Application Layer Overview

Manas Ranjan Lenka
School of Computer Engineering,
KIIT University

Application Architecture

Three types

- Client-Server
  - E.g. Web, Email, FTP
- Peer-to-Peer
  - file distribution(e.g., Bit-torrent), IPTV (e.g., PPLive)
- Hybrid of client-server and peer-to-peer
  - Skype, Instant messenger

Client-Server Architecture

Client:

- Initiates connection to server;
- dynamic IP address

Server:

- Provides specific services (e.g. google server provides search functionality);
- Always on; fixed IP address

Application services that are based on the client-server architecture are often infrastructure intensive and hence costly to provide.

Peer-to-Peer Architecture

- No server; end-systems coordinate to provide required service

- cost effective, since they normally don’t require significant server infrastructure and server bandwidth.

- can have dynamic IP addresses

- Scalable system, but tends to be complex

Hybrid Architecture

- Initial contact is to a central server
  - Used to determine info about other end-systems e.g., tracking the IP addresses of users.
- After this, end-systems talk directly

Application Protocols

- Define types of messages exchanged,
  - e.g., request, response
- Message syntax:
  - Fields in messages
- Message semantics
  - meaning of information in fields
- Rules for when to send and how to act on messages

Many protocols also have a companion protocol that specifies format of the data exchanged

– HTML is companion protocol of HTTP

– RFC 822 and MIME define format of email messages; companion of SMTP

Application Process/program

- Application programs use the application protocol to achieve intended task
  - E.g. Internet Explore, Chrome, Firefox all use HTTP to enable web access

- Processes identified by IP address:Port
  - Popular services have well know port numbers (e.g. 80 for web; 25 for email server)

Transport Services Available to Applications

The Upper layer uses the services of lower layer to achieve the requisite functionality.

Transport layer services are classified along four dimensions:

- reliable data transfer - guaranteed data delivery service
- throughput
  - – bandwidth-sensitive applications i.e. Applications that have throughput requirements
  - – elastic applications i.e. throughput as happens to be available. e.g. Email, file transfer, and Web transfers
- timing
  - – real-time applications require tight timing constraints. e.g. Internet telephony, teleconferencing.
  - – For non-real-time applications, lower delay is always preferable to higher delay, but no tight constraint. e.g. file transfer.
- security
  - – provide confidentiality, data integrity and end-point authentication

Today's Internet transport protocols does not provide the throughput or timing guarantees services.