Internet Protocols
Fall 2005

Lecture 1

Andreas Terzis
terzis@cs.jhu.edu

Course Description

- Learn about the protocols that make the Internet work
  - Understand design space and tradeoffs
  - Compare different protocols at the same level
    - Protocol examples: Ethernet, WiFi, IP, TCP, HTTP
- Learn how to write code for networked applications

Text Books

- Required: **Computer Networks: A Systems Approach (3rd ed.) by Peterson, Davie**
- Recommended: **UNIX Network Programming by Richard Stevens**
- Recommended: **Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching and Traffic Measurements**

Course Times

- Lectures: ThF 2:30-3:45 pm Shaffer 304
- Office Hours: ThF 1-2pm WP 417 and by appt
- TA Section: Every second week
- TA Office Hours: MT 11-12, WP 425

Grading Policies

- Grading
  - Homework 10%, Projects 35%
  - Midterm 20%, Final 30%
  - Class Participation 5%
  - Different curves for 349/449
- NO LATE HOMEWORK ACCEPTED (really)
- Project groups DO NOT change

Course Schedule

- W1 (9/8) Introduction, Internet Architecture
- W2 (9/15) Link Layer: Ethernet, PPP, IP
- W3 (9/22) MAC
- W4 (9/29) IP: Addressing, ARP/RARP, ICMP and examples
- W5 (10/6) Intra-domain routing
- W6 (10/13) Inter-domain routing, IP Mobility
- W7 (10/20) Review, **Midterm (10/21)**

Course Schedule (2)

- W8 (10/27) Transport layer
- W9 (11/3) Congestion Control
- W10 (11/10) QoS, Network Measurements
- W11 (11/17) DNS, HTTP
- W12 (11/24) Thanksgiving
- W13 (12/1) P2P, Streaming Media
- W14 (12/8) Wrap Up
- **Final TBA**

Course Projects

- Three implementation projects
  - Learn how to write TCP/IP networked applications (using the Unix API)
  - Teams of 2-3
  - For the last two projects you will create a software router that handles real traffic
    - http://yuba.stanford.edu/vns/

Logistics

- Course website
  - http://hinrg.cs.jhu.edu/cs349-fall05/
  - Check every day
    - Everything goes there
- Course email list
  - cs349@hinrg.cs.jhu.edu
  - To subscribe: https://hinrg.cs.jhu.edu/mailman/listinfo/cs349/

Short (P)Review

- Very high-level overview of networking
  - Show how the World Wide Web works
  - Present some of the key ideas behind the Internet architecture
    - Large Scale
    - Dealing with failures
    - Co-operation/Competition

World Wide Web

- We use it every day
- Millions of *websites*
- Webpages contain objects
  - HTML, Text, images, etc
  - Base HTML file contains reference to other objects
  - Each object addressable by a URL

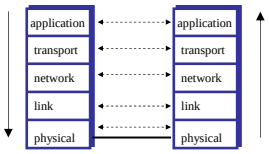
http://www.google.com/index.html

Questions

- How to express what we are looking for
- How to find what we are looking for
- How to transmit bits
- How to protect against errors
- How to direct bits towards the right destination
- How to share resources

Solution

- Divide and Conquer
  - Break the problem in smaller parts
  - Solve each part separately
  - Put the pieces back together
- Layering
- Protocol Stack

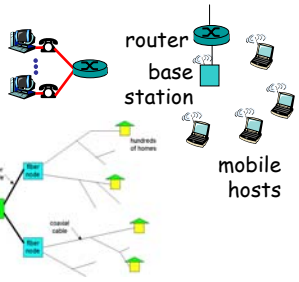


Physical Layer

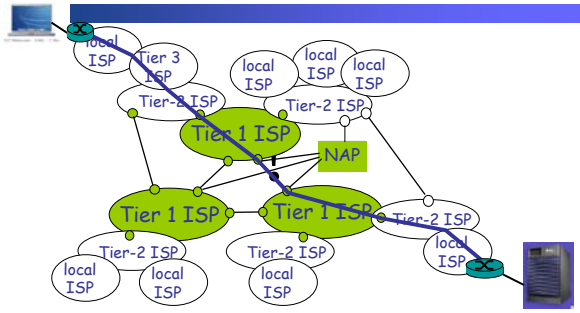
- Send bits between directly connected machines
- Physical Media
  - Twisted Pair
  - Coaxial Cable
  - Fiber Optics
  - Radio Link
- Bandwidth
- Dedicated vs. shared link

Data Link Layer

- Data Link Layer functions
  - Framing
  - Error Detection & Correction
  - Link sharing
- Examples
  - Dialup, ADSL
  - Cable Modems
  - Wireless (WiFi)



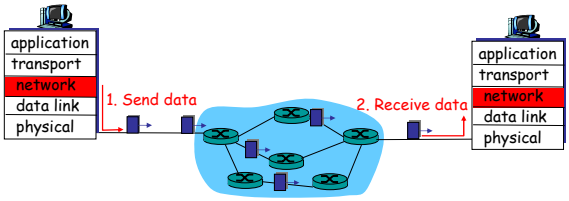
Internet: Network of Networks



ISP Topology

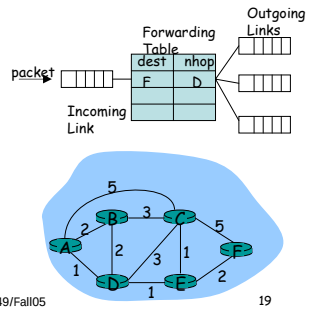
Network Layer

- Networking layer functions
  - Transport packets from sending to receiving hosts
  - *Packet Switching vs. Circuit Switching*



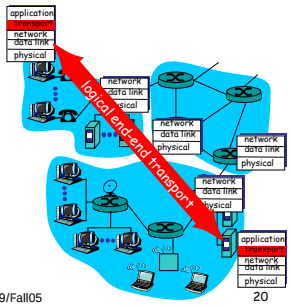
Forwarding vs. Routing

- *Forwarding*
  - *How to send packet to the next hop towards the destination*
- *Routing*
  - *How to compute the next hops*
- Routing algorithms
  - *Distance Vector (RIP)*
  - *Link State (OSPF)*



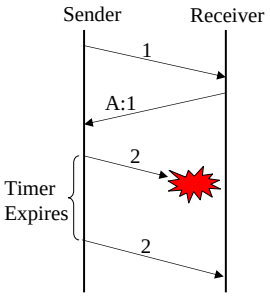
Transport Layer

- Packets get lost
  - Physical errors
  - Equipment failures
  - Congestion
- Transport layer functions
  - Deliver packets reliably
  - share resources among all network users



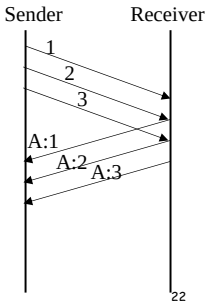
Reliable Transfer

- How to recover from losses
  - Introduce ACKs
  - Timers
  - Stop-and-Wait
    - Sender sends packet set timer
    - Receiver sends ACK
    - If no ACK received when timer expires sender retx



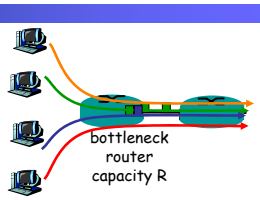
Reliable Transfer (2)

- *Stop-and*-Wait has low Utilization
- Pipelining
  - Senders allows multiple yet-to-be-acked packets (window)
- Two variants
  - *Go-Back-N*
  - *Selective repeat*



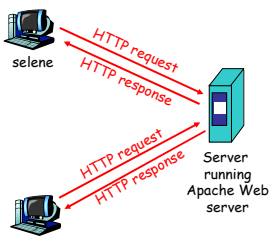
Congestion Control

- Congestion
  - Too many sources sending too much data too fast for *network* to handle
- Results
  - Lost packets
  - Long delays
- Solutions
  - End-to-End
  - Network assisted



Application Layer-HTTP

- So far: data reliably transferred end-to-end
- HTTP: hypertext transfer protocol
- client/server model
  - client: browser that requests, receives, "displays" Web objects
  - server: Web server sends objects in response to requests



Request/Response Timeline

We enter URL `www.google.com/index.html` (contains text, reference to 1 jpeg image)

1a. HTTP client initiates TCP connection to HTTP server (process) at www.google.com

1b. HTTP server at host www.google.com waiting for TCP connection. "accepts" connection, notifying client

2. HTTP client sends HTTP *request message* (containing URL). Message indicates that client wants index.html

3. HTTP server receives request message, forms *response message* containing requested object, and sends response

time

Request/Response Timeline (cont.)

time

5. HTTP client receives response message containing html file, displays html. Parsing html file, finds 1 referenced jpeg object

6. Steps 1-5 repeated for jpeg object

HTTP Request Message

HTTP request message:

- ASCII (human-readable format)

request line (GET, POST, HEAD commands)

header lines

```
GET /index.html HTTP/1.1
Host: www.google.com
User-agent: Mozilla/4.0
Connection: close
Accept-language:fr
```

Carriage return, line feed indicates end of message

(extra carriage return, line feed)

HTTP Response Message

status line (protocol status code status phrase)

header lines

```
HTTP/1.1 200 OK
Connection close
Date: Tue, 04 Mar 2003 07:28:29 GMT
Server: GWS/2.0
Content-Length: 2824
Content-Type: text/html

data data data data data ...
```

data, e.g., requested HTML file

Naming: DNS

- Translate from www.google.com to 216.239.51.101
- Domain Name System:
  - Distributed database implemented in hierarchy of many name servers
  - Application-layer protocol host, routers, name servers to communicate to resolve names (address/name translation)

Name Servers

- No server has all name-to-IP address mappings
- Local name servers:
  - Each ISP, company has local (default) name server
  - Host DNS query first goes to local name server
- Authoritative name server:
  - For a host: stores that host's IP address, name
  - Can perform name/address translation for that host's name

DNS Example

Root name server:

- may not know authoritative name server
- may know *intermediate name server:* who to contact to find authoritative name server
- Results are *cached*