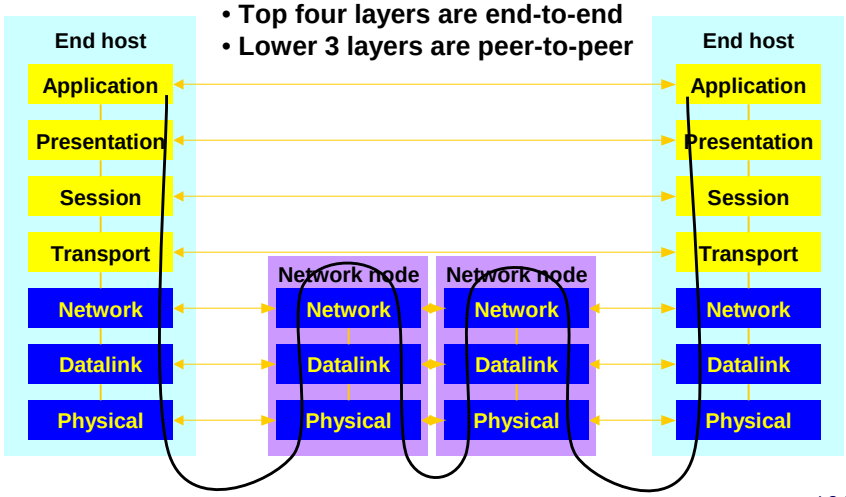
Internet Architecture

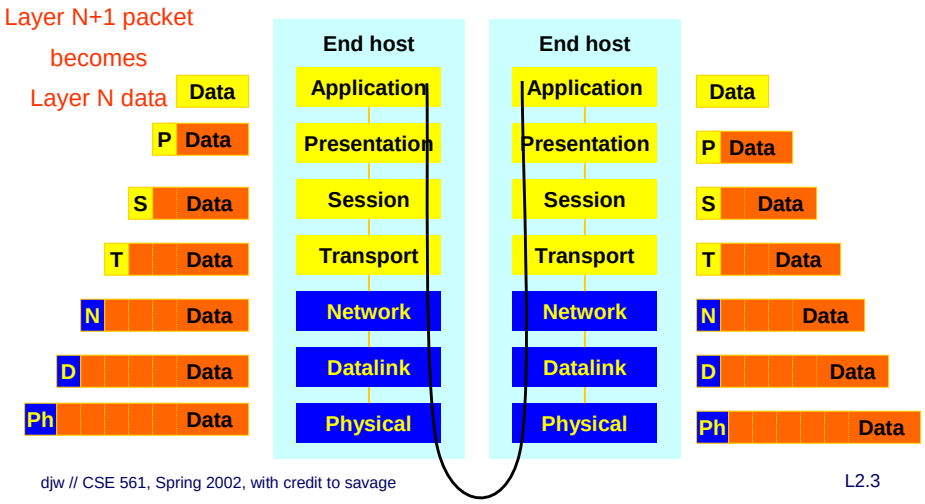
CSE 561 Lecture 2, Spring 2002
David Wetherall

The OSI layering Model

- Top four layers are end-to-end
- Lower 3 layers are peer-to-peer



Layer encapsulation

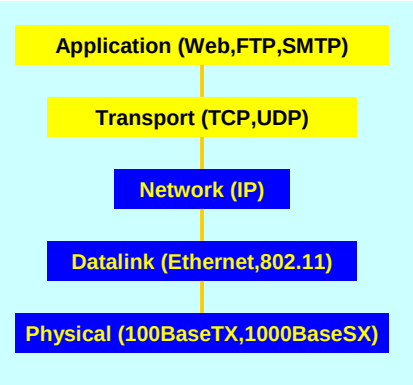


Layer Encapsulation (2)

- Typical Web packet

- Notice that layers add overhead
  - Space (headers), effective bandwidth
  - Time (processing headers, peeling the onion), latency

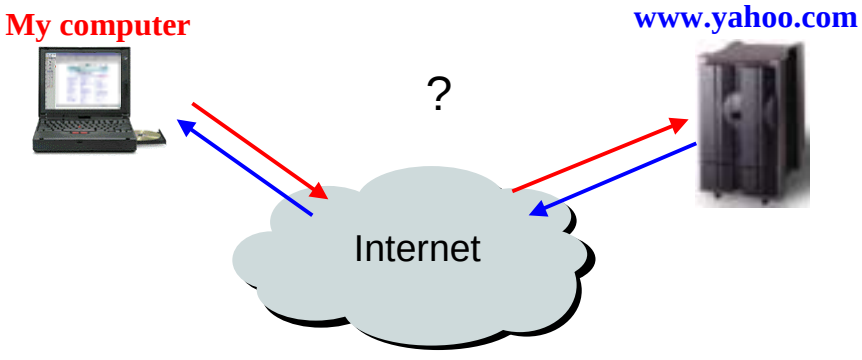
The Internet layering model



- So-called "hourglass" model
  - One network layer protocol
  - Significant diversity at other layers
- No presentation or session layers
- Implementations more important than interfaces

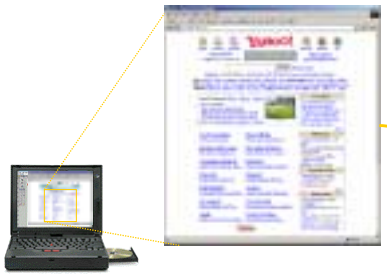
Layering by example…

- **ROUGHLY**, what happens when I click on a Web page from UCSD?



Application layer (HTTP)

- Turn click into HTTP request



```
GET http://www.yahoo.com/r/mp HTTP/1.1
Host: www.yahoo.com
Connection:keep-alive
…
```


Application layer? Name resolution (DNS)

- Where is www.yahoo.com?

My computer
(132.239.9.64)

What's the address for www.yahoo.com

Oh, you can find it at 64.58.76.177

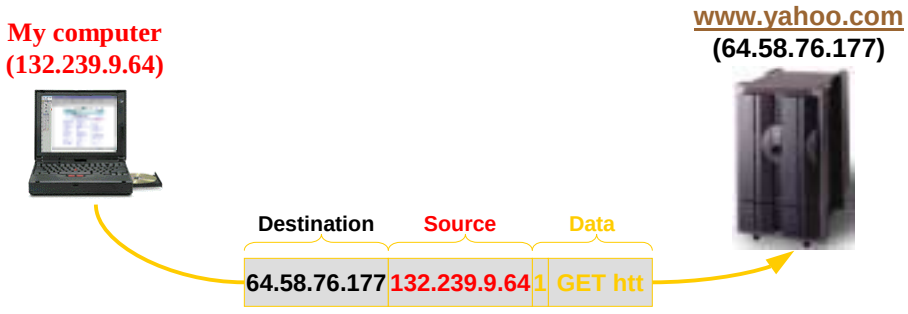
Local DNS server
(132.239.51.18)

Transport layer (TCP)

- Break message into packets (TCP segments)
- Should be delivered reliably & in-order

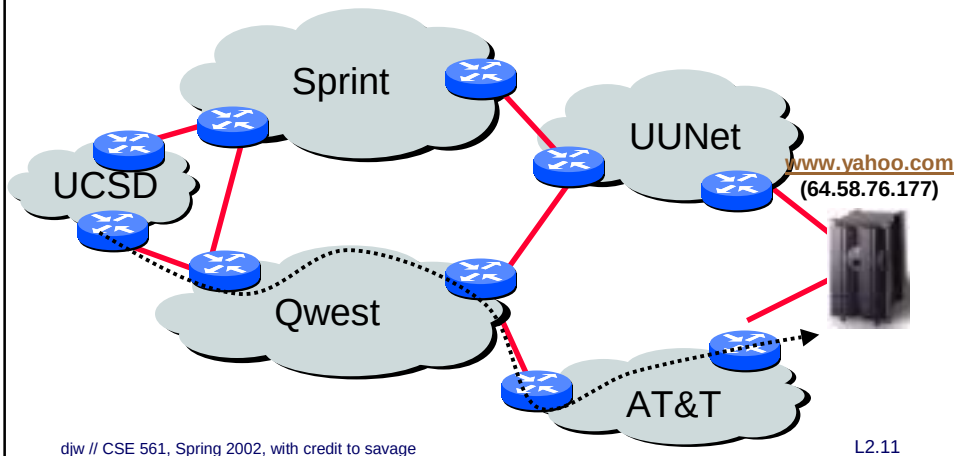
Network layer: IP Addressing

- Address each packet so it can traverse network and arrive at host



Network layer: IP Routing

- Each router forwards packet towards destination

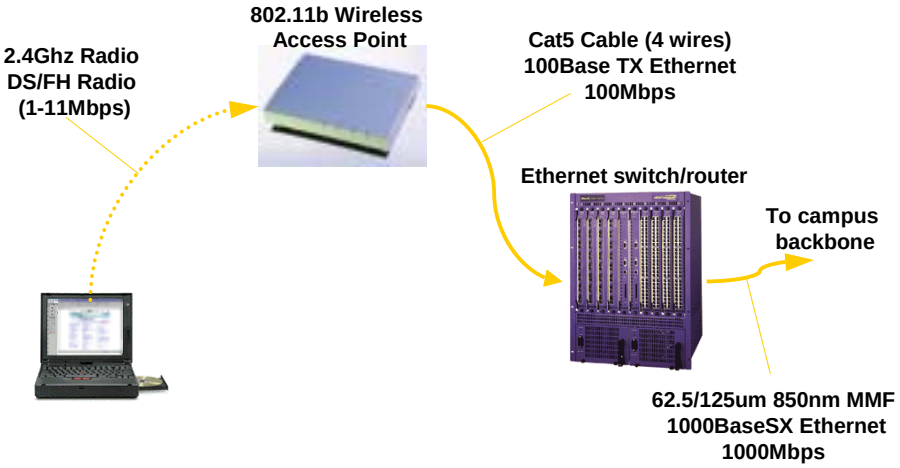


Datalink layer (Ethernet)

- Too boring for a picture (sorry)

- Break message into frames
- Media Access Control (MAC)
- Send frame

Physical layer



Clark88: Design Philosophy of the DARPA Internet Protocols

- Unique paper
  - Not many papers explaining the motivation and reasoning that went into the design of systems that we take for granted
- Note that this was written 15 years after the project began
  - And the paper itself is already 13 years old!
- The setting
  - Multiple research and military networks
  - How do we connect them so that they can talk to each other?
  - Hard to imagine, but this was *before* LANs

Some brief history

- DARPAnet (circa 1968*)
  - DARPA funded a lot of University mainframes
  - Wanted to saved money by sharing them
  - Bob Taylor, Larry Roberts to make it happen
- Internetworking/Internet (circa1978*)
  - SATNet, ARPAnet, Packet radio, Ethernet built
  - Each with a different network functionality
  - Painful to make them communicate
  - Cerf, Postel and Cohen split network functionality into single shared protocol – inventing internetworking

* Plug: "Where Wizards Stay Up Late" by Hafner and Lyon is the best account of early Internet History I've seen.

Meta-points…

- The Internet was designed
  - There is no natural law that says TCP/IP, network routing, etc.. had to look the way it does now
  - It could well have been done differently
  - It was guided by a particular set of design goals
- The Internet evolves
  - The Internet today is not the same Internet as 1988, 1973
  - TCP/IP have changed considerably over the years
  - We're using IPv4, with IPv6 (maybe) being deployed
  - The design goals are different now than yesterday

Primary Goal: Connect Stuff

- "Effective technique for multiplexed utilization of existing interconnected networks"
  - **Minimal** assumptions about underlying networks
    - No support for broadcast, multicast, real-time, reliability
    - Extra support could actually get in the way (X.25 example)
  - Packet switched, store and forward
    - Matched application needs, nets already packet switched
    - Enables **efficient resource sharing**/high utilization
  - "Gateways" interconnect networks
    - Routers/Switches in today's nomenclature

Why is this hard?

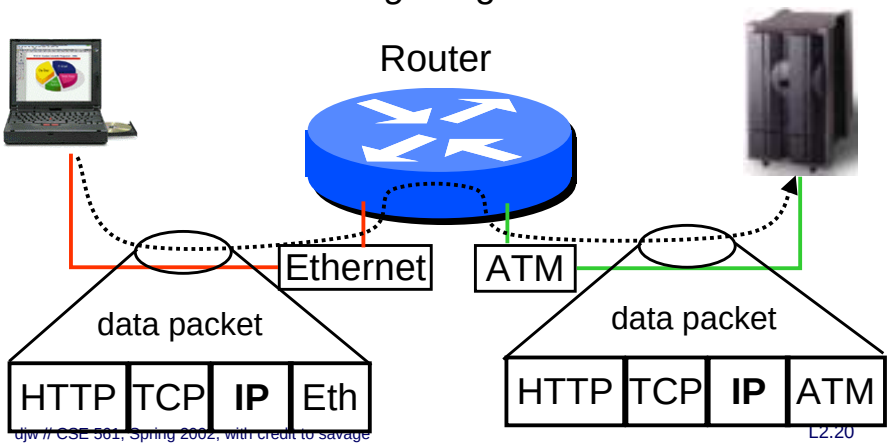
- **Heterogeneity**
  - Addressing
    - Each network media has a different addressing scheme
  - Bandwidth
    - Modems to terabits
  - Latency
    - Seconds to nanoseconds
  - Packet size
    - Dozens to thousands of bytes
  - Loss rates
    - Differ by many orders of magnitude
  - Service guarantees
    - Send and pray vs reserved bandwidth

Internetwork with a common network layer

- All nets communicate using a common format
  - Internet: IP over everything
  - To talk across networks, you send IP packets
  - Internal to a network, can use whatever you want
    - Ethernet, ATM, etc.
- Alternative: translate packets between different networks
  - Convert Ethernet to ATM
  - Convert IP to OSI CLNP
  - X.25 to NetBEUI

Heterogeneity: the power of IP

Separate physical networks communicate to form a *single* logical network



Goal #2: Survivability

- Internet
  - Assume anything can fail between two end points
    - Fate-sharing (put state with entity that requires it)
    - Direct consequence of end-to-end argument
  - Smart end-points, dumb routers, and packet switching
    - No need to manage connection state in routers; no replication
    - Key benefit is that it's proved easier to scale
- By contrast: POTS (the *other* global network)
  - Ultra reliable switches with self-healing
    - Hardware switch over in the middle of a phone call

Survivability Implications

- End points maintain all essential state
  - Routers are stateless ("soft state")
  - End points responsible for recovering from failures
- Host machines are trusted
  - Have to rely upon hosts to implement the protocols correctly
    - For performance as well as correctness
  - Easy to be malicious
    - Ex: source addresses (everything in an IP packet) are trusted (IP spoofing)
- Can be difficult to determine source of failures
  - Not much feedback from network back to end point
  - Makes performance optimizations more difficult
- Quality of Service
  - Hard to control resource allocations at a higher level

Goal #3: Types of Service

- Originally single protocol (NCP->TCP)
  - One set of service semantics not right for all applications
  - Split IP and TCP, and built other services
- Common denominator: IP
  - Best effort datagram service (send and pray)
- Other services built on top
  - Reliable data stream (TCP)
  - Unreliable message service (UDP)
  - Real-time delivery: network support?
  - Multicast: network support?

Key concepts and discussion

- Packet switching vs. circuit switching, virtual circuits
- Fate sharing
- Soft-state and flows
- The IP hourglass
- How have the design goals changed?
  - User empowerment vs. ISPs
  - Trust, accountability, multiple competing parties, …
- See:
  - Clark and Blumenthal, “Rethinking the Design of the Internet: end to end arguments vs. the brave new world”

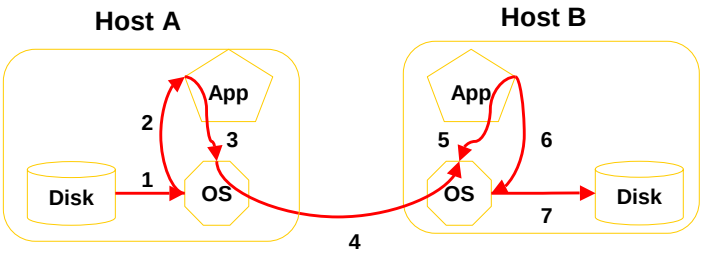
Saltzer84: End-to-End Argument

- **Key question**: Where should functionality be placed in the network architecture?
- **End-to-end argument**
  - Functionality should be implemented at a lower layer iff it can be **correctly** and **completely** implemented there
  - Incomplete versions of a function can be used as a performance enhancement, but not for correctness
- Early, and still relevant, example
  - ARPAnet provided reliable link transfers between switches
  - Packets could still get corrupted on host-switch link, or inside of the switches
  - Hence, still need reliability at higher layers

Reliable File Transfer

- From server disk over network to client disk
- Many places where errors can be introduced
  - Disk can introduce bit errors
  - Host I/O bus can introduce bit errors
  - Packets can be corrupted, dropped, reordered at any node
- Conclusion
  - Still need integrity checks on entire file, at application level, not per packet or per hop
  - Impossible to design “perfect” layers because perfect requires support from higher layers

Example: Reliable File Transfer



- Where can data be corrected?
- How to tell if data has been corrupted?
- Is there any value in lower-layer reliability?

Performance Optimizations

- Functionality at lower layer can enhance performance
  - Not required for correct operation
  - Can be required for reasonably efficient operation
- Back of the envelope
  - N hops (average hops on Internet route = 15 hops)
  - Prob(corrupted packet per link) = p
  - Prob(packet lost end to end)
    - p = 0.0001% -> Prob(e2e loss) = 0.0015%
    - p = 1% -> Prob(e2e loss) = 14%
- Tradeoff:
  - Higher layers have more information about service needs
  - Lower layers have more information about network capabilities

E2E Discussion

- E2E and network transparency
  - Extensibility?
- Engineering tradeoffs versus rules
  - Finding the endpoints
  - Performance tradeoffs
- What belongs in the network?
  - Multicast?
  - Firewalls, NAT boxes?
  - Web proxy caches?