Chapter2 - Application Layer

Topics:

- Principles of network applications
- Web and HTTP
- FTP
- Electronic Mail
- DNS
- P2P applications
- Socket programming with TCP
- Socket programming with UDP

Principles of network applications

1- Two application architectures

- client-server
  - a always-on host as server, others as clients. clients do not communicate with each other, but communicate with the server.
  - server having a permanent IP address, while clients may have dynamic IP.
  - server farms for scaling
- peer-to-peer(P2P)
  - no always-on server, arbitrary end systems directly communicate.
  - peers connect and change address dynamically.

- cost effective, high scalable but difficult to manage.

e.g Skype P2P to chat, c/s to register while coming online, or get IP addresses of others.

2- sockets

Socket: an API between processes and network to send and receive message, based on transport layer.

An user could control mostly on socket-application, seldom on socket-transport.

Assigning IP and port-number.

3- application requirements classification

- reliable data transfer/ loss-tolerant application
- throughput (throughput-sensitive/ elastic application)
- delay
- security

4- TCP and UDP

TCP	UDP
connection-oriented; reliable transport; flow control; congestion control	unreliable data transfer; connection less

NOT PROVIDE: security guarantee, delay, throughput

5- Application layer protocol

- types of messages
- message syntax
- message semantics
- rules

public domains: RFC; proprietary : the one of Skype

DNS(Domain Name System)

1- DNS

a kind of application based on UDP on port 53

- distributed database implemented in hierarchy of many name servers
- application-layer protocol

2- hierarchy

root DNS

Top-Level DNS

Zones and Authoritative DNS

local DNS

DNS cache

resolution: iterated query/ recursive query

Web and HTTP

web page: multiple objects

Web: C/S model

HTTP: client request for web and server send web, **stateless** for server does not contain information about clients.

TCP connection initialization request -> TCP connection set up -> HTTP -> TCP close

δ : HTTP/3 uses UDP

1- cookie

Files or strings contain server state relevant for client.

components:

- cookie header of HTTP response message
- cookie header of HTTP request message
- cookie file in user's host
- back-end database

2- performance: Page Load Time

Round Trip Time(RTT): a small packet travel from client to server then back

Page Load Time(PLT) = RTT(TCP connection) + RTT(HTTP request)+ file transmission time

For non persistent HTTP:

PLT=N*RTT+transmit time, N as the number of objects

3- web cache(proxy server)

- reduce response time
- reduce traffic on an institution's access link
- reduce load on origin servers

problem: object in web cache may be out of date

solution: conditional GET

FTP: file transfer protocol

FTP: c/s model

client: side that initiates transfer

server: remote host

1- control and data

control connection: persistent

data connection: non-persistent

Electronic Mail

1-components

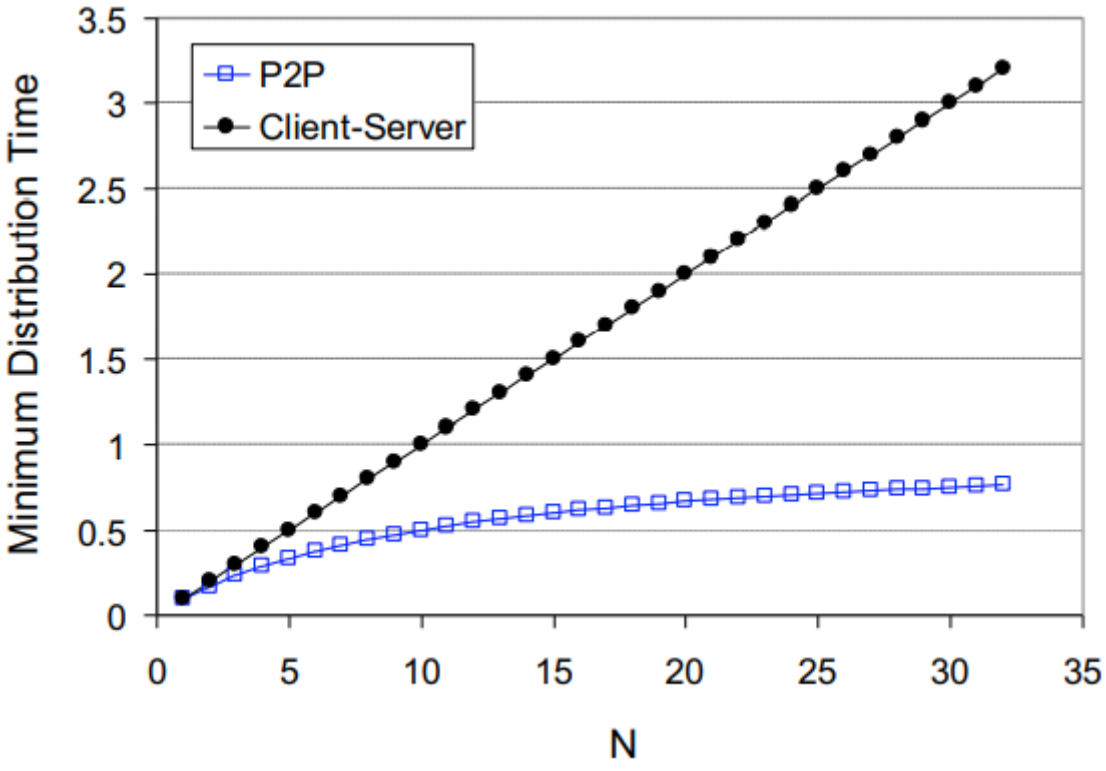
- user agents
- mail servers: mailbox for incoming messages, message queue for outgoing messages.
- SMTP(simple mail transfer protocol): between mail servers (based on TCP).

P2P applications

1- challenges

- No servers to rely on
- limited capacity
- decentralized indexing
- participation incentives

2- self scalability



3- BitTorrent

- tracker with tit-for-tat.
- trackerless with DHT