Life of a Packet

CS 640, 2015-01-22

Outline

- Recap: building blocks
- Application-to-application communication
- Process-to-process communication
- Host-to-host communication

Announcements

- Syllabus
- Should have received a welcome message on Piazza
- Read/watch posted materials before class; lots of in-class activities

Network Building Blocks

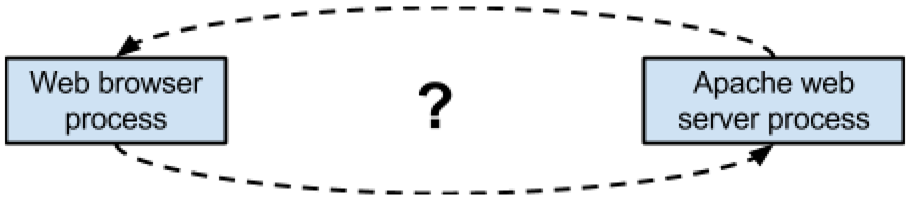
- Layers -- Internet model has 5 layers

Application
Transport
Network
Link
Physical

- Protocols -- concrete realizations of these layers
- Nodes and links
    - Nodes -- bridges, switches, routers, wireless access points, hosts (servers, desktops, laptops, mobile devices, "things")
    - Links -- wires (copper, optical fiber) or wireless (WiFi, cellular, bluetooth)
- Addresses -- domain names, IP addresses, MAC addresses

Scenario

- You want to view the New York Times webpage
    - Web browser running on a laptop connected to the CS department network ***[Show NYTimes website]***
    - NYTimes home page is an HTML file stored on a server in New York City



- *****Draw a picture that shows how the building blocks are put together to enable this scenario***
- Parallel scenario

    - You want a travel brochure for New York City
    - Paper form from NYC Visitors Bureau to request the brochure you want
    - NYC Visitors Bureau has travel brochures they will send

Applications

- Web browser process
    - You provide the URL; browser requests the file(s) and renders the page
    - *Requirement*: a way to issue a request and receive a response
    - Browser process does not need to know what server holds the file, or how to reach that server
- Web server process
    - Receives user requests; retrieves files from the file system; sends files to user
    - File may be large → *requirement*: a way to send the file in pieces
    - Server process does not need to know how to reach the client, it just needs to know how to process a request and provide a response
- Browser and server need some agreement on the structure of the request and response
    - HyperText Transfer Protocol (HTTP)
    - Protocol -- defines a standard interface for communication, i.e., a standard message format
    - Request includes method (GET), URL (/), host (nytimes.com), and User-Agent
    - Reply includes response code (200 OK), content-length (193756 bytes), content-type (text/html), and data
        - Differentiate meta information (header) from the file (payload)
    - Parallel scenario
        - Form to request brochure is request
        - Brochure is payload; a letter saying brochure is enclosed is the reply header
    - ***[Show HTTP reply/response in Firebug]***

Operating System

- *Requirement:* a way to determine the path to the server
    - *****Who should determine the path?*** -- options: web browser/server, <u>operating system, switches/routers</u>
- *Requirement:* reliable delivery mechanism
    - *****Who should provide this reliable delivery mechanism?*** -- options: web browser/server, physical devices, <u>operating system</u>
- Operating system provides capabilities that are needed by many applications
    - Reliable communication channel
    - Determine "next hop" to reach destination
    - Resolve human-readable server name to numeric server identifier

Reliable Communication Channel

- Formed between two processes
- One process needs to initiate the channel (client), and one process needs to be waiting for a channel to be initiated (server) → *requirement:* formal setup (and teardown) steps
- Many different processes on a machine -- we specifically want to talk to the web server

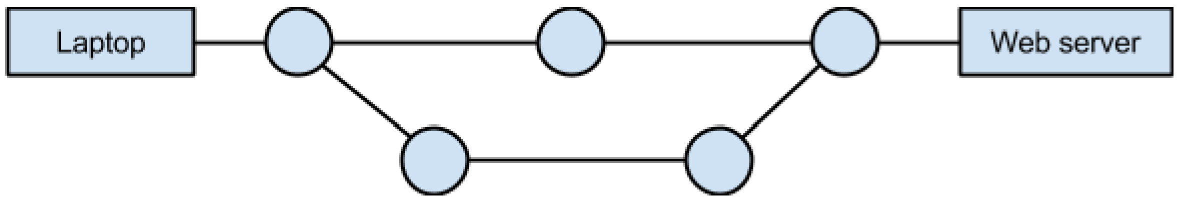
    - *****How do we identify the process we want to talk to?*** -- options: process id, executable name, <u>port number</u>
- Identify a specific process with a “port number” -- processes that use the same application protocol usually use a specific port, e.g., port 80 for HTTP
- *****How do we make sure data is not lost? Not reordered?*** -- options: <u>acknowledgements, sequence numbers</u>
- Transmission control protocol (TCP) -- standard for reliable communication channel
    - TCP setup: “three-way handshake” -- SYN (synchronize), SYN+ACK (acknowledge), ACK
    - TCP teardown: “four-way handshake” -- FIN (finish), ACK, FIN, ACK
    - Reliability: “sequence number” assigned to each byte of data, ACK for the highest number byte you got without having any gaps
    - Retransmit if you don’t get an ACK after some amount of time
- Need to communicate sequence numbers and control messages
    - Add another header -- called “encapsulation”
    - Parallel scenario: number each page of the request form and each page of the brochure
- Operating system provides “sockets” interface for applications; we’ll talk about this next class
- Now that we have process-process communication, we need host-to-host communication

Numeric Addressing
- How do we identify the host we want to talk to? -- gave the browser a URL with domain name
- *****What are the advantages of using a domain name for NYTimes server?*** -- easy to remember
- *****What are the disadvantages?*** -- might want more than one server, could be many names for the same thing (“New York Times”, “NY Times”), could have multiple languages (“New York”, “Nueva York”), strings are harder for computers to deal with
- ****Why is it important for everyone to agree on the system we use for identifying a host?**
    - Parallel scenario: what would happen if some people addressed letters using street addresses and some people addressed letters using latitude and longitude coordinates?
- Internet Protocol (IP) addresses -- common system of numeric addresses
    - Identifies a host and encodes a location (not necessarily physical location)
    - Version 4: 2^{32} = 4.3 billion addresses; not enough
    - Version 6: $2^{128} = 3.4 \times 10^{38}$; a billion billion IP addresses available for every square mm of the Earth's surface
    - IP addresses are assigned in “blocks”
        - 128.105.0.0/16 is assigned to CS department
        - 170.149.172.0/22 is NYTimes
    - IP is another header we add -- remember encapsulation?
        - Put both destination IP address, and source IP address -- source is needed so server knows who to send reply back to
        - Parallel scenario: put request form or brochure in an envelope, and put destination street address and return address on envelope

- Domain names are easy to remember, numerical addresses are not
    - Parallel scenario: easy to say you want to contact NYC Tourism Bureau, but postal address is hard to remember
    - Want a service where you can look it up -- phonebook
- Domain name system
    - Converts a hostname into a numerical address
    - Hierarchy of servers that provide the service -- ask DNS server in CS department, campus, ISP, root name server for .com, NYTimes server
- Next: how do we use the address to get the message to its destination?

Forwarding

- If two hosts are directly connected, then send over physical link
    - Parallel scenario: If you live across the street from the NYC Tourism Bureau, just walk across the street and drop-off your envelope (save yourself 49¢!)
- Most communicating hosts are at least "one hop" away
    - Parallel scenario: Your envelope needs to go through at least one post office



- Each node does not know the full path, only the "next hop"
    - Parallel scenario:
        - You don't know exactly which post offices will handle your envelope
        - Your postal carrier just knows how to get your envelope to the local post office
        - The local post office just knows how to get your envelope to the regional office
        - Etc.
- *****How do you determine the next hop?*** -- send to everyone you're physically connected to, <u>keep some type of mapping</u>
- Forwarding table
    - A mapping between addresses and next hops
    - Every host, router, and switch has a forwarding table
    - Forwarding table does not contain an entry for every possible address -- leverage the hierarchical nature of IP addresses
        - Parallel scenario:
            - If your envelope is going to someone in your block, you put it in their mailbox; otherwise you give it to your postal carrier to go the post office
            - Local post office will sort envelopes based on city+state+zip:
              envelopes going to same neighborhood are given to a carrier for delivery;
              envelopes going to another neighborhood go to a city-wide post office
            - City-wide post office also sorts envelopes based on city+state+zip:
              envelopes going to a neighborhood in the same city each go to a specific local post office;
              envelopes going to another city each go to a specific city post office

        - Bigger post offices have a larger number of post offices they know about and distribute envelopes to
    - ***[Show routing table on laptop]***
- There are tools for us to discover the path taken by a packet
    - ***[Run traceroute on laptop to show full path]***

Link/Physical Layer

- Add one more header (Ethernet) with another address (MAC address) -- skip over this for now
- OS sends packet through driver to network interface
    - Packets nowadays get copied directly to NIC from memory without the OS needing to copy each byte -- direct memory access (DMA)
- NIC encodes and sends packet as an electrical signal