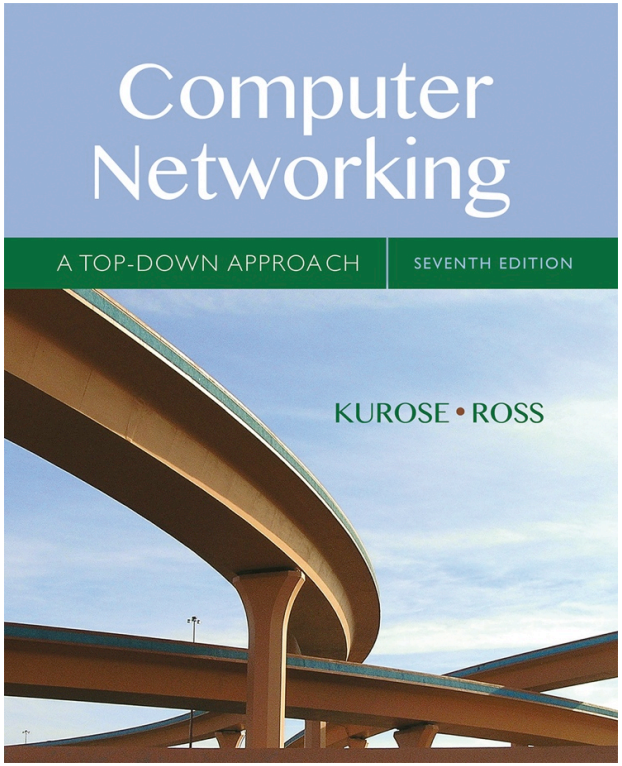
CN-Basic
L13

Application Layer Overview

Dr. Ram P Rustagi
rprustagi@ksit.edu.in
http://www.rprustagi.com
https://www.youtube.com/rprustagi

Resources Acknowledgement

Chapter 2
Application Layer



Computer Networking: A Top Down Approach

7th edition
Jim Kurose, Keith Ross
Pearson/Addison Wesley
April 2016

A note on the use of these Powerpoint slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Application Evolution in Network

- 1980s
  - Text based Email, File Transfer, Remote login, Newsgroup
- 1990s
  - Web surfing, web search, e-commerce
  - Killer Applications
    - P2P file sharing, instant messaging
- 2000s
  - Voice and Video applications (Skype),
  - Rich multimedia Apps, User generated video contents
- 2010s
  - Social computing apps, Video Streaming (NetFlix)
  - Multi-player games (SecondLife, WarCraft, ...)
  - Mobile Apps

Chapter 2: Application Layer

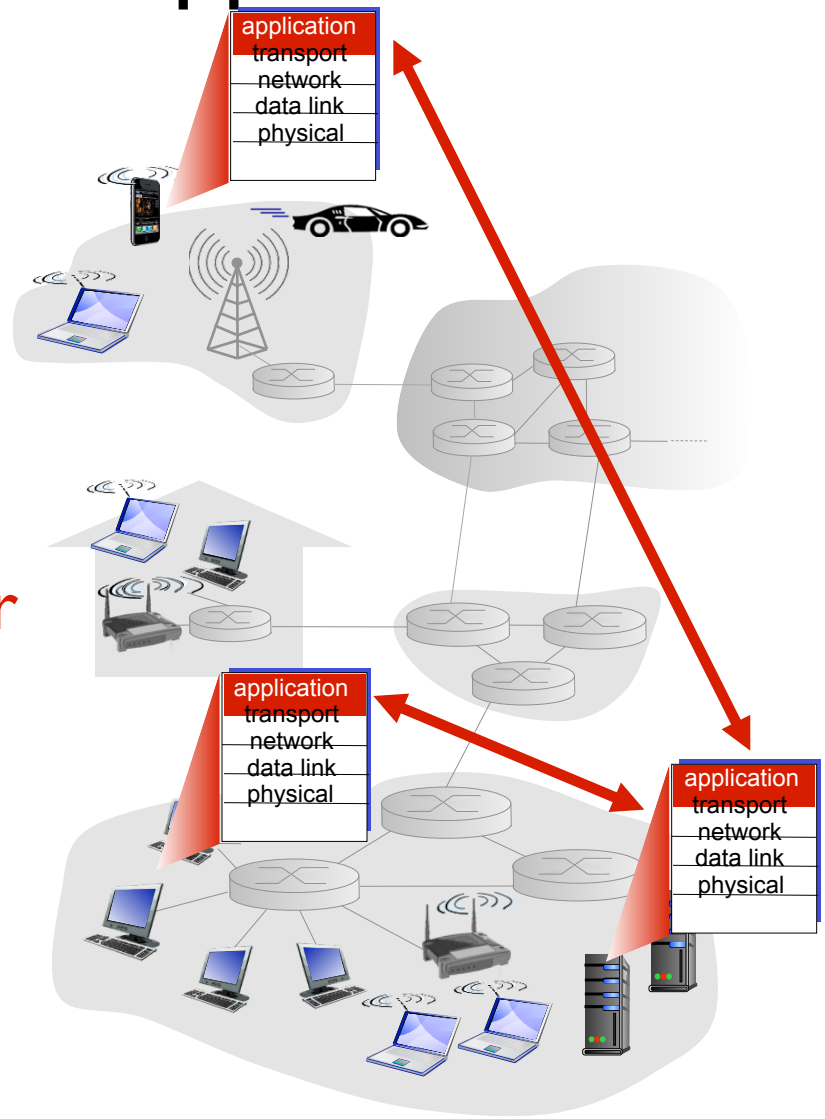
- Goals:
- Conceptual, implementation aspects of network application protocols
  - Transport-layer service models
  - Client-server paradigm
  - Peer-to-peer paradigm
  - Content Distribution Networks
- Learn about protocols by examining popular application-level protocols
- HTTP
- SMTP / POP3 / IMAP
- DNS
- Creating network applications
- Socket API

Some network apps

- 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
- …
- …

Creating a network app

- write programs that:
- run on (different) *end systems*
- communicate over network
- e.g., web server software communicates with browser software
- no need to write software for network-core devices
- network-core devices do not run user applications
- applications on end systems allows for rapid app development, propagation

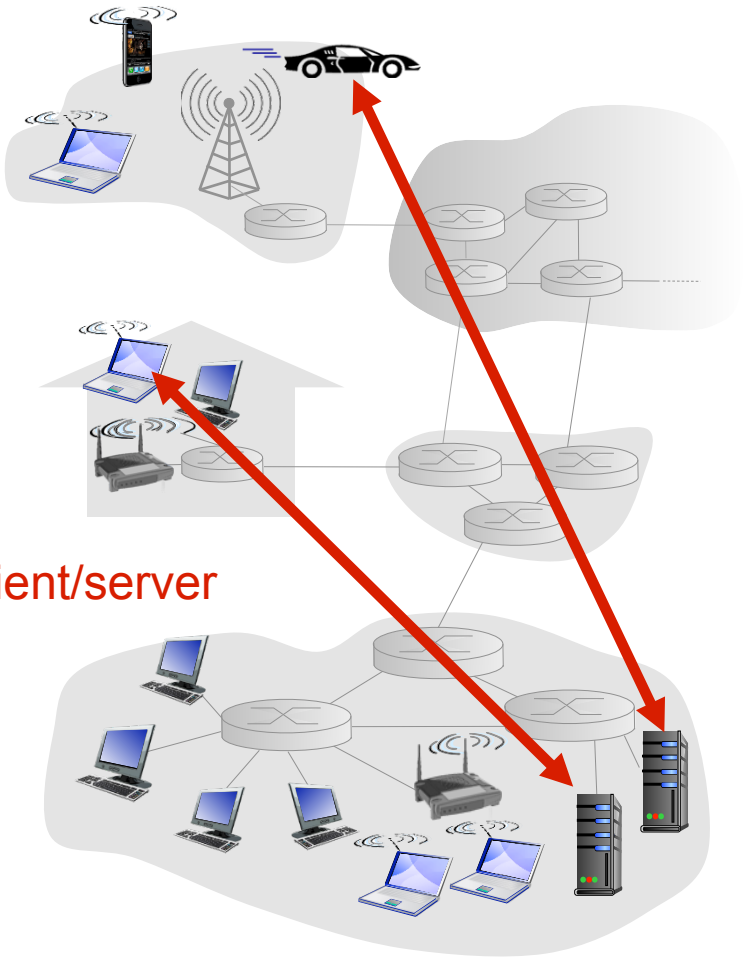


Application architectures

- 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:

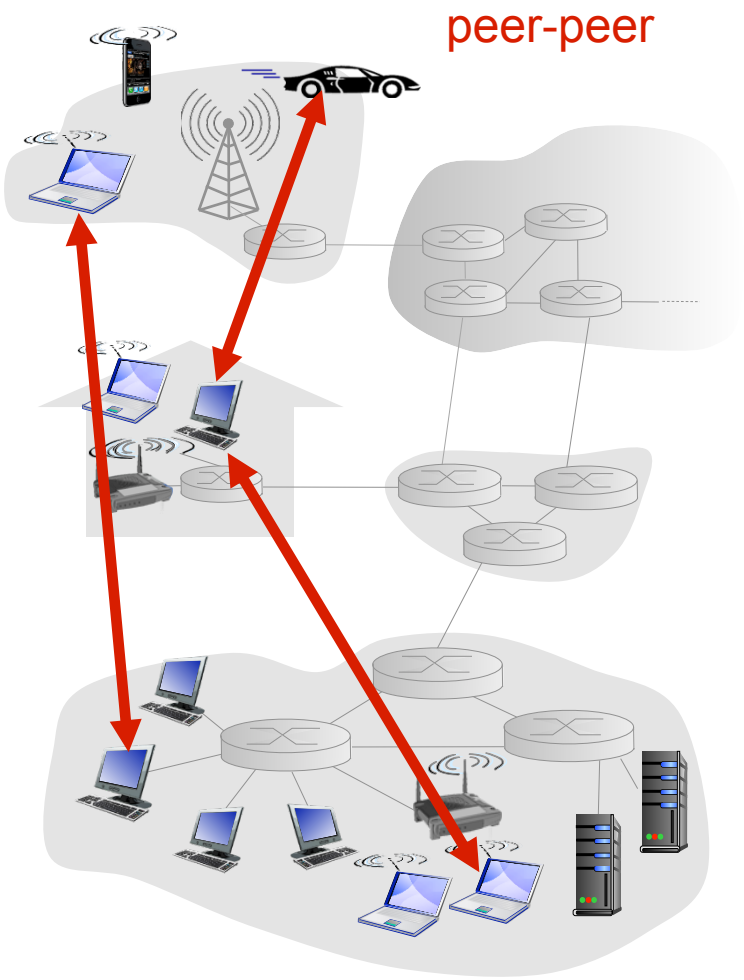
- communicate with server
- may be intermittently connected
- may have dynamic IP addresses
- do not communicate directly with each other

Application Architecture Paradigm

- Client-Server architecture
  - Client initiates requests to server
  - Clients do not talk to each other
  - server examples
    - Web server, FTP Server, Mail server,
  - Applications typically provided by service provider
    - Gmail, Yahoo
    - Google, Bing
    - Amazon, EBay, Flipkart
    - Netflix, Redbox
    - WhatsApp
  - Hosted in data centers

P2P architecture

- *no* always-on server
- arbitrary end systems directly communicate
- 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
  - complex management



Application Architecture Paradigm

- Peer-to-peer architecture
  - No reliance on dedicated servers
  - Direct communication between pairs of hosts
    - Could be via intermittent hosts
  - Peers (desktop, PC, smartphones etc) not owned by service provider
  - Self scalable
    - Each peer adds service capacity to the system
  - Application examples
    - BitTorrent
    - Skype

Peer to Peer Architecture

- Challenges to future applications
  - Asymmetric access to end user (ADSL)
    - P2P video will have issues
  - Security
    - Being distributed in nature, how to secure them
  - Incentives to users
    - How to convince new users to join

App-layer protocol defines

- types of messages exchanged,
  - e.g., request, response
- message syntax:
  - what fields in messages & how fields are delineated
- 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, WhatsApp

What transport service does an app need?

- Data Integrity, Timing, Throughput, Security

- data integrity
  - some apps (e.g., file xfer, web transactions) require 100% reliable data transfer
  - other apps (e.g., audio) can tolerate some loss

- throughput
  - some apps (e.g., multimedia) need minimum amount of throughput to be "effective"
  - other apps ("elastic apps") make use of whatever throughput they get

- timing
  - some apps (e.g., Internet telephony, interactive games) require low delay to be "effective"

- security
  - encryption, data integrity, …

Transport service requirements: common apps

application	data loss	throughput	time sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: 5kbps-1Mbps	yes, 100's msec
stored audio/video	loss-tolerant	video: 100kbps-5Mbps	yes, few secs
interactive games	loss-tolerant	same as above	yes, 100's msec
text messaging	no loss	few kbps up	yes and no
		elastic	

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,

- Q: why bother? Why is there a UDP?

Application layer protocols

- Applications/processes communication
  - via sockets
- Structure of communication
  - What are various fields
  - When to send messages
  - What kind of messages
- Application layer protocol defines
  - Type of messages :
    - send, receive
  - Syntax of various message types
    - Fields of messages
  - Semantics of fields
  - Rules for determining when to send msg

Application layer protocols

- Example applications
- Web Application
  - Components
    - Web browser, server, HTML Page, HTTP protocol
  - HTTP
    - Application layer protocol
- Email applications
  - Components
    - Mail server, mail client, SMTP, POP3, IMAP
  - SMTP, POP3, IMAP4
    - Application layer protocols

Internet apps: 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(RFC 3261), RTP, proprietary (e.g., Skype)	TCP or UDP

Exercise 01

- List three applications that are
  - Time sensitive
  - Time insensitive
- List three applications that can
  - Tolerate some data loss
  - Can't tolerate any data loss
- Research on when TCP provides reliable service, why do we need UDP protocol at transport layer.

Summary

- Application architecture
  - Client-Server
  - Peer to Peer
- Service requirements from Transport layer
- Examples of application layer protocols