CS305 Midterm Review

Introduction

1. Internet protocol stack (Application, Transport, Network, Link, Physical) ISP (Internet Service Provider) IXP(Internet Exchange Point)
2. Models (Client/Server model, P2P model, End system) end_system+server+client
3. How to access Network (Cable network, DSL(digital subscriber line), home network, Wireless LAN, Ethernet(企业级网络服务))
4. Network Core
    1. The **mesh() of packet** switches and links that interconnects the Internet's end systems. (Two ways: Packet and Circuit switching)

       Packet: breaks down into packets;each packet in full capacity; store-and-forward(independently);not reserved -> queue delay, packet loss

       Circuit: reserved; buffer; guarantee constant-speed (Frequency Division Multiplexing(FDM), Time Division Multiplexing(TDM))
5. delay, loss, throughput

$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

Application Layer

1. Architecture: (p2p or client-server)
2. Transport layer protocol (TCP or UDP) : **Process** and **Socket**. Processes send and receive through Socket

   client process: process that initiates communication

   server process: process that waits to be contacted
3. HTTP over TCP
    1. HTTP is stateless
    2. non-persistent(2RTT + transmission time) & persistent (第一次需要1个RTT establish connection)
    3. Request & Response
    4. HTTP/1.0(GET , POST , HEAD) HTTP/1.1 (GET， POST , HEAD , PUT , DELETE)
    5. ☆ Cookies (header line of HTTP response, header line of HTTP next request, cookie file kept in user's browser, back-end database at Web server), 相当于keep了一个state
    6. Web caching
        1. proxy(既是client也是server)

    2. proxy会询问server if-modified-since(conditional GET), 如果没有modify就直接可以直接让client获取(304),否则可以进行更新
7. 可以解决的问题：ISP之间占有率(access link)比较高，LAN利用率比较低；将部分数据存储到local cache proxy能大大降低delay以及合理利用带宽(lower link utilization)
8. MSS（Maximum Segment Size）是TCP（传输控制协议）中的一个重要参数，用于指示在TCP连接中单个数据段的最大大小。

4. Electronic mail
    1. Main Components: user agent, mail server, SMTP(two type commands and response)
    2. SMTP (Simple Mail Transfer Protocol, over TCP): deliver to receiver's server (PUSH operation), **3个RTT完成最终handshake** since we should establish TCP connection first.
    3. HTTP 把每个对象都封装到它自己的 HTTP 相应报文当中，即一个对象对应一个报文，而 SMTP 则把所有报文对象放在一个报文当中
    4. Mail access protocol: (PULL operation)
        1. POP3(Post Office Protocol 3): Authorization + Transaction + Update + stateless
        2. IMAP(Internet Mail Access Protocol): keep state
        3. HTTP: Web-base, Push or Pull.
5. DNS:
    1. Domain Name System, stateless
    2. Services: translation(host name to IP address), host and mail server aliasing(别名, 不同的域名指向同一个IP地址), load distribution(不同IP可以对应一个Web server),
        - [x] Use UDP(Port 53)
    3. Structure: 分发式、多层结构

        Why not centralize DNS? • Single point of failure • Traffic volume • Distant centralized database • Maintenance: huge database, update frequently
        1. root server -> TLD(Top-level Domain) server(.com, .cn, .edu, ...) -> Authoritative server, local DNS server 对应一个ISP
        2. Iterative query(由local server 完成对root,TLD, Authoritative服务器的询问，最终返回给client)
        3. Recursive query(按照顺序直接询问, client -> local -> root -> TLD -> Authoritative, 最后倒序返回给client)
        4. Local DNS会有TLD级的cache, root服务器不总是被访问，TTL(Time To Live), DNS会将网站的IP存储在缓存中以便下一次快速读取，加速域名解析
        5. 资源记录是四元组**Name, Value, Type, TTL**
    4. Protocol: 认证时候加上dns.xxx.xxx

        没有认证先是NS消息，然后是A消息
6. P2P vs Client-Server
    1. Client-Server: $Time \geq max\{NF/u_s, F/d_{min}\}$(N copies need to be sent), where F is the size and u_s is server upload capacity and d_i is the user download capacity.
    2. P2P: clients既要上传也要下载, $Time \geq max\{F/u_s, F/d_{min}, NF/(u_s + \Sigma u_i)\}$
7. BitTorrent
    1. tracker: 追踪哪些peer参与了对等交换, torrent: 对等体交换文件碎片

2. client先向tracker register，然后得到peer-list，然后开始交换，从rarest missing parts 优先开始获取
3. tit-for-tat交换策略

8. Video streaming
    1. coding(compression)：use redundancy within and between images to decrease # bits used to encode image, 用冗余空间进行压缩
    2. CBR(constant bit rate), VBR(variable bit rate)
    3. DASH(Dynamic, Adaptive, Streaming over HTTP)
        1. ☆ Client decides: when to request, what encoding rate, where to request
9. CDN(Content Distribution Network)
    1. 在ISP或者IXP地方存储有copy
    2. CDN节点上会存储拉取策略(需要时候才会去拉取)、从哪里拉取的指令(Cluster selection)、具体操作步骤
    3. CDN operation
10. Socket communication

    A socket is one endpoint of a two-way communication link between two programs running on the network.
    1. UDP: socket(AF_INET(*IPv4*),SOCK_DGRAM(*UDP socket*))
    2. TCP: connection_setup(cost two-way handshake, 1RTT) socket(AF_INET,SOCK_STREAM(*TCP socket*))
11. URL (Uniform Resource Locator)

Transport Layer

1. UDP: unreliable, unordered, error-checking. TCP: reliable, in-order, congestion control, flow control, connection setup
2. ☆ multiplexing and demultiplexing

    相当于是复用，多个应用层数据流共享一个传输层协议，然后根据他们的header信息进行demultiplexing
    1. multiplexing at sender: handle data from multiple sockets, add transport header (later used for demultiplexing)
    2. demultiplexing at receiver: use header info to deliver received segments to correct socket
        1. UDP: UDP socket is fully identified by a **two-tuple** consisting of a **destination IP address** and a **destination port number**
        2. TCP: Each TCP socket identified by its own 4-tuple: source IP address, source port number, dst IP address, dst port number(dst port 80 but demux to different sockets) **There is not always a one-to-one correspondence between connection sockets and processes**
3. UDP
    1. User Datagram Protocol, connectionless, no congestion control, used in DNS.
    2. Checksum 1s complement of the sum of segment contents, 计算checksum时候记得最后如果多出了carry要加到最前面
    3. 一些情况下**其实是header加上DNS信息**

4. RDT(reliable data transfer)
    1. rdt1.0 在绝对安全的管道下传输
    2. rdt2.0 引入error detection (ACK,NAK)、重传输会stop and wait
    3. rdt2.1 (假设不丢失数据包)由于rdt2.0的问题是无法保证ACK,NAK不损坏，引入序列号识别丢失和重复的包，收到重复直接丢弃 two seq #(0,1) added，只需要两个序列号就足够了，解决了重复导致pkt混淆的情况
    4. rdt2.2 只使用ACK，用重复发送的ACK来表示需要重复传输
    5. rdt3.0 (假设数据包依旧有可能丢失)等待一定时间让他返回ACK，否则直接重新发送(都需要说明序列号)
    6. rdt3.0利用率很低，用pipeline方式改进
        1. Go-Back-N cumulative ACK，从没有收到ACK回复的地方开始重传（不需要buffer），如果expect不等于收到的包，那么receiver重复发送收到的上一个包裹的ACK(cumulative ACK)发送方当某一个包裹没收到ACK就从那里开始retransmit, 接收方**不需要buffer**，因为是cumulative ACK，收到不expect就重复发送上一个ACK
        2. Selective-repeat individual ACK **需要buffer**，只重新传输没收到ACK回复的包 Only retransmit the unpacked pkt (SR)，receiver只管回发收到了什么包裹的ACK，发送方有buffer记录哪些包裹没收到ACK，就只重复发送那些(Individual ACK)
        3. The window size must be less than or equal to half the size of the sequence number space for SR protocols, 否则会出现识别错误

    rdt1.0完全reliable，rdt2系列提供bit error and detection，rdt3有包损失

5. TCP(Transmission Control Protocol) 几个特性
    1. 流量控制(flow control)：利用滑动窗口控制，灵活调整窗口大小
    2. 拥塞控制(congestion control)：避免过于拥堵导致丢包和延迟，控制流量速度

Tip

1. we use 2 hex-numbers to represent 1 byte.
2. RFC request for comment