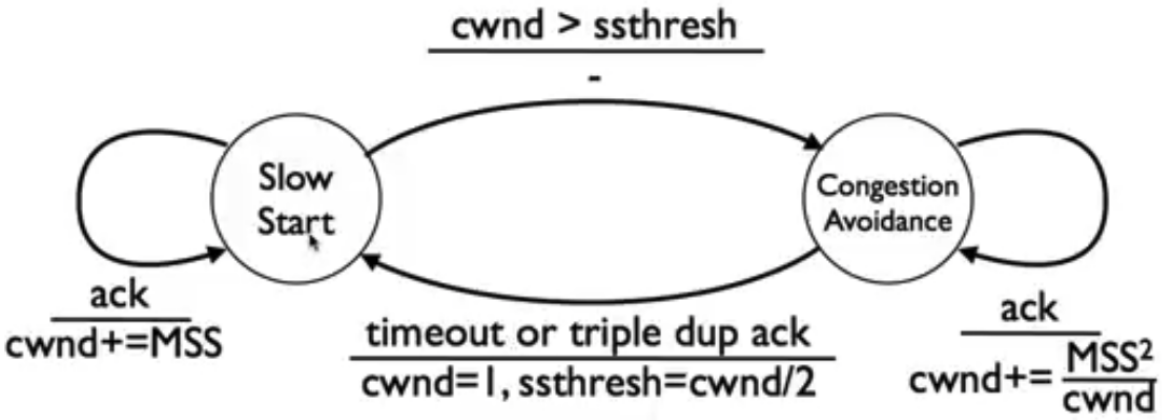


# How big should the buffer be?

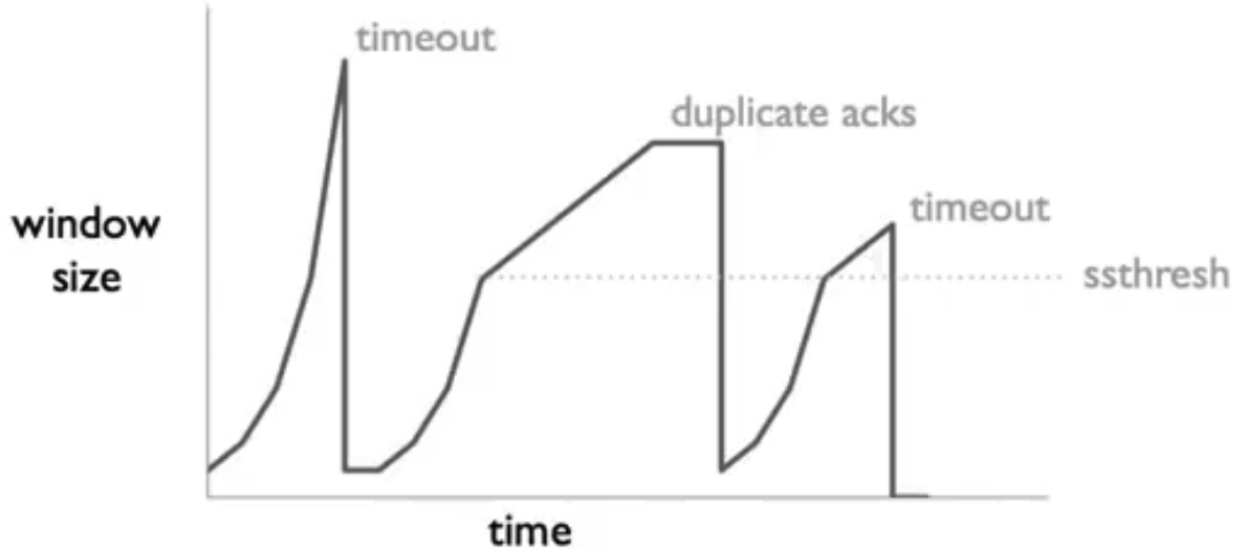



**when should you send new data?** TCP Tahoe

# TCP Tahoe FSM



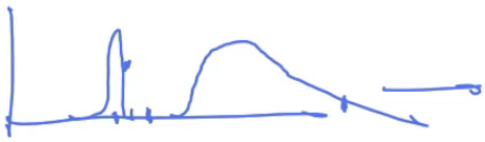
# TCP Tahoe Behavior




**when should you send data retransmissions?** `RTT Estimate`

# TCP Tahoe Timeouts

- r is RTT estimate, initialize to something reasonable
- g is the EWMA gain (e.g., 0.25)
- m is the RTT measurement from most recently acked data packet
- Error in the estimate e = m-r
- $r = r + g \cdot e$
- Measure variance $v = v + g(|e| - v)$
- Timeout = $r + \beta v$ ($\beta$=4)
- Exponentially increase timeout in case of tremendous congestion



**when should you send acknowledgments?** `self-clocking`

# Self-Clocking Principle

- Only put data in when data has left
  - Want to prevent congestion -- too much data in network
- Send new data in response to acknowledgments
- Send acknowledgments aggressively -- important signal

**More Modern TCP:** `TCP Reno`

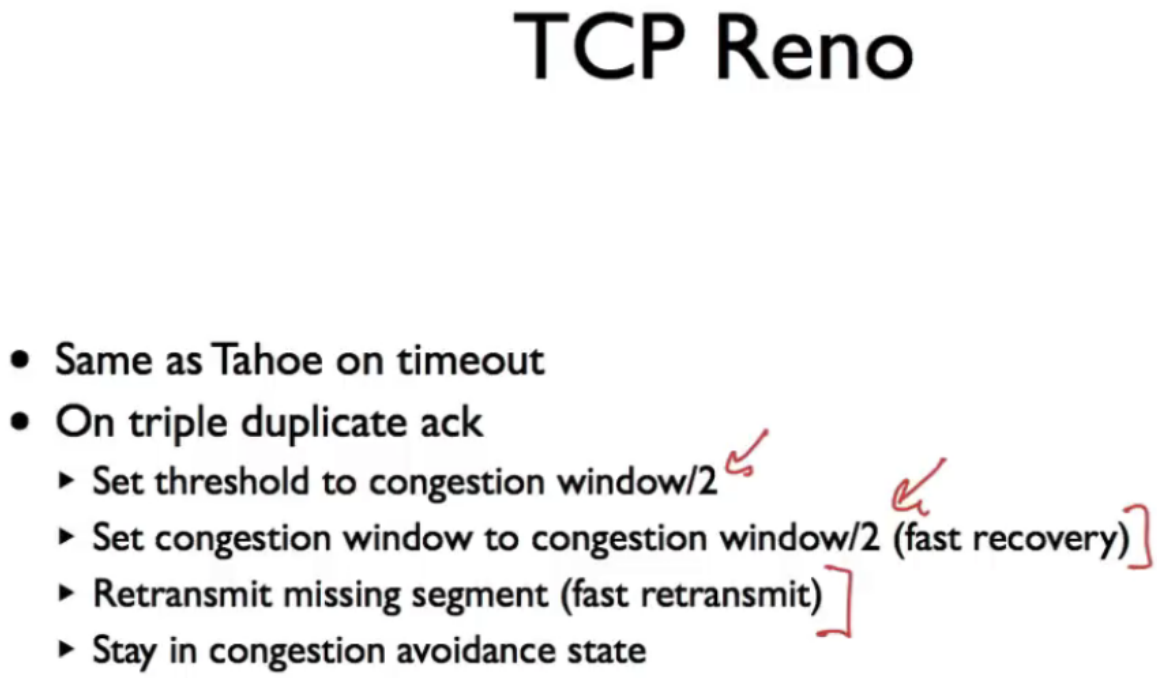
对 `timeout` 和 `triple same ack` 做不同处理

- 快恢复
- 快重传
- Tahoe

# TCP Tahoe

- On timeout or triple duplicate ack (implies lost packet)
  - Set threshold to congestion window/2
  - Set congestion window to 1
  - Enter slow start state

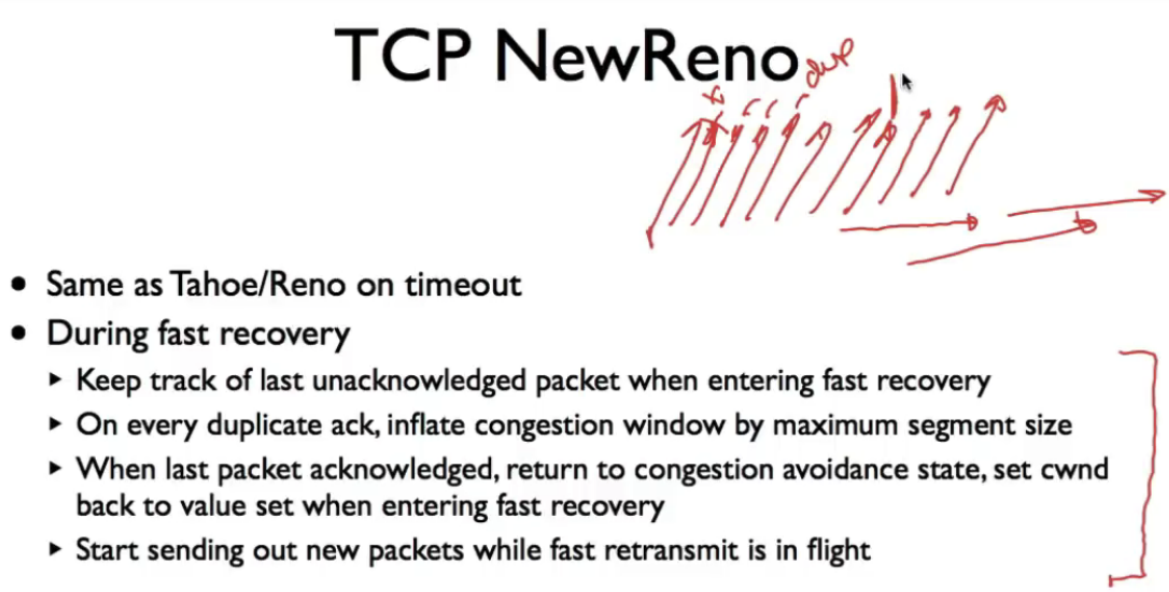
- Reno




**TCP NewReno**

在快速重传中发送新的数据包

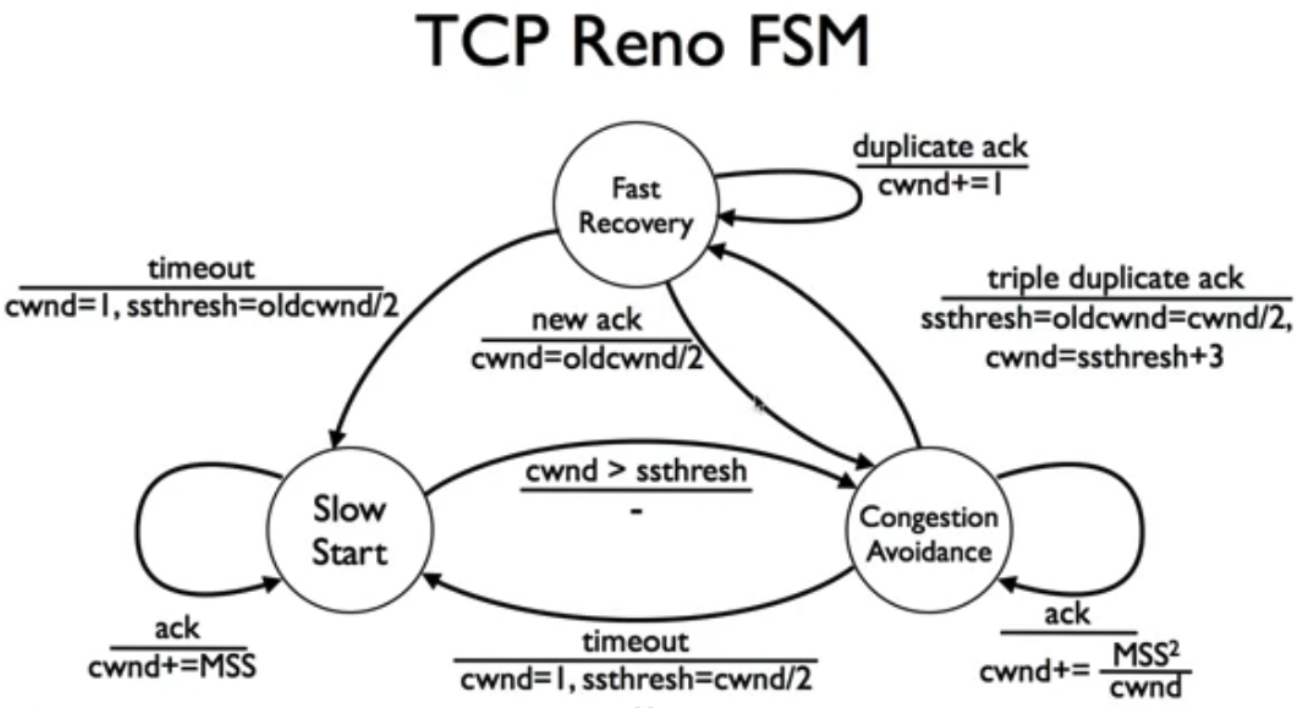
- New Reno



# Congestion Control

- One of the hardest problems in robust networked systems
- Basic approach: additive increase, multiplicative decrease
- Tricks to keep pipe full, improve throughput
  - Fast retransmit (don't wait for timeout to send lost data)
  - Congestion window inflation (don't wait an RTT before sending more data)

**性能提升：**`fast retransmit`、`fast recovery`




AIMD flow, max-min fairness ???

## application and NAT

**NAT**

Strong end to end principle

NAT ip port 到 local machine ip port 的映射

Types of NATs

- Full Cone NAT (NAT IP、NAT Port)

- Restricted Cone NAT (NAT IP、NAT Port、External Sender IP)
- Port Restricted NAT (NAT IP、NAT Port、External Sender IP、External Sender Port)
- Symmetric NAT (Symmetric Mapping 内网同一个endpoint向外网不同的endpoint12建立链接，需要建立不同的映射关系)
- Hairpining

**分类**

- Mapping Behavior (出):
    - Endpoint Independent Mapping
    - Address Dependent Mapping
    - Address and Port Dependent Mapping
- Filtering Behavior (进):
    - Endpoint Independent Filtering
    - Address Dependent Filtering
    - Address and Port Dependent Filtering

NAT 打洞，peer to peer 通信

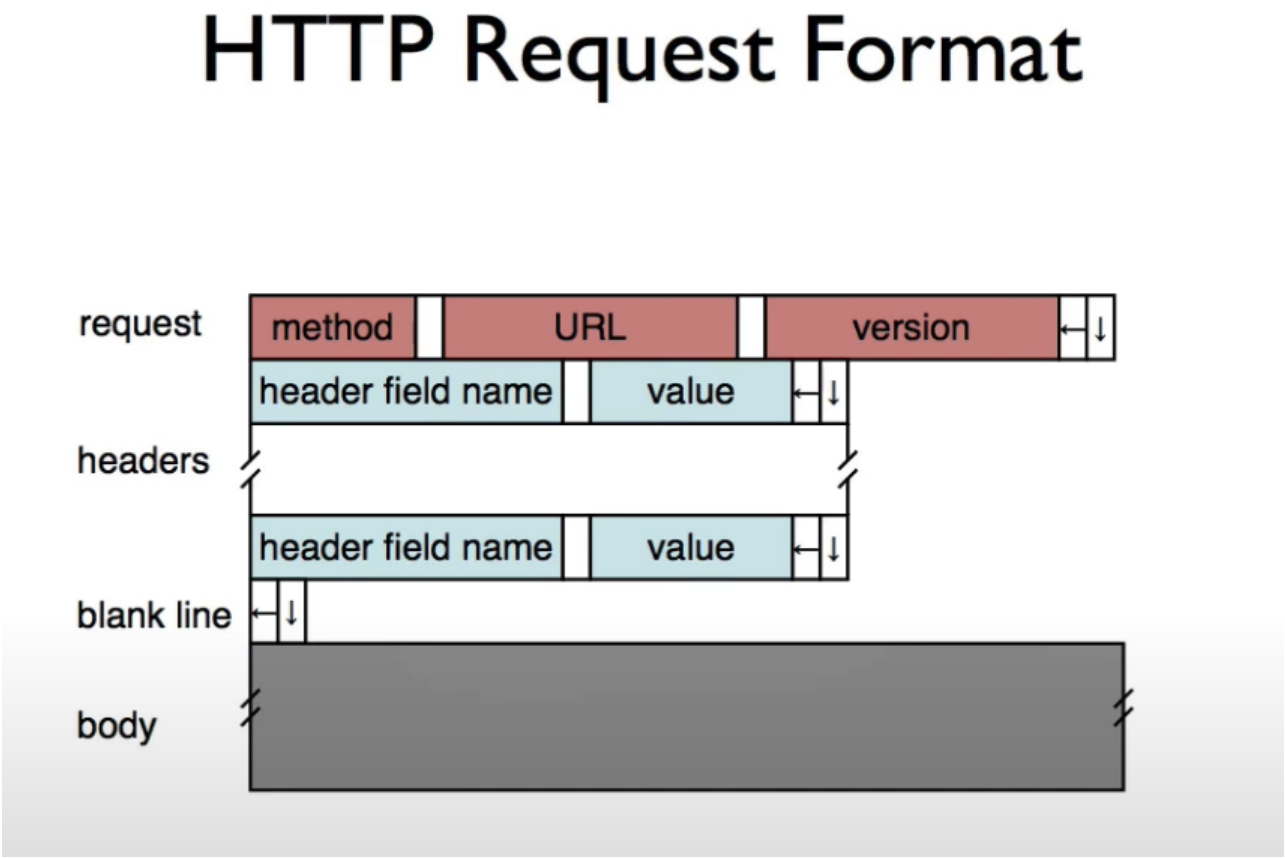
Full Cone NAT、Restricted Cone NAT、Port Restricted NAT（三者都是Endpoint Independent Mapping，Filtering 策略不同） 都可以实现 NAT 打洞

Symmetric NAT (X Dependent Mapping, Port Restricted Filtering) 打洞需要用到"试探"

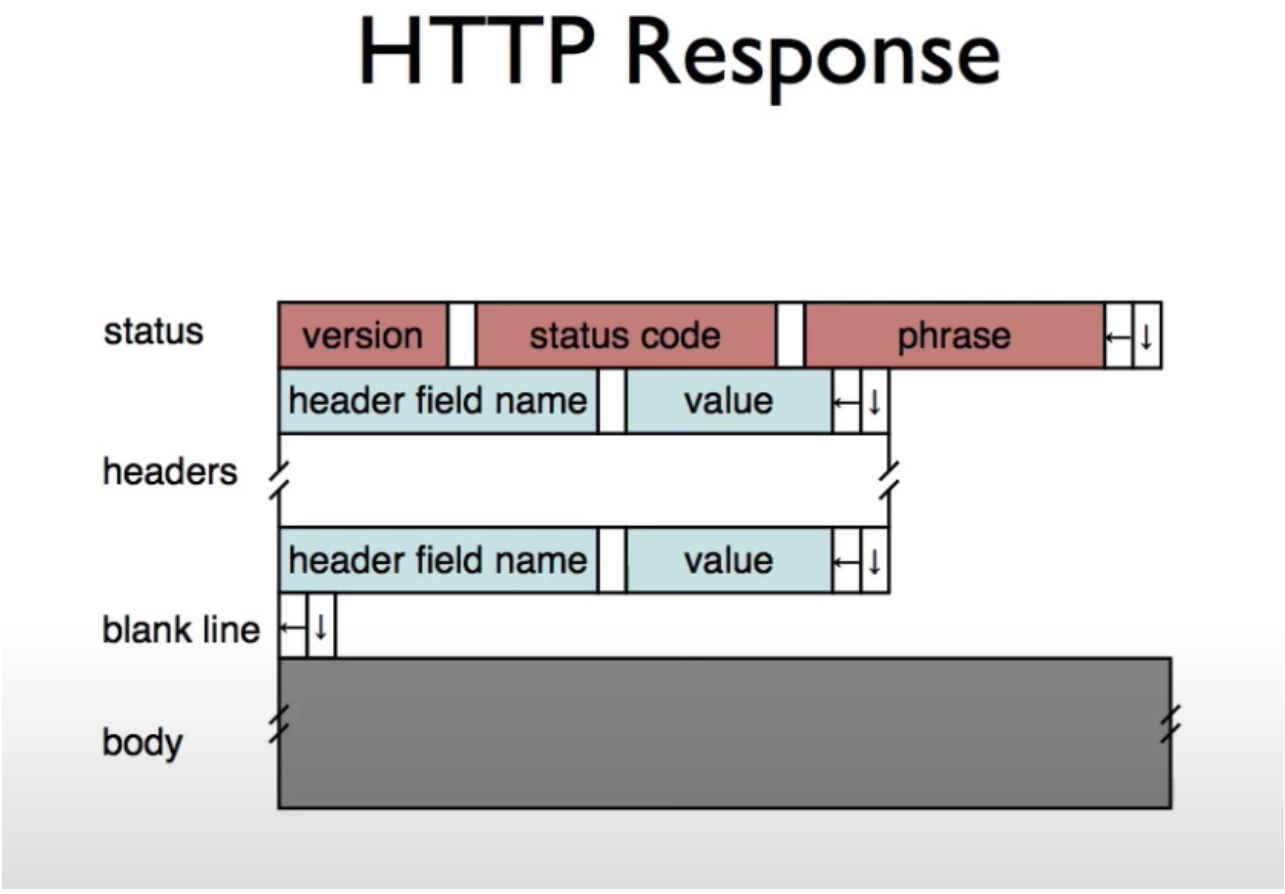
点击查看具体分析

**HTTP**

- Request Format




- Response Format



http1.0

# HTTP/1.0

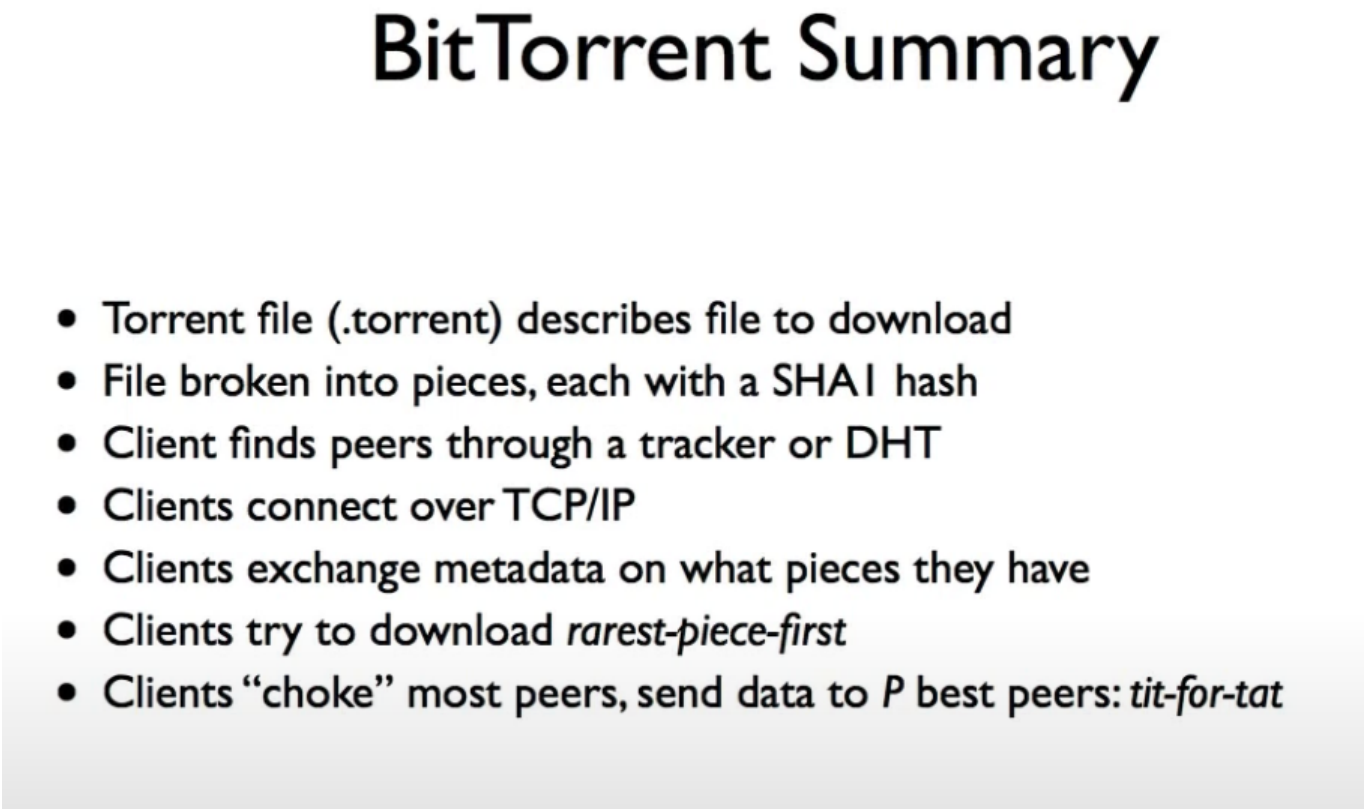
- Open connection
- Issue GET
- Server closes connection after response
- *Opening many connections is slow*
- *Many transfers are small, doesn't let TCP window grow*

http1.1

# HTTP/1.1

- Added Connection header for requests
  - keep-alive: tells the server "please keep this connection open, I'll request more"
  - close: tells the server to close the connection
  - Server can always ignore
- Added Connection header for responses
  - keep-alive: tells the client it'll keep the connection open
  - close: tells the client it's closing the connection
- Added Keep-Alive header for responses
  - Tells client how long the connection may be kept open

**Bit Torrent**




**DNS**

**DHCP**

need：

- ip address
- subnet mask
- gateway router
- dns server's ip address

## 6 Routing

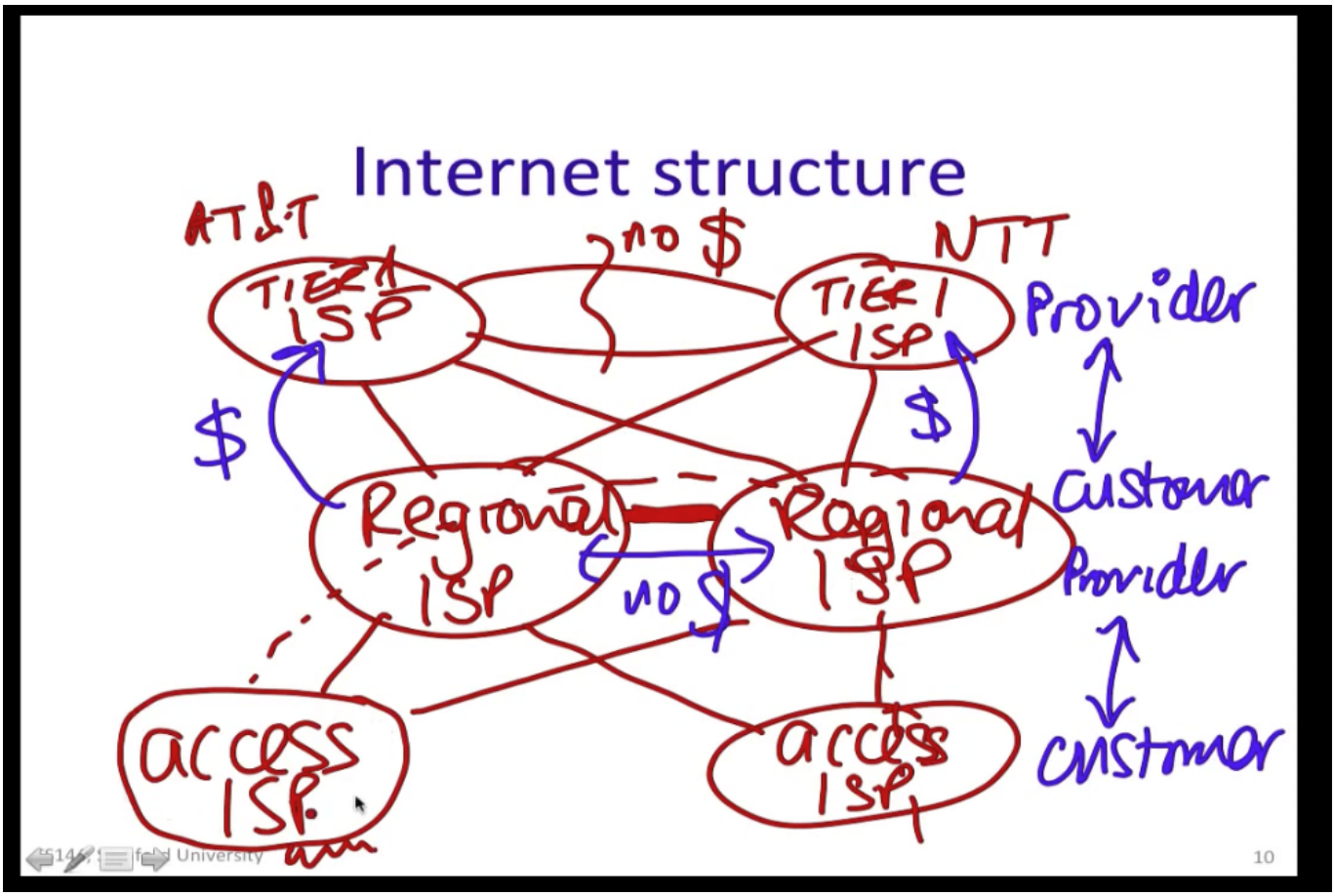
Approaches

- `Flooding`
- `Source Routing` 规定好整条路径 运营商必须开放内部节点信息 也不利于网络自己维护路由表
- `Forwarding Table`
  - `Bellman-Ford distance vector` used by RIP
  - `Dijkstra link state` used by OSPF
- `Spanning Tree` for Ethenet Switch

minimum cost spanning tree

interior routing

- Bellmanford（replaced by the latter）
- dijkstra




**BGP Border Gateway Protocol**

- uses "Path Vector"
- gives AS_PATH( that is the path vector )

# Border Gateway Protocol (BGP-4)

## *Basics*

BGP is not a link-state or distance-vector routing protocol.
- Instead, BGP uses what is called a "Path vector"

BGP routers advertise complete paths (a list of AS's).
- Also called AS_PATH (this is the path vector)
- Example of path advertisement:

"The network 171.64/16 can be reached via the path {AS1, AS5, AS13}"

Paths with loops are detected locally and ignored.
Local policies pick the preferred path among options.
When a link/router fails, the path is "withdrawn".

# BGP Messages

**Open** : Establish a BGP session.

**Keep Alive** : Handshake at regular intervals.

**Notification** : Shuts down a peering session.

**Update** : Announcing new routes or withdrawing previously announced routes.

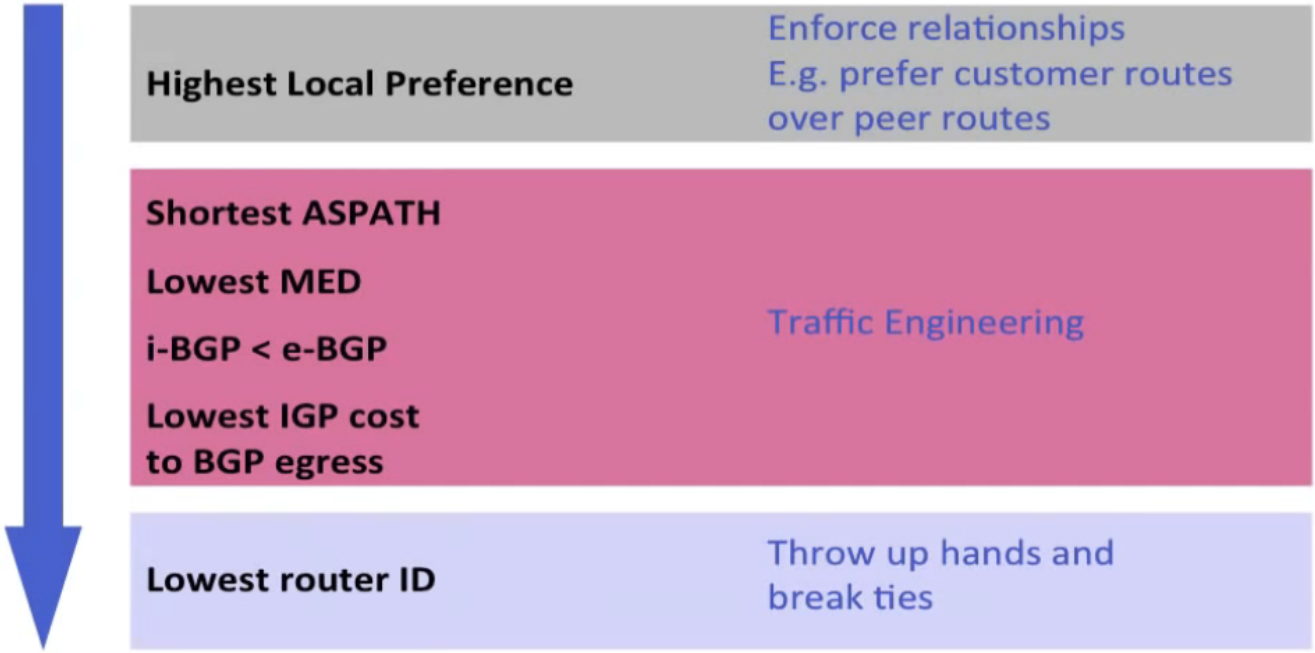
BGP announcement = prefix + path attributes

Path attributes

Include: next hop, AS Path, local preference, Multi-exit discriminator, ...

Used to select among multiple options for paths.

# BGP Route Selection Summary



# Summary

All AS's in the Internet must connect using BGP-4.

BGP-4 is a path vector algorithm, allowing loops to be detected easily.

BGP-4 has a rich andcomplex interface to let AS's choose a local, private policy.

Each AS decides a local policy for traffic engineering, security and any private preferences.

**Multicast Routing**

Broadcast Reverse Path Broadcast( RPB )

**Spanning Tree Protocol For Ethenet**

# Preventing loops

*Spanning Tree Protocol*

The topology of switches is a graph.

The Spanning Tree Protocol finds a a subgraph that spans all the vertices without loops.

- Spanning: all switches are included.
- Tree: no loops.

The distributed protocol decides:

1. Which switch is the Root of the tree, and
2. Which ports are allowed to forward packets along the tree.

# How it works

1. Periodically, all switches broadcast a "Bridge Protocol Data Unit" (BPDU) (ID of sender, ID of root, distance from sender to root).
2. Initially, every switch claims to be Root: sets distance field to 0.
3. Every switch broadcasts until it hears a "better" message:
   - A root with a smaller ID
   - A root with equal ID, but with shorter distance
   - Ties broken by smaller ID of sender.
4. If a switch hears a better message, retransmit message (add 1 to distance).

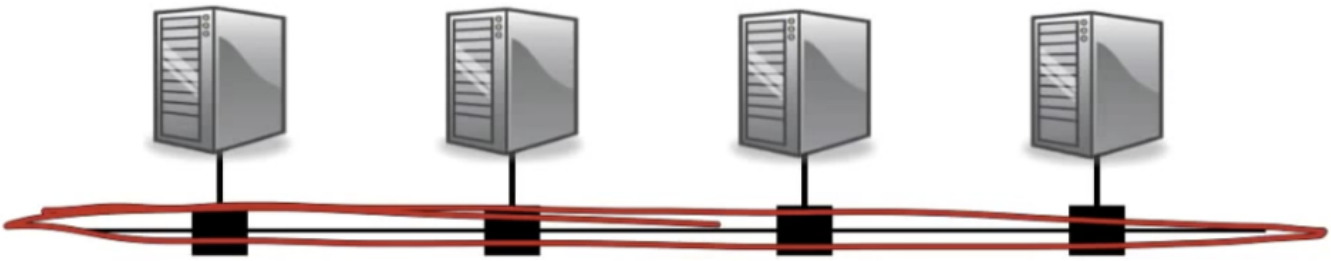
Root port: The port on a switch that is closest to the Root.
Designated port: The port neighbors agree to use to reach the Root.
All other ports are blocked from forwarding (but still send/receive BPDUs).

Eventually:
- Only the root originates configuration messages (others retransmit them).
- Locally, switch only forwards on ports.

7 Physical and Link Principles

link layer CSMA/CD Protocol

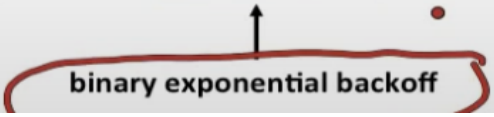
# CSMA/CD Protocol



All hosts transmit & receive on one channel
Packets are of variable size.

When a host has a packet to transmit:

1. Carrier Sense: Check the line is quiet before transmitting.
2. Collision Detection: Detect collision as soon as possible. If a collision is detected, stop transmitting; wait a random time, then return to step 1.




**Ethenet**

# Ethernet Switch

## Forwarding and Learning:

1. Examine the header of each arriving frame.
2. If the Ethernet DA is in the forwarding table, forward the frame to the correct output port(s).
3. If the Ethernet DA is not in the table, broadcast the frame to all ports (except the one through which the frame arrived).
4. Entries in the table are learned by examining the Ethernet SA of arriving packets.

## Topology:

Run Spanning Tree Protocol (STP) to create loop free topology.

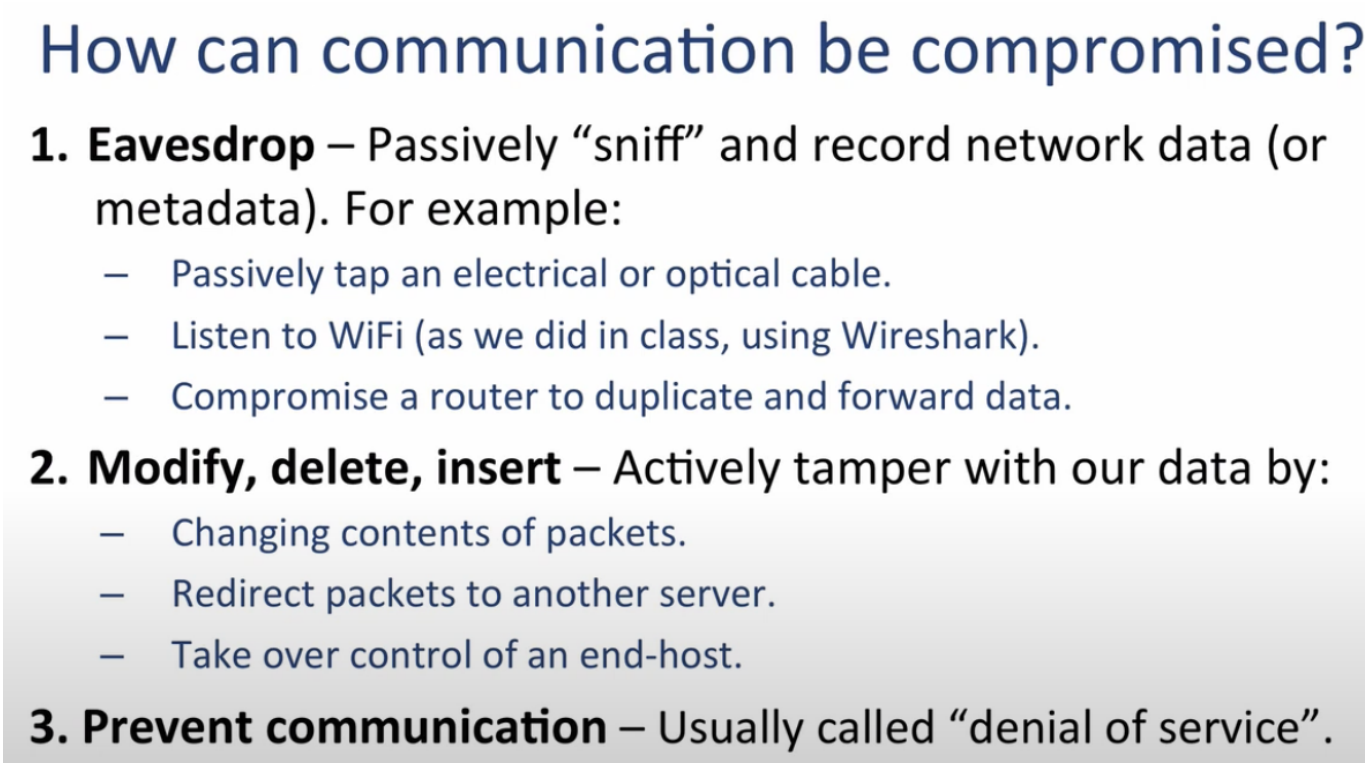
**Fragment and Assembly**

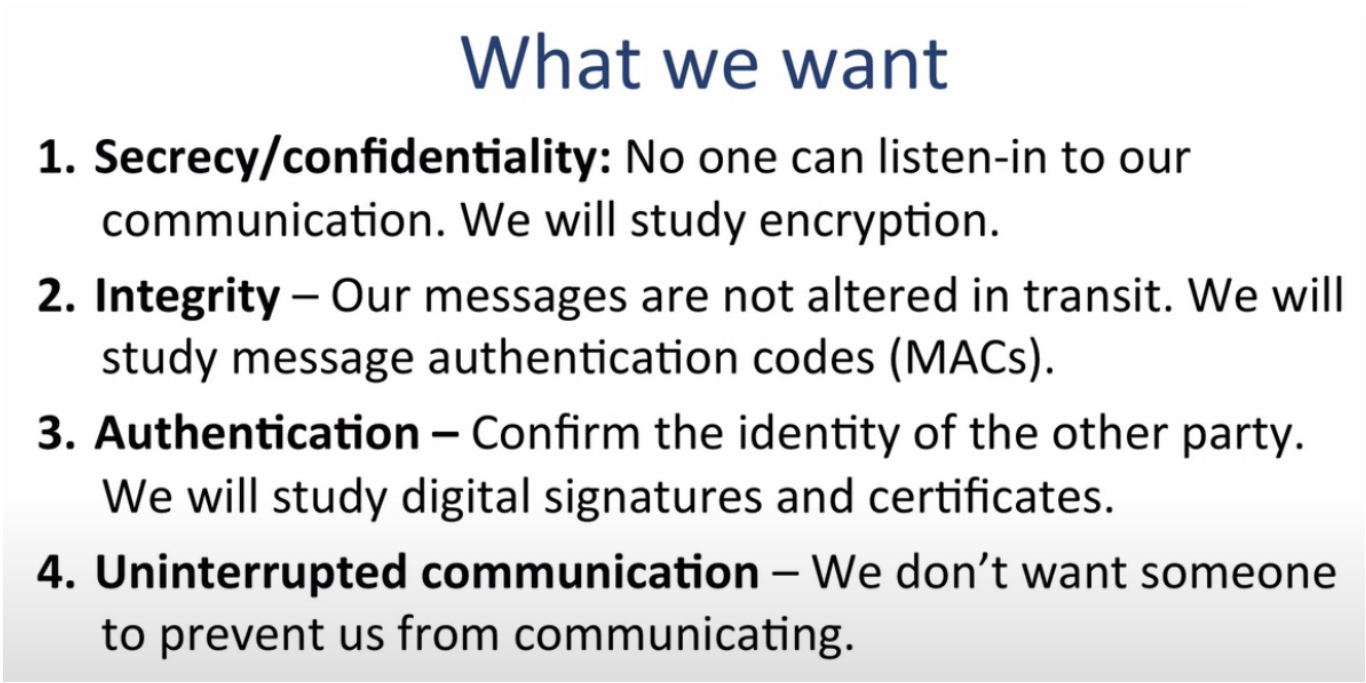
# Fragmentation and Assembly

- Problem occurs when higher layer's data unit is too large for lower layer
- Fragmentation: taking a large data unit and breaking it into smaller chunks
- Assembly: combining chunks into original data unit
- Examples:
  - Transport: TCP takes stream of bytes and breaks into TCP segments
  - Network: IP takes packets too big for a link and breaks them up into IP fragments
  - Link: 6lowpan takes IPv6 packets and breaks them into link fragments if needed

usual mtu 1500 byte

8 security



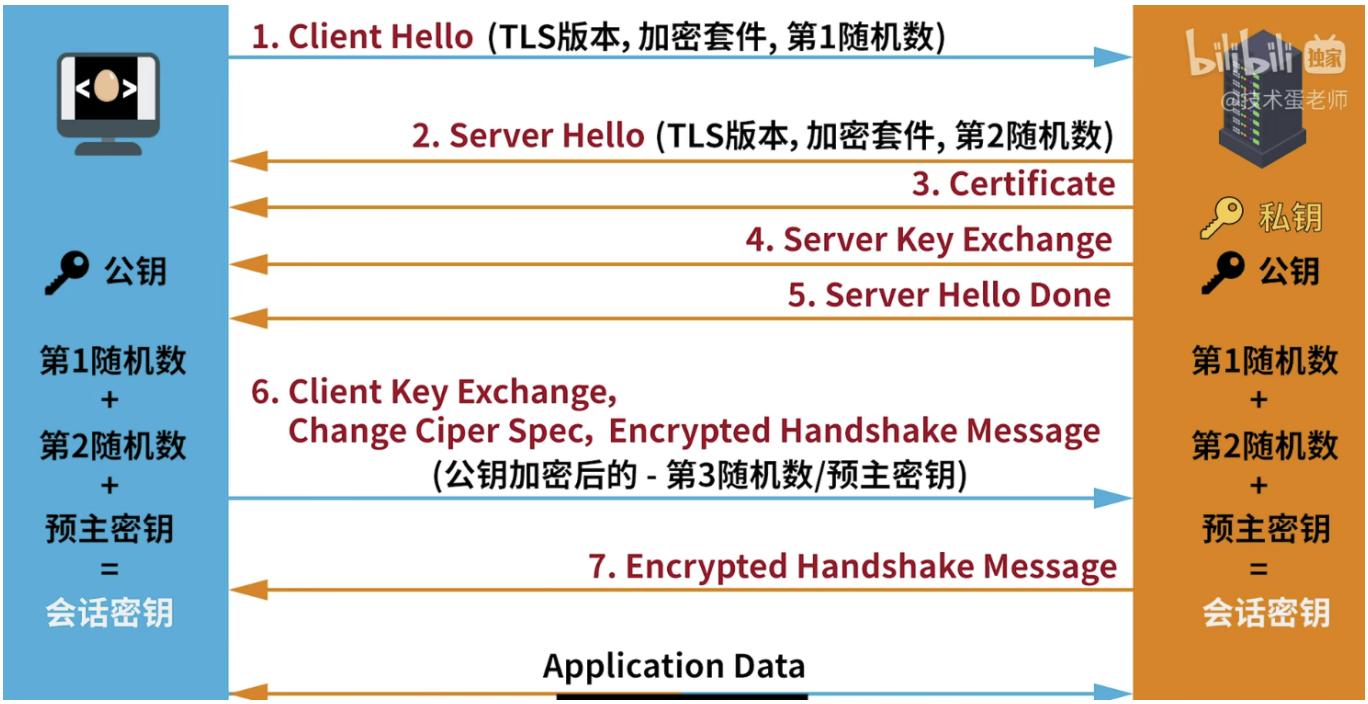

三大特性

- confidentiality、integrity、authenticity

**Transport Layer Security (TLS)**

RFC5246:you use it every day

**TLS1.2**


1. Client Hello (TLS版本, 加密套件, 第1随机数)
2. Server Hello (TLS版本, 加密套件, 第2随机数)
3. Certificate
4. Server Key Exchange
5. Server Hello Done
公钥
私钥
公钥
第1随机数
+
第2随机数
+
预主密钥
=
会话密钥
6. Client Key Exchange,
Change Ciper Spec, Encrypted Handshake Message
(公钥加密后的 - 第3随机数/预主密钥)
7. Encrypted Handshake Message
Application Data
第1随机数
+
第2随机数
+
预主密钥
=
会话密钥
@技术蛋老师