Foundations of the Web: TCP/IP and HTTP

Qiang Yang
Simon Fraser University
Thanks: Prof. Francis Lau (HKU), Ethan Cerami (NYU), Tom Anderson (U Washington)

Some Definitions

- Host -- computer, PDA, toaster, ...
- Link -- transmit bits
  - wire or wireless
  - broadcast or switched (or both!)
- Switch -- move bits between links
  - packet switching: stateless store&forward
  - circuit switching: stateful, cut through

Internet -- network of networks

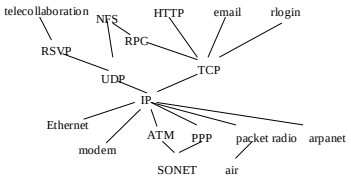
- network delivers packets (& locates nodes)
- router (gateway) moves packets between networks
- IP interoperability on top of any potential network or link layer
  - modem, Ethernet, token ring, cell phone, ADSL, cable modem, smoke signals, ...
- Minimum possible requirements on underlying networks

Protocols

- Protocol: agreement between two parties as to how information is to be transmitted
  - more valuable with more users
    - economic incentive to develop standards => lots and lots and lots of protocols
  - Standardize protocols vs. standardize interfaces?

Layering

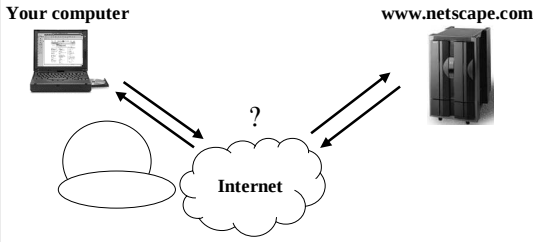
- Build complex services on top of simpler ones



OSI Model: 7 Protocol Layers

- Physical -- how to transmit bits
- Data link -- how to transmit frames
- Network -- how to route packets
- Transport -- how to send packets reliably
- Session -- how to tie groups together
- Presentation -- byte ordering, security
- Application -- everything else!

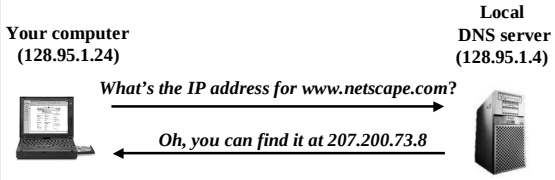
What happens when you click on a Web link?



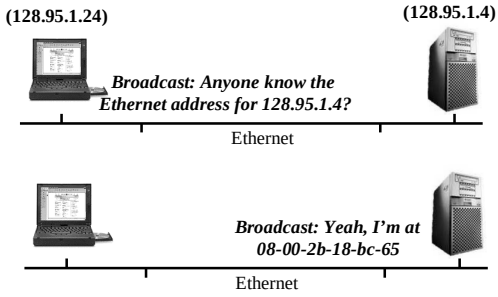
Different kinds of addresses

- Domain name (e.g. *www.netscape.com)*
  - Global, human readable
- IP Address (e.g. 207.200.73.8)
  - Global, works across all networks
- Ethernet (e.g. *08-00-2b-18-bc-65)*
  - Local, works on a particular network

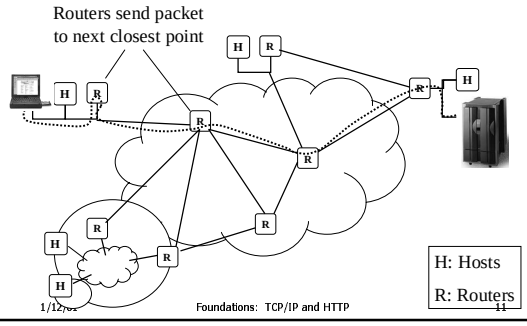
Finding the right IP address: Domain Naming System (DNS)



address: Address Resolution (ARP)



How does a packet get through the Internet?



TCP/IP

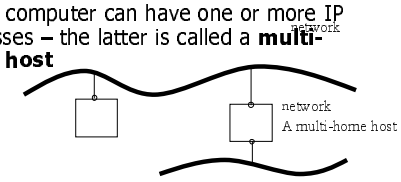
- They go hand in hand – a **suite**
- Major protocol suites:
  - NetWare IPX/SPX
  - **TCP/IP**
  - AppleTalk
- TCP offers to higher levels a **reliable end-to-end** delivery service
- The actual routing through the maze is done by IP
- Two issues: **addressing** and **routing**

OSI vs. DoD

Application		Process/Application
Presentation		
Session		
Transport		Host-to-host (Transport)
Network		Internet(work)
Data Link		Network
Physical		

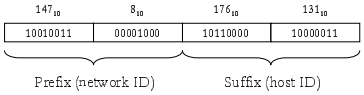
IP Addresses

- The prevailing version is **IP Version 4** (**IPv4**)
- **IPv6/IPng** is on the horizon
- Every protocol layer has its own addressing scheme
- A host computer can have one or more IP addresses – the latter is called a **multi-home host**



IP Address Structure

- An **IP address** has 32 bits, divided into 4 **octets** (bytes)
- IP address = prefix + suffix
- Prefix identifies a network; suffix a computer within the network
- An example:

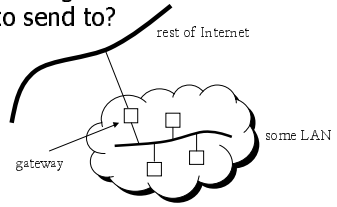


IP Address Structure (cont'd)

- Prefix = **network ID/number**; suffix = **host ID/number**
- **Host** means a computer (a network node) that runs applications and/or supports users
- Network IDs assigned by some Internet administrative body; host IDs by the administrator of the network (e.g., an ISP)

IP Address to Physical Address

- IP is not the lowest level
- IP packet arriving at a network: which computer to send to?



IPv6

- (Version 5 had been assigned to some experimental protocol)
- We will soon run out of IP addresses (is "IP on Everything" a fantasy?):
  - Growth in number of IP hosts
  - Inefficient use of (classful) IP addresses
- **Classless addresses** as a temporary solution
  - Could solve the second problem above

IPv6 – A Much Larger Space

- IPv6 uses 128 bits
- Number of hosts > 10^{36}
- IPv6 will not use the DDN, but a **hexadecimal notation** (base 16); for example:

7865:67E9:4510:37AF:ED07:0:1980:FFFF

1080::8:800:200C:417A

Why TCP?

- IP is unreliable
- TCP builds on the unreliable datagram service offered by IP to provide a reliable data delivery service to application programs
- Most applications cannot tolerate any or a combination of
  - Duplicated – Out-of-order – Littered – Lost (DOLL)
- So IP for efficiency (timeliness), and TCP for reliability (accuracy)

Major Functions of TCP

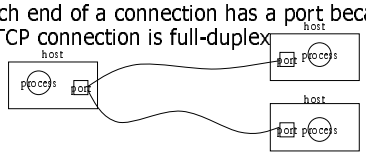
- **Streaming**: TCP data is organized as a stream of bytes
- **Reliable delivery** using sequence numbers and retransmissions
- **Network adaptation**: TCP dynamically learns the delay characteristics of a network and adjusts its operations accordingly
- **Flow control**: TCP manages data buffers and coordinate traffic so that slow receivers won't get run over by fast senders

TCP Addressing

- **Endpoint** = (**host**, **port**) where **host** is an IP address, and **port** is a special number on that host
- A port is attached to a **socket**
- A socket is a software channel for an application to read data from
  - Socket = "telephone"; port = "telephone number"
  - **Socket address** = (host, port)
- Two endpoints make a connection:
  - connection = (source endpoint, destination endpoint)
- A TCP connection is **two-way** and **full-duplex**

Ports and Connections

- Multiple connections can share a port, but at most one connection between one set of ports
- A port is for receiving data, not for sending
- Each end of a connection has a port because a TCP connection is full-duplex



TCP Ports

- A port is specified using 16 bits
- TCP ports are classified into
  - **Well-known ports** (0 through 1023)
  - **Registered ports** (1024 through 49151)
  - **Dynamic** or **private ports** (49152 through 65535)
- Well-known ports are controlled by the Internet Assigned Number Authority (IANA)

Examples of Well-Known Ports

- The official assigned numbers document ftp://ftp.isi.edu/in-notes/iana/assignments/port-numbers

7	ECHO	Echo
13	DAYTIME	Daytime
17	QUOTE	Quote of the Day
21	FTP	File Transfer Protocol
23	TELNET	Terminal Connection
25	SMTP	Simple Mail Transport Protocol
42	NAMESERVER	Host Name Server
53	DOMAIN	Domain Name Server
79	FINGER	Finger

HTTP Overview

- HTTP: HyperText Transfer Protocol
- Developed by Tim Berners Lee, 1990
- Enables web clients to request documents from web servers
- Stateless Protocol
  - each HTTP request is completely independent.
  - Web Servers do not retain any memory of related requests.

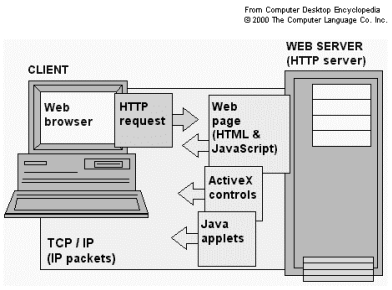
HTTP Client/Server

- Client/Server Architecture
- Client: web browser that requests a document.
  - Examples: Microsoft Internet Explorer, Netscape Navigator
- Server: web server that returns a document
  - Examples: Apache, Netscape Enterprise Server, etc.

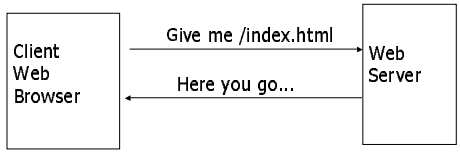
Request and Response

- A client's **request** consists of a request header which specifies the HTTP method (command) to be used and other things, and data (if any)
  - `GET /path/to/file/index.html HTTP/1.0`
- A server's **response** consists of a response header consisting of a **status code** indicating whether the transaction was successful, and data (if any)
  - "HTTP/1.0 404 Not Found" being the most famous response line; 404 is the status code
- Refer to the HTTP/1.0 spec or the HTTP/1.1 spec for details

HTTP



Http Client/Server



HTTP via Telnet

- You can run HTTP via the UNIX Telnet command.
- Instructions
  - Log into your UNIX account
  - telnet www.yahoo.com 80
  - GET /index.html
- Good method to learn the details of HTTP

Sample Telnet Session

```
bash-2.04$ telnet www.yahoo.com 80
Trying 216.32.74.50...
Connected to www.yahoo.akadns.net.
Escape character is '^]'.
GET /
HTTP/1.0 200 OK
Content-Length: 15582
Content-Type: text/html

<html><head><title>Yahoo!</title><base href=http://www.yahoo.com/><meta
   http-equiv="PICS-Label" content='(PICS-1.1
   "http://www.rsac.org/ratingsv01.html" l gen true for
   "http://www.yahoo.com" r (n 0 s 0 v 0 l))'></head><body><center><form
   action=http://search.yahoo.com/bin/search><map name=m><area
   coords="0,0,52,52" href=r/a1><area coords="53,0,121,52"
   href=r/p1><area coords="122,0,191,52" href=r/m1><area
...
```


Example HTTP Session

- Client requests the following URL: http://hypothetical.ora.com:80/
- Anatomy of the Request:
  - http:// HyperText Transfer Protocol; other options: ftp, mailto.
  - hypothetical.ora.com: host name
  - :80: Port Number. 80 is reserved for HTTP. Ports can range from: 1-65,535
  - / Root document

The Client Request

- Actual Browser Request:

```
GET / HTTP/1.1
Accept: image/gif, image/x-xbitmap,
  image/ jpeg, image/pjpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE
  5.01; Windows NT)
Host: hypothetical.ora.com
Connection: Keep-Alive
```


Anatomy of the Client Request

- GET / HTTP/1.1
  - Requests the root / document.
  - Specifies HTTP version 1.1.
  - HTTP Versions: 1.0 and 1.1 (more on this later…)
- Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
  - Indicates what type of media the browser will accept.

Anatomy of the Client Request

- Accept-Language: en-us
  - Browser's preferred language
- Accept-Encoding: gzip, deflate
  - Accepts compressed data (speeds download times.)
- User-Agent: Mozilla/4.0 (compatible; MSIE 5.01; Windows NT)
  - Indicates the browser type.

Anatomy of the Client Request

- Host: hypothetical.ora.com
  - Required for HTTP 1.1
  - Optional for HTTP 1.0
  - A Server may host multiple hostnames. Hence, the browser indicates the host name here.
- Connection: Keep-Alive
  - Enables "persistent connections". Faster performance (more later…)

Server Response

```
HTTP/1.1 200 OK
Date: Mon, 06 Dec 1999 20:54:26 GMT
Server: Apache/1.3.6 (Unix)
Last-Modified: Fri, 04 Oct 1996 14:06:11 GMT
Content-length: 327
Connection: close
Content-type: text/html
<title>Sample Homepage</title>
<img src="/images/oreilly_mast.gif">
<h1>Welcome</h2>Hi there, this is a simple web page.
   Granted, it may not be as elegant as some other web
   pages you've seen on the net, but there are some
   common qualities...
```


Anatomy of Server Response

- HTTP/1.1 200 OK
  - Server Status Code
  - Code 200: Document was found
  - We will examine other status codes shortly.
- Date: Mon, 06 Dec 1999 20:54:26 GMT
  - Date on the server.
  - GMT (Greenwich Mean Time)

Anatomy of Server Response

- Last-Modified: Fri, 04 Oct 1996 14:06:11 GMT
  - Indicates the time when the document was last modified.
  - Very useful for browser caching.
  - If a browser already has the page in its cache, it may not need to request the whole document again (more later…)

Anatomy of Server Response

- Content-length: 327
  - Number of bytes in the document response.
- Connection: close
  - Indicates that the server will close the connection.
  - If the client wants to send another request, it will need to open another connection to the server.

Anatomy of Server Response

- Content-type: text/html
  - Indicates the MIME Type of the return document.
  - Multi-Purpose Internet Mail Extensions
  - Enables web servers to return binary or text files.
  - Other MIME Categories:
    - audio, video, images, xml
  - Full list of MIME Types available in WebMaster in a Nutshell, pages 401-407.

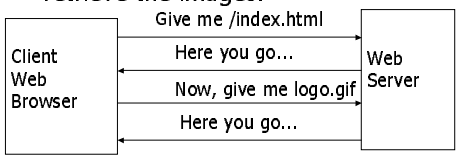
Anatomy of Server Response

The actual HTML document:

```
<title>Sample Homepage</title>
<img src="/images/oreilly_mast.gif">
<h1>Welcome</h2>Hi there, this is a
  simple web page.  Granted, it may not
be as elegant as some other web pages
  you've seen on the net, but there are
some common qualities…
```


Getting Images

- Once a browser receives an HTML page, it makes separate connections to retrieve the images.



HTTP 1.0 v. 1.1

- HTTP 1.0:
  - For each request, you must open a new connection with the server.
- HTTP 1.1
  - For each request, the default action is to maintain an open connection with the server.
  - Faster, Persistent Connections
  - Supported by most browsers and servers.

Keep-Alive

- Suppose a client accesses a page containing 10 inline images; to display the page completely would require 11 HTTP sessions
- Some browsers/servers support a feature called **keep-alive** which can keep the connection open until explicitly it is torn down

HTTP 1.0 v. 1.1

- HTTP 1.0: Get HTML Page plus Images
  - Open Connection: GET /index.html
  - Open Connection: GET /logo.gif
  - Open Connection: GET /button.gif
- HTTP 1.1: Get HTML Page plus Images
  - Open Persistent Connection: GET /index.html
  - GET /logo.gif
  - GET /button.gif

Server Responses

- Every server response includes three parts:
  - Response line: HTTP version number, three digit status code, and status message.
  - Header: Information about the server
  - Entity Body: The actual data.

Server Status Codes

- 100-199 Informational
- 200-299 Client Request Successful
- 300-399 Client Request Redirected
- 400-499 Client Request Incomplete
- 500-599 Server Errors

Some Important Status Codes

- 200: OK
  - Request was successful.
- 301: Moved Permanently
  - Server redirects client to a new URL.
- 404 Not Found
  - Document does not exist
- 500 Internal Server Error
  - Error within the Web Server
- All other status codes are noted in WebMaster in a Nutshell, pages 384-388

HTTP Proxy

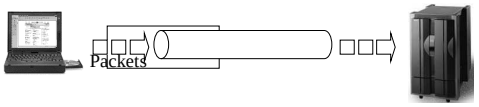
- An HTTP proxy is a program that acts as an intermediary between a client and a server
  - It receives requests from clients, and forwards those requests to the intended servers
  - Sometimes it serves the requests itself, e.g., when it has a copy of the requested resource
  - The responses pass back through it in the same way
  - Thus, a proxy has functions of both a client and a server

HTTP Proxy (cont'd)

- Requests use the complete URL of the resource being requested, instead of just the path:
  - GET http://www.somehost.com/path/file.html HTTP/1.0
- That way, the proxy knows which server to forward the request to

Network Metrics: Measurements

Network is a pipe connection two computers



Basic Metrics

- Bandwidth, delay, overhead, error rate and message size

Network metrics

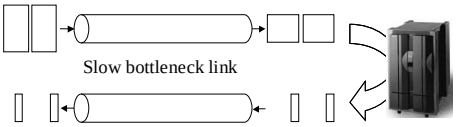
- Bandwidth
  - Data transmitted at a rate of R bits/sec
- Delay or Latency
  - Takes D seconds for bit to progagate down wire
- Overhead
  - takes O secs for CPU to put message on wire
- Error rate
  - Probability P that messsage will not arrive intact
- Message size
  - Size M of data being transmitted

How long to send a message?

- Transmit time T = M/R + D
  - 10Mbps Ethernet LAN (M=1KB)
    - M/R=1ms, D ~=5us
  - 155Mbps cross country ATM (M=1KB)
    - M/R = 50us, D ~= 40-100ms
- R*D is "storage" of pipe

How to measure bandwidth?

Measure how slow link increases gap between packets



How to measure delay?

Measure round-trip time