Internet Architecture

Computer Networks

Collections of machines exchanging information with each other

Principles and algorithms by which communication software & hardware are organized

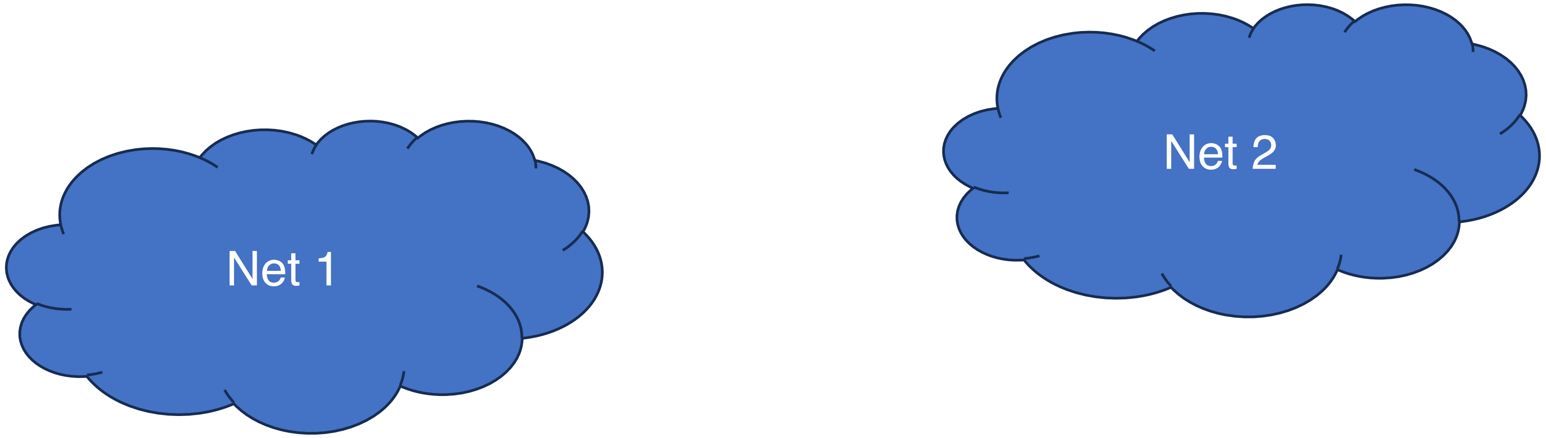
Design of foundational artifacts of the Internet and other modern networks

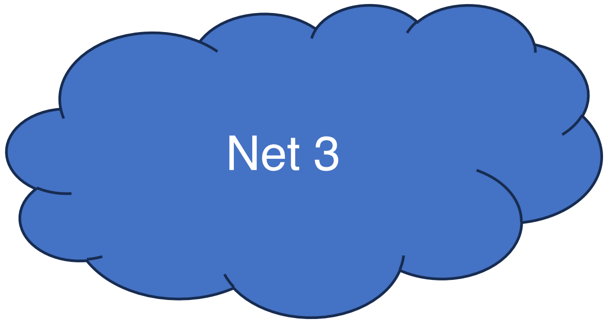
There’s a science to communication

- What language should machines speak?
- How to partition functionality? “Who” should do “what”?
- How to make communication effective?
    - Achieve better scale, performance, resource efficiency
    - Evolve to address new needs over time
- How to grow organically? include more communicating parties
    - Make it easy for humans to build and manage?
- How to make communication worthy of societal trust?

A brief history of Internet Architecture

Prior to the Internet

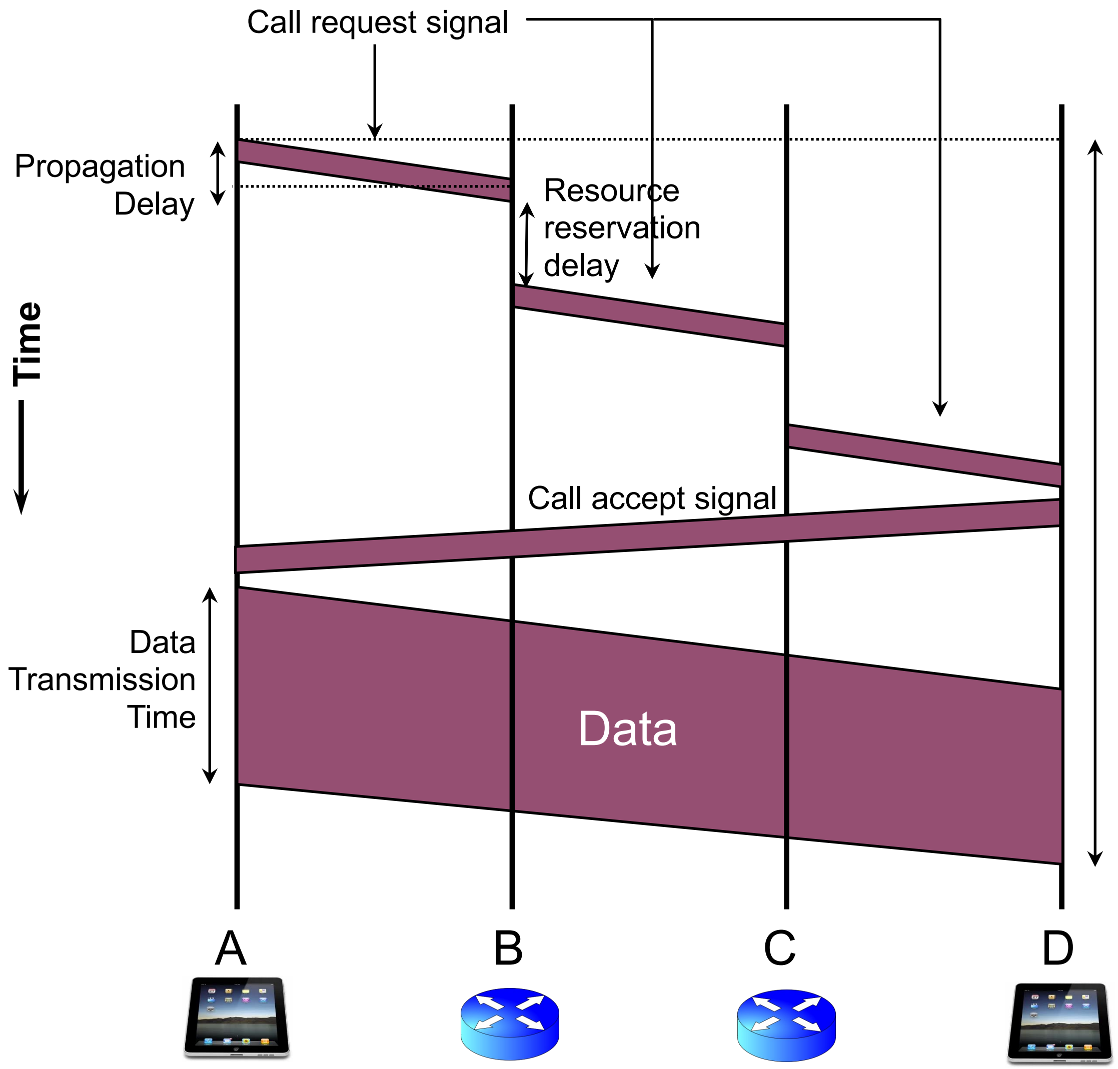




Different technologies:
wireless/wired
slow/fast,
un/reliable,
switching techniques
Different administration

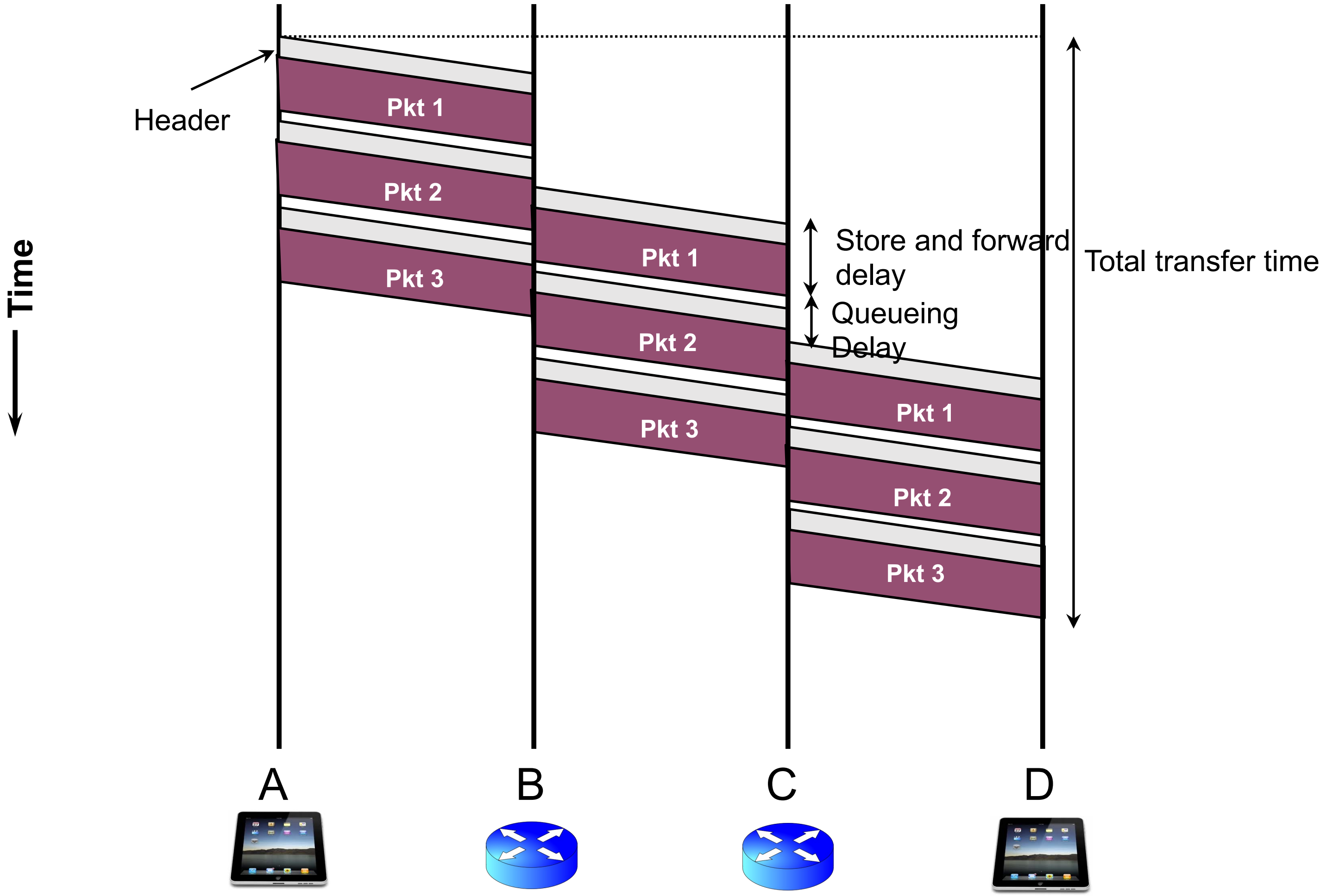
How to interconnect these existing networks?
Focus more on practicality and usefulness rather than “clean design”

Circuit switching



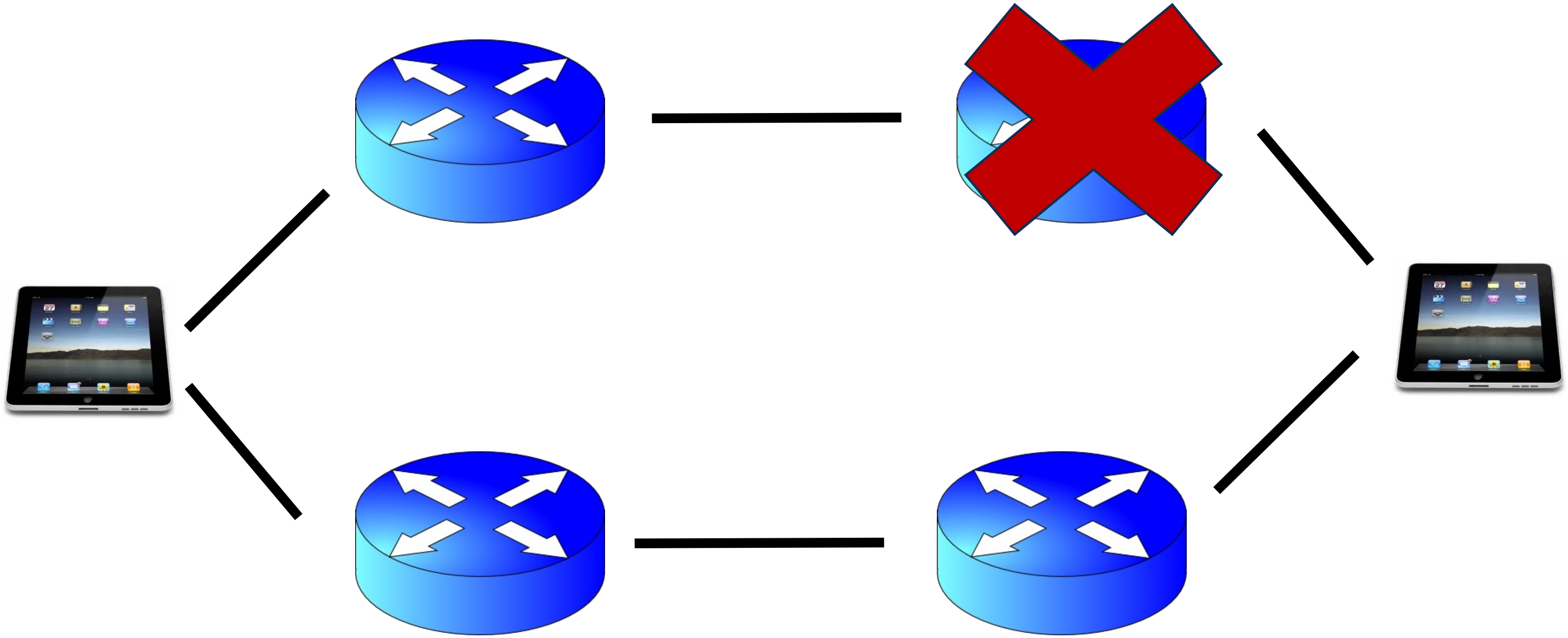
1. Setup: Control message sets up a path from origin to destination
2. Accept signal informs source that data transmission may proceed
3. Data transmission begins
4. Entire path remains allocated to the transmission (whether used or not)
5. When transmission is complete, source releases the circuit

Packet switching



Packet switching is simpler and required less from existing networks that were interconnected.

Survivability



Should applications care?
No
Who has the “context” of the communication? Endpoint vs. Network

Tradeoffs:
Replicate, still can’t guarantee
Instead, fate sharing. Simpler to engineer

The Internet puts the context of conversations in the endpoints. Call this transport information.

Networks provide datagram service.

Endpoints are trusted to implement the right algorithms to provide any sort of guarantees about communication.

Distributed Management

Don’t require single administrative entity.
The Internet is federated.

Two-tier management: within an administration, across administrations

Consequences of datagram service

- A simple building block for different kinds of applications:
  - Bulk file transfer
  - Conversational (real time)
  - Try very hard not to constraint what can be implemented atop network

- No explicit guarantees required from interconnecting networks
  - Accommodate a variety of existing networks

- Services explicitly not assumed from the network:
  - reliable or sequenced delivery, network level broadcast or multicast, priority ranking of transmitted packet, support for multiple types of service, and internal knowledge of failures, speeds, or delays.
  - Only see packets, no higher-level context

Layering and Protocols

Software/hardware organization at hosts

Communication functions broken up and “stacked”

Each layer depends on the one below it.

Each layer supports the one above it.

The interfaces between layers are well-defined and standardized.

Internet software and hardware are arranged in layers.

Layering provides modularity

Each layer: well-defined function & interfaces to layers above & below it.

Functionality is implemented in protocols.

Protocols: The “rules” of networking

- Protocols consist of two things

- Message format
  - structure of messages exchanged with an endpoint

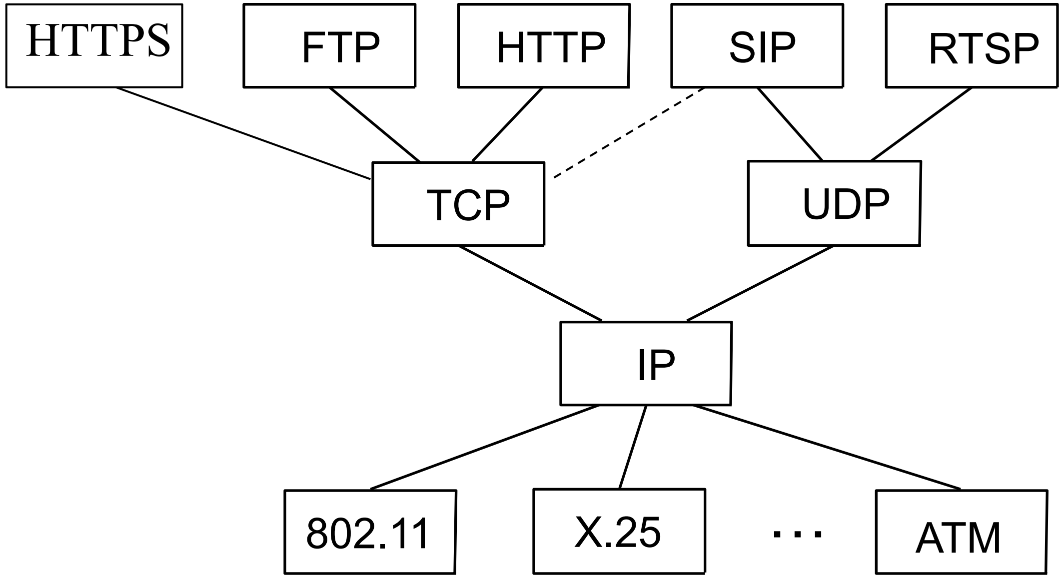
- Actions
  - operations upon receiving, or not receiving, messages

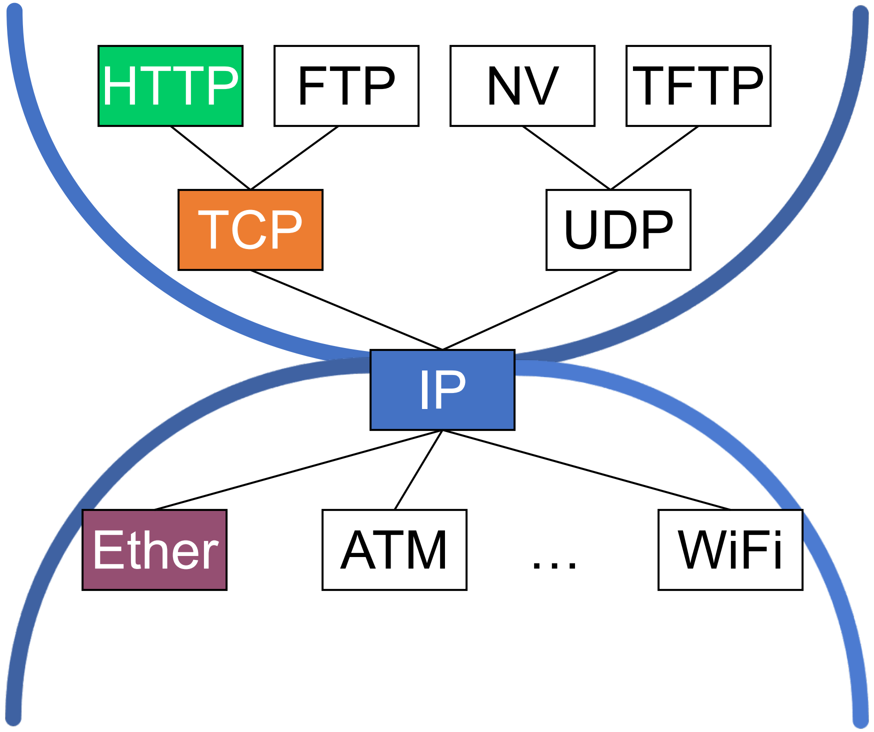
- Example of a Zoom conversation:
  - Message format: English words and sentences
  - Actions: when a word is heard, say “yes”; when nothing is heard for more than 3 seconds, say “can you hear me?”

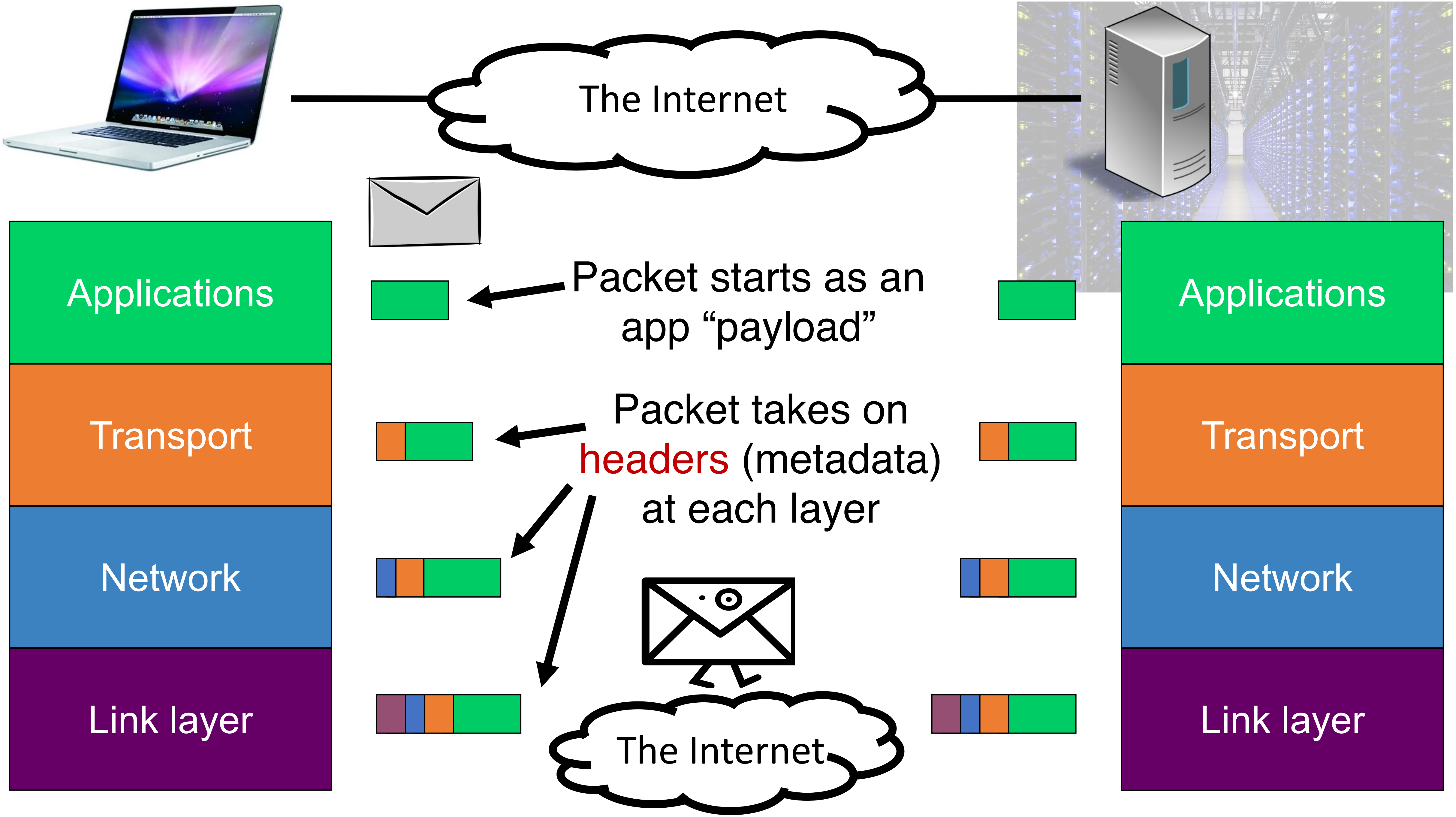
The protocols of the Internet

- Standardized by the Internet Engineering Task Force (IETF)
  - through documents called RFCs ("Request For Comments")

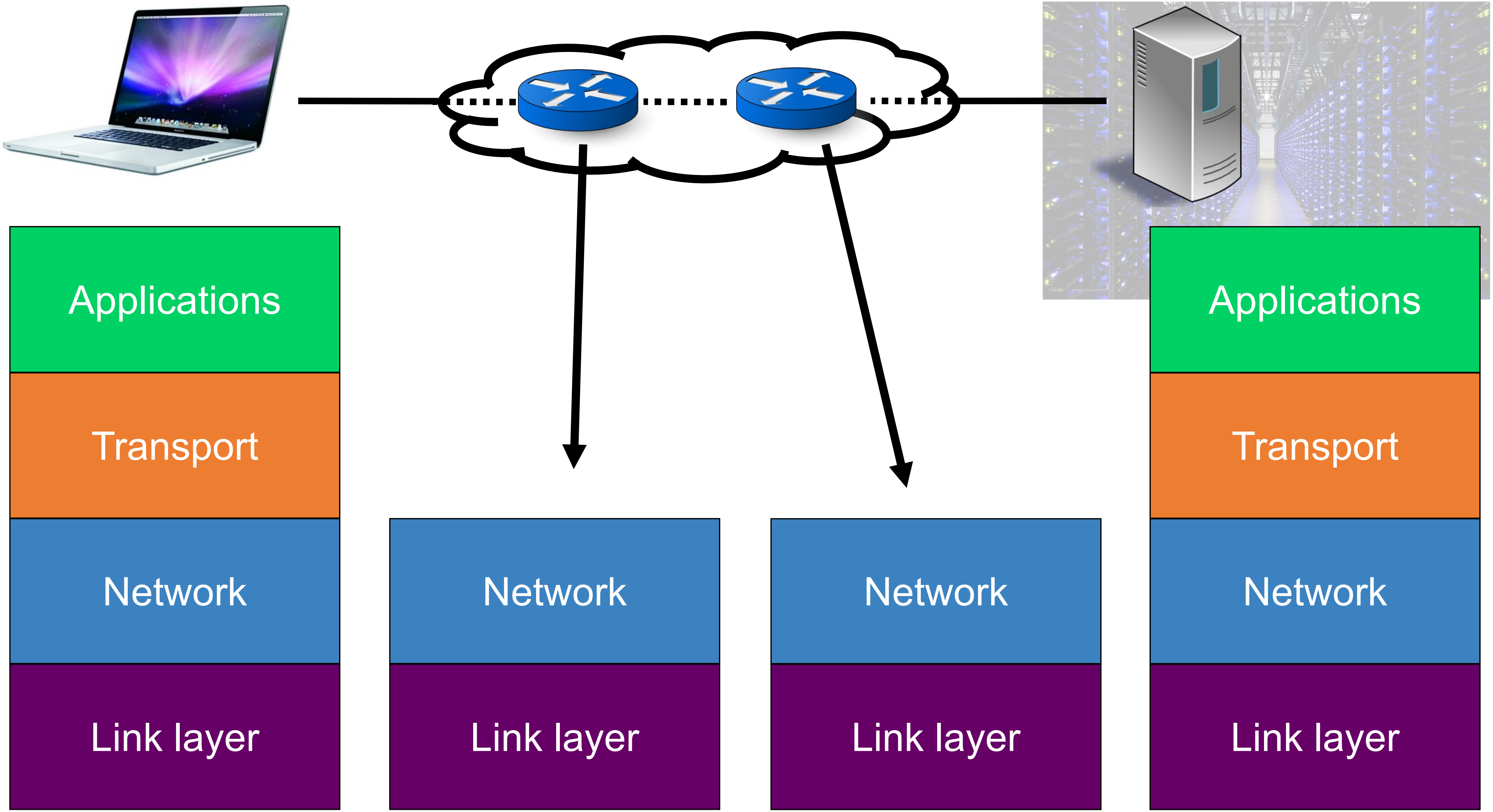
- Layering of protocols

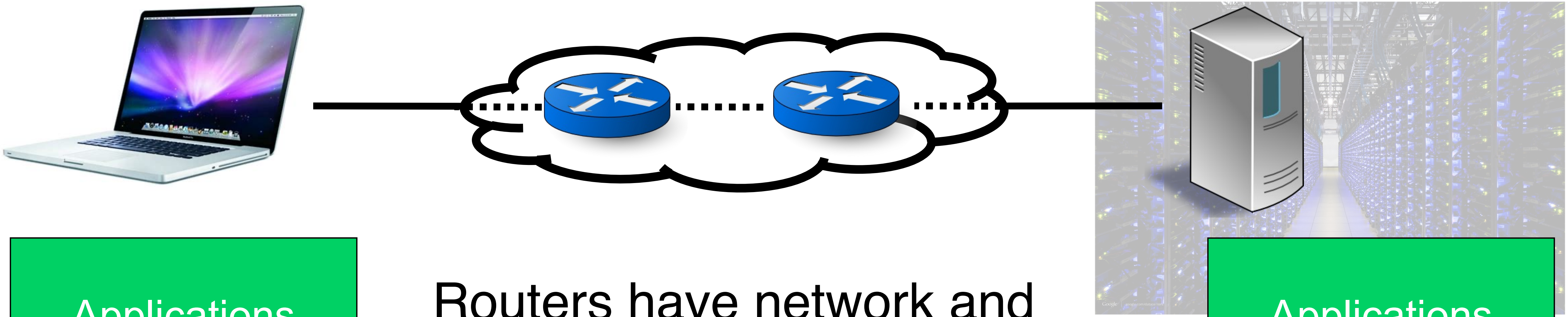






The Internet
Applications
Transport
Network
Link layer
Packet starts as an app "payload"
Packet takes on headers (metadata) at each layer
The Internet
Applications
Transport
Network
Link layer


Applications
Transport
Network
Link layer
Network
Link layer
Network
Link layer
Applications
Transport
Network
Link layer



Routers have network and link layers too!

Applications			Applications
Transport			Transport
Network	Network	Network	Network
Link layer	Link layer	Link layer	Link layer

Layering

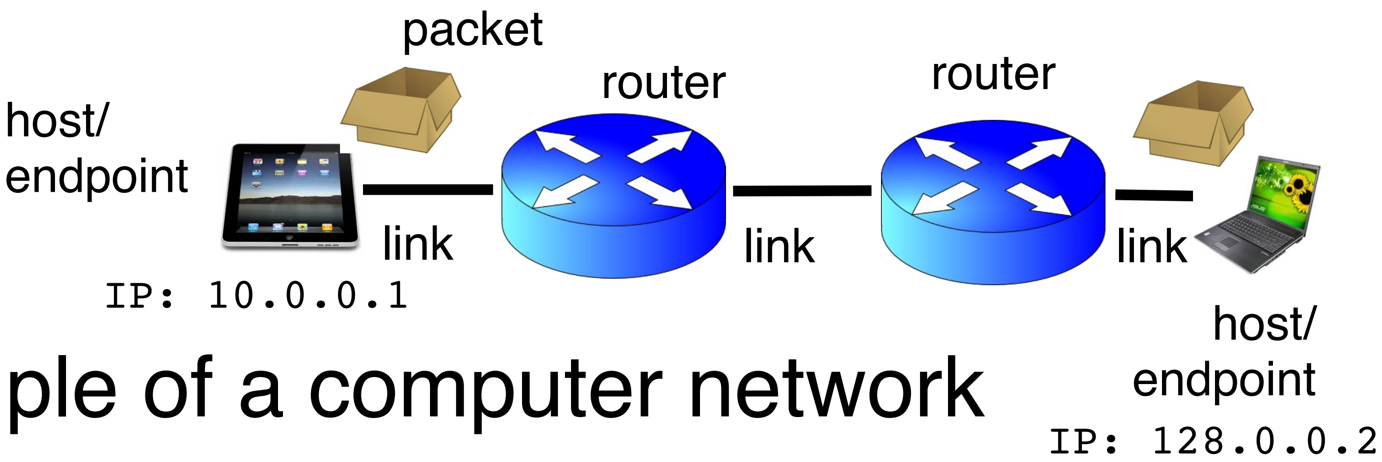
- Communication over the Internet is a complex problem.
- Layering simplifies understanding, testing, maintaining
- Easy to improve or replace protocol at one layer without affecting others

Many open and partially solved problems

- Resource management in distributed routing
- Not cost effective: small packets have high header overheads
- Retransmitting lost packets end to end can be inefficient (network recovery could have made things simpler)
- Cost of attaching a host to the network is somewhat high: all necessary software must reside on the host (e.g., transport).
- No inherent network mechanisms to account for or control resource usage
    - e.g., Putting important packets ahead in queues

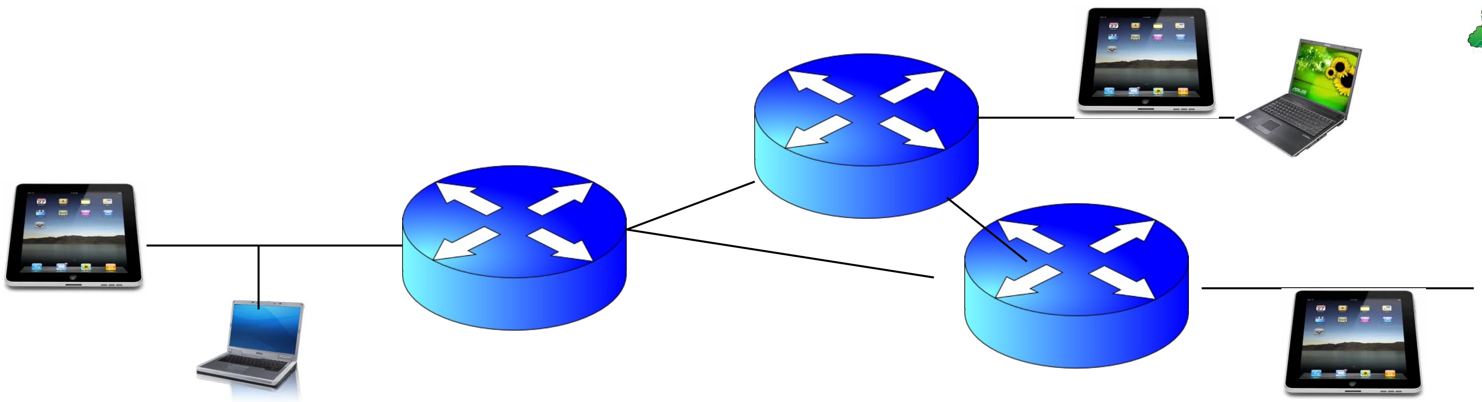
Some fundamental problems

Some definitions

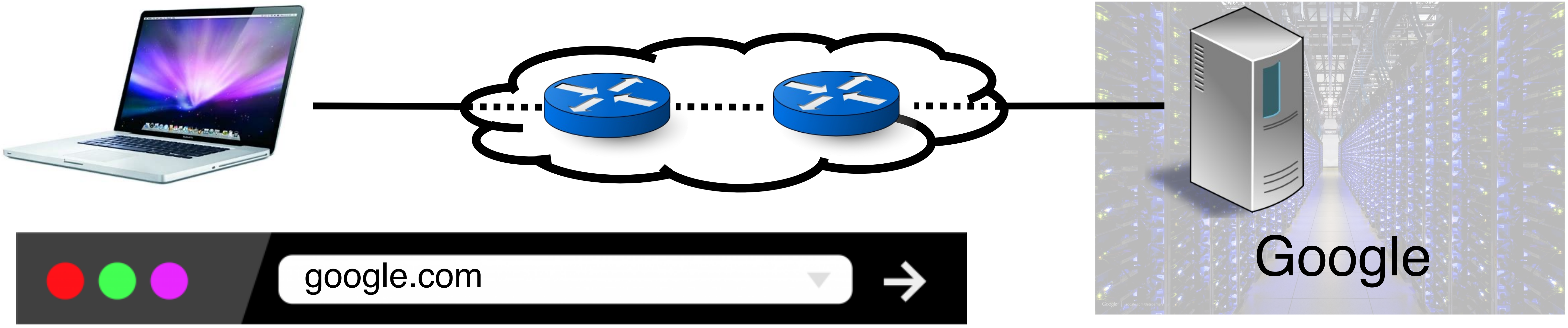


- The Internet is an example of a computer network
- Endpoint or Host: Machine running user application
- Packet: a unit of data transmission (ex: 1500 bytes)
- Link: physical communication channel between two or more machines
- Router: A machine that processes packets moving them from one link to another towards a destination
- Network: Collection of interconnected machines
- Address: a unique name given to a machine

(0) Naming & Resolution



- Communication requires naming the endpoints
  - Addresses
- Internet addresses (IP addresses) allocated hierarchically
  - Machine readable, not easy for humans to remember
- Link addresses are tied to the hardware on the endpoint
- Name resolution: how to turn human-readable names (google.com) into routable addresses?



Machines communicate using IP addresses and ports
IP addresses: ~12 digits (IPv4) or more
Ports: fixed based on application (e.g., 80: web)

Need a way to turn human-readable addresses into Internet addresses.

Ask someone	Ask everyone	Tell everyone
Directory service	Query broadcast	Information flooding

Asking “someone” could involve asking many machines…

Domain Name Service

DOMAIN NAME	IP ADDRESS
spotify.com	98.138.253.109
cs.rutgers.edu	128.6.4.2
www.google.com	74.125.225.243
www.princeton.edu	128.112.132.86

<Client IP, CPort, DNS server IP, 53>

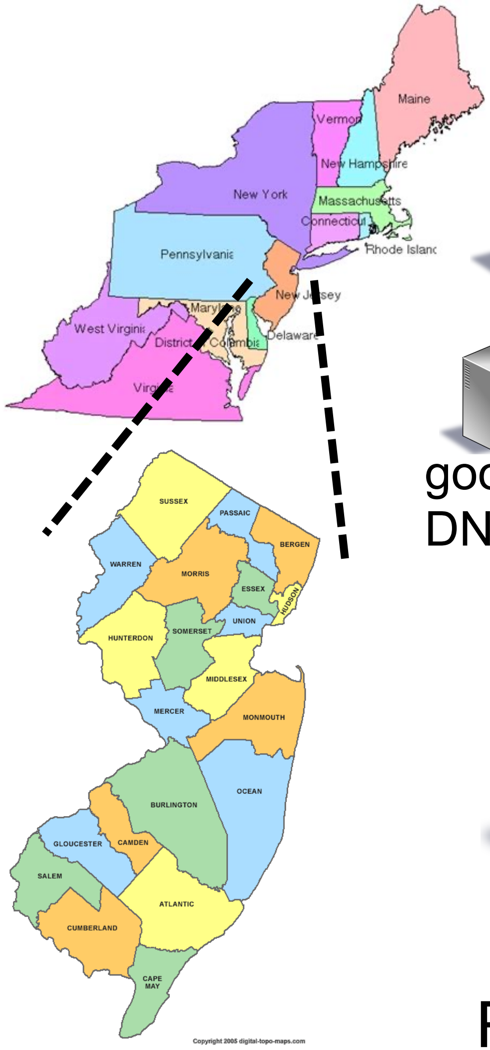
QUERY cs.rutgers.edu

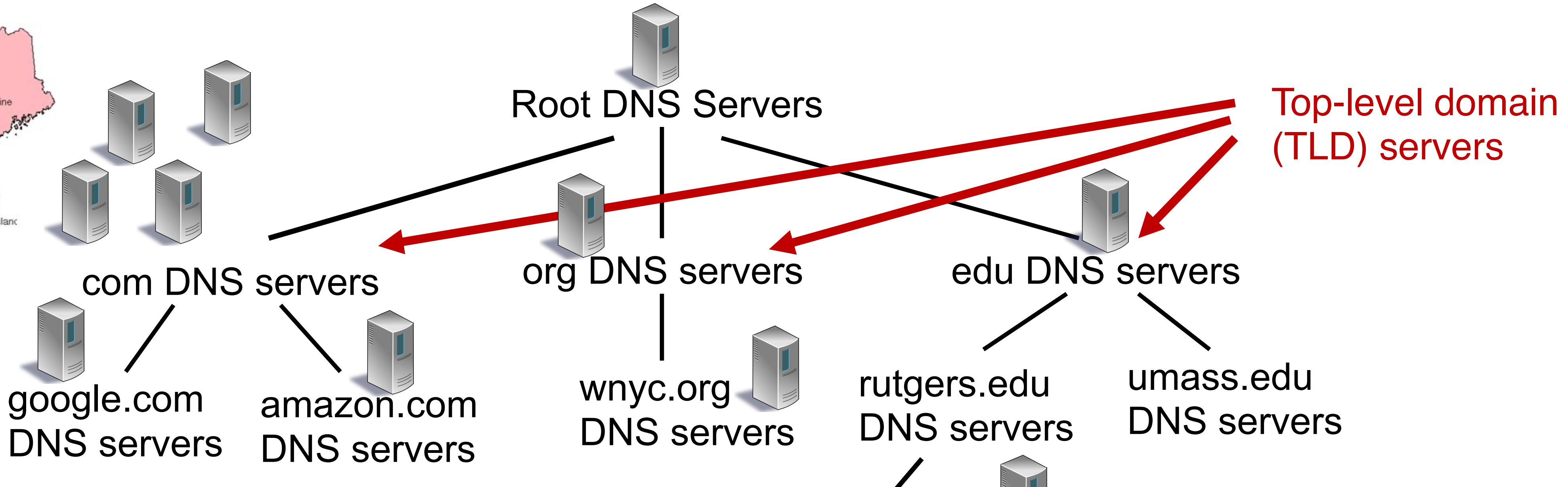
<DNS server, 53, Client IP, Cport>

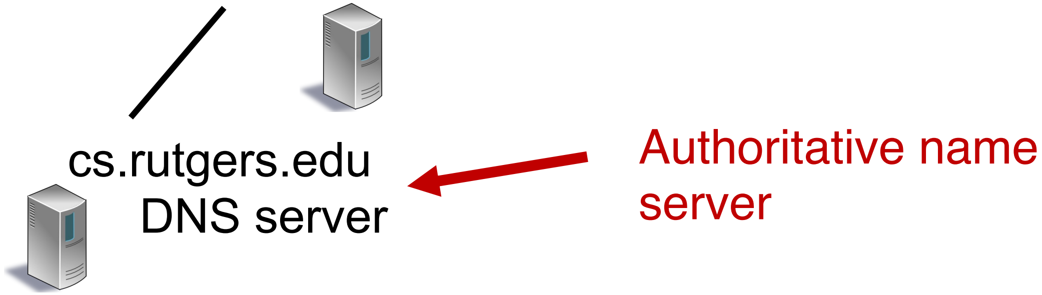
RESPONSE 128.6.4.2

- Key idea: Implement a server that looks up a table.
- Will this scale?
  - Every new (changed) host needs to be (re)entered in this table
  - Performance: can the server serve billions of Internet users?
  - Failure: what if the server or the database crashes?
  - Security: What if someone “takes over” this server?

Distributed and hierarchical database







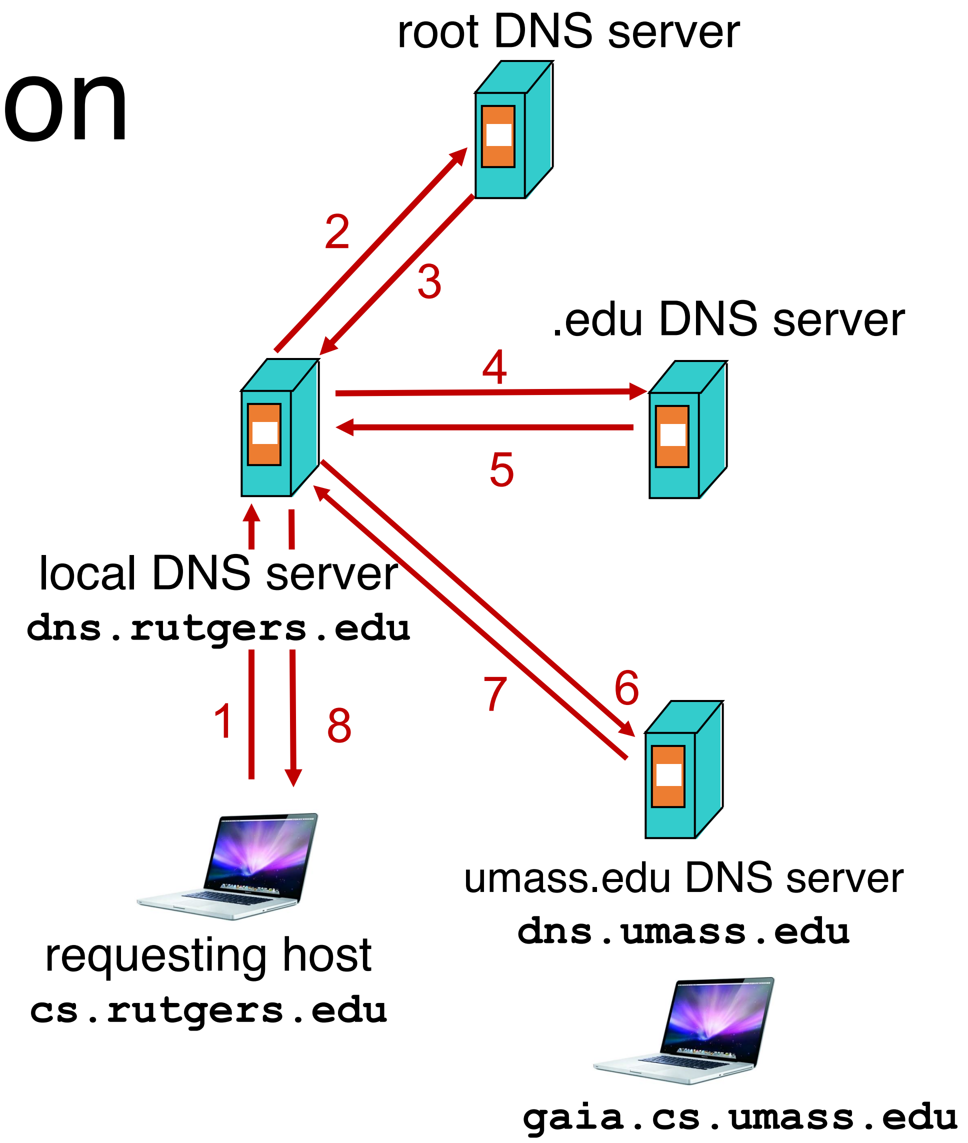
RFC 1034

Hierarchy Replication Indirection

DNS name resolution

- Host at cs.rutgers.edu wants IP address for gaia.cs.umass.edu

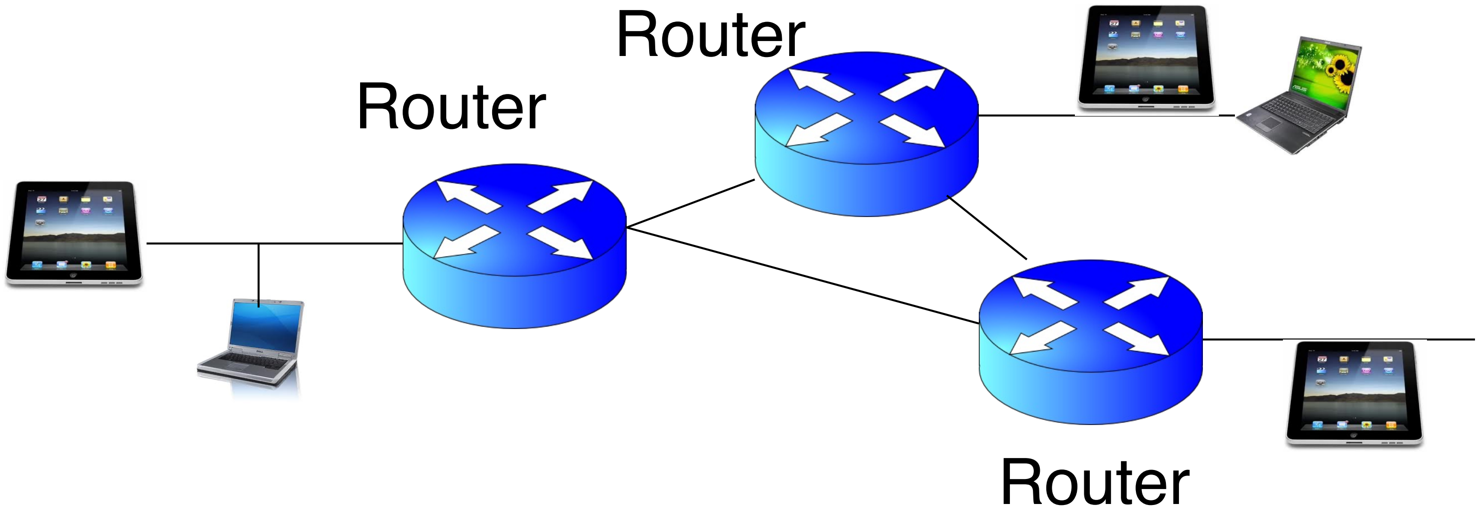
- Local DNS server
- Root DNS server
- TLD DNS server
- Authoritative DNS server



Example DNS interactions

- `dig <domain-name>`
- `dig +trace <domain-name>`
- `dig @<dns-server> <domain-name>`

(1) Routing



- Networks must move data between different hosts
- Need to figure out how to move packets from one host to another host, e.g., how to reach google.com from your laptop
- Known as the routing problem

Routing

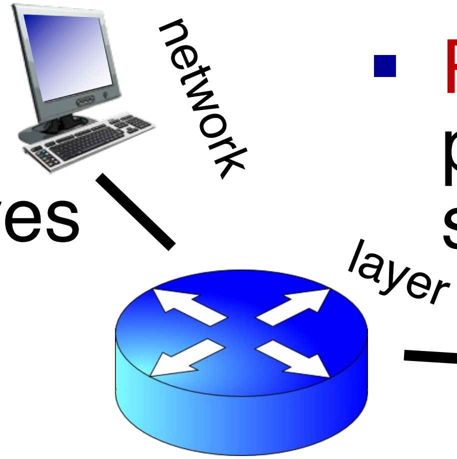

Norwegian Sea
Iceland
Sweden
Finland
Norway
Russia
Denmark
Ireland
Poland
Belarus
Ukraine
Austria
Romania
Kazakhstan
Spain
Portugal
Turkey
Turkmenistan
Sea of Japan
Sea of Okhotsk
China
Japan
Tunisia
Syria
Iraq
Iran
Afghanistan
Pakistan
Morocco
Algeria
Libya
Egypt
Nepal
Western Sahara
Saudi Arabia
India
Oman
Myanmar (Burma)
Mauritania
Mali
Niger
Chad
Sudan
Yemen
Gulf of Aden
Arabian Sea
Bay of Bengal
Philippines
Burkina Faso
Guinea
Nigeria
Ethiopia
South Sudan
Ghana
Laccadive Sea
Gulf of Guinea
Somalia
Gabon
Kenya
Indonesia
DR Congo
Tanzania
Banda Sea
Papua New Guinea
Arafura Sea

Two key network-layer functions

- Forwarding: move packets from router's input to appropriate router output

- Routing: determine route taken by packets from source to destination
  - routing algorithms

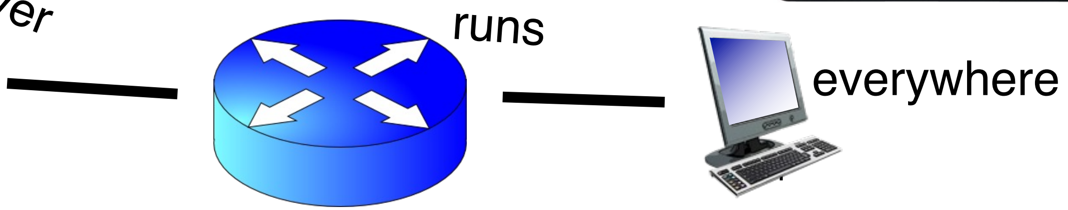
- The network layer solves the routing problem.



Analogy: taking a road trip

- Forwarding: process of getting through single exit

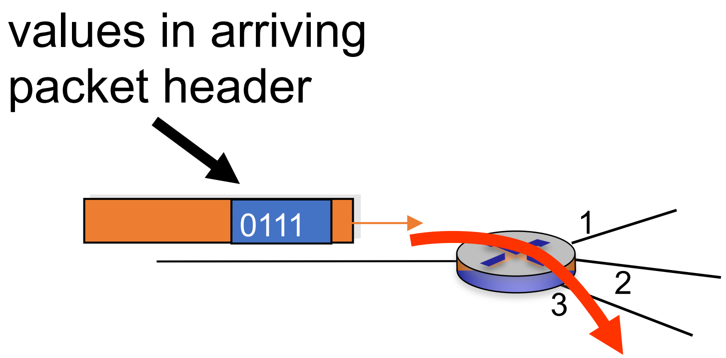
- Routing: process of planning trip from source to destination



Control/Data Planes

Data plane = Forwarding

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port



Control plane = Routing

- network-wide logic
- determines how datagram is routed along end-to-end path from source to destination endpoint
- two control-plane approaches:
  - Distributed routing algorithm running on each router
  - Centralized routing algorithm running on a (logically) centralized machine