Outline

- Announcements
- Markup Languages
- Networks
- Student Presentations
- Akamai Case
Announcements

- Reading
  - Messerschmitt Ch 10.1 -10.2
  - American Airlines
Markup languages

- Click to add text
Documents and XML

- Enterprises create/store/exchange documents
- When using DBMS, documents are stored in their entirety
  - BUT their internal structure is not visible
- The WWW as a communication medium
  - Non-traditional database
  - Billions of documents stored/exchanged
  - Need for identifying/processing documents' internal structure
A **markup language** describes the structure of a document
- Based on tags
- **Tags** denote structural elements like sections, subsections, figures, etc

**Internationally standardized**, so application independent
Family lineage

SGML
- Standardized in mid 80s by ISO
- Purpose- and industry-specific extensions

HTML
- Introduced in Early 90s
- Both are standardized by W3C
- Emphasizes formatting and presentation of documents

XML
- Proposed in mid 90s
- Emphasizes structure of documents
- Both are standardized by W3C

XQuery, XPath
- Query languages for XML data (W3C standards)
Example: HTML

```
<html>
  <h1> Super Widget </h1>
  <h2> Widgets Incorporated</h2>
  <em> 123456789 </em>
  <br>
  <p> $300</p>
</html>
```
Networks
What are some examples of communications networks?

- Public Telephone Network
- Internet
- LANs (Local Area Networks)
What does a network do?

- **Transport data from one host to another**
  - Host allocation
  - Routing

- **Millions of users/applications/hosts share the same network**
  - Resource sharing
  - Congestion control
Network Architecture

Network architectures are layered
Each layer
- uses the services of the layers below
- To offer more advanced services to layer above

Allows layers to be designed independently
We will talk about 3 layers next...

- Network
- Link
- Physical
Link Layer

Make a Frame link out of a bit link
- Instead of endless sequence of 1s and 0s, we want distinct “packages” of data that are separate from each other

Say we want to send 2 Frames with data
- 01010101010111010 and 101010101011010
- Concatenate them and send them as a sequence?

How can the receiver tell where the new frame begins?

Solution: insert a special sequence at the start of frame: for example: 01111110
Link Layer (cont’d)

Also does error detection/correction

- Insert extra information that helps the receiver to determine if the data has been corrupted.
- Example: parity bit
  - Sender adds a 1 or zero to end of data so number of ones is always odd
    - 10011 or 10000
  - If receiver counts an even number of ones, than it knows the data was corrupted.
Error Detection

- **Simple parity** bit scheme problem?
  - Cannot detect if two or any even number of bits were modified during the transmission.

- **CRC (for cyclic redundancy code)**: more sophisticated error detection code
  - sending node: calculates the CRC bits (typically 32 bits) from the previous bits in the packet
  - receiving node: performs the same calculation and compares the result with the error detection bits in the packet;
  - if they differ, the receiving node knows that some error occurred and it discards the corrupted packet.
More Link Layer.. -- Ethernet

Want to allow multiple hosts to *share a link*

How do they avoid talking at the same time?

Don’t transmit if you hear another host transmitting

If there is a collision, stop wait a random amount of time, and try again

This is a **Medium Access Control (MAC)** Protocol
Ethernet Continued

How do the hosts on this Ethernet identify each other?

Each host (actually each interface)

- has a globally unique *MAC address*
- *Cannot be changed*
Ethernet Hub

Hub broadcasts packets on a link to all others
As if all hosts connected to single link
- We say it is a Single collision domain
Only one host can talk at a time
Routing in the Internet

Many feasible paths from source to destination.
Recall: three ways of locating things (Ch. 11)

Name
  • “John Smith”

Address
  • “1156 High St., Santa Cruz, CA”

Reference
  • “Postmaster of UCSC, Santa Cruz, CA”
Name services

1. name

Name service

2. address or reference

3. interaction

E.g. Map a host name (URL) to its IP address
Domain Names

IP addresses are inconvenient for people
  - e.g. 128.55.156.273
  - 32 bits hard to remember
  - 128 bits very hard to remember

Domain names
  - e.g. argus.eecs.berkeley.edu

  - Easier to remember than IP addresses

  - However, we need some way of mapping domain names to IP addresses.
Domain Name System (DNS)

EDU
Name Server
e.g. 128.X.X.X

BERKELEY
Name Server
e.g. 128.55.X.X

EECS
Name Server
e.g. 128.55.156.X

UCSC
Name Server
e.g. 128.114.X.X

“Wild cards”

SoE
Name Server
e.g. 128.114.48.X

e.g. 128.55.48.20
Hierarchy in Addresses vs. Names

Addresses hierarchical in topology
- Maximize “wild cards” and distribute address administration

Names hierarchical in administration
- Single administered organizations often distributed topologically (e.g. ibm.com)
Routing

Routing
- Updating the routing table
- Objective: each packet gets closer to destination

Packet forwarding
- Transmitting each packet on the appropriate output link
- Based on routing table
Routing Algorithms

Routers talk to each other to build their routing tables

“I am accepting Traffic to 114.211.1.X”

Wild Card
Routing Table has Wild Cards

Host A

Host B

Host C

HUB

<table>
<thead>
<tr>
<th>ROUTING TABLE</th>
<th>Link 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>114.211.1.X</td>
<td></td>
</tr>
<tr>
<td>200.261.19.X</td>
<td></td>
</tr>
</tbody>
</table>
Issues In Networking

- Sharing of Limited Resources
  - How Should A and B share a link with limited bit rate?

```
Source A

C bits per second

Source B

Destination A

Destination B
```

C bits per second
Issues In Networking

- Time Division Multiplexing
  - Gives each connection the use of the link a fixed fraction of time
  - Fixed fraction of resources reserved for each connection
  - Technology called *circuit switching*.

- Problem
  - When A is silent, A’s fraction of link goes unused.
Transport Protocols

The Internet is unreliable
- It will make a “best effort” to get your packet to its destination

Packets can be lost because of
- Congestion
- Link errors
- Routing problems
Transmission Control Protocol (TCP)

Retransmit mechanism for reliability

- Receiver sends acknowledgements to sender
- If a packet is lost, source fails to get ACK, and then retransmits.

Congestion control

- If congestion perceived (by lost packets)
- Source reduces its send rate
  - When loss, sender reduces send rate by half
  - Otherwise slowly increases
Network congestion

Traffic can overload links
- Failure of statistical multiplexing

Congestion must be limited in some fashion
Increasing portion of network traffic is resent packets.

Congestion instability

Network “capacity”

Offered traffic

Social optimum

Carried traffic
When networks are congested, certain sessions (Source-destination pairs) should reduce offered rates.

- Today all TCP sessions slow down when they detect packet losses.
- UDP sessions do not slow down.

What are some alternative strategies?

- Have those whose applications aren’t as sensitive slow down more?
  - How would we know which are less sensitive
We count in base 10 by powers of 10:

\[ 10^1 = 10, \quad 10^2 = 10 \times 10 = 100, \quad 10^3 = 10 \times 10 \times 10 = 1000, \]
\[ 10^6 = 1000000 \]

Computers count by base 2:

\[ 2^1 = 2, \quad 2^2 = 2 \times 2 = 4, \quad 2^3 = 2 \times 2 \times 2 = 8, \quad 2^{10} = 1024, \quad 2^{20} = 1048576 \]

So in computers, the following units are used:

<table>
<thead>
<tr>
<th>Unit</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 kilobyte (KB)</td>
<td>1,024 bytes</td>
</tr>
<tr>
<td>1 megabyte (MB)</td>
<td>1,048,576 bytes (1,024*1,024)</td>
</tr>
<tr>
<td>1 gigabyte (GB)</td>
<td>1,073,741,824 bytes</td>
</tr>
<tr>
<td>1 terabyte (TB)</td>
<td>1,099,511,627,776 bytes</td>
</tr>
<tr>
<td>1 petabyte (PB)</td>
<td>1,125,899,906,842,624 bytes</td>
</tr>
</tbody>
</table>
Akamai Case (cont’d)
Bottlenecks in the Internet architecture

- **First mile**
  - Web and application server performance
  - Router and Switch capacity.

- **Backbone**
  - Many routers for data to go through (17 to 20 routers)

- **Peering**
  - Dumping traffic on other NSPs

- **Last mile**
  - IAP’s router or modem capacity
  - Limited transmission rates
  - Older computers

2 major problems:
- loss of packets (information)
- slow connections
Freeflow

- Deployed in 1999

- Akamai Infrastructure
  - 13000 servers in 954 networks by 2001

- Customers
  - Large Commercial Websites
Competition

- Hosting firms (substitute)
  - Exodus
- Other CDNs
  - Sandpiper, Adero, Mirror Image
- Content Alliances
  - Akamai's competitors banded together to share networks
Akamai Case

- How did Akamai differentiate from its competitors after the dot-com recession of 2000?

- Was it a successful move? Was it a costly decision for Akamai? Why?
2001 Market Changes

Bad

- Dot-coms bust
- Customers leave
  - "churn rate goes to 22% per quarter"

Good

- Hosting firms go bust (exodus)
- Some CDN competitors go bust.
- Competing CDN alliances mired in problems
EdgeSuite

- Assemble and Present dynamic pages at edges rather than just deliver heavy objects

- Pricing – higher than old service

- Soon EdgeSuite dominated revenue