Lecture 2
Protocol Stacks and Layering

David Andersen
School of Computer Science
Carnegie Mellon University

15-441 Networking, Fall 2006
http://www.cs.cmu.edu/~srini/15-441/F06/

Last Time

- The Big Picture
  - Goals:
    - Efficiency
    - "ilities" (scalability, manageability, availability),
    - Ease of creating applications
  - Challenges:
    - Scale
    - Geography
    - Heterogeneity (** today's focus!)
- A few specific details:
  - Circuits vs. packets
  - Little bit about routing
  - Service model and how to construct services (** today!)

Today's Lecture

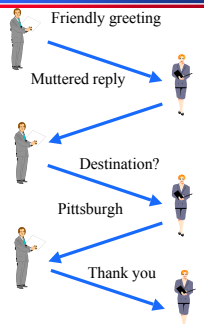
- Last time: "Big picture"
- Today:
  - General architectural principles for networks
  - Introduces a few concrete models & examples
- Where we are going:
  - Tuesday: Application examples (still high level)
  - Thursday: Socket programming review++ (for project)
  - After that: Burrowing into the details, ground up
- Today's specifics:
  - What is a protocol.
  - Protocol stacks.
  - Some history.
  - Standards organizations.
  - Application layer.

Why protocols and layering?

- Interoperability
- Reuse
- Hiding underlying details

What is a Protocol

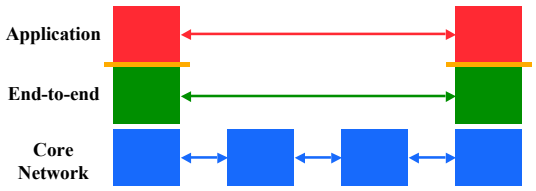
- An agreement between parties on who communication should take place.
- Protocols may have to define many aspects of the communication.
- Syntax:
  - Data encoding, language, etc.
- Semantics:
  - Error handling, termination, ordering of requests, etc.
- Protocols at hardware, software, *all* levels!
- Example: Buying airline ticket by typing.
- Syntax: English, ascii, lines delimited by "\n"



Interfaces

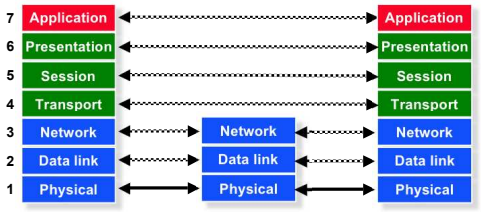
- Each protocol offers an interface to its users, and expects one from the layers on which it builds
  - Syntax and semantics strike again
    - Data formats
    - Interface characteristics, e.g. IP service model
- Protocols build upon each other
  - Add value
    - E.g., a reliable protocol running on top of IP
  - Reuse
    - E.g., OS provides TCP, so apps don't have to rewrite

Protocol and Service Levels



A Layered Network Model

The Open Systems Interconnection (OSI) Model.



OSI Motivation

- Standard way of breaking up a system in a set of components, but the components are organized as a set of layers.
  - Only horizontal and vertical communication
  - Components/layers can be implemented and modified in isolation
- Each layer offers a service to the higher layer, using the services of the lower layer.
- "Peer" layers on different systems communicate via a protocol.
  - higher level protocols (e.g. TCP/IP, Appletalk) can run on multiple lower layers
  - multiple higher level protocols can share a single physical network
- "It's only a model!" - TCP/IP has been crazy successful, and it's not based on a rigid OSI model. But the OSI model has been very successful at shaping thought.

OSI Functions

- (1) Physical: transmission of a bit stream.
- (2) Data link: flow control, framing, error detection.
- (3) Network: switching and routing.
- (4) Transport: reliable end to end delivery.
- (5) Session: managing logical connections.
- (6) Presentation: data transformations.
- (7) Application: specific uses, e.g. mail, file transfer, telnet, network management.

Multiplexing takes place in multiple layers

Looking at protocols

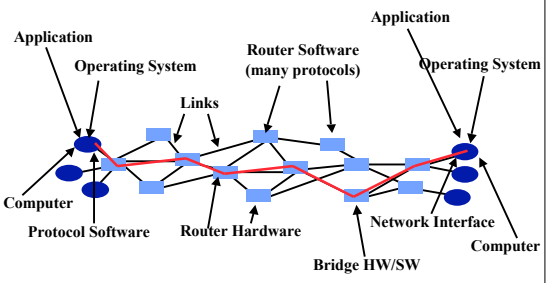
- Hop by hop / link protocols
  - Ethernet
- End-to-end protocols
  - TCP, apps, etc.
- Management / "control plane" protocols
  - Routing, etc.
    - Can be either link or e2e themselves
    - Definition somewhat vague.
- Standards
  - File formats, etc.
    - E.g., JPEG, MPEG, MP3, …

Categories not solid / religious, just a way to view things.

Different Sources of Components

- Application: web server/browser, mail, distributed game,..
- Presentation/session.
  - Often part of application
  - Sometimes a library
- Transport/network.
  - Typically part of the operating system
- Datalink.
  - Often written by vendor of the network interface hardware
- Physical.
  - Hardware: card and link

Motivation: Many many Network Components



Protocols for Interoperability

- Many implementations of many technologies:
- Hosts running FreeBSD, Linux, Windows, MacOS, …
- People using Mozilla, Explorer, Opera, …
- Routers made by cisco, juniper, …
- Hardware made by IBM, Dell, Apple, …
- And it changes all the time.
- Phew!

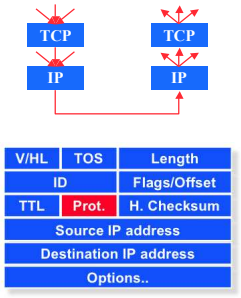
- But they can all talk together because they use the same protocol(s)
  - Application level protocols: HTTP, SMTP, POP, IMAP, etc.
  - Hardware protocols (ethernet, etc)

Protocols for Abstraction & Reuse

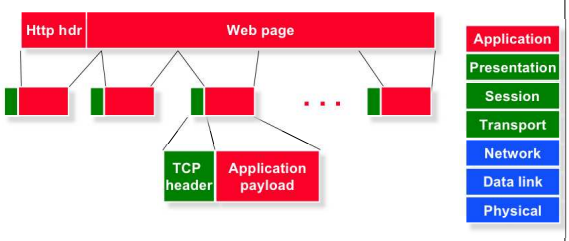
- Multiple choices of protocol at many layers
  - Physical: copper, fiber, air, carrier pigeon
  - Link: ethernet, token ring, SONET, FDDI
  - Transport: TCP, UDP, SCTP
- But we don't want to have to write "a web (HTTP) browser for TCP networks running IP over Ethernet on Copper" and another for the fiber version…
  - Reuse! Abstraction!
  - Protocols provide a standard interface to write to
  - Layers hide the details of the protocols below

Multiplexing and Demultiplexing

- There may be multiple implementations of each layer.
  - How does the receiver know what version of a layer to use?
- Each header includes a demultiplexing field that is used to identify the next layer.
  - Filled in by the sender
  - Used by the receiver
- Multiplexing ooccurs at multiple layers. E.g., IP, TCP, …



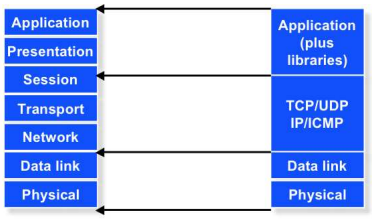
Example: Sending a Web Page



Limitations of the Layered Model

- Some layers are not always cleanly separated.
  - Inter-layer dependencies in implementations for performance reasons
  - Some dependencies in the standards (header checksums)
- Higher layers not always well defined.
  - Session, presentation, application layers
- Lower layers have "sublayers".
  - Usually very well defined (e.g., SONET protocol)
- Interfaces are not really standardized.
  - It would be hard to mix and match layers from independent implementations, e.g., windows network apps on unix (w/out compatability library)
  - Many cross-layer assumptions, e.g. buffer management

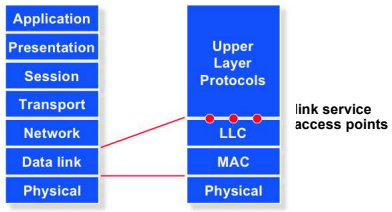
The TCP/IP Model



19

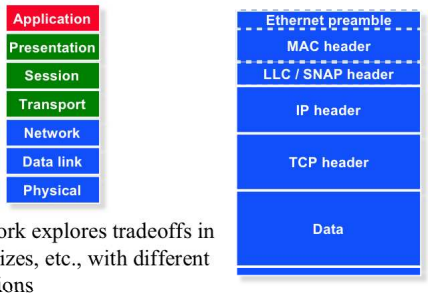
Local Area Network Protocols

IEEE 802 standards "refine" the OSI data link layer.



20

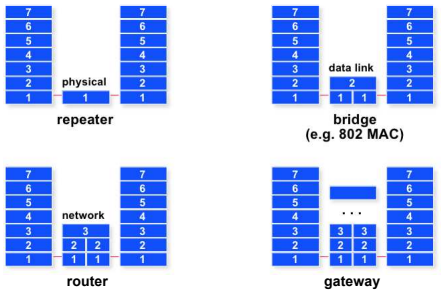
A TCP / IP / 802.3 Packet



Homework explores tradeoffs in header sizes, etc., with different applications

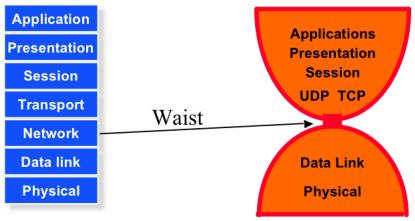
21

Internetworking Options



22

The Internet Protocol Suite



The waist facilitates Interoperability.

The Hourglass Model

23

Some History: The Early Days

- **Early packet switching networks (61-72).**
  - Definition of packet switching
  - Early DARPA net: up to tens of nodes
    - single network
    - discovery of "interesting" applications
- **Internetworking (72-80).**
  - Multiple networks with inter-networking: networks are independent, but need some rules for interoperability
  - Key concepts: best effort service, "stateless" routers, decentralized control (very different from telephones!)
  - Basis for Internet: TCP, IP, congestion control, DNS, …
  - Rapid growth: 10 to 100000 hosts in 10 years
    - Driven by NSF net, research communigy

24

Recent History: Commercialization

- Industry interest in networking encourages first commercial network deployment.
  - In part also encouraged by NSFNET policies
- Introduction of the Web makes networks more accessible.
  - Killer application
  - Good user interface that is accessible to anybody
  - Network access on every desktop and in every home
  - Shockingly recent - 1989, caught on in '92 or so

25

Standardization

- Key to network interoperability.
- A priori standards.
  - Standards are defined first by a standards committee
  - Risk of defining standards that are untested or unnecessary
  - Standard may be available before there is serious use of the technology
- De facto standards.
  - Standards is based on an existing systems
  - Gives the company that developed the base system a big advantage
  - Often results in competing "standards" before the official standard is established

26

Relevant Standardization Bodies

- ITU-TS - Telecommunications Sector of the International Telecommunications Union.
  - government representatives (PTTs/State Department)
  - responsible for international "recommendations"
- T1 - telecom committee reporting to American National Standards Institute.
  - T1/ANSI formulate US positions
  - interpret/adapt ITU standards for US use, represents US in ISO
- IEEE - Institute of Electrical and Electronics Engineers.
  - responsible for many physical layer and datalink layer standards
- ISO - International Standards Organization.
  - covers a broad area

27

The Internet Engineering Task Force

- The Internet society.
  - Oversees the operations of the Internet
- Internet Engineering Task Force.
  - decides what technology will be used in the Internet
  - based on working groups that focus on specific issues
  - encourages wide participation
- Request for Comments.
  - document that provides information or defines standard
  - requests feedback from the community
  - can be "promoted" to standard under certain conditions
    - consensus in the committee
    - interoperating implementations
  - Project 1 will look at the Internet Relay Chat (IRC) RFC

28

Higher Level Standards

- Many session/application level operations are relevant to networks.
  - encoding: MPEG, encryption, ...
  - services: electronic mail, newsgroups, HTTP, ...
  - electronic commerce, ...
- Standards are as important as for "lower-level" networks: interoperability.
  - defined by some of the same bodies as the low-level standards, e.g. IETF

29

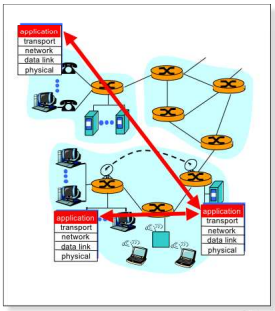
How do you select protocols?

- Application layer
  - Client-server
  - Application requirements
- Application level communication
  - TCP vs. UDP
  - Addressing
- Application examples (Lecture 4).
  - ftp, http
  - End-to-end argument discussion

30

Applications and Application-Layer Protocols

- Application: communicating, distributed processes
  - Running in network hosts in "user space"
  - Exchange messages to implement app
  - e.g., email, file transfer, the Web
- Application-layer protocols
  - One "piece" of an app
  - Define messages exchanged by apps and actions taken
  - Use services provided by lower layer protocols
- Saw sockets API last time



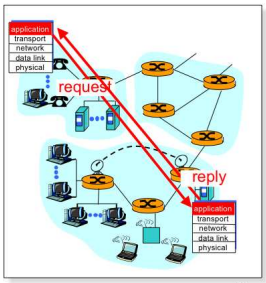
31

Client-Server Paradigm

Typical network app has two pieces: *client* and *server*

Client:
- Initiates contact with server ("speaks first")
- Typically requests service from server,
- For Web, client is implemented in browser; for e-mail, in mail reader

Server:
- Provides requested service to client
- e.g., Web server sends requested Web page, mail server delivers e-mail
- (We'll cover p2p at semester end)



32

What Transport Service Does an Application Need?

Data loss
- Some applications (e.g., audio) can tolerate some loss
- Other applications (e.g., file transfer, telnet) require 100% reliable data transfer

Timing
- Some applications (e.g., Internet telephony, interactive games) require low delay to be "effective"

Bandwidth
- Some applications (e.g., multimedia) require a minimum amount of bandwidth to be "effective"
- Other applications ("elastic apps") will make use of whatever bandwidth they get

33

User Datagram Protocol(UDP): An Analogy

UDP	Postal Mail
Single socket to receive messages	Single mailbox to receive letters
No guarantee of delivery	Unreliable ☺
Not necessarily in-order delivery	Not necessarily in-order delivery
Datagram – independent packets	Letters sent independently
Must address each packet	Must address each reply

Example UDP applications
Multimedia, voice over IP

34

Transmission Control Protocol (TCP): An Analogy

TCP	Telephone Call
Reliable – guarantee delivery	Guaranteed delivery
Byte stream – in-order delivery	In-order delivery
Connection-oriented – single socket per connection	Connection-oriented
Setup connection followed by data transfer	Setup connection followed by conversation

Example TCP applications
Web, Email, Telnet

35

Transport Service Requirements of Common Applications

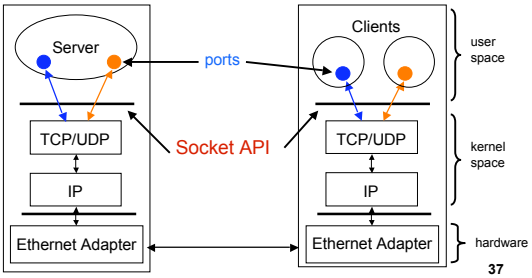
Application	Data loss	Bandwidth	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
web documents	no loss	elastic	no
real-time audio/ video	loss-tolerant	audio: 5Kb-1Mb video:10Kb-5Mb	yes, 100's msec
stored audio/video	loss-tolerant	same as above	yes, few secs
interactive games	loss-tolerant	few Kbps	yes, 100's msec
financial apps	no loss	elastic	yes and no

- Interactions between layers are important.
  - persistent HTTP
  - encryption and compression
  - MPEG frame types. Loss & real-time video.

36

Server and Client

Server and Client exchange messages over the network through a common Socket API



37

Readings

- **Read two papers on the motivations for the Internet architecture:**
  - **"End-to-end arguments in system design", Saltzer, Reed, and Clark, ACM Transactions on Computer Systems, November 1984.**
  - **"The design philosophy of the DARPA Internet Protocols", Dave Clark, SIGCOMM 88.**
- **In-class discussion:**
  - **Briefly next Thursday**
  - **Revisit the topic in the second half of the semester**

38