CSC 450 COMPUTER NETWORKS

Lecture 2 - Protocol Stacks

Last Lecture: What is the Objective of Networking?

- Enable communication between applications on different computers
  - Web
  - Peer to Peer
  - Audio/Video Conferencing

- Must understand application needs/demands (next Lecture)
  - Traffic data rate
  - Traffic pattern (bursty or constant bit rate)
  - Traffic target (multipoint or single destination, mobile or fixed)
  - Delay sensitivity
  - Loss sensitivity

Last Lecture:
Lots of Functions Needed

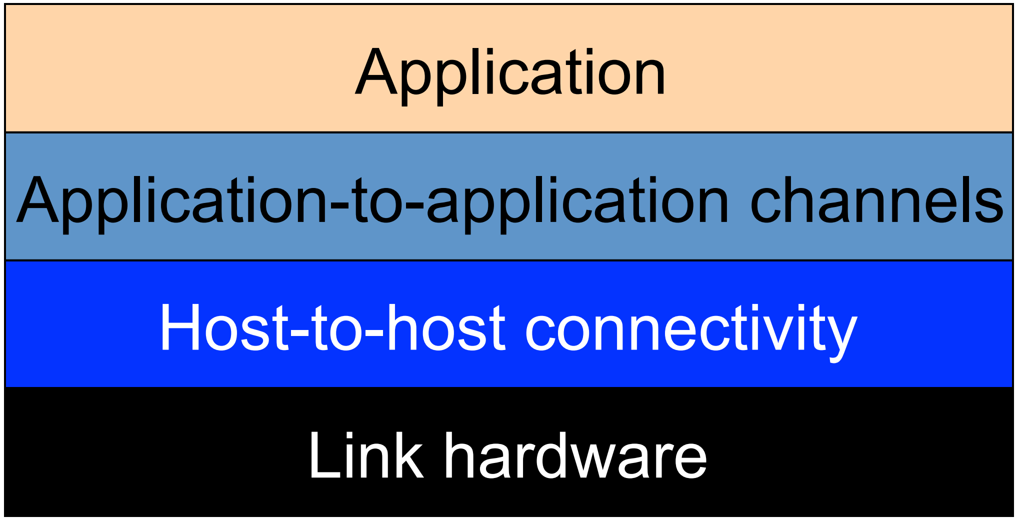
- Link
- Multiplexing
- Routing
- Addressing/naming (locating peers)
- Reliability
- Flow control
- Fragmentation
- Etc….

Today's Lecture

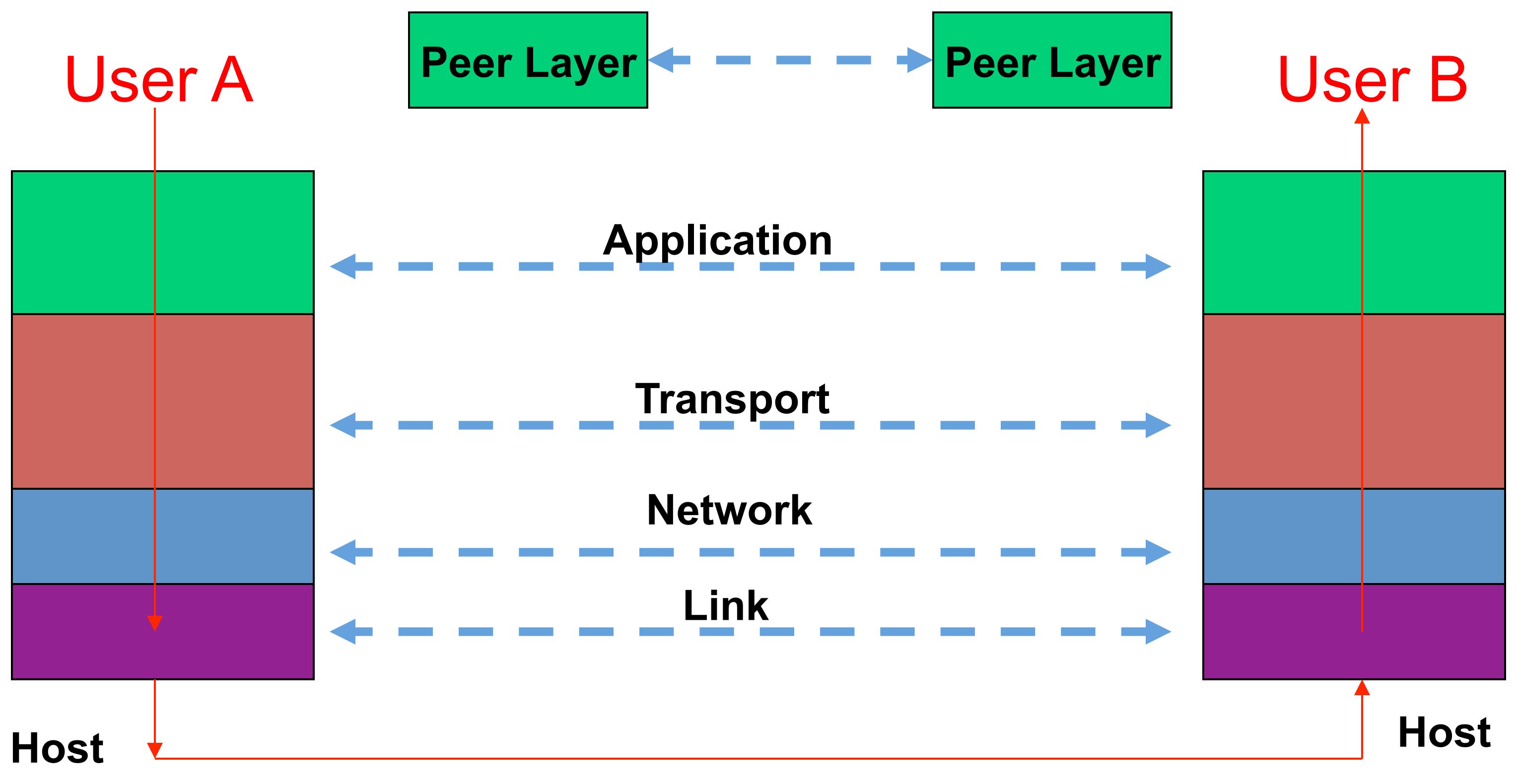
- Layers and protocols
- Design principles in internetworks

What is Layering?

- Modular approach to network functionality
- Example:



What is Layering?



Modular approach to network functionality

Layering Characteristics

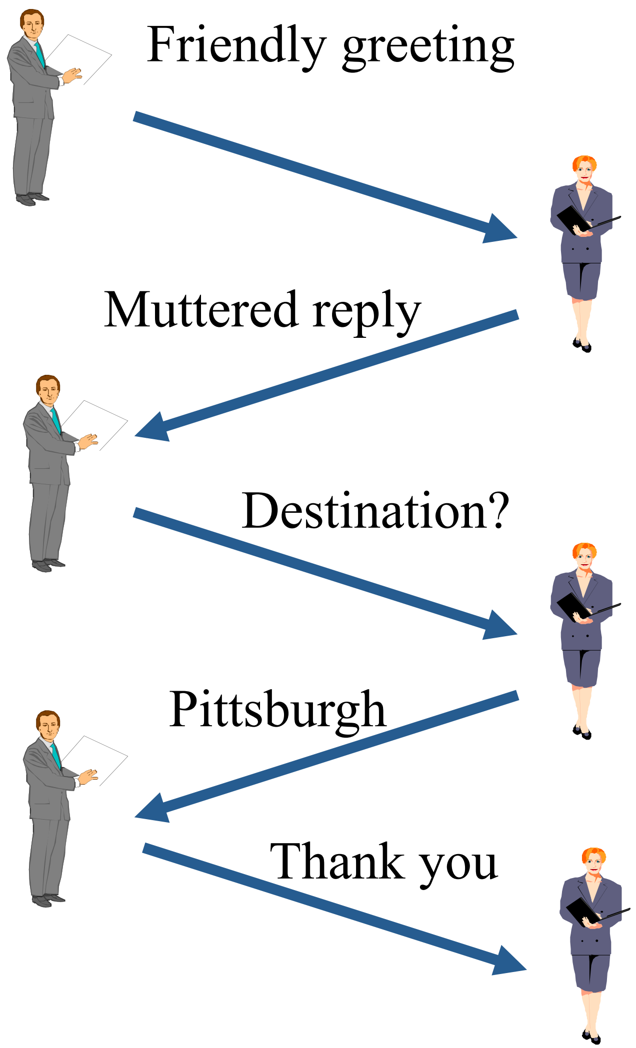
- Each layer relies on services from layer below and exports services to layer above
- Interface defines interaction with peer on other hosts
- Hides implementation - layers can change without disturbing other layers (black box)

Is Layering Harmful?

- Layer N may duplicate lower level functionality (e.g., error recovery)
- Layers may need same info (timestamp, MTU)
- Strict adherence to layering may hurt performance
- Some layers are not always cleanly separated.
  - Inter-layer dependencies in implementations for performance reasons
  - Some dependencies in the standards (header checksums)
- Interfaces are not really standardized
  - It would be hard to mix and match layers from independent implementations, e.g., windows network apps on unix (w/out compatibility library)
  - Many cross-layer assumptions, e.g. buffer management

What are Protocols?

- An agreement between parties on how communication should take place
- Module in layered structure
- Protocols define:
  - Interface to higher layers (API)
  - Interface to peer (syntax & semantics)
    - Actions taken on receipt of a messages
    - Format and order of messages
    - Error handling, termination, ordering of requests, etc.
- Example: Buying airline ticket



The Internet Engineering Task Force

- Standardization is key to network interoperability
  - The hardware/software of communicating parties are often not built by the same vendor → yet they can communicate because they use the same protocol

- Internet Engineering Task Force
  - Based on working groups that focus on specific issues

- Request for Comments
  - Document that provides information or defines standard
  - Requests feedback from the community
  - Can be "promoted" to standard under certain conditions
    - consensus in the committee
    - interoperating implementations

Other Relevant Standardization Bodies

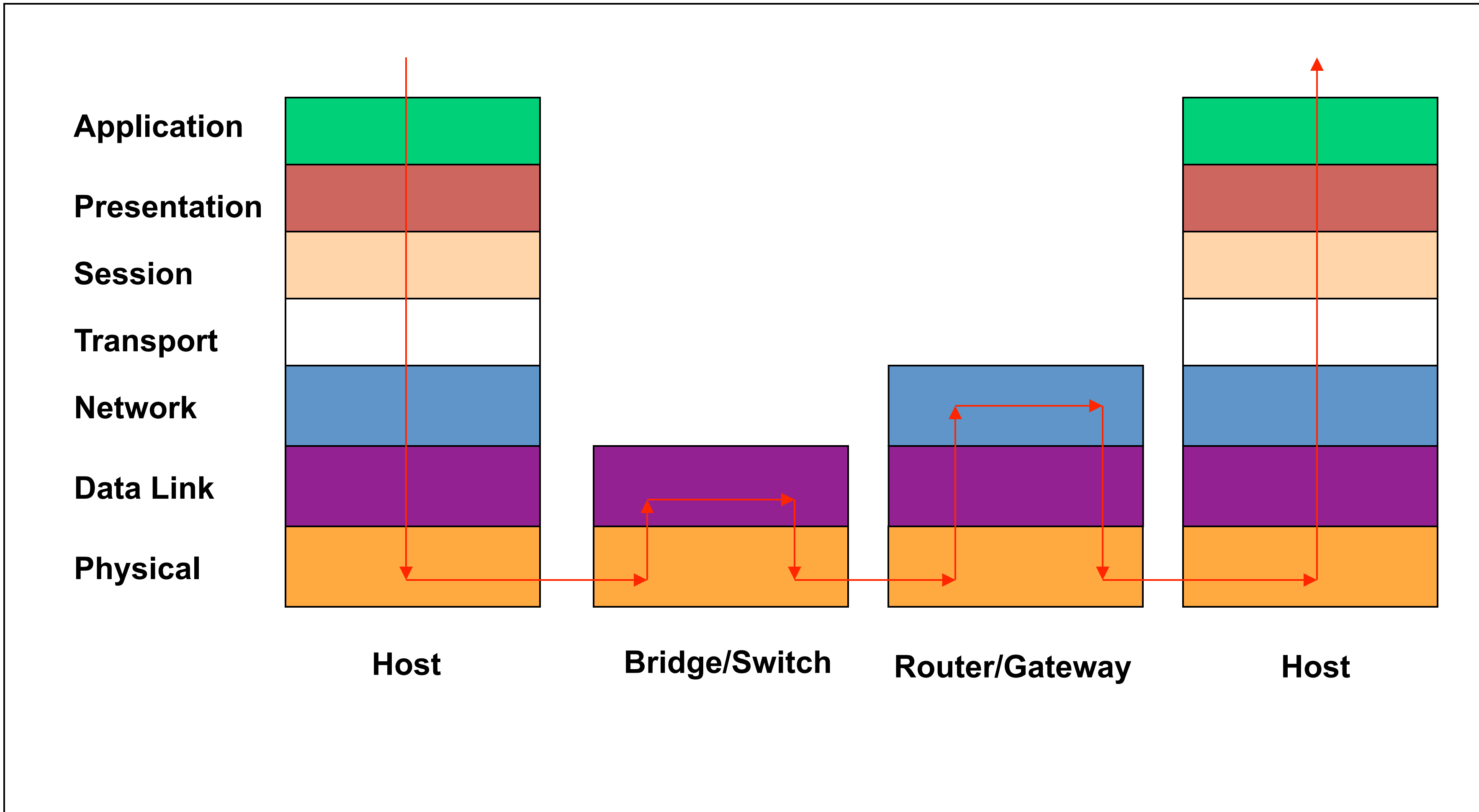
- ITU-TS - Telecommunications Sector of the International Telecommunications Union.
  - government representatives (PTTs/State Department)
  - responsible for international "recommendations"
- T1 - telecom committee reporting to American National Standards Institute.
  - T1/ANSI formulate US positions
  - interpret/adapt ITU standards for US use, represents US in ISO
- IEEE - Institute of Electrical and Electronics Engineers.
  - responsible for many physical layer and datalink layer standards
- ISO - International Standards Organization.
  - covers a broad area

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 end2end
- Session: how to tie flows together
- Presentation: byte ordering, security
- Application: everything else

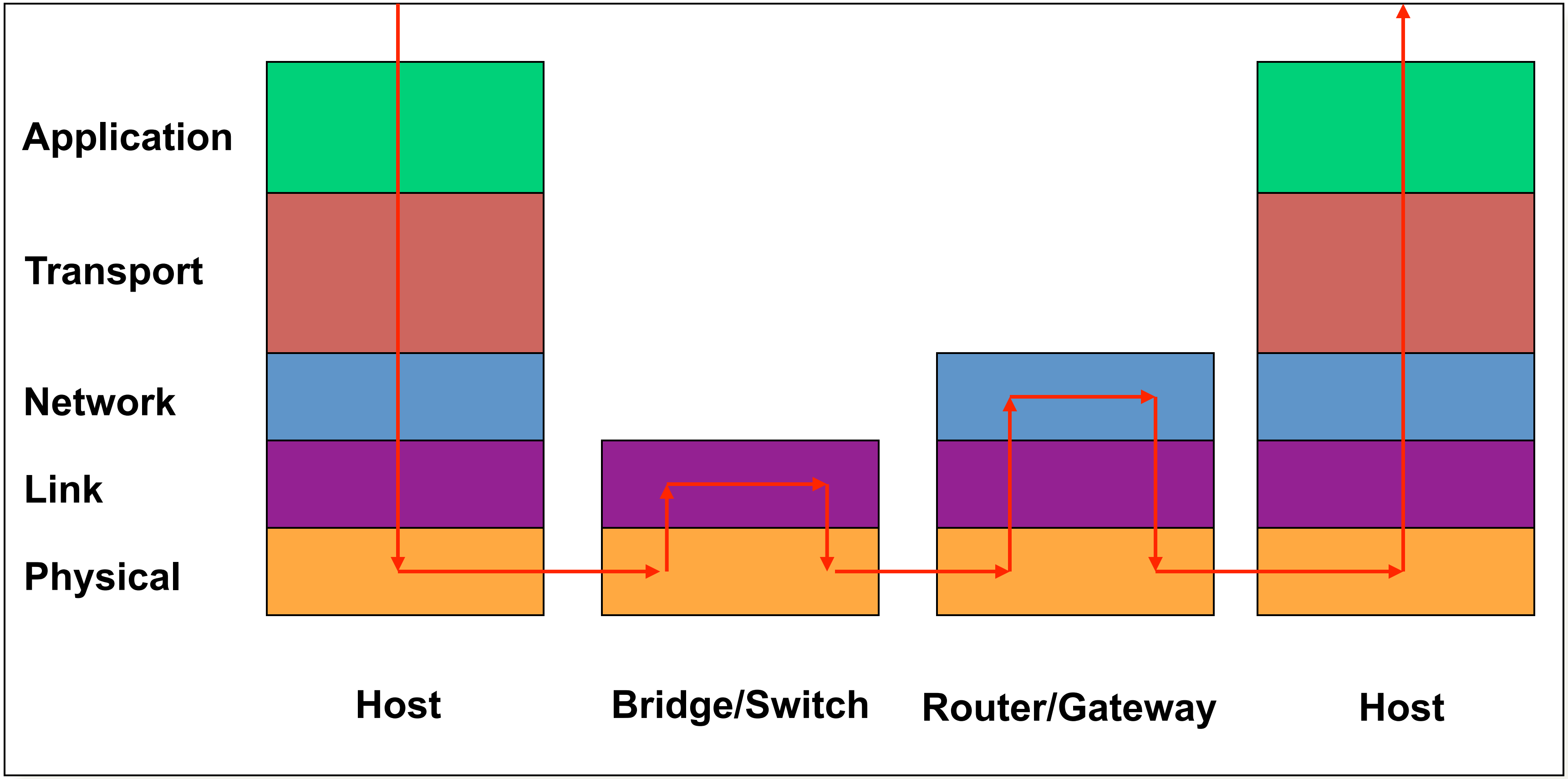
- TCP/IP has been amazingly successful, and it's not based on a rigid OSI model. The OSI model has been very successful at shaping thought

OSI Layers and Locations

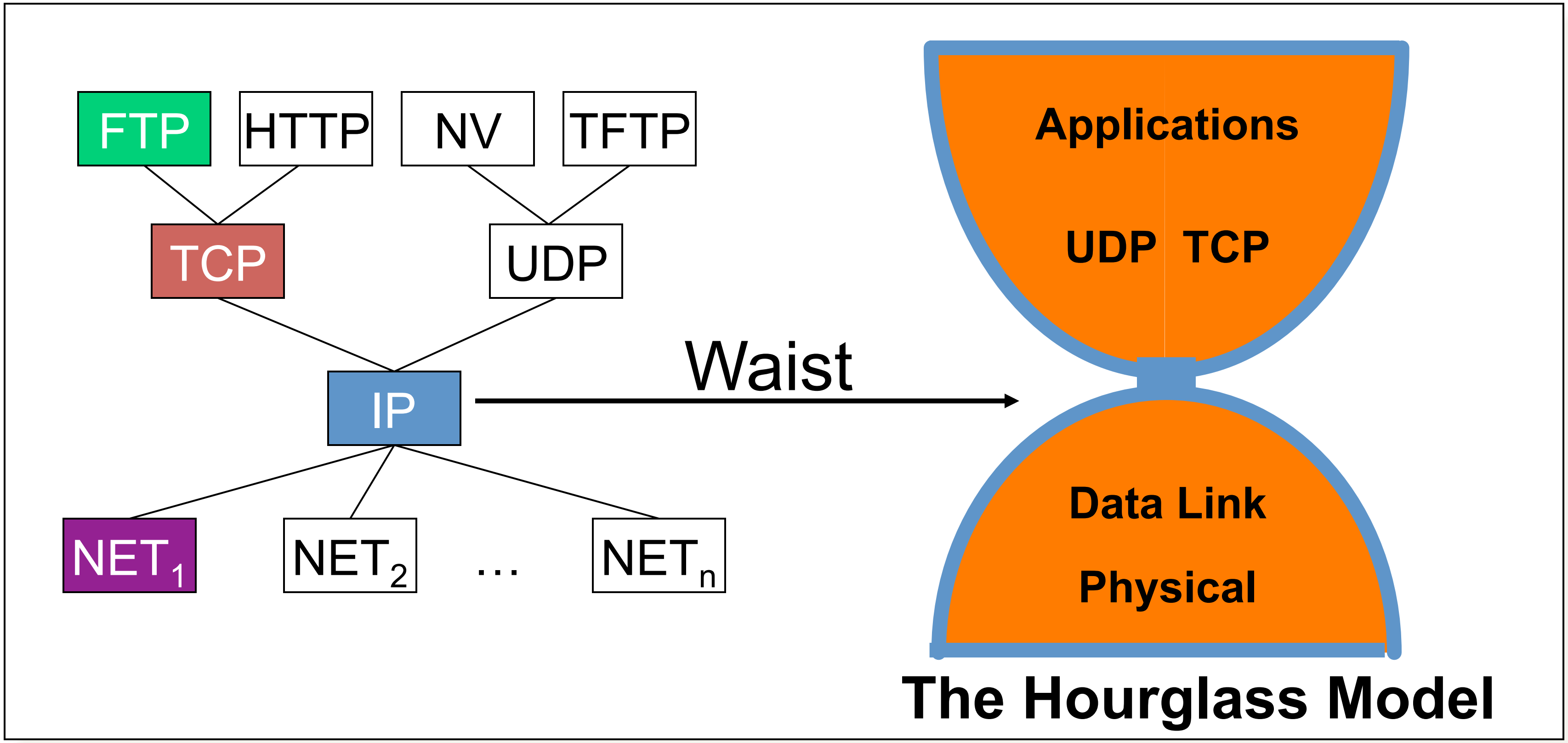


IP Layering

- Relatively simple

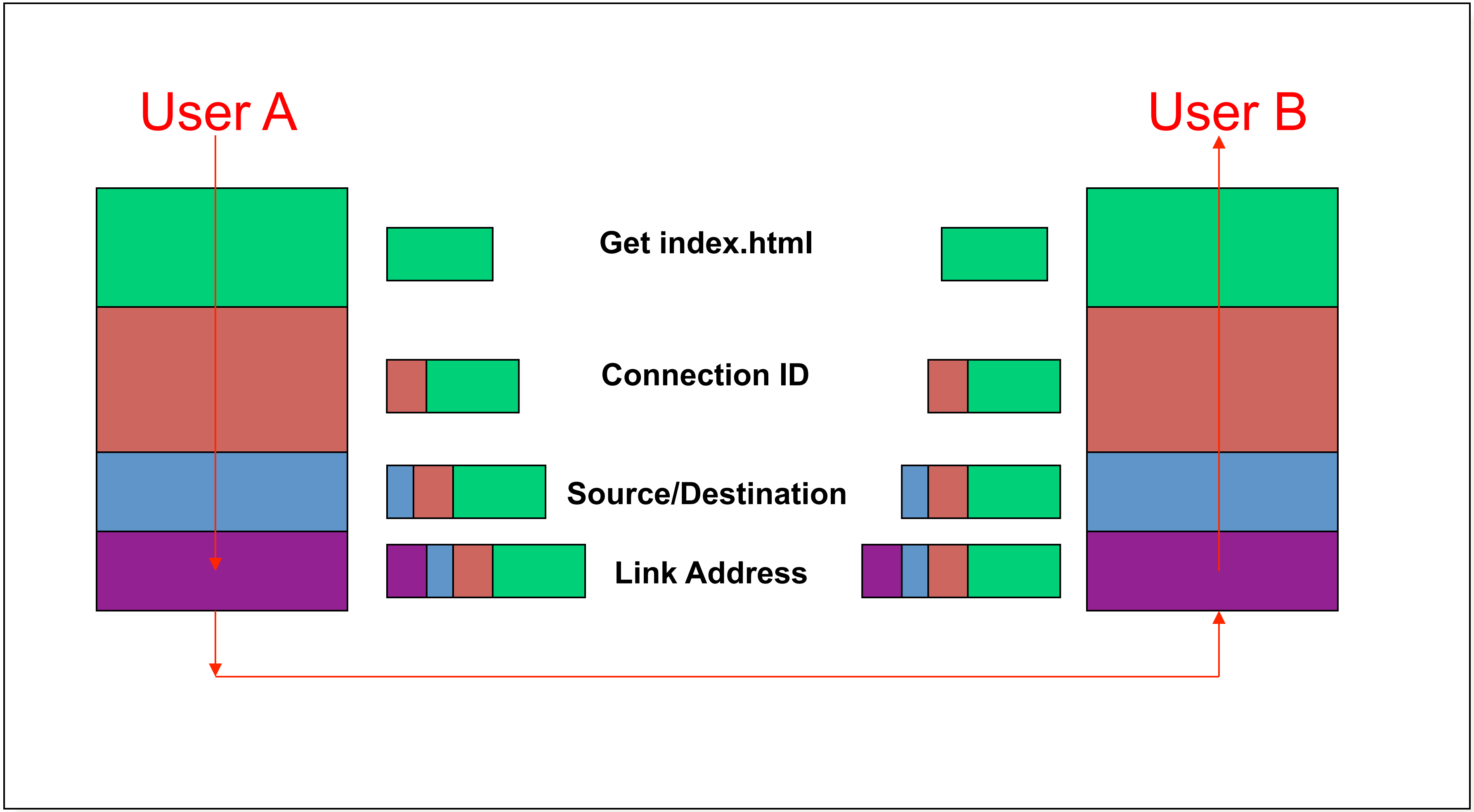


The Internet Protocol Suite



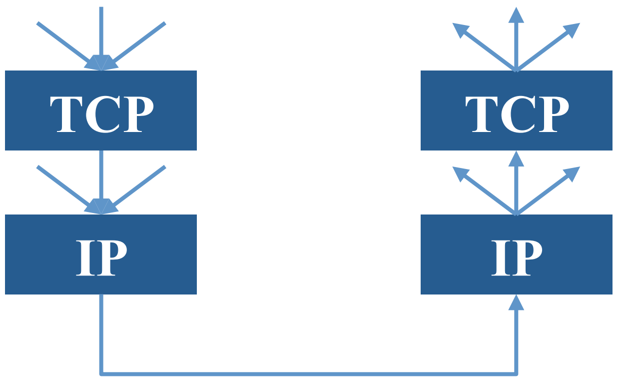
The waist facilitates interoperability

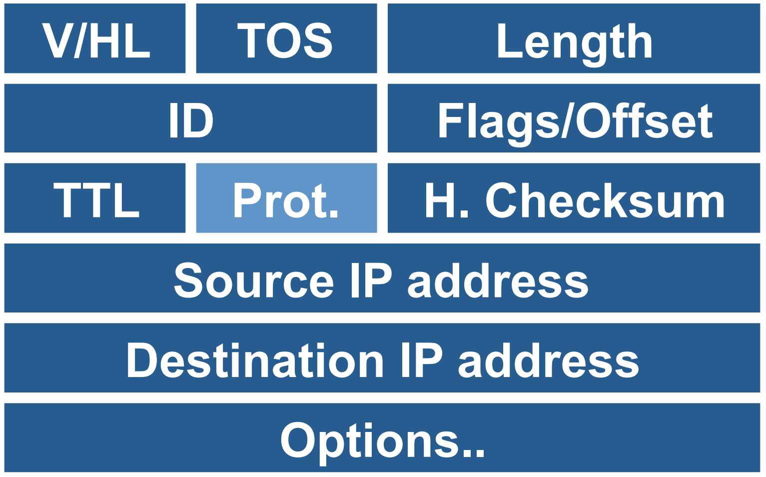
Layer Encapsulation



Multiplexing and Demultiplexing

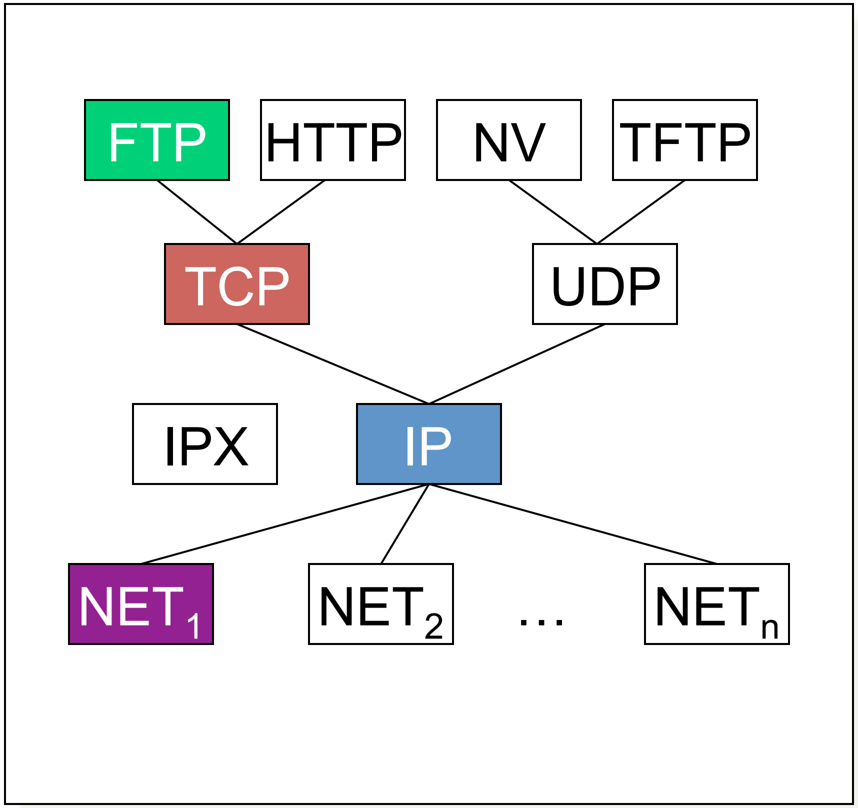
- There may be multiple implementations of each layer.
  - How does the receiver know what version of a layer to use?
- Each header includes a demultiplexing field that is used to identify the next layer.
  - Filled in by the sender
  - Used by the receiver
- Multiplexing occurs at multiple layers. E.g., IP, TCP, …

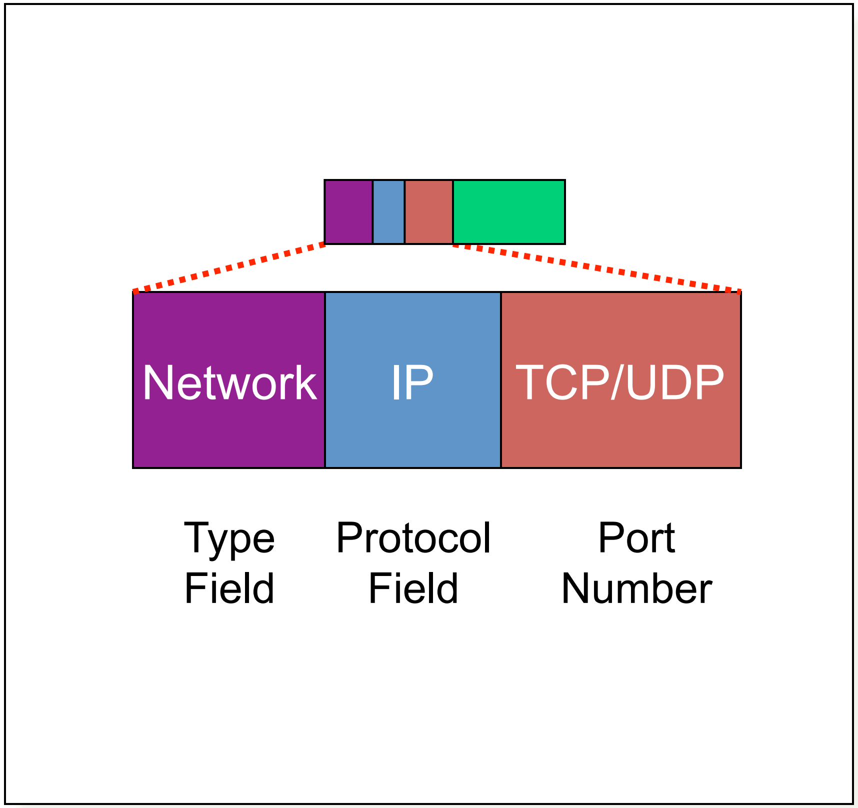




Protocol Demultiplexing

- Multiple choices at each layer





Today’s Lecture

- Layers and protocols
- Design principles in internetworks

Goals [Clark88]

0 **Connect existing networks**

initially ARPANET and ARPA packet radio network

1. Survivability

ensure communication service even in the presence of network and router failures

2. Support multiple types of services
3. Must accommodate a variety of networks
4. Allow distributed management
5. Allow host attachment with a low level of effort
6. Be cost effective
7. Allow resource accountability

Priorities

- The effects of the order of items in that list are still felt today
  - E.g., resource accounting is a hard, current research topic
- Let's look at them in detail

Goal 0: Connecting Networks

- How to internetwork various network technologies
  - ARPANET, X.25 networks, LANs, satellite networks, packet networks, serial links…
- Many differences between networks
  - Address formats
  - Performance – bandwidth/latency
  - Packet size
  - Loss rate/pattern/handling
  - Routing

Challenge 1: Address Formats

- Map one address format to another?
  - Bad idea → many translations needed
- Provide one common format
  - Map lower level addresses to common format

Challenge 2: Different Packet Sizes

- Define a maximum packet size over all networks?
  - Either inefficient or high threshold to support
- Implement fragmentation/re-assembly
  - Who is doing fragmentation?
  - Who is doing re-assembly?

Gateway Alternatives

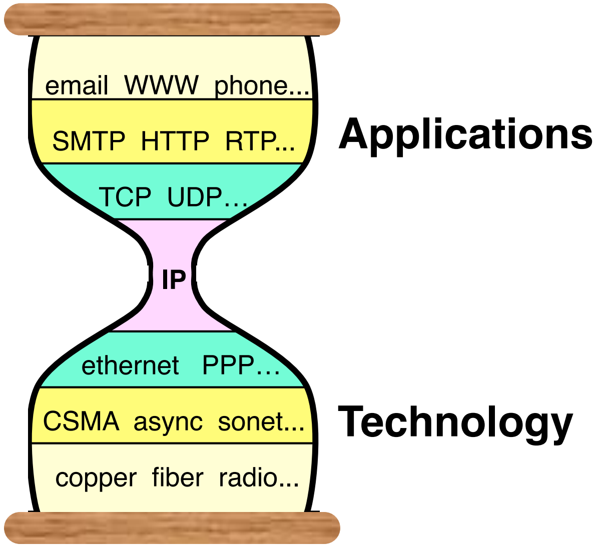
- Translation
  - Difficulty in dealing with different features supported by networks
  - Scales poorly with number of network types (N^2 conversions)
- Standardization
  - "IP over everything" (**Design Principle 1**)
  - Minimal assumptions about network
  - Hourglass design

IP Standardization

- Minimum set of assumptions for underlying net
  - Minimum packet size
  - Reasonable delivery odds, but not 100%
  - Some form of addressing unless point to point
- Important non-assumptions:
  - Perfect reliability
  - Broadcast, multicast
  - Priority handling of traffic
  - Internal knowledge of delays, speeds, failures, etc
- Also achieves Goal 3: Supporting Varieties of Networks

IP Hourglass

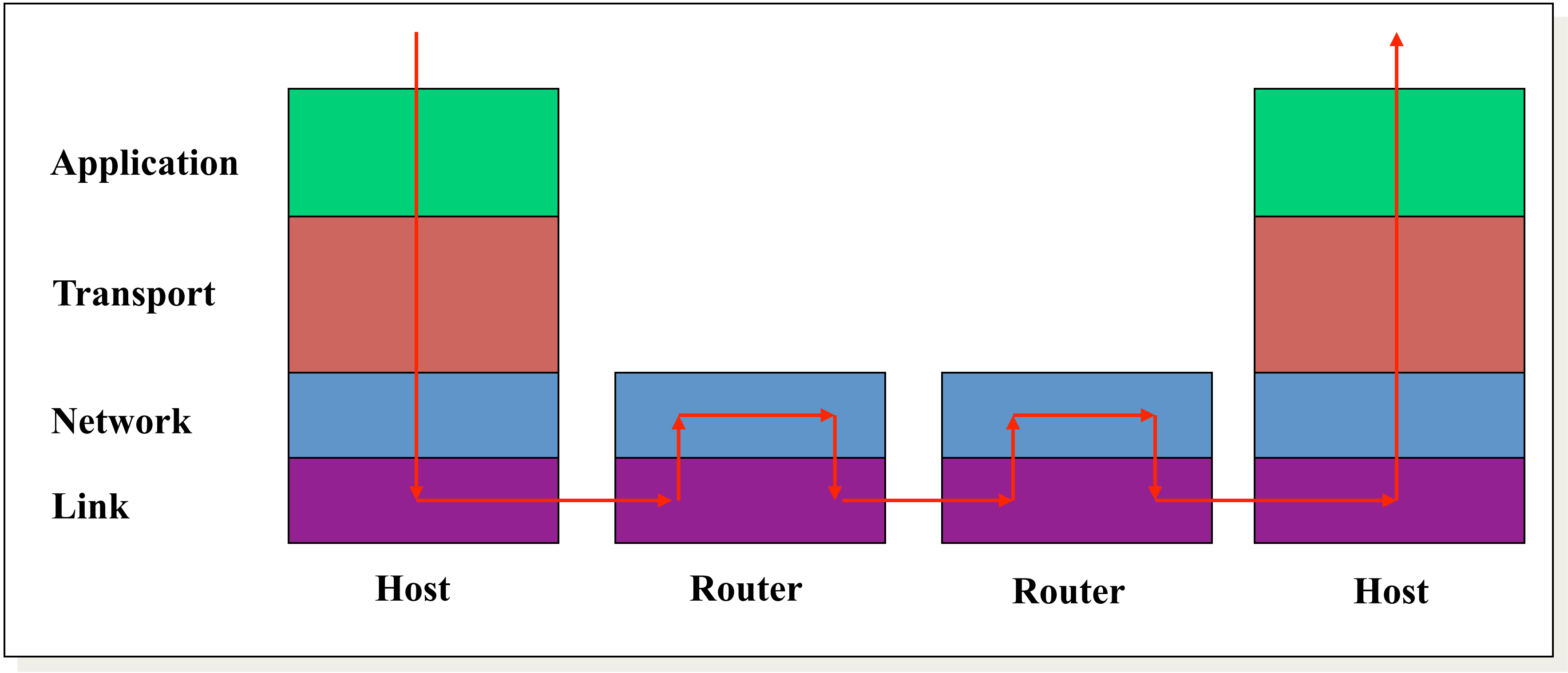
- Need to interconnect many existing networks
- Hide underlying technology from applications
- Decisions:
  - Network provides minimal functionality
  - “Narrow waist”



Tradeoff: **No assumptions, no guarantees.**

IP Layering (Principle 2)

- Relatively simple
- Sometimes taken too far



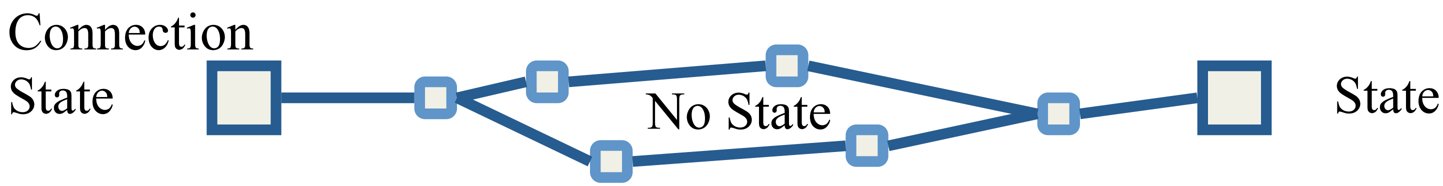
Goal 1: Survivability

- If network is disrupted and reconfigured…
  - Communicating entities should not care!
  - No higher-level state reconfiguration

- How to achieve such reliability?
  - Where can communication state be stored?

	Network	Host
Failure handing	Replication	"Fate sharing"
Switches	Maintain state	Stateless
Host trust	Less	More

Principle 3: Fate Sharing



- Lose state information for an entity if and only if the entity itself is lost.
- Examples:
  - OK to lose TCP state if one endpoint crashes
    - NOT okay to lose if an intermediate router reboots
  - Is this still true in today's network?
    - NATs and firewalls

- Tradeoffs
  - Survivability: Heterogeneous network → less information available to end hosts and Internet level recovery mechanisms
  - Trust: must trust endpoints more

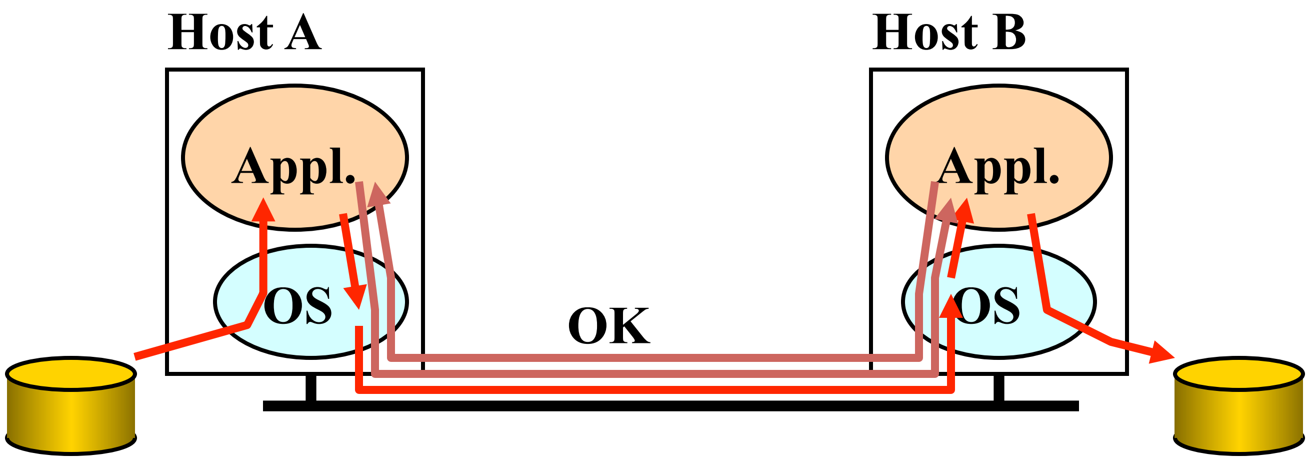
Principle 4: Soft-state

- Soft-state
  - Announce state
  - Refresh state
  - Timeout state
- Penalty for timeout – poor performance
- Robust way to identify communication flows
  - Possible mechanism to provide non-best effort service
- Helps survivability

Principle 5: End-to-End Argument

- Deals with where to place functionality
  - Inside the network (in switching elements)
  - At the edges
- Argument
  - There are functions that can only be correctly implemented by the endpoints – do not try to completely implement these elsewhere
  - Guideline not a law

Example: Reliable File Transfer



- Solution 1: make each step reliable, and then concatenate them
- Solution 2: end-to-end check and retry

E2E Example: File Transfer

- Even if network guaranteed reliable delivery
  - Need to provide end-to-end checks
  - E.g., network card may malfunction
  - The receiver has to do the check anyway!
- Full functionality can only be entirely implemented at application layer; no need for reliability from lower layers
- Is there any need to implement reliability at lower layers?

Discussion

- Yes, but only to improve performance
- If network is highly unreliable
  - Adding some level of reliability helps performance, not correctness
  - Don’t try to achieve perfect reliability!
  - Implementing a functionality at a lower level should have minimum performance impact on the applications that do not use the functionality

Examples

- What should be done at the end points, and what by the network?
  - Reliable/sequenced delivery?
  - Addressing/routing?
  - Security?
  - What about Ethernet collision detection?
  - Multicast?
  - Real-time guarantees?

Goal 2: Types of Service

- **<u>Principle 6</u>**: network layer provides one simple service: best effort datagram (packet) delivery
  - All packets are treated the same

- Relatively simple core network elements
- Building block from which other services (such as reliable data stream) can be built
- Contributes to scalability of network

- No QoS support assumed from below
  - In fact, some underlying nets only supported reliable delivery
    - Made Internet datagram service less useful!
  - Hard to implement without network support
  - QoS is an ongoing debate…

Types of Service

- TCP vs. UDP
  - Elastic apps that need reliability: remote login or email
  - Inelastic, loss-tolerant apps: real-time voice or video
  - Others in between, or with stronger requirements
  - Biggest cause of delay variation: reliable delivery
    - Today's net: ~100ms RTT
    - Reliable delivery can add *seconds*.
- Original Internet model: "TCP/IP" one layer
  - First app was remote login…
  - But then came debugging, voice, etc.
  - These differences caused the layer split, added UDP

Goal 4: Decentralization

- **Principle 7:** Each network owned and managed separately
  - Will see this in BGP routing especially

- **Principle 7’:** Be conservative in what you send and liberal in what you accept
  - Unwritten rule
- Especially useful since many protocol specifications are ambiguous
- E.g. TCP will accept and ignore bogus acknowledgements

The “Other” goals

5. Attaching a host
   - Host must implement hard part ☹ → transport services
     - Not too bad

6. Cost effectiveness
   - Packet overhead less important by the year
   - Packet loss rates low
   - Economies of scale won out
   - Internet cheaper than most dedicated networks

- But…

7. Accountability

- Huge problem

- Accounting
  - Billing? (mostly flat-rate. But phones have become that way also - people like it!)
  - Inter-ISP payments
    - Hornet's nest. Complicated. Political. Hard.

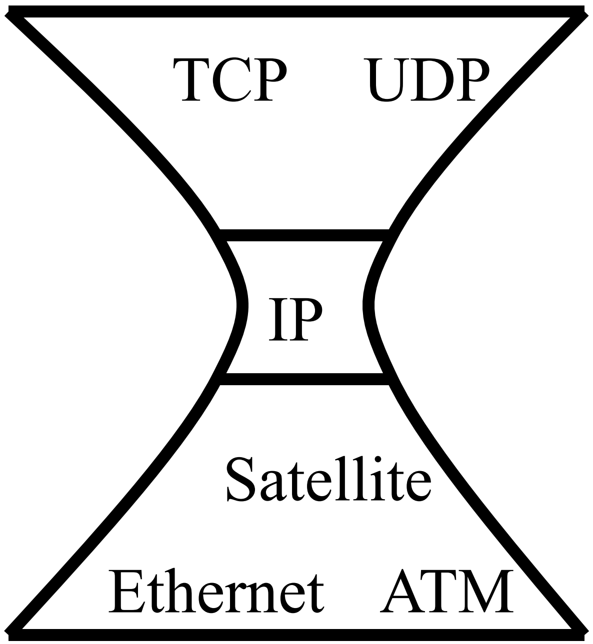
- Accountability and security
  - Huge problem.
  - Worms, viruses, etc.
    - Partly a host problem. But hosts very trusted.
  - Authentication
    - Purely optional. Many philosophical issues of privacy vs. accountability
  - Greedy sources aren't handled well

Other IP Design Weaknesses

- Weak administration and management tools
- Incremental deployment difficult at times
  - Result of no centralized control
  - No more “flag” days

Summary: Internet Architecture

- Packet-switched datagram network
- IP is the “compatibility layer”
  - Hourglass architecture
  - All hosts and routers run IP
- Stateless architecture
  - no per flow state inside network



Summary: Minimalist Approach

- Dumb network
  - IP provide minimal functionalities to support connectivity
    - Addressing, forwarding, routing
- Smart end system
  - Transport layer or application performs more sophisticated functionalities
    - Flow control, error control, congestion control
- Advantages
  - Accommodate heterogeneous technologies (Ethernet, modem, satellite, wireless)
  - Support diverse applications (telnet, ftp, Web, X windows)
  - Decentralized network administration

Summary

- Successes: IP on everything!
- Drawbacks…

  but perhaps they're totally worth it in the context of the original Internet. Might not have worked without them!

"This set of goals might seem to be nothing more than a checklist of all the desirable network features. It is important to understand that these goals are in order of importance, and **an entirely different network architecture would result if the order were changed**."

Changes Over Time → New Principles?

- Developed in simpler times
  - Common goals, consistent vision
- With success came multiple goals – examples:
  - ISPs must talk to provide connectivity but are fierce competitors
  - Privacy of users vs. government's need to monitor
  - User's desire to exchange files vs. copyright owners
- Must deal with the tussle between concerns in design

- Provide choice → allow all parties to make choices on interactions
  - Creates competition
  - Fear between providers helps shape the tussle