CS/ECE 4457

Computer Networks:
Architecture and Protocols

Lecture 4/5
- Three Architectural Principles
- Design Goals

Qizhe Cai

Goals for Today's and next Lecture

- **Recap: Three architectural principles:**
    - Layering
    - End-to-end principle
    - Fate Sharing principle

- **Design goals for computer networks:**
    - Eight of them

Quick recap from last lecture

Three Architectural Principles

Network Modularity Decisions

- How to break system into modules?
  - Classic decomposition into tasks
- Where are modules implemented?
  - Hosts?
  - Routers?
  - Both?
- Where is state stored?
  - Hosts?
  - Routers?
  - Both?

Leads to three design principles

- How to break system into modules
    - **Layering**
- Where are modules implemented
    - **End-to-End Principle**
- Where is state stored?
    - **Fate-Sharing**

Layering

Breakdown end-to-end functionality into tasks

- Bits on wire
- Packets on wire
- Deliver packets between hosts in a “local” network (eg, within UVA)
- Routing & forwarding packets across networks (eg, from UVA to UIUC)
- Deliver data reliably between processes (applications)
- Do something with the data

Breakdown end-to-end functionality into tasks

- **Bits on wire**
- Packets on wire
- **Deliver packets between hosts in a local network**
- **Routing and forwarding (packets) across networks**
- **Deliver data reliably between processes**
- Do something with the data

Resulting Modules (Layers)

- **Bits on wire (Physical)**
- Packets on wire
- **Deliver packets between hosts in a local network (Datalink)**
- **Routing and forwarding (packets) across networks (Network)**
- **Deliver data reliably between processes (Transport)**
- Do something with the data (Application)

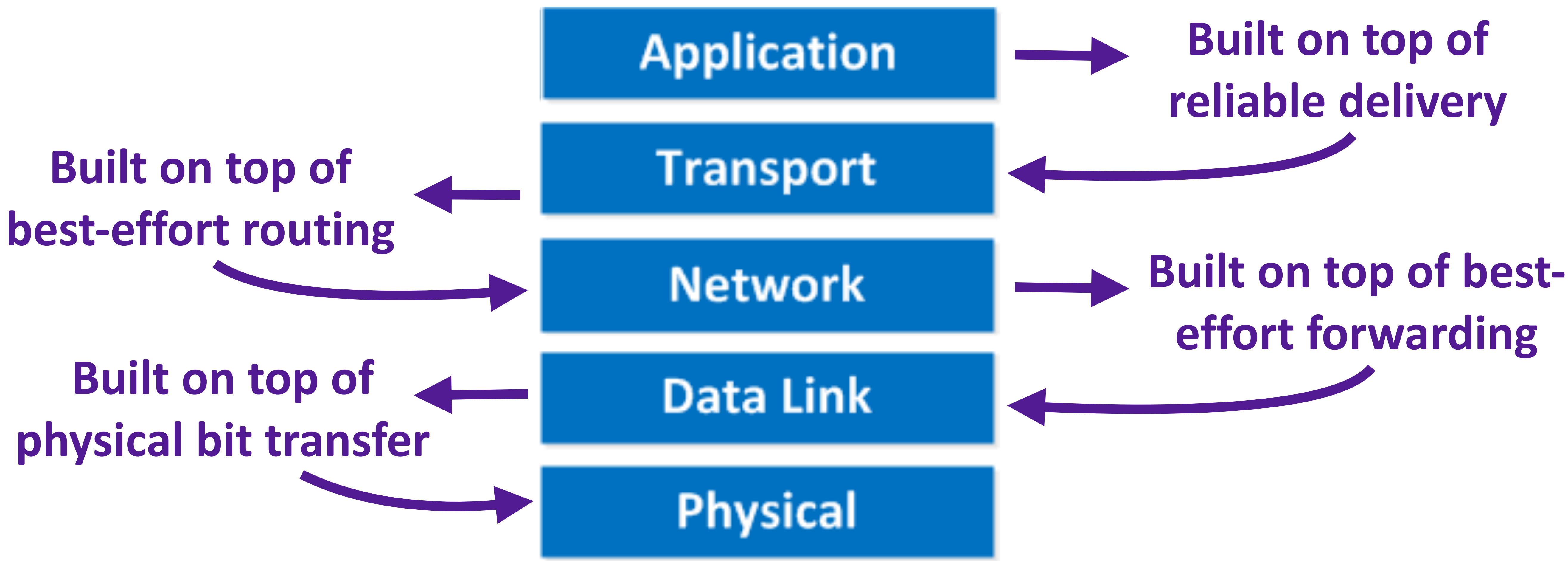
Resulting Modules (Layers)

- **Bits on wire (Physical, Layer1)**
- Packets on wire
- **Deliver packets to hosts across local network (Datalink, Layer2)**
- **Routing and forwarding (packets) across networks (Network, Layer3)**
- **Deliver data reliably between processes (Transport, Layer4)**
- Do something with the data (Application)

Five Layers (Top - Down)

- Application: Providing network support for apps
- **Transport (L4):** (Reliable) end-to-end delivery
- **Network (L3):** Routing and forwarding across networks
- **Datalink (L2):** Forwarding within a local network
- **Physical (L1):** Bits on wire

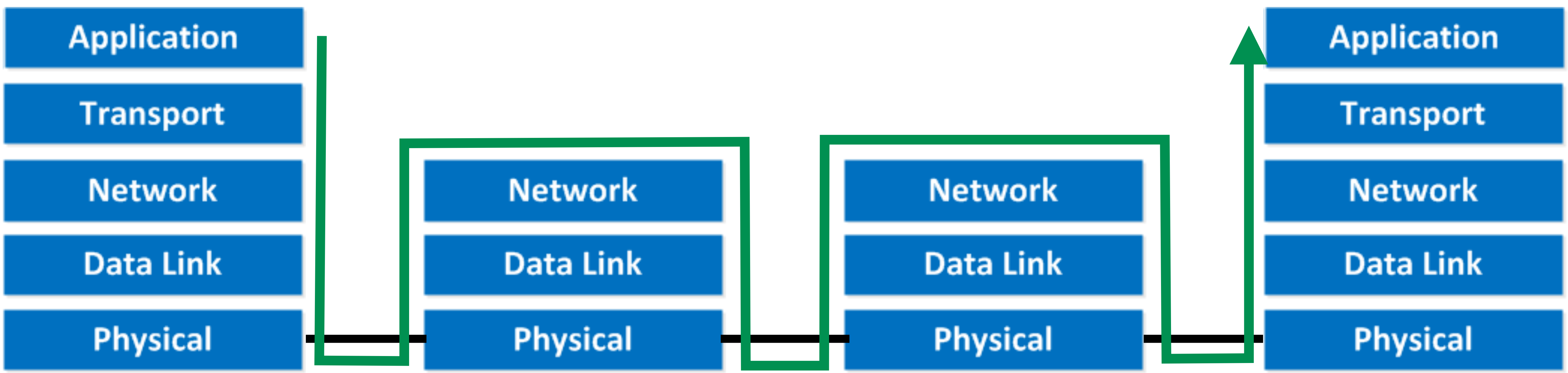
Layering



- **A kind of modularity**
  - Functionality separated into layers
  - Layer n **interfaces with only layer n-1 and layer n+1**
    - Hides complexity of surrounding layers

An end-to-end view of the layers

- Application: Providing network support for apps
- **Transport (L4):** (Reliable) end-to-end delivery
- **Network (L3):** Routing and forwarding across networks
- **Datalink (L2):** Forwarding within a local network
- **Physical (L1):** Bits on wire



Why does the packet go all the way to network layer at each hop?

Questions?

Three Internet Design Principles

- How to break system into modules?
  - Layering
- Where are modules implemented?
  - **End-to-End Principle**
- Where is state stored?
  - Fate-Sharing

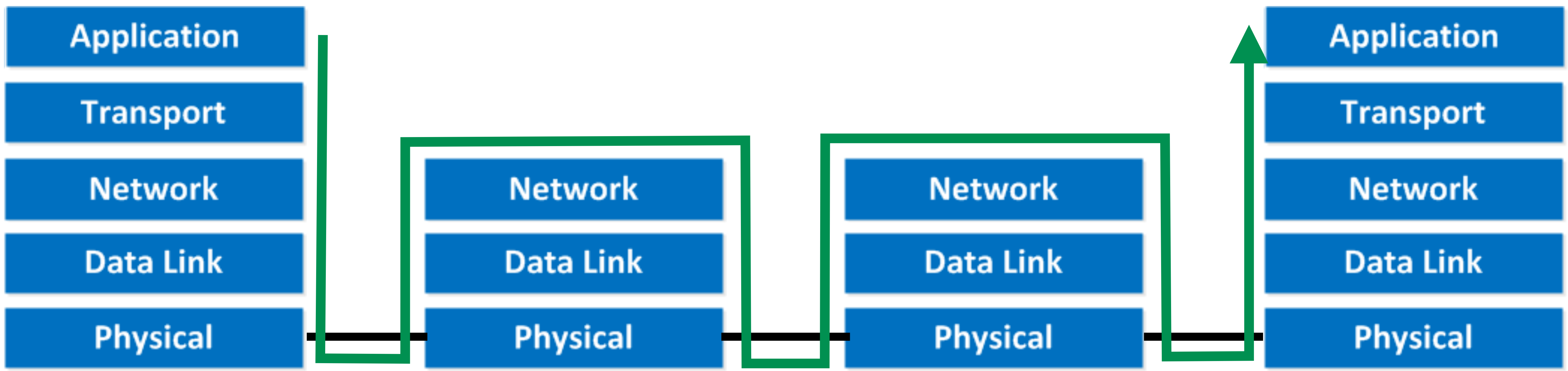
End-to-end Principle

If a function can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system,

then providing that function as a feature of the communication system itself is not possible.

Sometimes providing an incomplete version of that function as a feature of the communication system itself may be useful as a performance enhancement.

End-to-end Principle: an example



- **Suppose each link layer transmission is reliable**
  - Does that ensure end-to-end (application-to-application) reliability?
- **Suppose network layer is reliable**
  - Does that ensure end-to-end (application-to-application) reliability?

End-to-end Principle: lets read again

If a function can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system,

then providing that function as a feature of the communication system itself is not possible.

Sometimes providing an incomplete version of that function as a feature of the communication system itself may be useful as a performance enhancement.

End-to-end Principle (Interpretation)

Assume the condition (IF) holds. Then,

- **End-to-end implementation**
    - Correct
    - Generalized, and simplifies lower layers
- **In-network implementation**
    - Insufficient
    - May help — or hurt — performance

Questions?

Three Internet Design Principles

- How to break system into modules?
  - Layering

- Where are modules implemented?
  - End-to-End Principle

- **Where is the state stored?**
  - **Fate-sharing**

General Principle: Fate-Sharing

- When storing state in a distributed system, colocate it with entities that rely on that state
  - e.g, Connection states vs. Transport layers

- Only way failure can cause loss of the critical state is if the entity that cares about it also fails …
  - … in which case it doesn’t matter

- Often argues for keeping network state at end hosts rather than inside routers
  - E.g., packet switching rather than circuit switching

Questions?

Decisions and their Principles

- How to break system into modules
  - **Dictated by layering**
- Where modules are implemented
  - **Dictated by End-to-End Principle**
- Where state is stored
  - **Dictated by Fate Sharing**

From Architecture to Design:

Design Goals

David Clark

- Wrote a paper in 1988 that tried to capture why the Internet turned out as it did (Layer 3)
- It described an ordered list of priorities that informed the decision
- What do you think those priorities were?

Internet Design Goals (Clark ’88)

- **Connect existing networks**
- Robust in face of failures
- Support multiple types of delivery services
- Accommodate a variety of networks
- Allow distributed management
- Easy host attachment
- Cost effective
- Allow resource accountability

#1: Connect Existing Networks

Want one protocol that could be used to connect any pair of (existing) networks

- Different networks may have different needs
    - For some: reliable delivery more important
    - For others: performance more important
    - **But there is one need that every network has: connectivity**
- The Internet Protocol (IP) is that unifying protocol
    - All (existing) networks must be able to implement it

#2: Robust in Face of Failures

As long as network is not partitioned, two hosts should be able to communicate (eventually)

- Must **eventually recover** from failures

- Very successful in the past; unclear how relevant now
    - **Availability** is becoming increasingly important than **recovery**

#3: Support Multiple Types of Delivery Services

Different delivery services (applications) should be able to co-exist

- Already implies an application-neutral framework
- Build lowest common denominator service
  - **Again: connectivity**
  - For Reliability,
    - Applications that need reliability may use it
    - Applications that do not need reliability can ignore it
- **This isn't as obvious as it seems…**
  - What would applications in 2050 need?

Questions?

#4: Variety of Networks

Must be able to support different networks with different hardware

- **Incredibly successful!**
    - Minimal requirements on networks
    - No need for reliability, in-order, fixed size packets, etc.
    - A result of aiming for lowest common denominator
- **Again: Focus on connectivity**
    - Let networks do specific implementations for other functionalities
    - Automatically adapt: WiFi, LTE, 3G, 4G, 5G ….

#5: Decentralized Management

No need to have a single “vantage” point to manage networks

- Both a curse and a blessing
    - Important for easy deployment
    - Makes management hard today

- Recent efforts have improved management of individual networks
    - But no attempt to manage the Internet as a whole…
    - What might make this complex?

#6: Easy Host Attachment

The mechanism that allows hosts to attach to networks must be made as easy as possible, but no easier

- Clark observes that cost of host attachment may be higher because hosts had to be smart
- But the administrative cost of adding hosts is very low, which is probably more important
    - Plug-and-play kind of behavior…
- And now most hosts are smart for other reasons
    - So the cost is actually minimal…

#7: Cost Effective

Make networks as cheap as possible, but no cheaper

- Cheaper than circuit switching at low end
- More expensive than circuit switching at high end
- Not a bad compromise:
    - Cheap where it counts (low-end)
    - More expensive for those who can pay…

#8: Resource Accountability

Each network element must be made accountable for its resource usage

- Failure!

Internet Motto

"We reject kings, presidents and voting. We believe in rough consensus and running code."

- - David Clark

Real Goals

- **Build something that works**
- Connect existing networks
- Robust in face of failures
- Support multiple types of delivery service
- Accommodate a variety of networks
- Allow distributed management
- Easy host attachment
- Cost effective
- Allow resource accountability

Questions to think about

- What goals are missing from this list?
    - **Suggestions?**

- What would the resulting design look like?

Some of the missing issues

- **Performance**
- Security
    - Resilience to attacks (denial-of-service)
    - Endpoint security
    - Tracking down misbehaving users
- Privacy
- Availability
- Resource sharing (fairness, etc.)
- ISP-level concerns
    - Economic issues of interconnection

Questions?

Next lecture

- Beginning of “Design of computer networks”
- Start with Layer 1 and Layer 2
    - Physical bits (very little)
    - Local best-effort forwarding
    - Lots of interesting aspects
    - Lots of group activities
    - …