COMPUTER NETWORK SCE 4303

Application Layer

Dr. Yahaya Coulibaly

Topics Covered

1 Principles of network applications

2 Web and HTTP

3 FTP

4 electronic mail

- SMTP, POP3, IMAP

5 DNS

Learning Objectives

- conceptual, implementation aspects of network application protocols
  - transport-layer service models
  - client-server paradigm
  - peer-to-peer paradigm

- learn about protocols by examining popular application-level protocols
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS

1 Principles of Network Applications

Some network applications

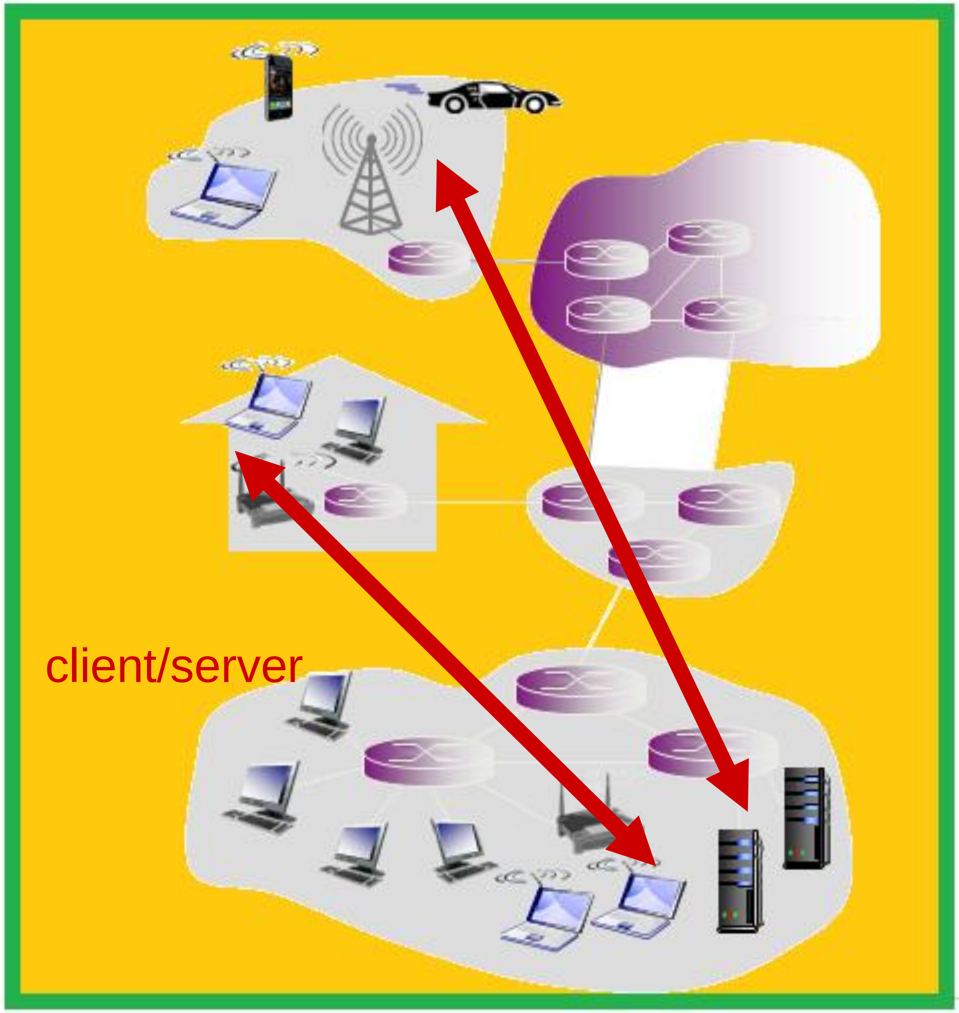
- e-mail
- web
- text messaging
- remote login
- P2P file sharing
- multi-user network games
- streaming stored video (YouTube, Hulu, Netflix)
- voice over IP (e.g., Skype)
- real-time video conferencing
- social networking
- search
- …
- …

Application architectures

possible structure of applications:

- client-server
- peer-to-peer (P2P)

Client-server architecture



server:

- always-on host
- permanent IP address
- data centers for scaling

clients:

- communicate with server
- may be intermittently connected
- may have dynamic IP addresses
- Not allowed to communicate directly with each other

P2P architecture

Characteristics of P2P:

- *not* always-on server
- end systems directly communicate in an arbitrary manner
- 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 and change IP addresses.
- Potential issues:
  - complex management

Application-layer protocol defines

- types of messages exchanged,
  - e.g., request, response
- message syntax:
  - what fields in messages & how fields are delineated (explained)
- message semantics
  - meaning of information in fields
- rules for when and how processes send & respond to messages

open protocols:

- defined in RFCs
- allows for interoperability
- e.g., HTTP, SMTP

proprietary protocols:

- e.g., Skype

Internet transport protocols services

TCP service:

- *reliable transport* between sending and receiving process
- *flow control:* sender won't overwhelm receiver
- *congestion control:* throttle sender when network overloaded
- *does not provide:* timing, minimum throughput guarantee, security
- *connection-oriented:* setup required between client and server processes

UDP service:

- *unreliable data transfer* between sending and receiving process
- *does not provide:* reliability, flow control, congestion control, timing, throughput guarantee, security, or connection setup,

Internet applications: application, transport protocols

application	application layer protocol	underlying transport protocol
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	HTTP (e.g., YouTube), RTP [RFC 1889]	TCP or UDP
Internet telephony	SIP, RTP, proprietary (e.g., Skype)	TCP or UDP

2 Web and HTTP

Web and HTTP

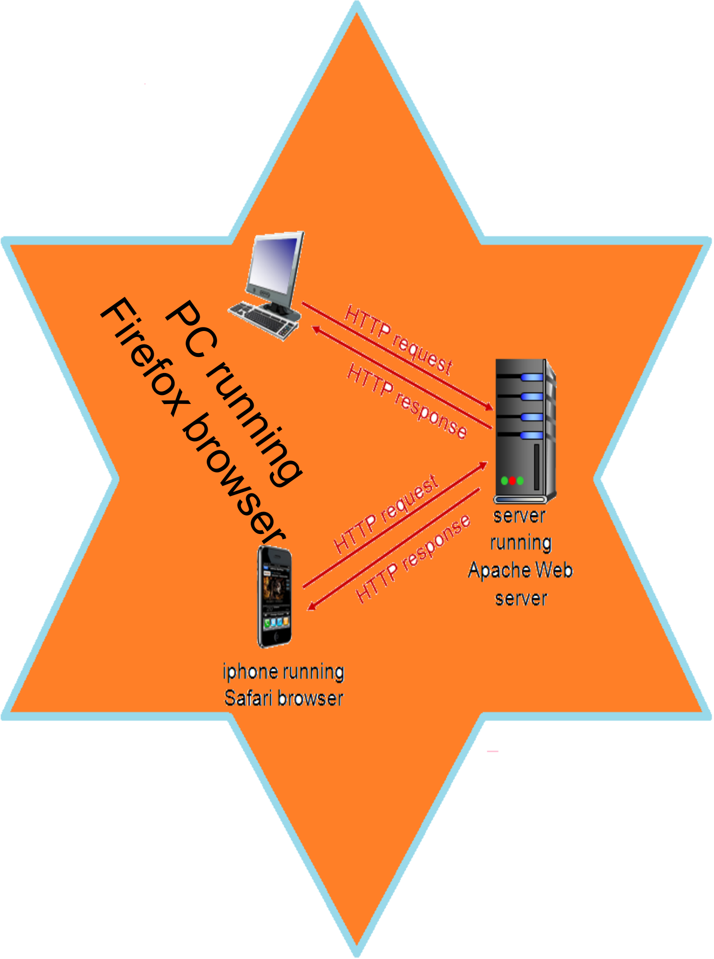
- *web page* consists of *objects and* object can be HTML file, JPEG image, Java applet, audio file,…
- each object is addressable by a *URL,* e.g.,

$$\underbrace{\texttt{www.comp.utm.my}}_{\text{host name}}\underbrace{\texttt{someDept/coulibaly}}_{\text{path name}}$$

HTTP overview

HTTP: hypertext transfer protocol

- It is the application layer protocol for the Web
- It is client/server model
  - *client:* browser that requests, receives, (using HTTP protocol) and "displays" Web objects
  - *server:* Web server sends (using HTTP protocol) objects in response to requests



HTTP overview

uses TCP:

- client initiates TCP connection (creates socket) to server, port 80
- server accepts TCP connection from client
- HTTP messages (application-layer protocol messages) exchanged between browser (HTTP client) and Web server (HTTP server)
- TCP connection closed

It is important to note that HTTP is "stateless";**this means that server maintains no information about past client requests**

HTTP connections

There are two types of HTTO connection:

1. persistent HTTP

- multiple objects can be sent over single TCP connection between client, server

2. non-persistent HTTP

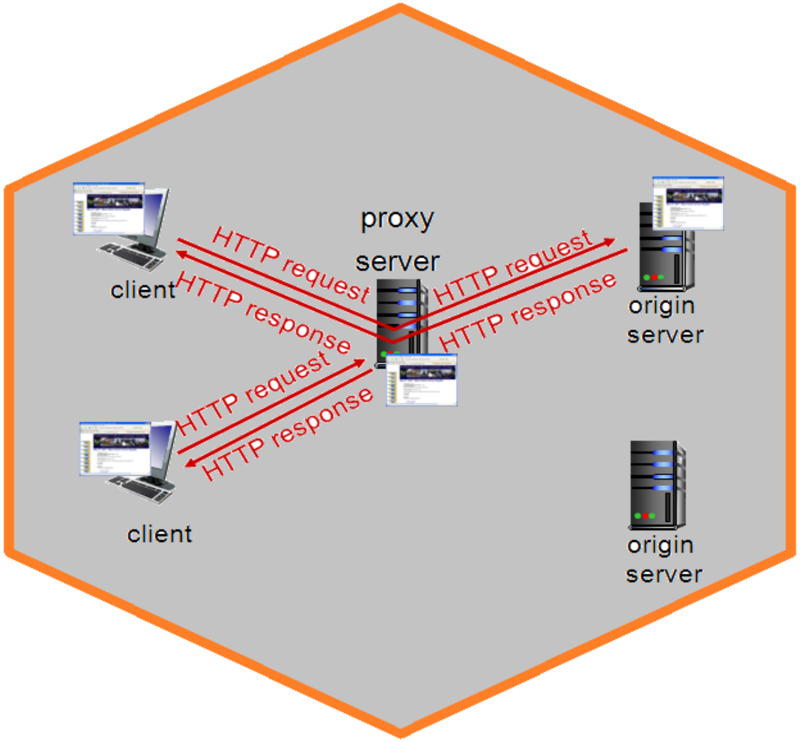
- at most one object sent over TCP connection
  - – connection then closed
- downloading multiple objects required multiple connections

Web caches (proxy server)

Goal: The aim of a proxy server is to satisfy client request without involving origin server

When a Web cache is configured:

- user sets browser: Web accesses via cache
- browser sends all HTTP requests to cache
  - object in cache: cache returns object
  - else cache requests object from origin server, then returns object to client



3. FTP :File Transport Protocol

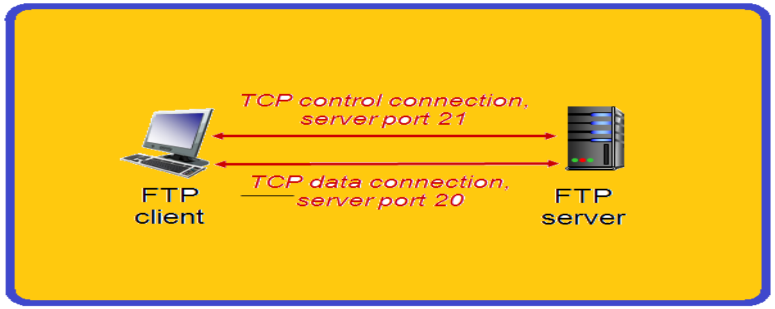
FTP: the file transfer protocol

- transfer file to/from remote host
- client/server model
  - *client:* side that initiates transfer (either to/from remote)
  - *server:* remote host

- RFC 959 defines FTP
- FTP server uses port 21

FTP: separate control, data connections

- FTP client contacts FTP server at port 21, using TCP
- client authorized over control connection
- client browses remote directory, sends commands over control connection
- when server receives file transfer command, *server* opens 2nd TCP data connection (for file) *to* client
- after transferring one file, server closes data connection


TCP control connection, server port 21
TCP data connection, server port 20
FTP client
FTP server

- server opens another TCP data connection to transfer another file
- Thus, control connection is said to be *"out of band"*
- FTP is a state-full protocol as server maintains "state": current directory, earlier authentication

4. Electronic mail: SMTP, POP3 , IMAP

Electronic mail

Three major components:

- user agents
- mail servers
- simple mail transfer protocol: SMTP

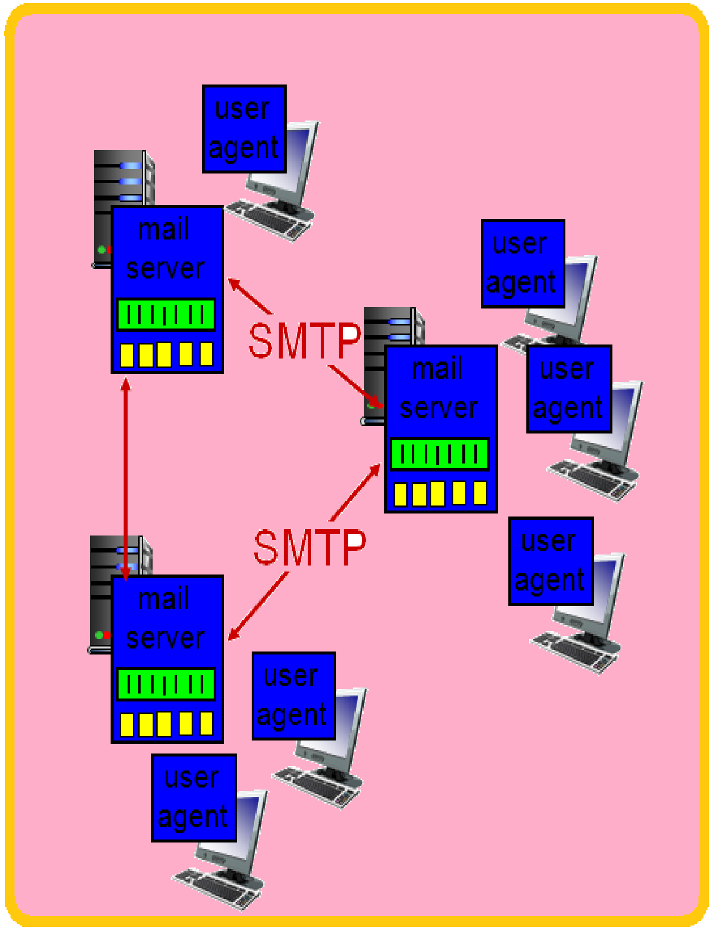
User Agent

- a.k.a. “mail reader”
- composing, editing, reading mail messages
- e.g., Outlook, Thunderbird, iPhone mail client
- outgoing, incoming messages stored on server

Electronic mail: mail servers

mail servers:

- *mailbox* contains incoming messages for user
- *message queue* of outgoing (to be sent) mail messages
- *SMTP protocol* between mail servers to send email messages
  - client: sending mail server
  - "server": receiving mail server



Mail message format

SMTP: protocol for exchanging email msgs

RFC 822: standard for text message format:

- header lines, e.g.,
  - To:
  - From:
  - Subject:

different from SMTP MAIL FROM, RCPT TO: commands!

- Body: the "message"
  - ASCII characters only

5. DNS (Domain Name Service (Server)

DNS: domain name system

Domain Name System:

- *distributed database* implemented in hierarchy of many *name servers*
- *application-layer protocol:* hosts, name servers communicate to *resolve* names (address/name translation)
  - note: core Internet function, implemented as application-layer protocol
  - complexity at network' s "edge"

TLD, authoritative servers

top-level domain (TLD) servers:

- responsible for com, org, net, edu, aero, jobs, museums, and all top-level country domains, e.g.: uk, fr, ca, jp
- Network Solutions maintains servers for .com TLD
- Educause for .edu TLD

authoritative DNS servers:

- organization’s own DNS server(s), providing authoritative hostname to IP mappings for organization’s named hosts
- can be maintained by organization or service provider

Local DNS name server

- does not strictly belong to hierarchy
- each ISP (residential ISP, company, university) has one
  - also called "default name server"
- when host makes DNS query, query is sent to its local DNS server
  - has local cache of recent name-to-address translation pairs (but may be out of date!)
  - acts as proxy, forwards query into hierarchy