计算机网络期中复习纲要

第一章 计算机网络和因特网

1、网络的构成：资源子网，通信子网
2、基本概念：服务和协议，可靠数据传输（reliable data delivery）和尽力交付（best-effort）
3、TCP 和 UDP 的特征，运行 TCP 和 UDP 的程序和应用层协议
4、网络边缘（network edges）：接入技术，传输介质，C/S 结构和 P2P 结构
5、网络核心（network core）：两种交换方式，电路交换当中复用的类型，报文交换当中的存储转发模式
6、四种延时，流量强度（traffic intensity），丢包（packet loss），吞吐量（throughput）
7、计算机网络的层次结构，每一层的功能，报文的封装（encapsulation）和解封装（decapsulation）

第二章 应用层

1、应用的三种结构，socket，IP 地址
2、应用层依靠的下层（传输层）提供的服务
3、HTTP 协议和 web 服务：
   1）默认采用 80 端口，传输层采用 TCP 协议，无状态（stateless）协议
   2）请求（request）报文和响应（response）报文
   3）持续（persistent）HTTP（包括带流水线 pipelining 和不带流水线）和非持续（nonpersistent）HTTP
   4）RTT（round trip time），连接的建立与释放
   5）请求报文和响应报文头部格式，请求报文的方法（method），响应报文的状态码（status code）
   6）cookie，网页缓存 web cache（代理服务器 proxy server），条件 GET
4、FTP 协议：**控制连接 21 端口**，数据连接 20 端口，传输层 TCP 协议，控制信息带外（out of band）传送，有状态协议，数据非持久连接
5、E-mail 服务，SMTP，POP3，IMAP 协议
   1）用户代理（user agent）和邮件服务器（mail server）
   2）邮件服务器间协议：SMTP；25 端口；TCP 协议；ASCII 码表示；直接发送无中转；持续连接；推协议
   3）邮件访问协议：POP3 协议（110 端口）及相关命令，IMAP 协议，基于 web 的电子邮件
6、DNS 服务：53 端口，UDP 协议，分布式设计
   1）DNS 提供的服务
   2）DNS 服务器的层次结构
   3）本地 DNS，递归查询（recursive query）和迭代查询（iterative query），DNS 缓存
   4）DNS 资源记录（resource record），插入资源记录

第三章 传输层

1、服务模型
   1）IP 的服务模型
   2）运输层的服务
   3）TCP 的附加服务
2、复用与分用的实现，socket 的标识
3、流行的因特网应用及其采用的运输层协议
4、UDP
   1）UDP 所具有的传输层服务
   2）UDP 的优点
   3）UDP 报文段结构
   4）UDP 检验和（checksum）
5、可靠数据传输（reliable data transfer）的原理（注意：以下原理仅是为了解释 TCP 协议方便，不代表在 TCP 协议中完整使用了以下原理）
   1）可靠数据传输协议 rdt

2）停等（stop-and-wait）协议 VS 流水线（pipelining）协议
3）解决流水线差错：后退 N 步（go-back-N）协议
4）避免大量重传：选择重传（selective repeat）
5）总结：保证可靠数据传输的技术

6、TCP
1）面向连接（connection-oriented），全双工（full duplex），点对点（point-to-point）
2）TCP 报文段结构
3）顺序号和确认号
4）RTT 的估计（estimate）与超时的确定
5）加倍超时间隔，3 个冗余 ACK 导致的快速重传（fast retransmit）
6）TCP 解决差错的措施
7）流量控制（flow control）
8）TCP 连接的建立（三次握手 three-way handshaking）与关闭

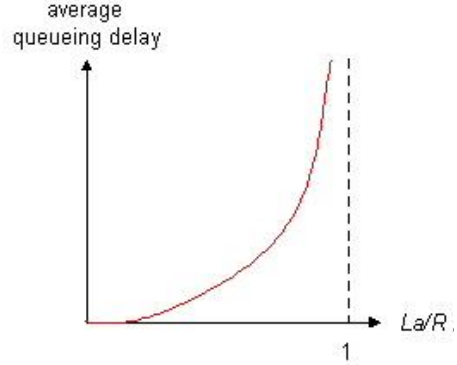
7、拥塞控制
1）AIMD（加性增乘性减）
2）慢启动（slow start）
3）TCP 的拥塞控制方法，阈值（threshold）

8、TCP 公平性

复习重点内容

Chapter 1 Computer Networks and the Internet

1. 什么是协议：A protocol defines the format and the order of messages exchanged between two or more communicating entities, as well as the actions taken on the transmission and/or receipt of a message or other event.一个协议定义了在两个或多个通信实体之间交换的报文格式和次序，以及在报文传输和/或接收或其他事件方面所采取的动作。
2. 使用 TCP 的服务和协议：web 服务（HTTP），文件传输服务（FTP），远程登录服务（telnet），电子邮件服务（SMTP，POP3，IMAP）
使用 UDP 的服务：流媒体（streaming media），远程会议（teleconferencing），域名解析（DNS），网络电话（Internet telephony）
3. 接入技术 住宅（home/residental）接入：拨号，DSL，HFC
公司（company/enterprise）接入：LAN 组网
无线（wireless）接入：WiFi，3G，WiMAX
传输介质：双绞线（twisted pair），同轴电缆（coaxial cable），光纤（fiber optics/optical fiber），无线信道（wireless channel）
4. 交换的两种类型：电路交换（circuit switch）分组交换（packet switch）。
交换当中涉及的问题：电路交换—复用（multiplexing） 分组交换—报文交换，延时，丢包，吞吐量
5. 复用的类型：频分多路复用（FDM）：按频率划分，时分多路复用（TDM）：划分时隙（slot），统计复用（Statistical Multiplexing）
6、存储转发（store-and-forward）：先将到达的比特存储，待同一个分组当中的所有比特都到达之后再进行下一跳转发。
7. 四种延时：节点处理延时（nodal processing delay），排队延时（queueing delay），传输延时（transmission delay），传播延时（propagation delay）。其中，传输延时指分组从设备发送到链路上所经历的时间，等于分组大小/网络带宽。传播延时等于分组从物理链路一端发送到另一端所经历的时间，等于链路长度/传播速度。排队延时常常通过流量强度进行估计。
流量强度（traffic intensity）=La/R



说明：内容仅供参考。

R=链路带宽 L=分组大小 a=平均分组到达速率

8. ISO/OSI 七层模型：物理层（physical layer），数据链路层（data link layer），网络层（network layer），传输层（transport layer），会话层（session layer），表示层（presentation layer），应用层（application layer）

各层的分组名称：应用层-消息 message 传输层-报文段 segment 网络层-数据报 datagram 数据链路层-帧 frame 物理层-原始比特流 raw bits stream 统称：packet

9. 各层的功能：传输层-进程间的逻辑通信 网络层-主机间的逻辑通信 数据链路层-相邻结点间的逻辑通信

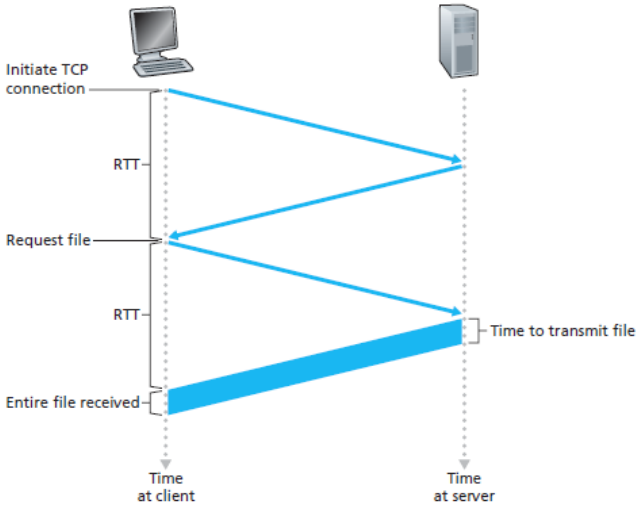
Chapter 2 Application Layer

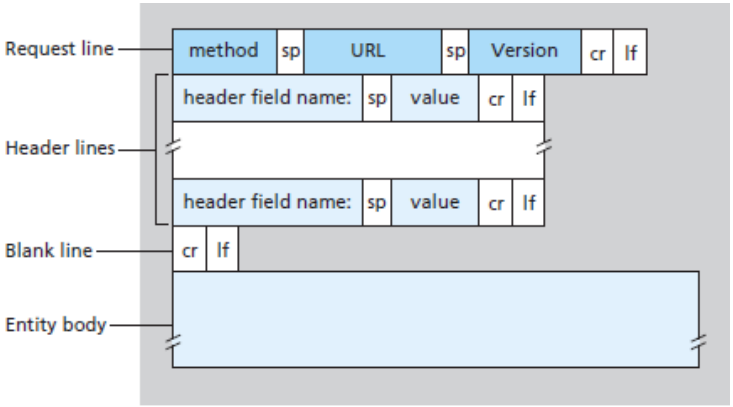
1. 应用的三种结构：客户机-服务器结构（client-server），对等结构（peer-to-peer），混合结构（hybrid of C/S and P2P）
2. 发起通信的一般是客户机，等待通信的一般是服务器。服务器长期在线等待通信。
3. Socket：主机进程与网络之间的接口。IP 地址：32 位二进制数，标识主机接口。
4. TCP 提供的服务：面向连接（connection-oriented）可靠数据传输（reliable transport），差错检测（check），分用（demultiplexing）和复用（multiplexing），流量控制（flow control），拥塞控制（congestion control）

   UDP 提供的服务：无连接（connectionless）数据传输，差错检测，分用和复用

   二者都不提供的服务：最小带宽保证（minimum bandwidth guarantee），最小延时保证（guarantee of the transmission time）
5. 网页（web page），对象（object），HTML 文件，URL。标准的 URL 格式为<协议>://<主机名>:<端口>/<路径>
6. HTTP 协议在服务器端默认采用 80 端口，传输层采用 TCP 协议。
7. HTTP 是无状态（stateless）协议，不保存客户状态信息。
8. 客户端（client）发送请求（request）报文，服务器（server）返回响应（response）报文。
9. 非持续（nonpersistent）HTTP：一个连接最多只能发送一个对象；持续（persistent）HTTP：一个连接可发送多个对象。不带流水线的持续 HTTP：收到前一个响应才能发出下一个请求。带流水线（pipelining）的持续 HTTP：收到响应报文之前就可以继续发送新的请求报文。HTTP/1.0 使用非持续 HTTP；HTTP/1.1 使用带流水线的持续 HTTP。
10. 往返时延 RTT（round trip time）。非持续连接中传输一个对象的时间为 2RTT+传输文件所需的时间（如图）。
11. HTTP 请求报文头部格式

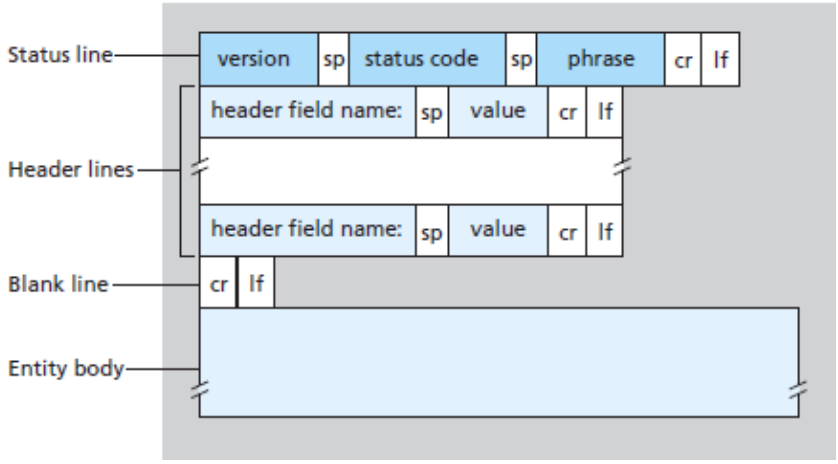
```
GET /somedir/page.html HTTP/1.1
Host: www.tjut.edu.cn
User-agent: Mozilla/4.0
Connection: close
Accept-language:cn
```





12. 主要的方法（method）有：GET，POST，HEAD，PUT，DELETE。前三个是 1.0 和 1.1 当中都有的，后两个只有 1.1 当中有。
13. 使用 GET 方法时实体主体（entity body）为空，当方法为 POST 时，实体主体中包含的是用户在表单字段输入的值。
14. HTTP 响应报文头部格式：

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 06 Aug 1998 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 1998 …...
Content-Length: 6821
Content-Type: text/html
```



15. 响应报文状态码（status code）的含义（仅供了解）：
    1xx 表示通知信息的，如请求收到了或正在进行处理。
    2xx 表示成功，如接受或知道了。
    3xx 表示重定向，表示要完成请求还必须采取进一步的行动。
    4xx 表示客户的差错，如请求中有错误的语法或不能完成。
    5xx 表示服务器的差错，如服务器失效无法完成请求。
    典型的状态码：
    `200 OK`
    `301 Moved Permanently`
    `304 Not Modified`
    `400 Bad Request`
    `404 Not Found`
    `505 HTTP Version Not Supported`
16. Cookies：保存用户信息；web 缓存：改善网络性能；条件 GET，`If-modified-since`，`301 Moved Permanently`，`304 Not Modified`。
17. FTP 的端口：21。（控制连接 21 端口，数据连接 20 端口），控制信息带外（out of band）传送（数据和控制信息不使用同一 TCP 连接），保留用户状态（state）信息，每个数据连接只能传送一个文件（非持久连接）。
18. 用户代理（user agent）与邮件服务器（mail server），邮件服务器间的协议：SMTP 协议，采用 TCP 连接 25 端口，报文队列（massage queue）
19. SMTP 比 HTTP 问世的时间要长的多。
20. SMTP 协议限制所有邮件报文的主体部分只能采用简单的 7 位 ASCII 码表示。
21. SMTP 的 TCP 连接是从发送方到接收方的直接相连，不通过中转。
22. SMTP 采用持久连接，如果发送邮件服务器有几个报文发往同一个接收邮件服务器，它可以通过同一个 TCP 连接发送所有这些报文。对每个报文，客户机都用一个新的 `MAIL FROM:`<发送方服务器名>开始，用一个独立的句点指示该邮件的结束，并且仅当所有邮件发送完后才发送 `QUIT`。
23. 一个 SMTP 的例子：

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

24. SMTP 与 HTTP 对比：

相同点：都是从一台主机向另一台主机发送文件，都可以使用持久连接；

不同点：1、HTTP 主要是一个拉协议（pull protocol），TCP 连接是由想获取文件的机器发起的；SMTP 基本上是一个推协议（push protocol），TCP 连接是由要发送文件的机器发起的；

2、SMTP 要求每个报文（包括他们的主体）都使用 7 位 ASCII 码格式，HTTP 则没有这个限制；

3、HTTP 把每个对象封装到它自己的 HTTP 响应报文中，SMTP 将所有报文对象放在一个报文当中；

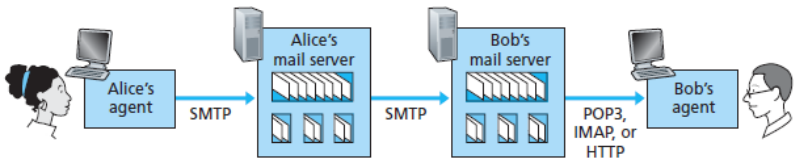
25. 发送的报文首部：

```
From: alice@crepes.fr
To: bob@hamburger.edu
```

接收到的报文将由 SMTP 接收服务器添加 `Received` 行。

26. MIME 通常使用 base64 对邮件当中的非文本信息进行编码。

27. 电子邮件协议及其通信实体：



28. POP3（110 端口）当中的服务器命令可能有两种：`+OK` 和 `-ERR`。特许阶段有两个特殊的命令：`user <user name>` 和 `pass <password>`。例如

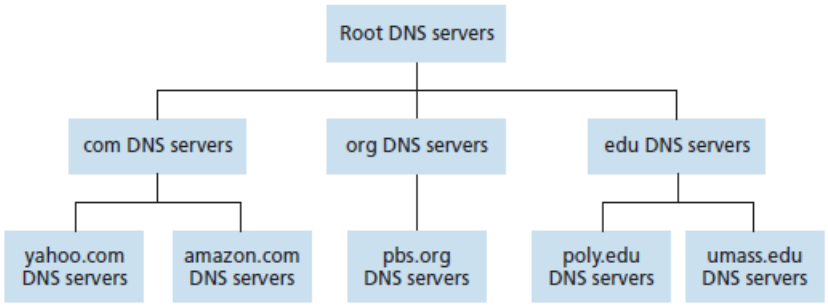
```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

29. 在特许阶段之后，用户代理仅使用四个命令：list，retr，dele 和 quit。例如

```
C: list
S: 1 498
S: 2 912
S: .
```

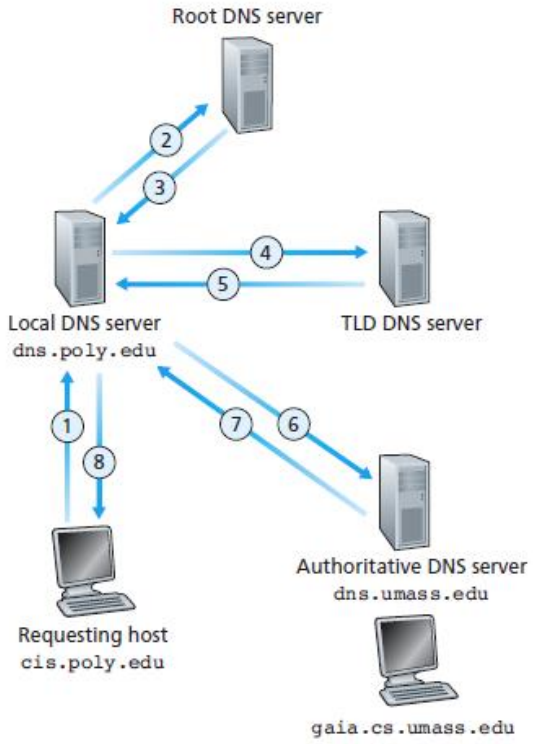
```
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

30. 所有的 DNS 请求和回答报文使用 UDP 数据报经端口 53 发送。DNS 采用分布式的设计方案。
31. DNS 提供的服务：主机名到 IP 地址转换，主机别名（host aliasing），邮件服务器别名（mail server alising），负载分配（load distribution）。多个服务器可以具有相同的别名。
32. DNS 层次结构：

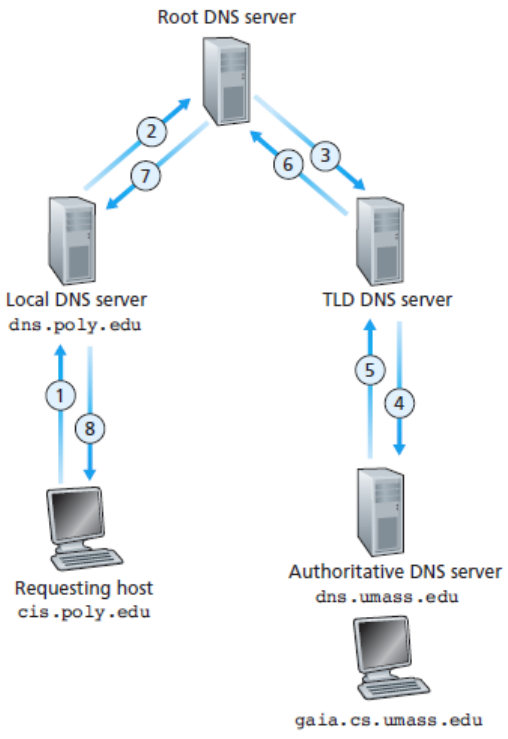


13 个根（root）DNS 服务器—顶级域（top-level domain）服务器—权威（authoritative）DNS 服务器

33. 本地（local）DNS：当本地 DNS 没有相应记录时做代理查询。
34. 递归查询（recursive query）和迭代查询（iterative query）



迭代查询



递归查询

35. DNS 资源记录是一个包含了下列字段的四元组：(name, value, type, ttl)

如果 Type=A，则 Value 是该主机名的地址。因此，一条类型为 A 的资源记录提供了标准的主机名到 IP 地址的映射；

如果 Type=NS，则 Name 是域，而 Value 是知道如何获得该域中主机 IP 地址的权威 DNS 的主机名；

如果 Type=CNAME，则 Value 是别名为 Name 的主机对应的规范主机名；

如果 Type=MX，则 Value 是别名为 Name 的邮件服务器的规范主机名。

TTL 是该记录的生存时间，它决定了资源记录应当从缓存中删除的时间。

36. 注册域名时需要插入的资源记录，例如

(networkutopia.com, dns1.networkutopia.com, NS)

(dns1.networkutopia.com, 212.212.212.1, A)

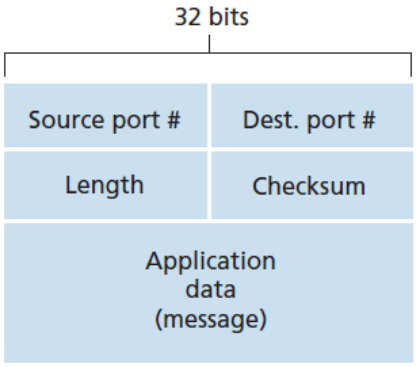
37. 发送 DNS 查询报文，使用 nslookup 命令。

Chapter 3 Transport Layer

1. IP 的服务模型：尽力而为交付服务（best-effort delivery service）—不可靠服务（unreliable service）
2. 运输层的服务：多路复用（multiplexing），多路分用（demultiplexing），数据交付（data delivery），差错检测（error detection）
3. TCP 的附加服务：可靠数据传输（reliable data transfer）：流量控制（flow control）、顺序号（sequence number）、确认号（acknowledgement number）、计时器（timer），拥塞控制（congestion control）
4. 分用和复用的数据标识：端口号（port number）：16 位二进制数 0-65535，周知端口号（well-known port number）0-1023。
5. 一个 UDP socket 是由一个包含目的 IP 地址和目的端口号的二元组来全面标识的。因此，如果两个 UDP 报文段有不同的源地址和/或源端口号，但具有相同的目的 IP 地址和目的端口号，那么这两个报文段将通过相同的目的 socket 定向到其中的目的进程。TCP socket 通过四元组来标识：源端口号，源主机 IP 地址，目的端口号，目的主机的 IP 地址。服务器主机可同时支持很多 TCP socket，每个 socket 与一个进程相联系。
6. 流行的因特网应用及其采用的传输层协议：

Application	Application-Layer Protocol	Underlying Transport Protocol
Electronic mail	SMTP	TCP
Remote terminal access	Telnet	TCP
Web	HTTP	TCP
File transfer	FTP	TCP
Remote file server	NFS	Typically UDP
Streaming multimedia	typically proprietary	UDP or TCP
Internet telephony	typically proprietary	UDP or TCP
Network management	SNMP	Typically UDP
Routing protocol	RIP	Typically UDP
Name translation	DNS	Typically UDP

7. UDP 所具有的传输层服务：复用分用，差错检测，数据交付。
8. UDP 优点：快，不用建立连接和断开连接，不保留连接状态，分组首部开销小。
9. 使用 UDP 的应用是可以实现可靠数据传输的。这可通过在应用程序自身中建立可靠性机制来完成。
10. UDP 报文段结构：

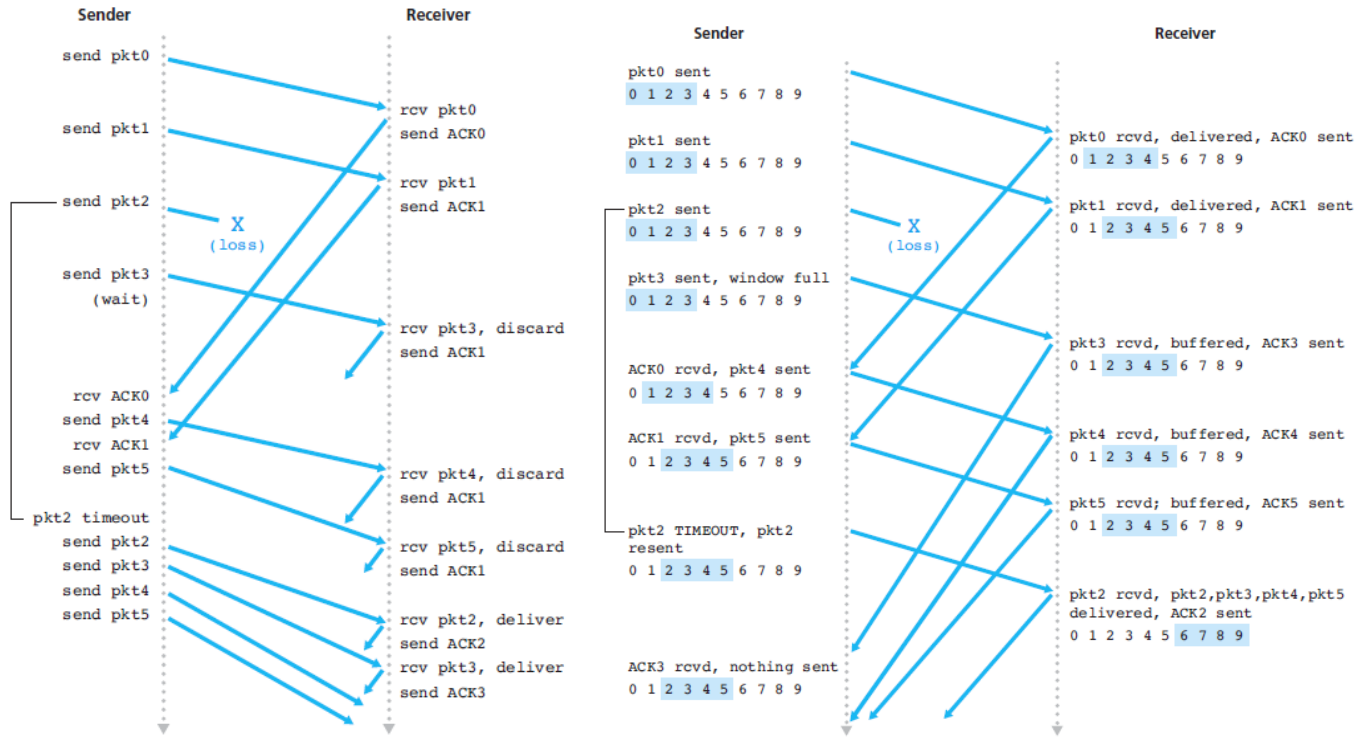


长度字段指明了包括首部在内的 UDP 报文段长度。

11. UDP 检验和（checksum）：在检验和字段中填入全 0，按 16 比特字相加（最高位回卷到最低位相加），得到的和的反码为检验和。在接收方将所有 16 比特字相加，在无差错的情况下将得到全 1。
12. 可靠数据传输协议（reliable data transfer protocol）
    1）rdt1.0：仅提供数据发送；
    2）rdt2.0：考虑比特差错。解决方法：差错检测（检验和），接收方反馈（ACK 和 NAK），重传。停等（stop-and-wait）协议。
    3）rdt2.1：考虑 ACK/NAK 分组本身的差错（顺序号和确认号），可能出现冗余分组（duplicate packet）。
    4）rdt2.2：不发送 NAK，而是发送一个对上次正确接收分组的 ACK（冗余 ACK）。
    5）rdt3.0：考虑丢包（packet loss）。解决方法：倒计时器（countdown timer）。发送方要能做到发送分组时启动倒计时器，响应定时器中断，终止定时器。因为分组序号在 0 和 1 之间交替，因此 rdt3.0 有时被称为比特交替协议（alternating-bit protocol）。
    6）rdt 总结：

rdt 版本	发送方状态数	接收方状态数
1.0	1	1
2.0	2	1
2.1	4	2
2.2	4	2
3.0	4	2

13. 停等协议需要等待上一个分组的确认才能发送下一个分组。流水线（pipelining）技术改善了这个问题。流水线技术可对可靠数据传输协议带来如下影响：必须增加序号范围，协议的发送方和接收方必须缓存多个分组，解决差错恢复。
14. 后退 N 步（go-back-N）协议：滑动窗口（sliding-window）协议。累积确认（cumulative acknowledgement）：接收方对目前收到的最后一个正确有序的分组发出确认，丢弃所有乱序分组。如果超时，发送方将重传所有已发送但还未被确认的分组。
15. 解决丢弃大量分组：选择重传（selective repeat）。单独确认。窗口下沿的分组收到（被确认）时滑动接收（发送）窗口，超时则重发发送窗口下沿的分组。**对于 SR 协议而言，窗口长度必须小于等于序号空间大小的一半。**

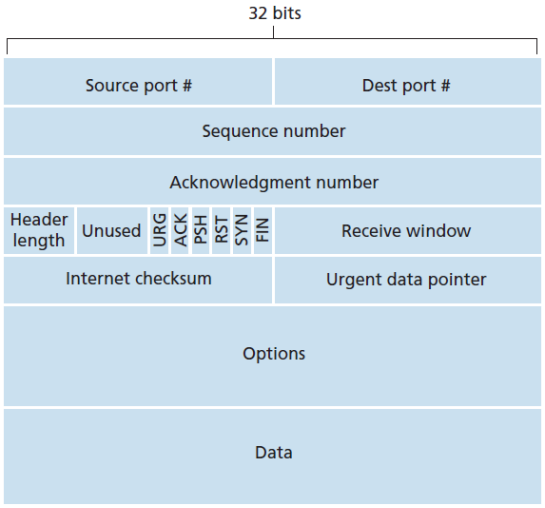


GBN SR

16. 可靠数据传输机制及其用途的总结

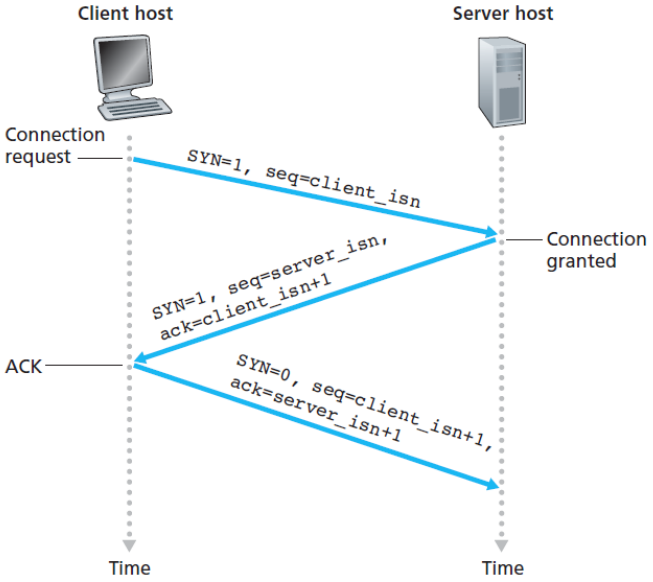
机 制	用途和说明
检验和	用于检测在一个传输分组中的比特错误
定时器	用于检测超时/重传一个分组，可能因为该分组（或其ACK）在信道中丢失了。由于当一个分组延时但未丢失（过早超时），或当一个分组已被接收方收到但从接收方到发送方的ACK丢失时，可能产生超时事件，所以接收方可能会收到一个分组的多个冗余拷贝
序号	用于为从发送方流向接收方的数据分组按顺序编号。所接收分组的序号间的空隙可使该接收方检测出丢失的分组。具有相同序号的分组可使接收方检测出一个分组的冗余拷贝
确认	接收方用于告知发送方一个分组或一组分组已被正确地接收到了。确认报文通常携带着被确认的分组或多个分组的序号。确认可以是逐个的或累积的，这取决于协议
否定确认	接收方用于告知发送方某个分组未被正确地接收。否定确认报文通常携带着未被正确接收的分组的序号
窗口、流水线	发送方也许被限制仅发送那些序号落在一个指定范围内的分组。通过允许一次发送多个分组但未被确认，发送方的利用率可在停等操作模式的基础上得到增加。我们很快将会看到，窗口长度可根据接收方接收和缓存报文的能力或网络中的拥塞程度这两种情况之一或全部来进行设置

17. TCP 是面向连接（connection-oriented）、点对点（point-to-point）的，提供全双工（full duplex）服务（如果一台主机上的进程 A 与另一台主机上的进程 B 存在一条 TCP 连接，那么应用层数据就可在从进程 B 流向进程 A 的同时，也从进程 A 流向进程 B）。
18. TCP 报文段结构：

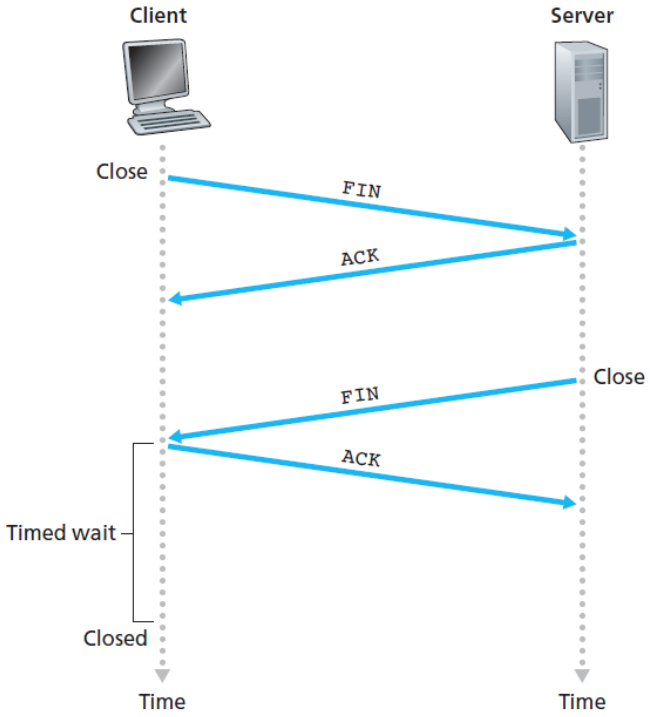


4 比特的首部长度（header length）字段指示了以 32 比特的字为单位的 TCP 首部长度。在选项（option）部分为空时，TCP 首部的长度为 20 字节，因此该字段中的值为（0101）$_2$=5。
接收窗口（receive window）字段指示接收方愿意接收的字节数量。
紧急数据指针（urgent data pointer）字段指示了紧急数据的最后一个字节。

19. 最大报文段长（maximum segment size， MSS）最大传输单元（maximum transmission unit，MTU）
20. 一个报文段的序号（sequence number）是该报文段首字节的字节流编号，确认号是期望收到的下一个报文段的顺序号。
21. TCP 被称为是提供累积确认。
22. TCP 确定重传超时间隔：$TimeoutInterval=EstimatedRTT+4\cdot DevRTT$
    TCP 决不为已重传的报文段计算 Sample RTT，而仅为传输一次的报文段测量 Sample RTT。
23. 加倍超时间隔：TCP 重传具有最小序号的还未被确认的报文段。但是每次 TCP 重传都会将下一次的超时间隔设为先前值的两倍。
24. 冗余 ACK（duplicate ACK）就是再次确认某个报文段的 ACK，而发送方先前已经收到对该报文段的确认。一旦收到三个冗余 ACK，TCP 就执行快速重传（fast retransmit），即在该报文段的倒计时器过期之前重传该丢失的报文段。
25. TCP 解决差错的措施：累积确认，缓存分组，至多重传一个报文段（确认丢失的那个）。
26. 流量控制（flow control）$RcvWindow= RcvBuffer-[LastByteRcvd - LastByteRead]$
    $LastByteSent - LastByteAcked \leq RcvWindow$
27. TCP 连接的建立：三次握手（three-way handshaking）

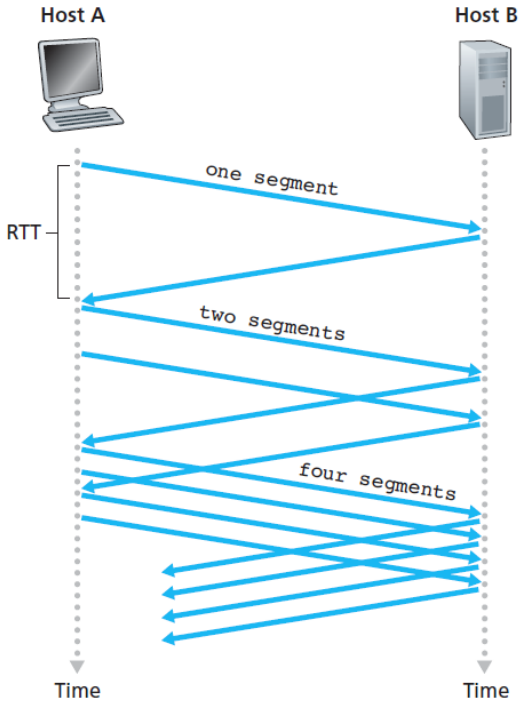


28. TCP 连接的关闭：参与 TCP 连建立的两个进程中的任何一个都能终止连接。



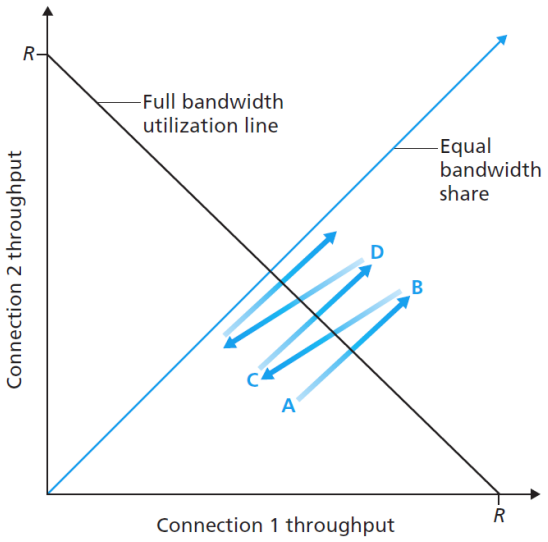
29. 加性增乘性减（additive-increase, multiplicative-decrease）：拥塞窗口（congestion window）线性增长，当发生丢包事件时降至当前值的一半。

30. TCP 慢启动（slow start）



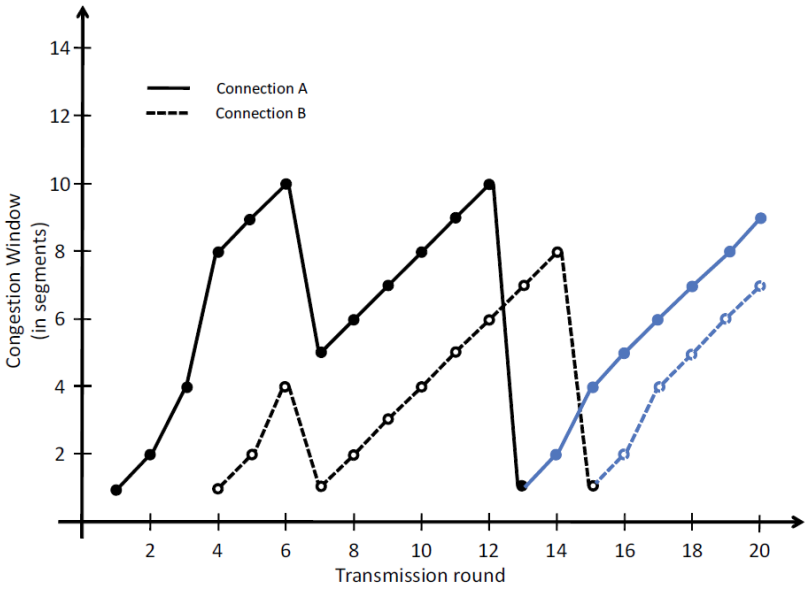
31. TCP 拥塞控制：TCP 维持一个阈值（threshold）。初始状态下阈值为一个较大的值，拥塞窗口从 0 开始慢启动（指数增长），到达阈值后加性增（线性增长/拥塞避免 congestion avoidance），遇到超时事件时将阈值设定为当前拥塞窗口的一半，拥塞窗口从 1 开始慢启动（重复原有过程），遇到三个冗余 ACK 时将阈值设定为当前拥塞窗口的一半，拥塞窗口从该阈值开始加性增（即快速恢复：取消了慢启动阶段。因为相比超时，冗余 ACK 说明网络至少还有能力投递 ACK 分组，拥塞不如超时情况下严重。这是 TCP Reno 方法。在相对早期的 TCP Tahoe 方法中，冗余 ACK 也将导致拥塞窗口从 1 开始慢启动）。

32. TCP 公平性：



典型例题

Consider the following plot of TCP window size as a function of time for two TCP connections A and B. In this problem we will suppose that both TCP senders are sending large files. We also assume that the packet loss events are independent in connection A and B.



Question 1: Considering the above values of congestion window (CongWin) for these connections, can we identify the type of TCP connections (Reno or Tahoe) that have been used by connection A and B? Justify your answers.

Ans： Considering the different changes of CongWin in the 6th and 12th transmission rounds, connection A uses TCP Reno, whereas we cannot say that connection B uses TCP Reno or Tahoe.

Question 2: What are the values of the Threshold parameter between the 1st and the 14th transmission rounds for each connection?

Ans： Connection A: The value of Threshold is 8 between the first and the sixth transmission round. It is 5 between the sixth and the fourteenth transmission round.

Connection B: With the above plot we cannot identify the exact value of Threshold for connection B between the first and

the sixth transmission round. It could have any value larger than 4. From the sixth to the fourteenth transmission round, it is 2 and at the fourteenth transmission round it is 4.

Question 3: At the 12th transmission round for connection A, is segment loss detected by a triple duplicate ACK or by timeout? Justify your answer.

Ans： It is detected by timeout, because CongWin has dropped to 1 at the 13th transmission round.

Question 4: Draw (on Figure 1) the CongWin values of both connections up to the 20th transmission round, considering that there is neither timeout nor duplicate ACK for any of the connections.

Question 5: Assume that the segment size is 1460 bytes and that a total of 87600 bytes have been successfully transmitted over connection A before the 13th transmission round. At which transmission round the cumulative amount of the successful transmitted data is equal to 163520 bytes? Again we assume that there is neither timeout nor duplicate ACK after the 13th transmission round.

Ans： 87600 is equal to 87600/1460 = 60 segments. We would like to know at which transmission round the 163520/1460 = 112 segment will be transmitted. Thus we have to find x such that: 1 + 2 + 4 + 5 + 6 + 7 + 8 +… + x = 112 - 60 = 52

x(x + 1)/2 - 3 = 52 x = 10.

This means that in the 21nd transmission round 163520 bytes will be transmitted.