CS4450

Computer Networks:
Architecture and Protocols

Lecture 20
The IP protocol
DNS, Discovery protocols
Putting ALL the Pieces Together

Rachit Agarwal

Goal of today's lecture

- **THE Internet Protocol**
  - Functionality: delivering the data
  - Three key ideas:
    - **Addressing** (IP addressing)
    - **Routing** (using a variety of protocols)
    - **Packet header as an interface** (Encapsulating data into packets)
  - Why do packet headers look like the way they look?
- **A brief introduction to Domain Name System**
- **Discovery protocols**
- **End-to-end: how everything fits together**

Network Layer

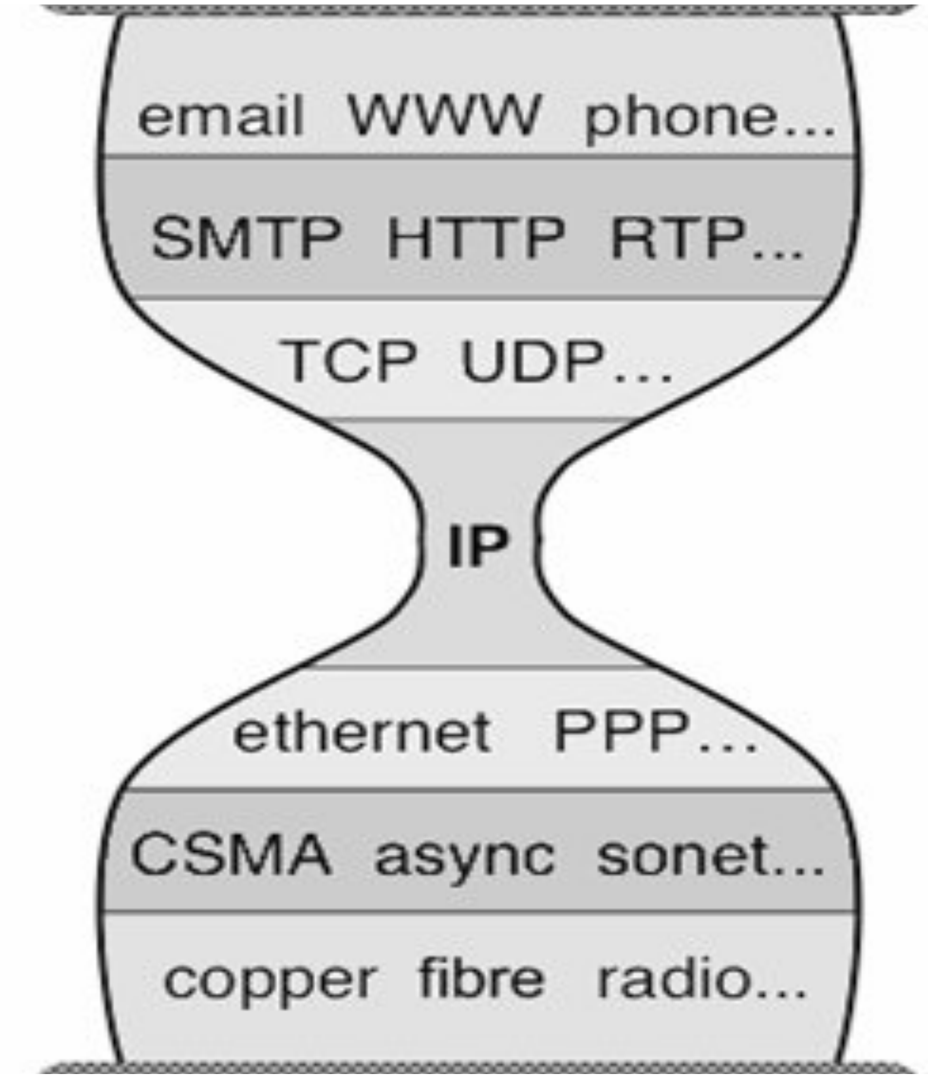
- THE functionality: **delivering the data**

- **THE protocol: Internet Protocol (IP)**

- **Achieves its functionality (delivering the data), using three ideas:**
    - **Addressing** (IP addressing)
    - **Routing** (using a variety of protocols)
    - **Packet header as an interface** (Encapsulating data into packets)

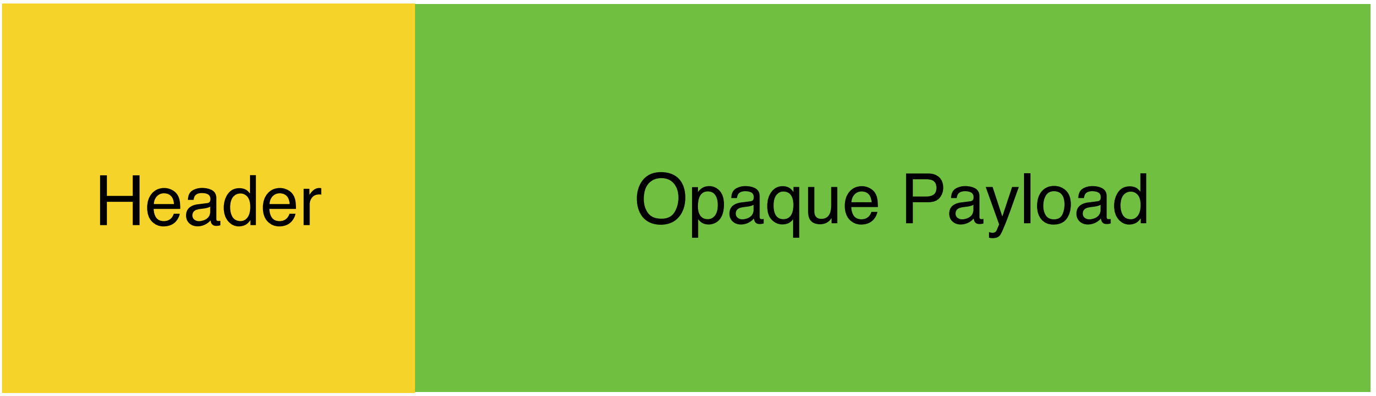
Internet Protocol

- THE functionality: **delivering the data**
- **THE protocol: Internet Protocol (IP)**
- Unifying protocol



What is Designing IP?

- Syntax: format of packet
    - Nontrivial part: packet “header”
    - Rest is opaque payload (**why opaque?**)



- Semantics: meaning of header fields
    - Required processing

Packet Header as Interface

- Think of packet header as interface
  - Only way of passing information from packet to switch
- Designing interfaces:
  - What task are you trying to perform?
  - What information do you need to accomplish it?
- Header reflects information needed for basic tasks

What Tasks Do We Need to Do?

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with the packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Reading Packet Correctly

- Where does the header end?
- Where the the packet end?
- What protocol are we using?
    - Why is this so important?

Getting to the Destination

- Provide destination address
- Should this be location or identifier (name)?
  - And what’s the difference?
- If a host moves should its address change?
  - If not, how can you build scalable Internet?
  - If so, then what good is an address for identification?

Getting Response Back to Source

- Source address
- Necessary for routers to respond to source
  - When would they need to respond back?
    - Failures!
  - Do they really need to respond back?
    - How would the source know if the packet has reached the destination?

Carry Data

- Payload!

Questions?

List of Tasks

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Telling Destination How to Process Packet

- Indicate which protocols should handle packet
- What layers should this protocol be in?
- What are some options for this today?
- How does the source know what to enter here?

Special Handling

- Type of service, priority, etc.
- Options: discuss later

Dealing With Problems

- Is packet caught in loop?
    - TTL
- Header corrupted:
    - Detect with Checksum
    - What about payload checksum?
- Packet too large?
    - Deal with fragmentation
    - Split packet apart
    - Keep track of how to put together

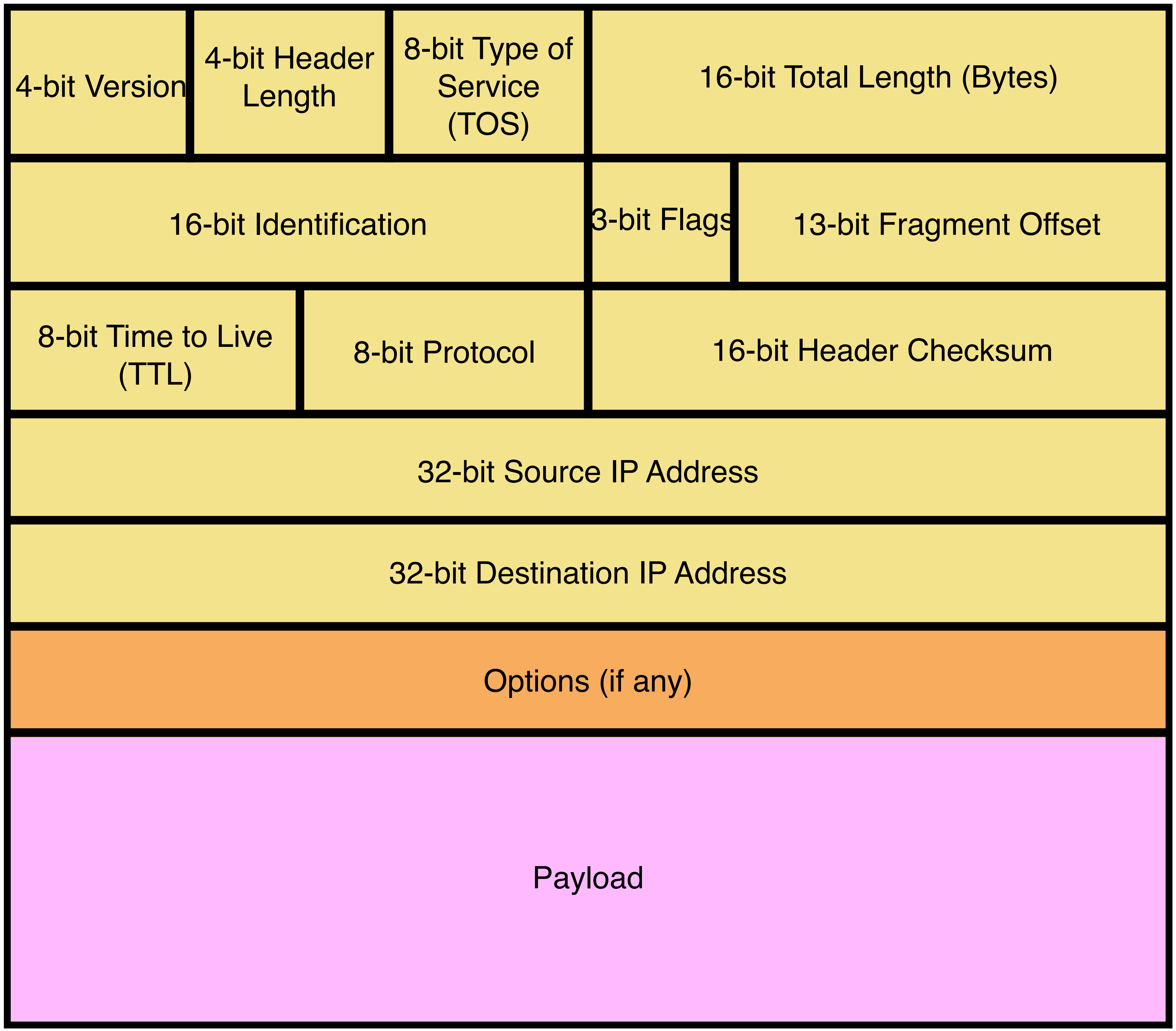
Are We Missing Anything?

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

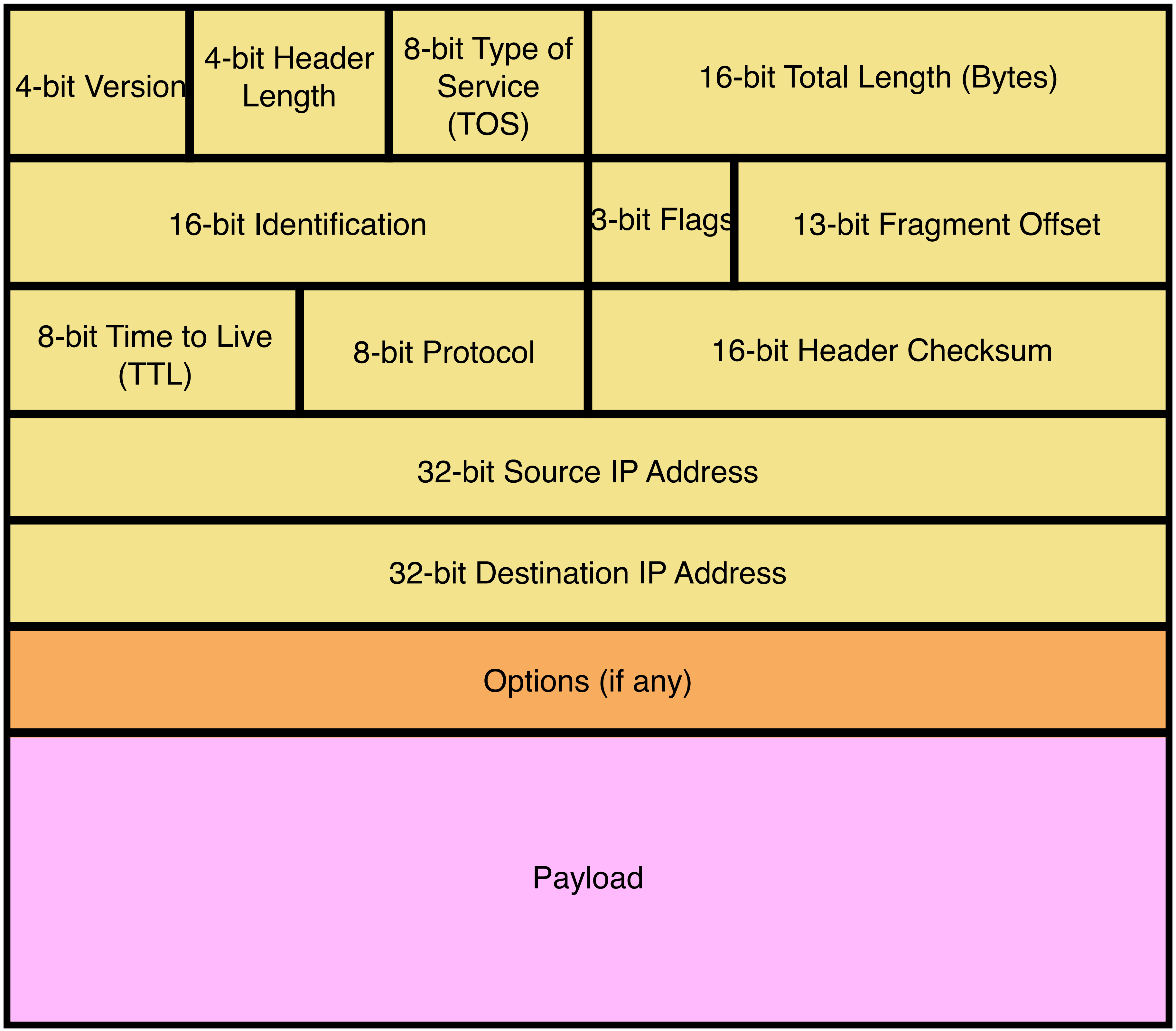
From Semantics to Syntax

- The past few slides discussed the information the header must provide
- Will now show the syntax (layout) of IPv4 header, and discuss the semantics in more detail

IP Packet Structure



20 Bytes of Standard Header, then Options



Next Set of Slides

- Mapping between tasks and header fields
- Each of these fields is devoted to a task
- Let’s find out which ones and why…

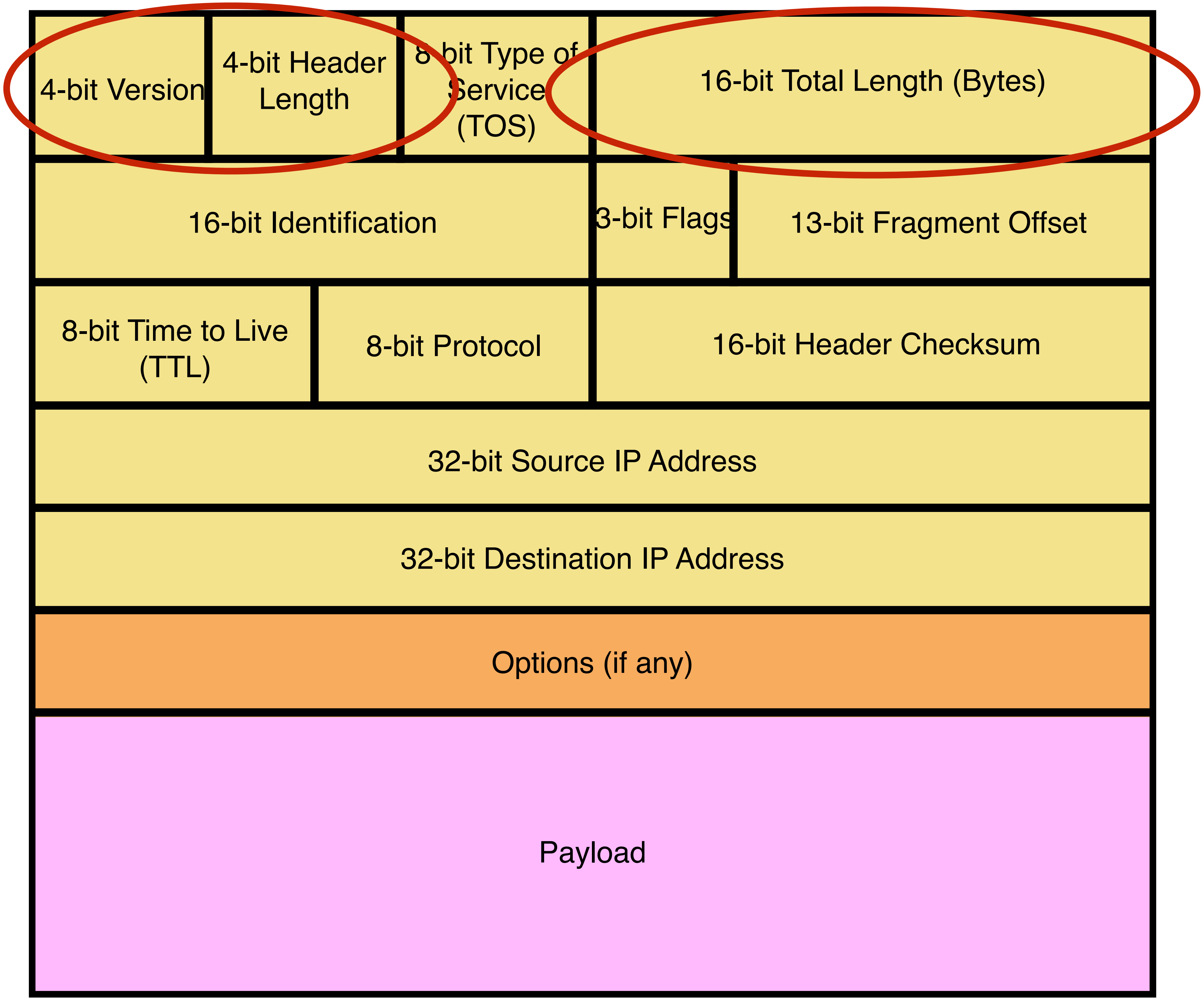
Go Through Tasks One-by-One

- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- Tell host what to do with packet once arrived
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Read Packet Correctly

- **Version number** (4 bits)
  - Indicates the version of the IP protocol
  - Necessary to know what other fields to expect
  - Typically “4” (for IPv4), and sometimes “6” (for IPv6)
- **Header length** (4 bits)
  - Number of 32-bit words in the header
  - Typically “5” (for a 20-byte IPv4 header)
  - Can be more when IP options are used
- **Total length** (16 bits)
  - Number of bytes in the packet
  - Maximum size is 65,535 bytes (2^16 -1)
  - … though underlying links may impose smaller limits

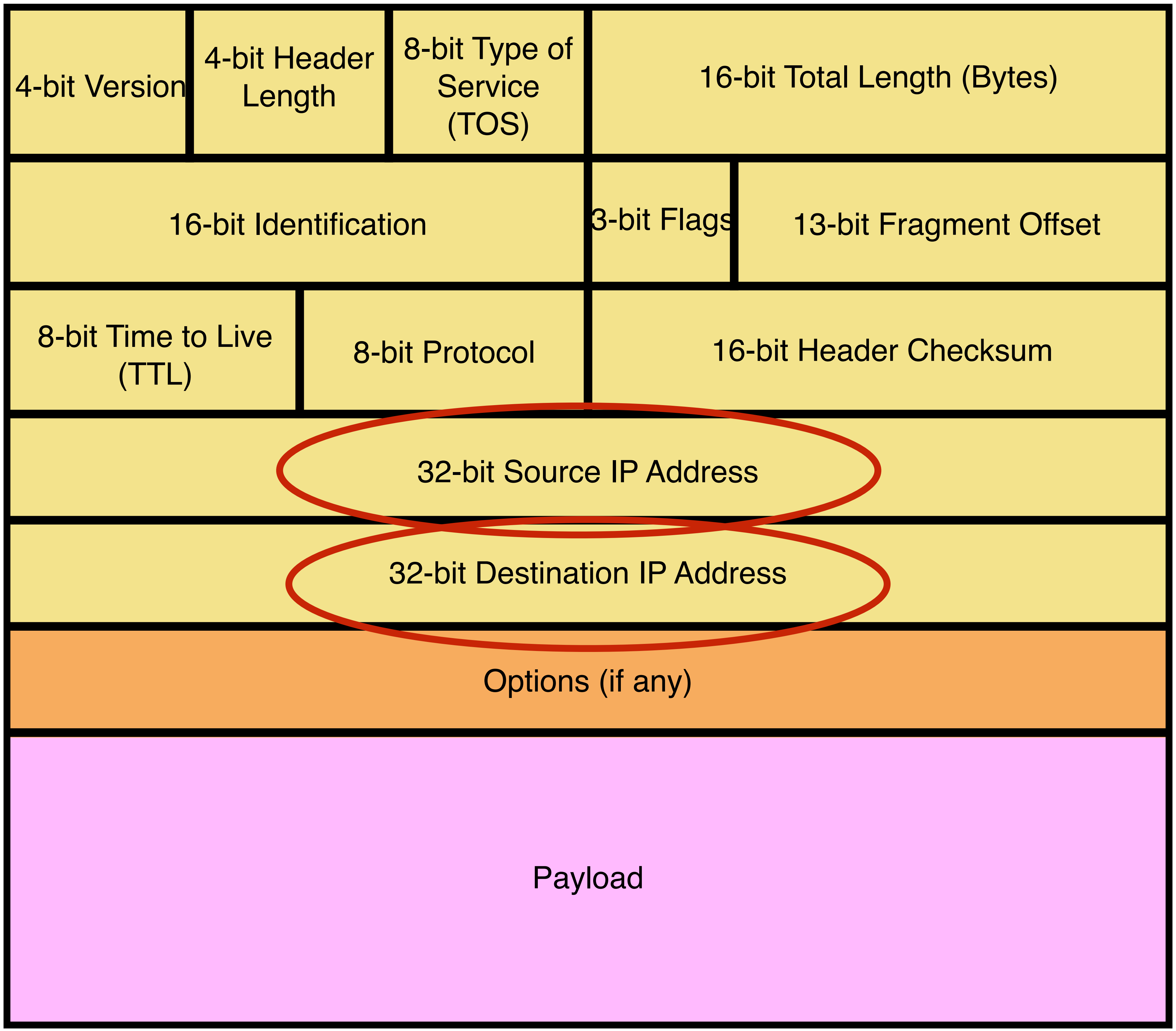
Fields for Reading Packet Correctly



Getting Packet to Destination and Back

- **Two IP addresses**
  - Source IP address (32 bits)
  - Destination IP address (32 bits)
- **Destination Address**
  - Unique locator for the receiving host
  - Allows each node to make forwarding decisions
- **Source Address**
  - Unique locator for the sending host
  - Recipient can decide whether to accept packet
  - Enables recipient to send a reply back to the source

Fields for Reading Packet Correctly



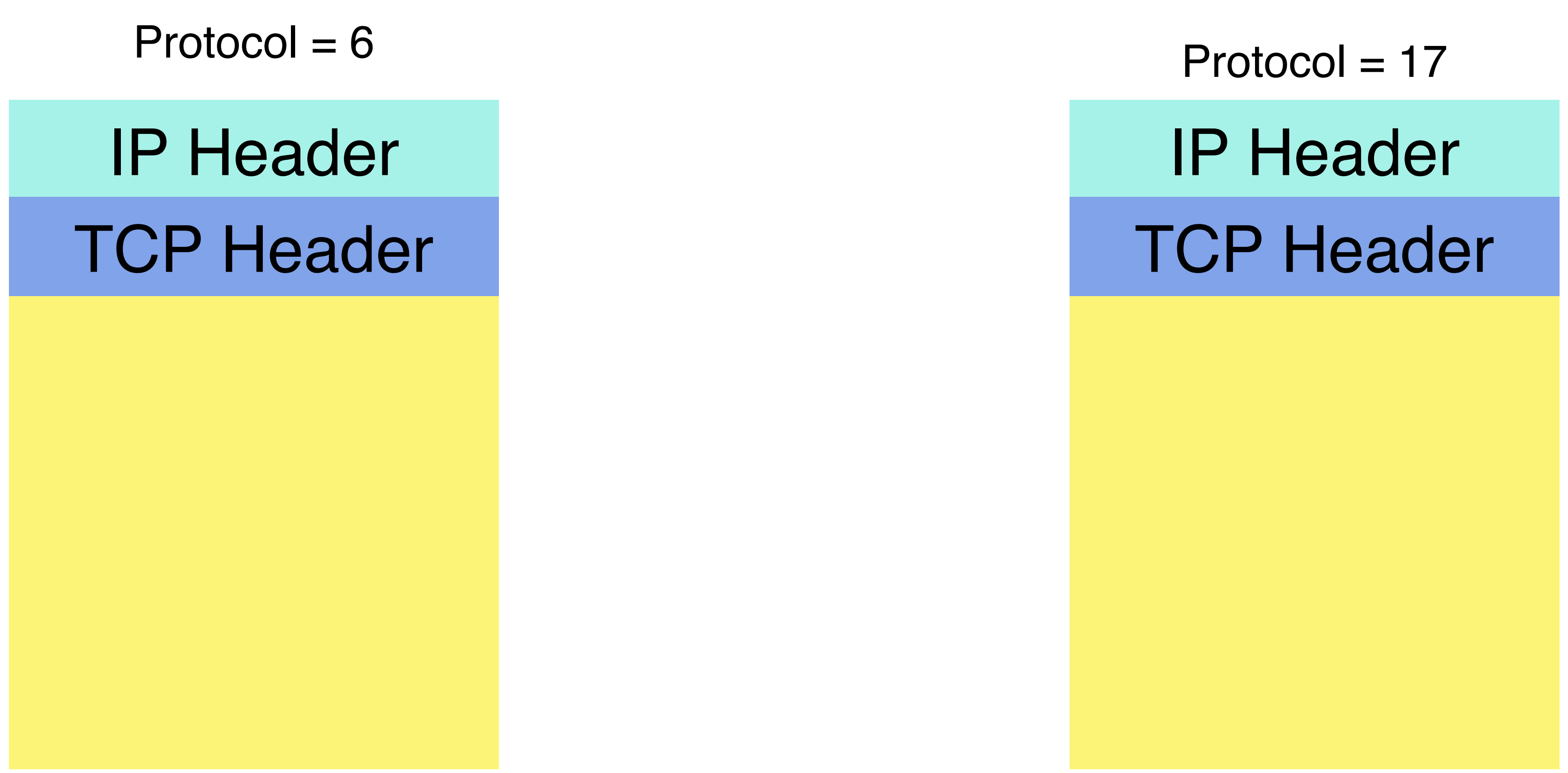
Questions?

List of Tasks

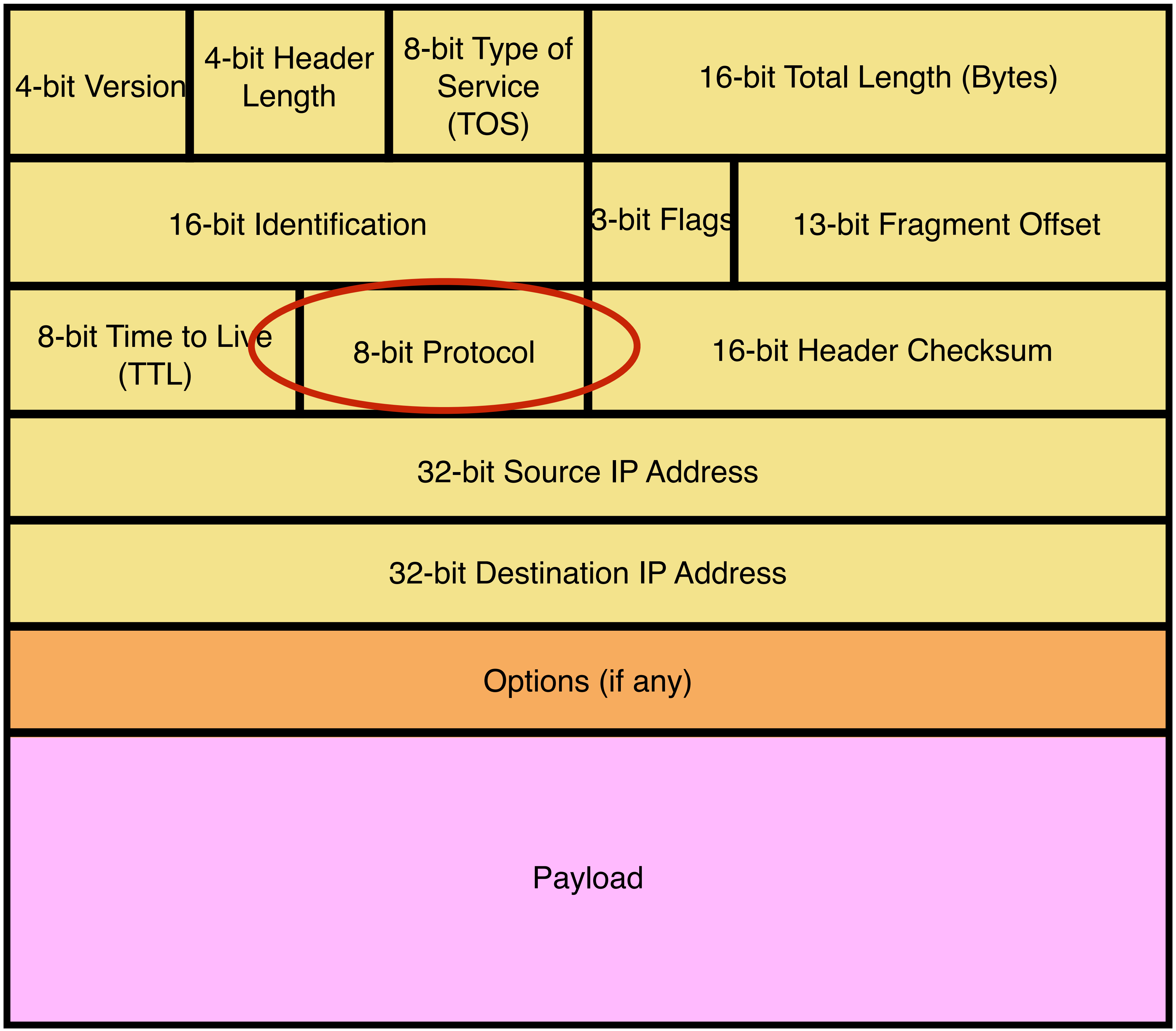
- Read packet correctly
- Get the packet to the destination
- Get responses to the packet back to source
- Carry data
- **Tell host what to do with packet once arrived**
- Specify any special network handling of the packet
- Deal with problems that arise along the path

Telling Host How to Handle Packet

- **Protocol (8 bits)**
  - Identifies the higher level protocol
  - Important for demultiplexing at receiving host
- **Most common examples**
  - E.g., “6” for the Transmission Control Protocol (TCP)
  - E.g., “17” for the User Datagram Protocol



Fields for Reading Packet Correctly



Special Handling

- **Type-of-Service (8-bits)**
  - Allow packets to be treated differently based on needs
  - E.g., low delay for audio, high bandwidth for bulk transfer
  - Has been redefined several times, no general use
- **Options**
  - Ability to specify other functionality
  - Extensible format

Examples of Options

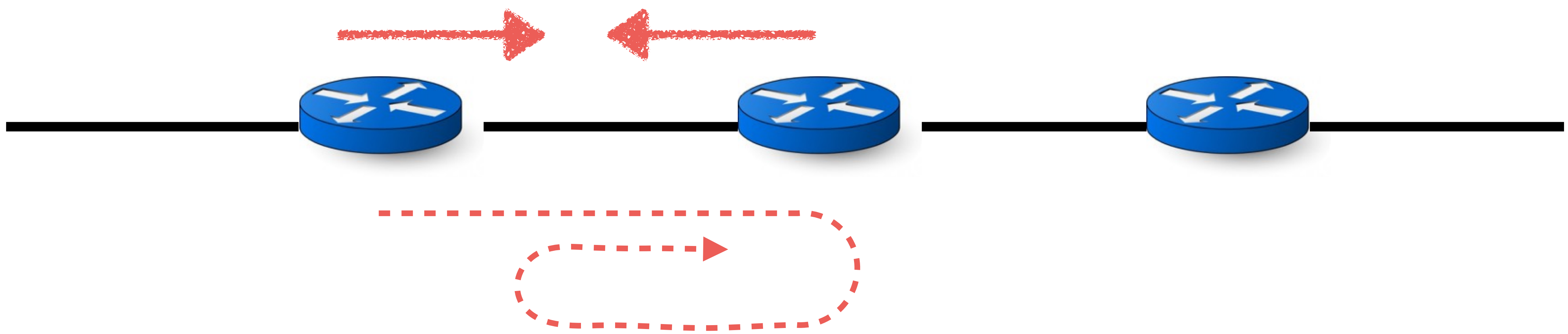
- Record Route
- Strict Source Route
- Loose Source Route
- Timestamp
- Traceroute
- Router Alert
- …

Potential Problems

- Header Corrupted: **Checksum**
- Loop: **TTL**
- Packet too large: **Fragmentation**

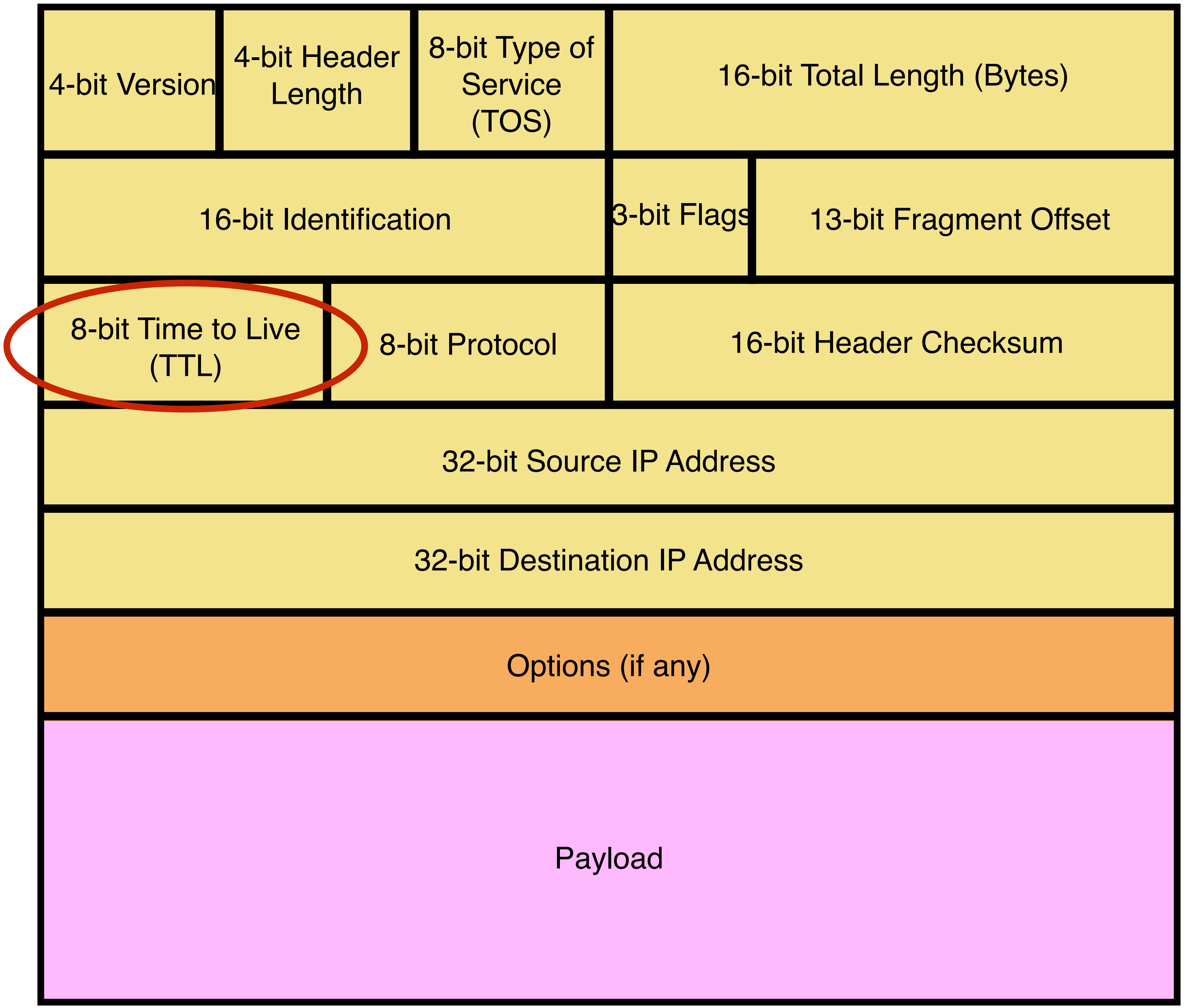
Preventing Loops

- Forwarding loops cause packets to cycle forever
    - As these accumulate, eventually consume all capacity



- Time-to-live (TTL) Field (8-bits)
    - Decremented at each hop, packet discarded if reaches 0
    - … and “time exceeded” message is sent to the source
        - Using “ICMP” control message; basis for traceroute

TTL Field

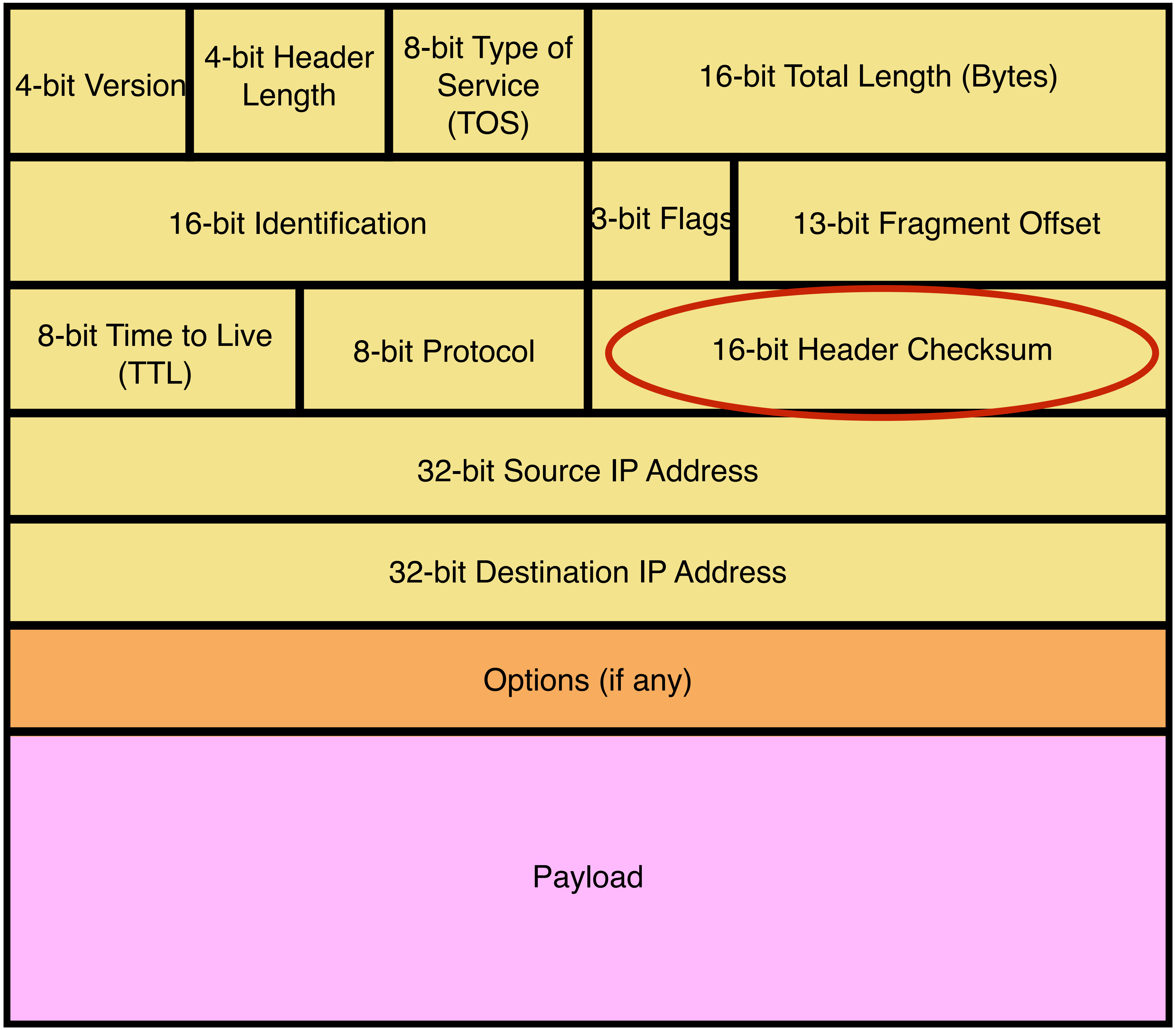


4-bit Version
4-bit Header Length
8-bit Type of Service (TOS)
16-bit Total Length (Bytes)
16-bit Identification
3-bit Flags
13-bit Fragment Offset
8-bit Time to Live (TTL)
8-bit Protocol
16-bit Header Checksum
32-bit Source IP Address
32-bit Destination IP Address
Options (if any)
Payload

Header Corruption

- Checksum (16 bits)
  - Particular form of checksum over packet header
- If not correct, router discards packets
  - So it doesn’t act in bogus information
- Checksum recalculated at every router
  - Why?
  - Why include TTL?
  - Why only header?

Checksum Field



Packet Header as an interface

- **Useless to learn the header format by heart**
    - If you remember the tasks that need to be performed …
    - Understanding **why** header format is what it is …
    - In general: if you understand the problem, solution is easy
    - As the problem evolves, you will know where to look for a solution
- **Transition from IPv4 to IPv6**
    - Gradually happening …
    - If you want to learn a bit, see backup slides

Domain Name System (DNS)

What is DNS?

- **User has name of entity she/he wants to access**
  - E.g., www.cnn.com
  - Content, host, etc.
- However, **Internet routes and forwards requests based on IP addresses**
  - Need to convert name (e.g., www.cnn.com) to an IP address
- **Domain Name System (DNS)**
  - **Provides the mapping from name to IP address**
  - User asks DNS: what is the IP address for www.cnn.com
  - DNS responds: 157.166.255.18

Correctness Requirements

- **Addresses can change underneath**
  - Move www.cnn.com to 4.125.91.21
  - Humans/Applications should be unaffected

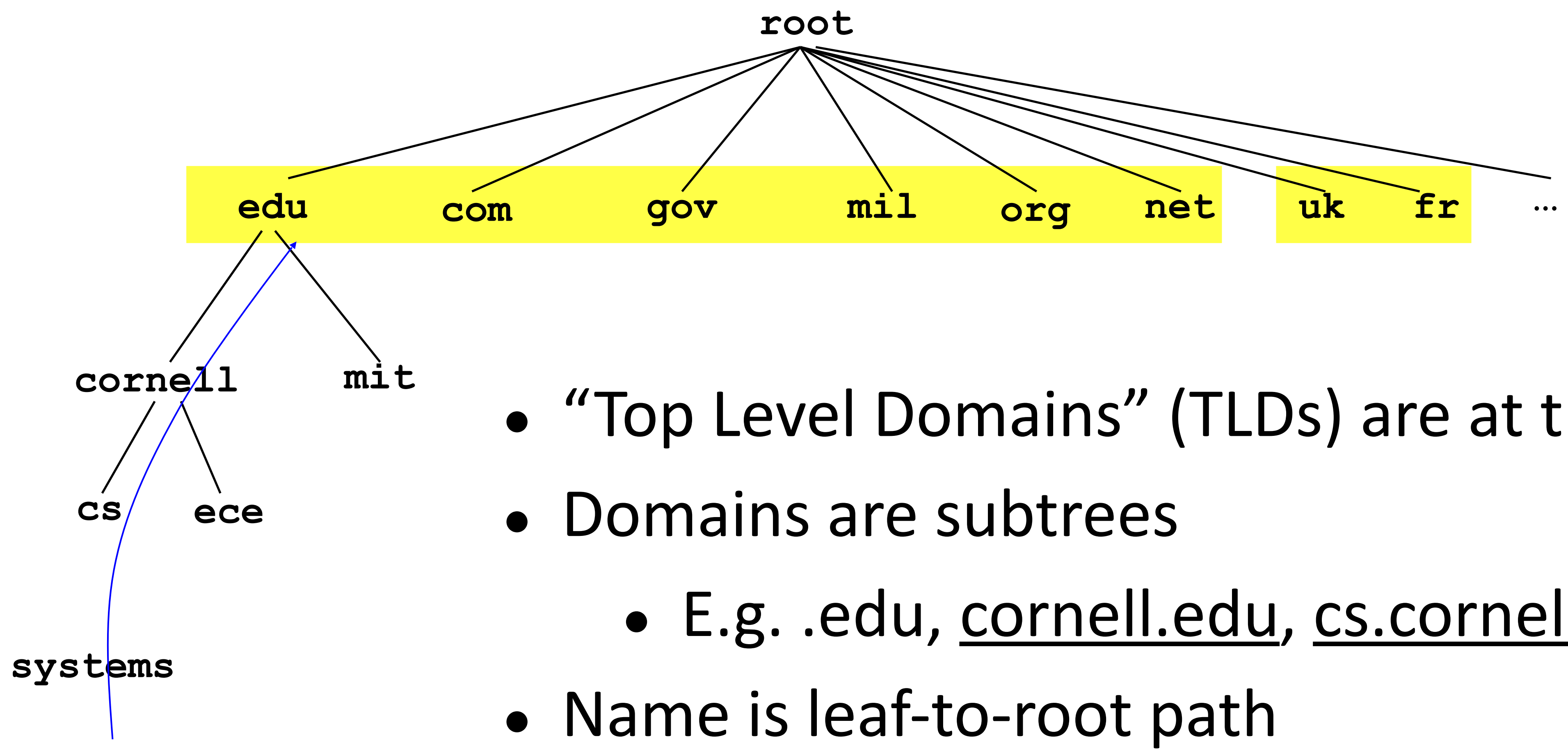
- **Name could map to multiple IP addresses**
  - www.cnn.com to multiple replicas to the Web site
  - To enable “load balancing” or reduced latency
    - Replicas may see different load (eg, due to geographic location)
    - Some replicas may be closer to the user

- **Multiple names for the same address**
  - E.g., www.cnn.com and cnn.com should map to same IP addresses

Goals and Approach

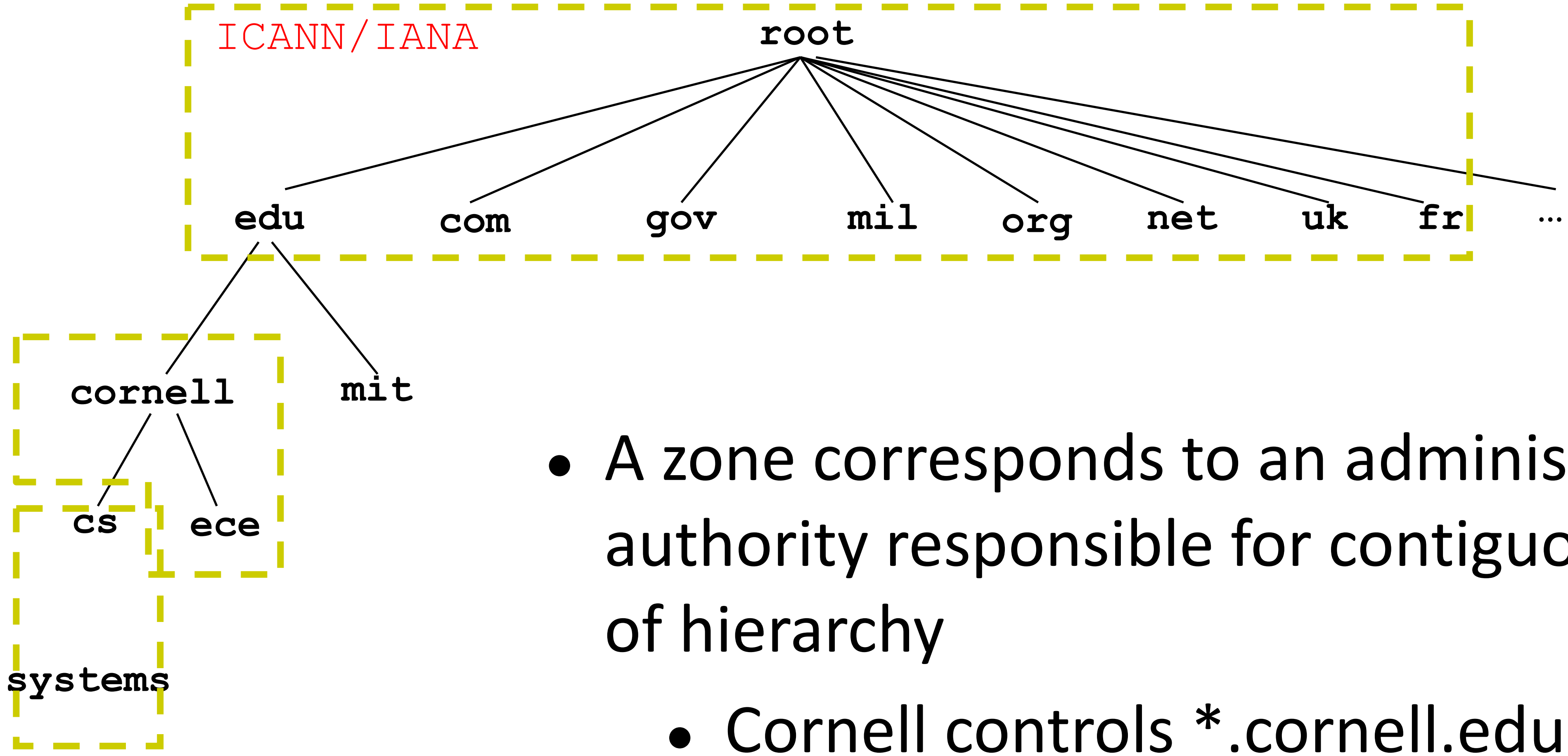
- **Goals**
  - Correctness (from previous slide)
  - Scaling (names, users, updates, etc.)
  - Ease of management (uniqueness of names, etc.)
  - Availability and consistency
  - Fast lookups
- **Approach: Three intertwined hierarchies**
  - Hierarchical Namespace: exploit structure in names
  - Hierarchical Administration: hierarchy of authority over names
  - Hierarchical Infrastructure: hierarchy of DNS servers

Hierarchical Namespace



- "Top Level Domains" (TLDs) are at the top
- Domains are subtrees
  - E.g. .edu, cornell.edu, cs.cornell.edu
- Name is leaf-to-root path
  - systems.cs.cornell.edu

Hierarchical Administration



- A zone corresponds to an administrative authority responsible for contiguous portion of hierarchy
  - Cornell controls *.cornell.edu
  - CS controls *.cs.cornell.edu
- Name collisions trivially avoided
  - Each domain can ensure this locally

Hierarchical Infrastructure

- Top of hierarchy: root
  - Location hardwired into other servers
- Next level: Top Level Domain (TLD) servers
  - .com, .edu, etc.
- Bottom level: Authoritative DNS servers
  - Actually do the mapping
  - Can be maintained locally or by a service provider

Who Knows What?

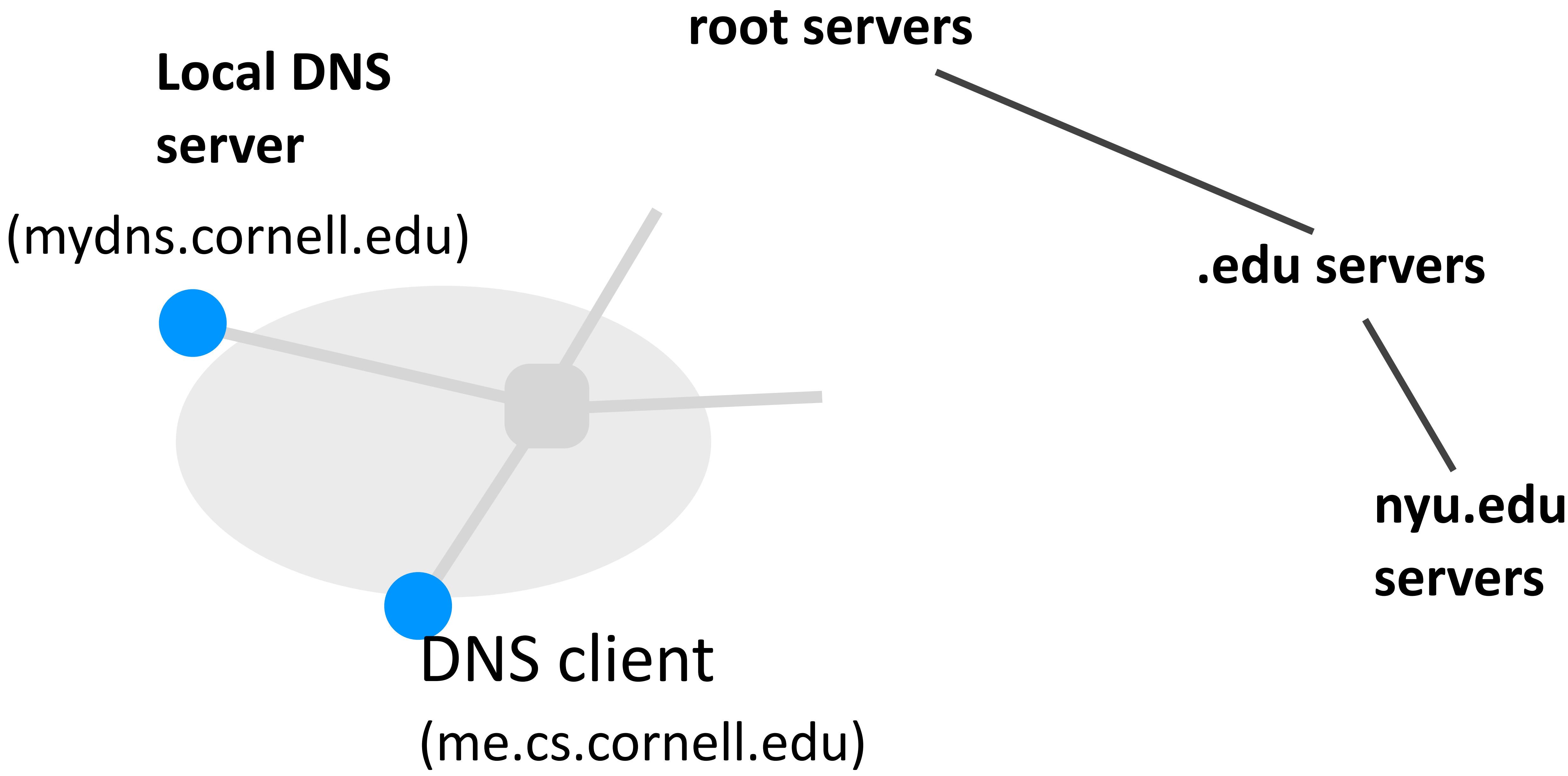
- Every server knows address of root name server
- Root servers know the address of all TLD servers
- Every node knows the address of all children
- An ***authoritative*** DNS server stores name-to-address mappings (“resource records”) for all DNS names in the domain that it has authority for
- Therefore, each server:
  - Stores only a subset of the total DNS database (scalable!)
  - Can discover server(s) for any portion of the hierarchy

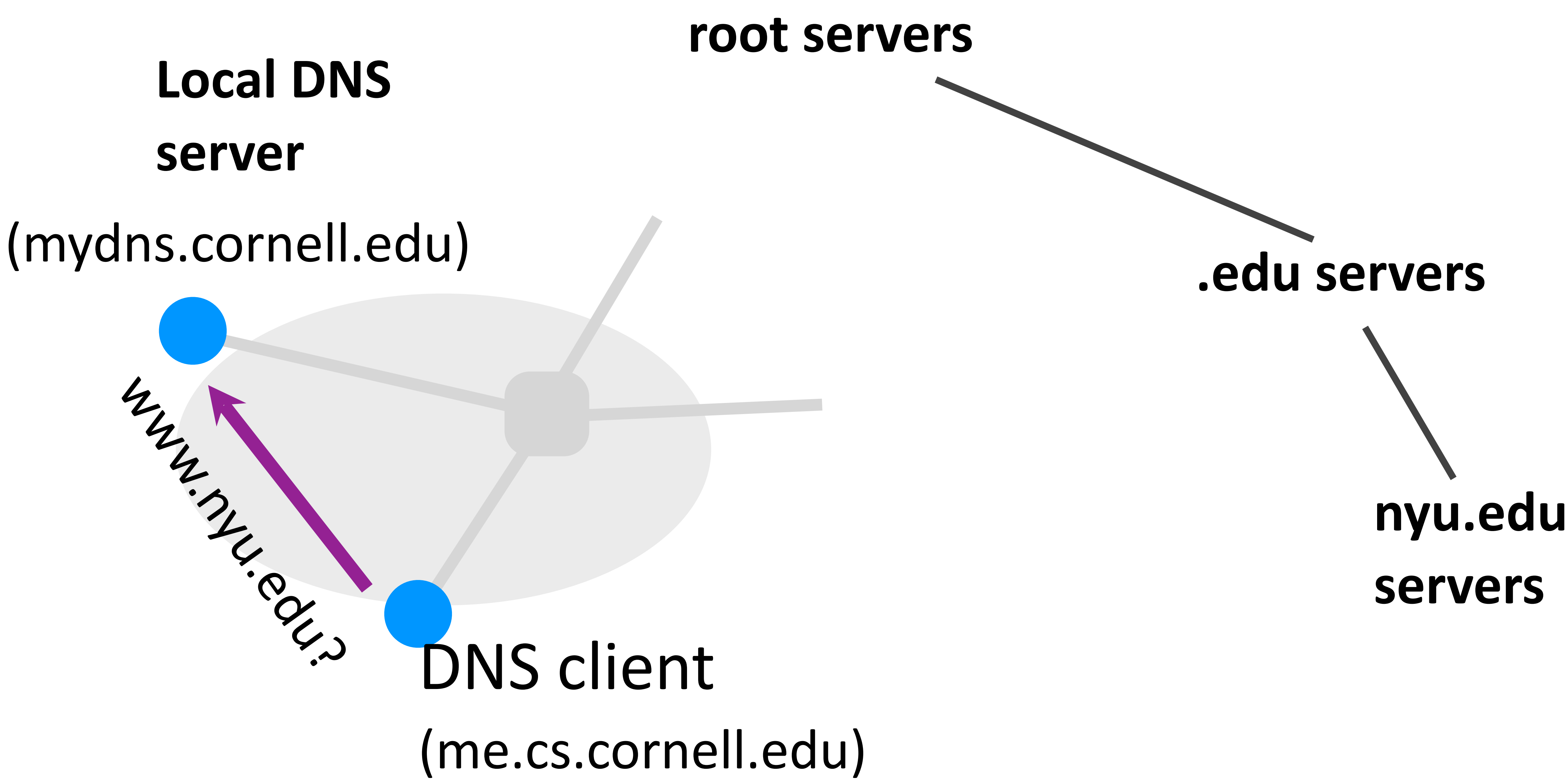
Using DNS

- Two components
  - Local DNS servers
  - Resolver software on hosts

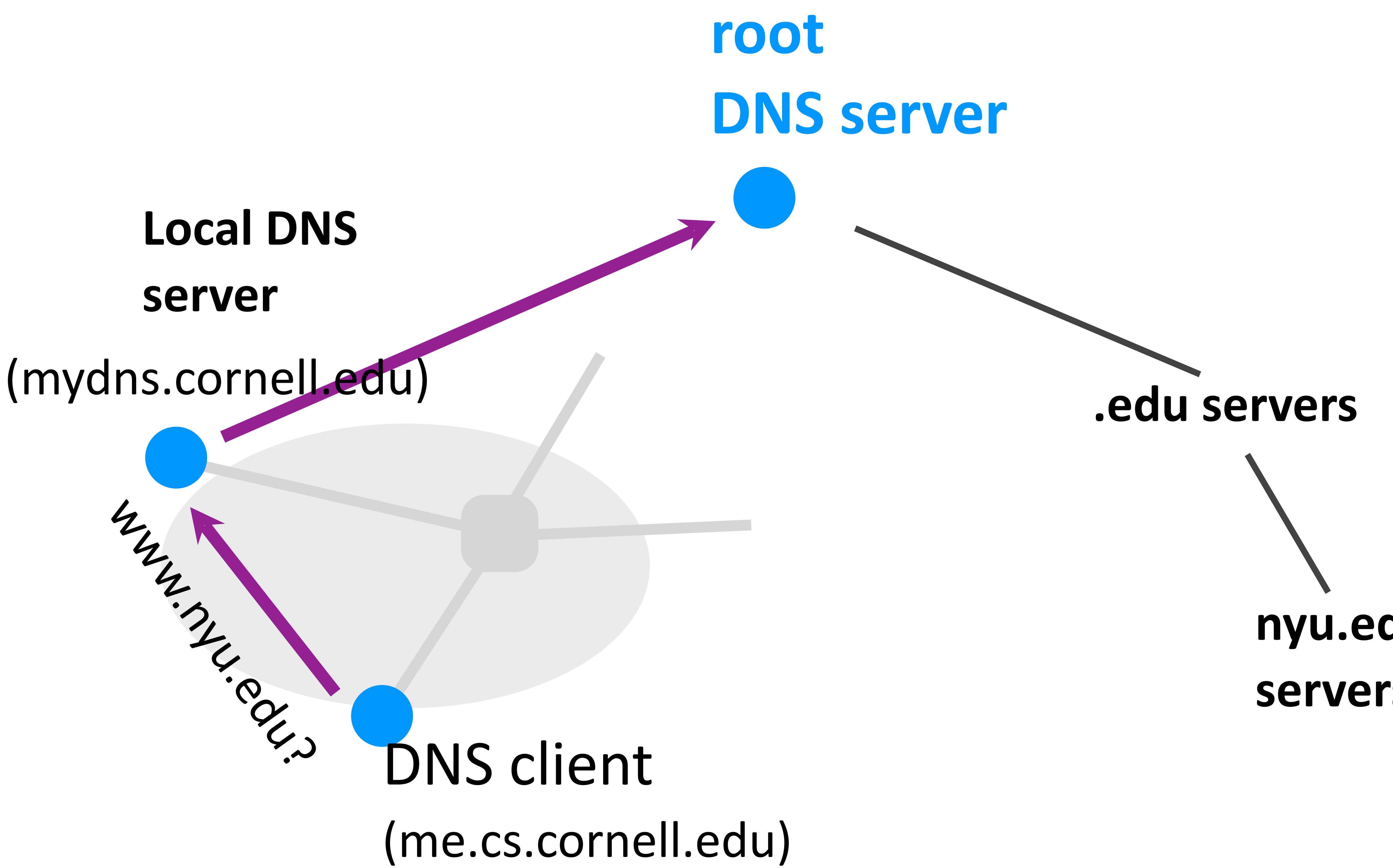
- Local DNS server ("default name server")
  - Usually near the end hosts that use it
  - Local hosts configured with local server (e.g, /etc/resolv.conf) or learn server via DHCP (to be discussed soon)

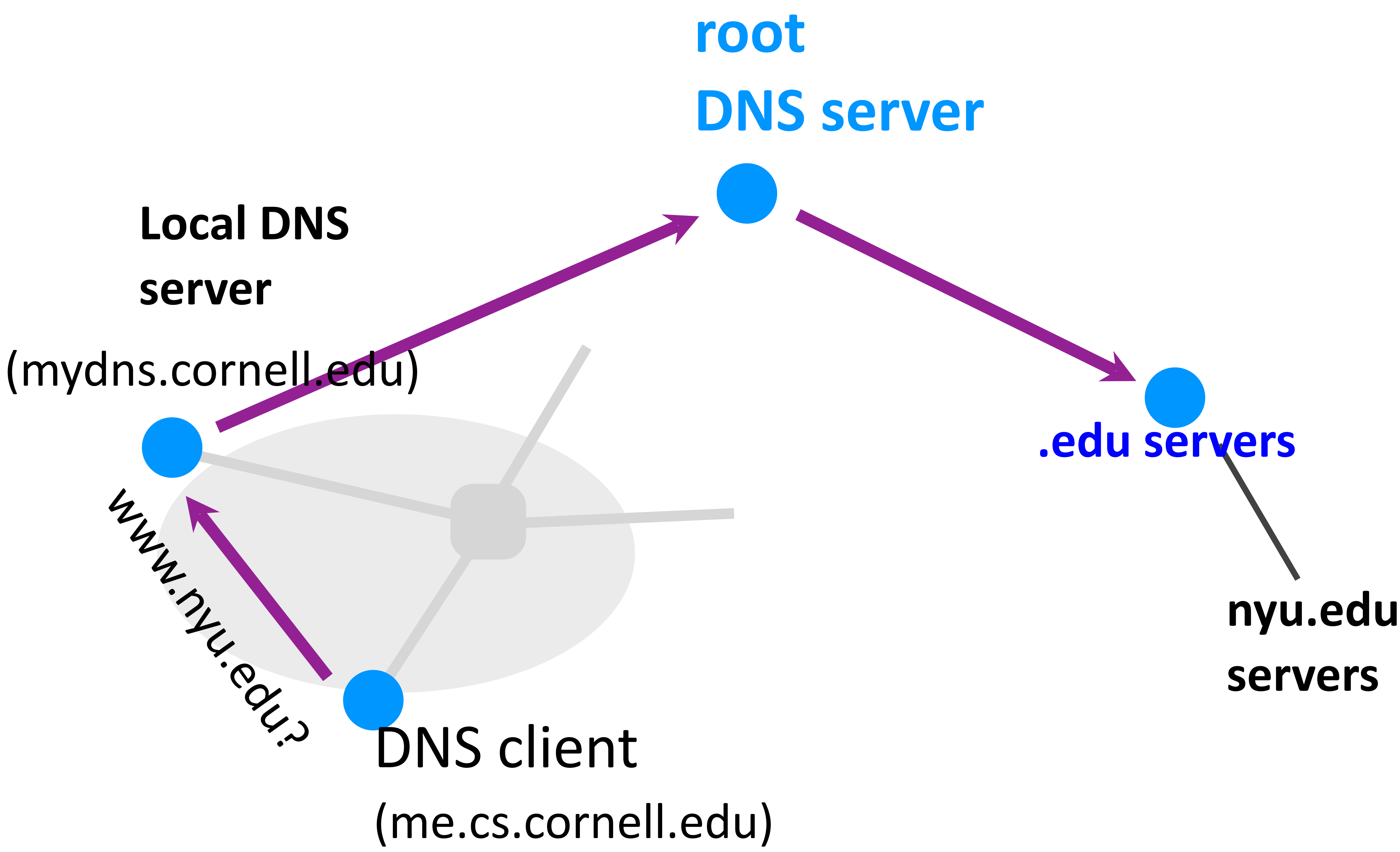
- Client application
  - Obtain DNS name (e.g., from the URL)
  - Do **gethostbyname()** to trigger resolver code
  - Which then sends request to local DNS server


root servers
Local DNS
server
(mydns.cornell.edu)
.edu servers
nyu.edu
servers
DNS client
(me.cs.cornell.edu)

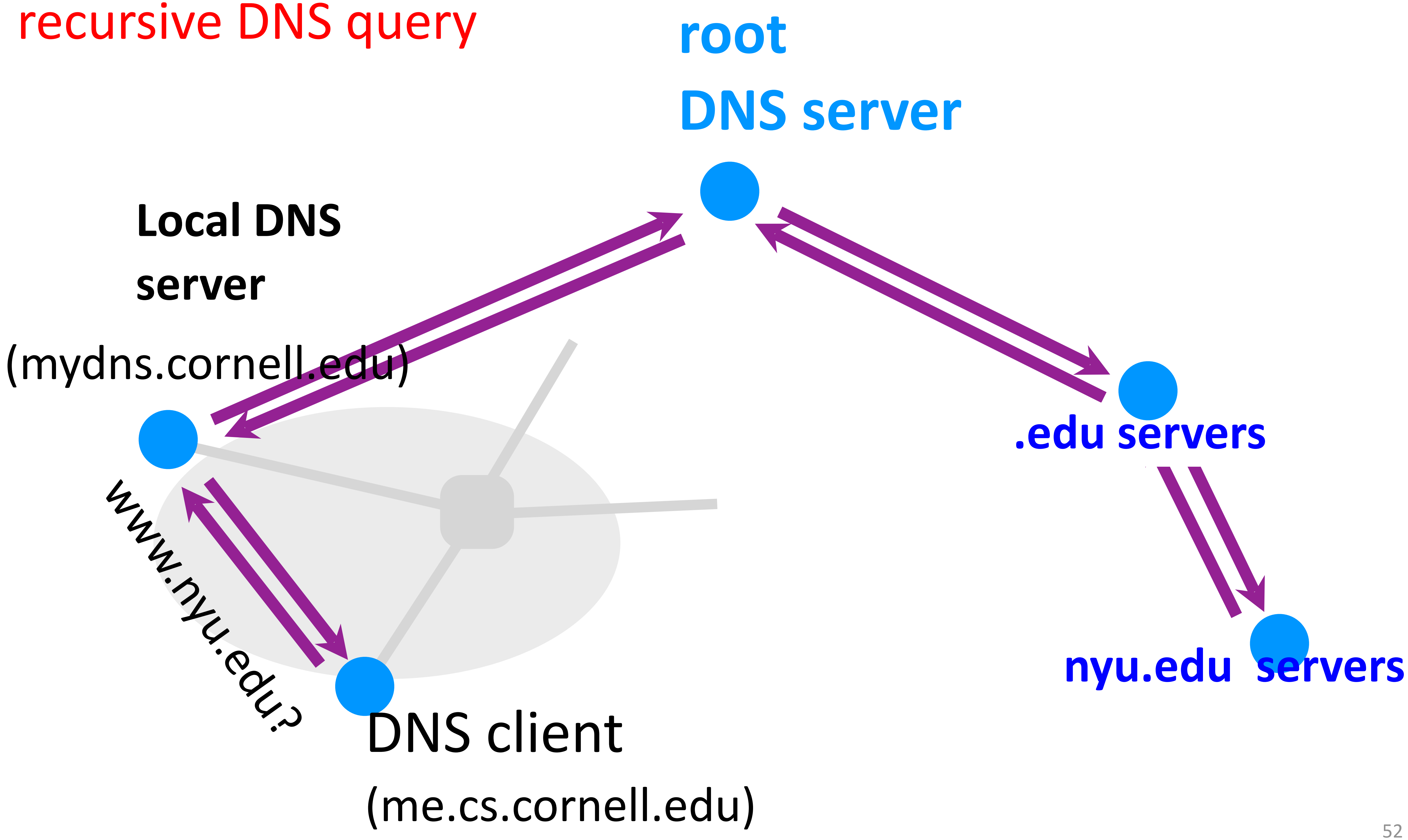


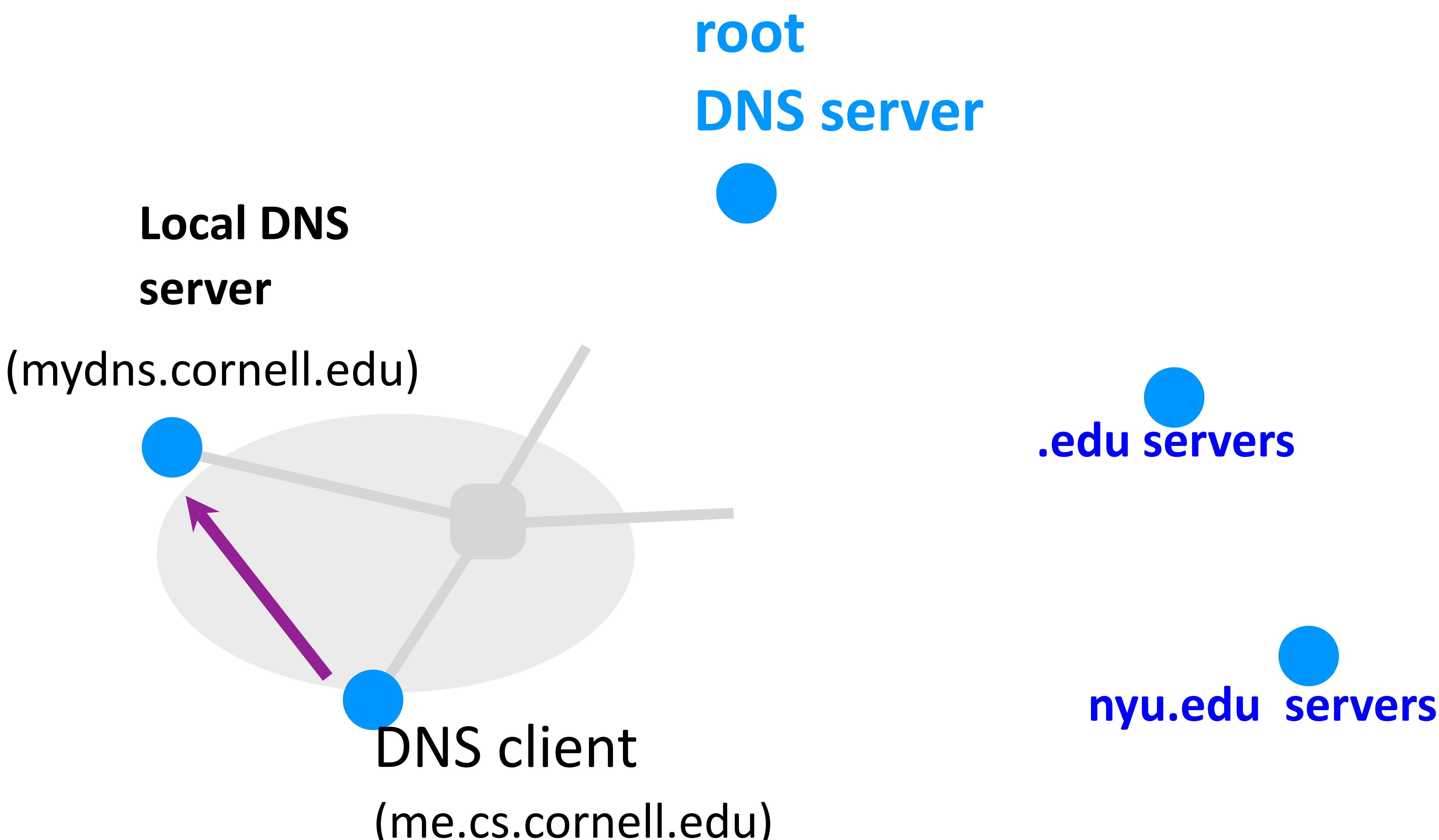
root servers
Local DNS
server
(mydns.cornell.edu)
.edu servers
www.nyu.edu?
nyu.edu
servers
DNS client
(me.cs.cornell.edu)


root
DNS server
Local DNS
server
(mydns.cornell.edu)
.edu servers
www.nyu.edu?
nyu.edu
servers
DNS client
(me.cs.cornell.edu)

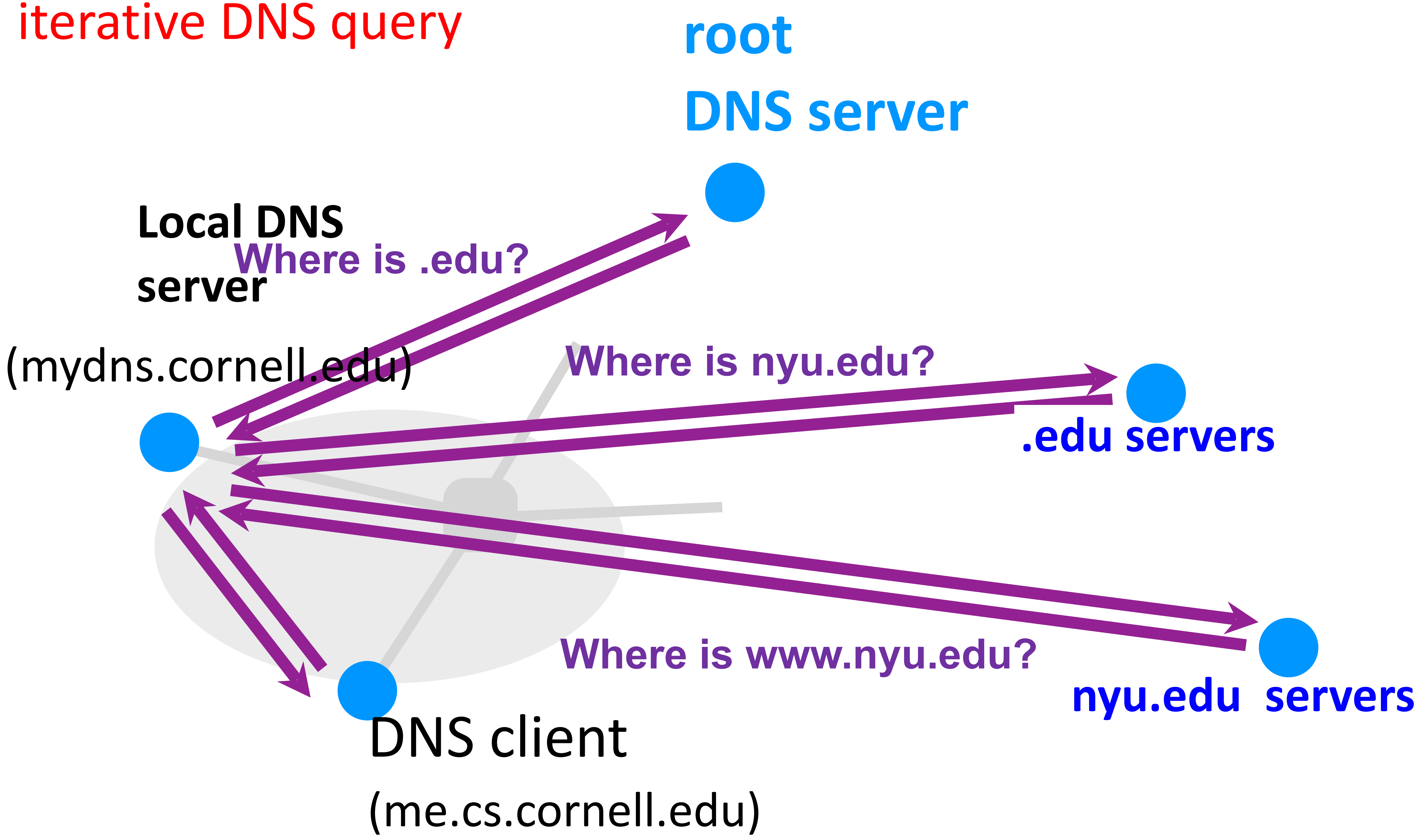

root
DNS server
Local DNS
server
(mydns.cornell.edu)
.edu servers
nyu.edu
servers
www.nyu.edu?
DNS client
(me.cs.cornell.edu)

recursive DNS query




root
DNS server
Local DNS
server
(mydns.cornell.edu)
.edu servers
nyu.edu servers
DNS client
(me.cs.cornell.edu)

iterative DNS query



Discovery Protocols

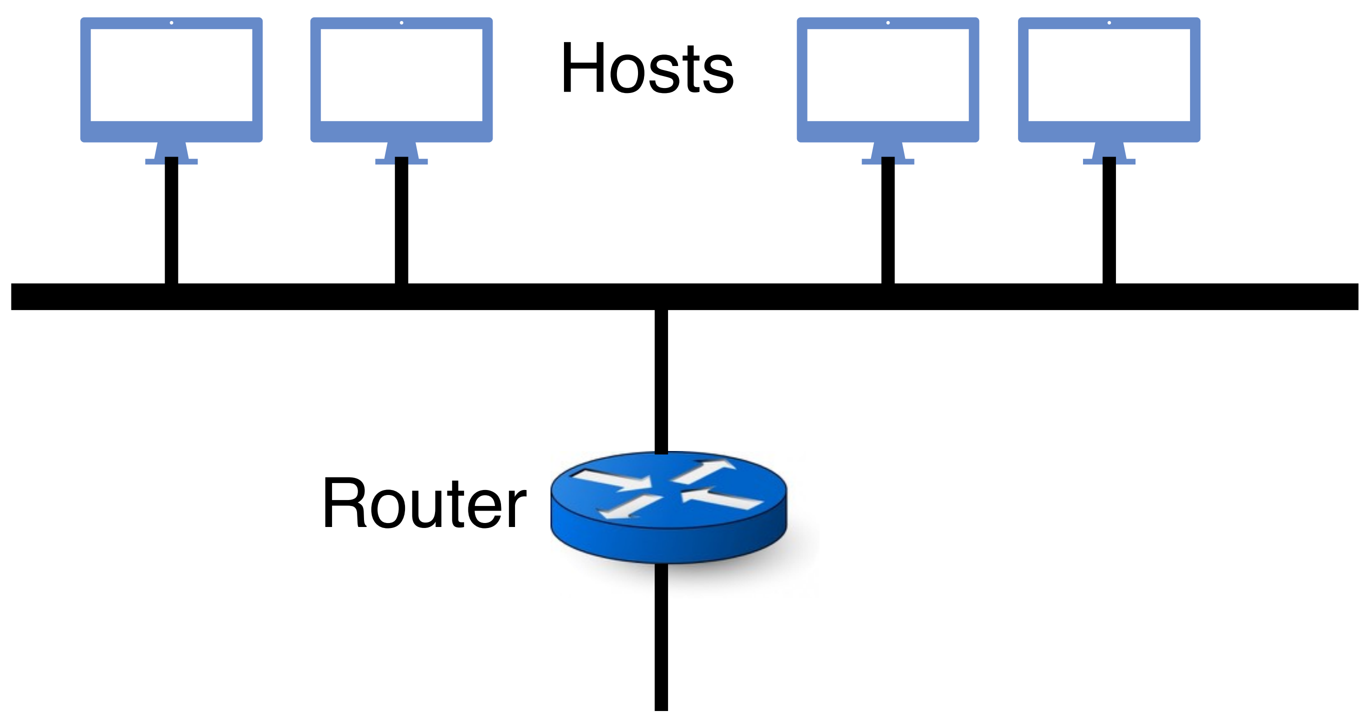
Suppose Host A wants to communication with Host B

Discovery

- Suppose I am host A
- I want to communicate with B (say, www.google.com)
- I was “born” knowing **only** my name — my MAC address :-)
- Must discover some information before I can communicate with B
    - What is my IP address?
    - What is B’s IP address?
        - Using DNS
    - Is B within my LAN?
    - If yes, what is B’s MAC address?
    - If not, what is the address of my first-hop router to B?
    - …

DHCP and ARP

- Link layer discovery protocols
  - DHCP — Dynamic Host Configuration Protocol
  - ARP — Address Resolution Protocol
  - Configured to a single LAN
  - Rely on broadcast capability



DHCP and ARP

- Link layer discovery protocols
- Serve two functions
  1. Discovery of local end-hosts
     - For communication between hosts on the same LAN
  2. Bootstrap communication with remote hosts
     - What’s my IP address?
     - Who/where is my local DNS server?
     - Who/where is my first hop router?

DHCP

- Dynamic Host Configuration Protocol
  - Defined in RFC 2131
- A host uses DHCP to discover
  - Its own IP address
  - Subnet masks — allows to test whether an IP address is local or not
  - IP address(es) for its local DNS name server(s)
  - IP address(es) for its first-hop “default” router(s)

DHCP: operation

1. One or more local DHCP servers maintain required information
   - IP address pool, netmask, DNS servers, etc.
   - Application that listens on UDP port 67

DHCP: operation

1. One or more local DHCP servers maintain required information

2. Client broadcasts a DHCP discovery message
    - L2 broadcast, to MAC address FF:FF:FF:FF:FF:FF

DHCP: operation

1. One or more local DHCP servers maintain required information

2. Client broadcasts a DHCP discovery message

3. One or more DHCP servers respond with a DHCP “offer” message
    - Proposed IP address for client, lease time
    - Other parameters

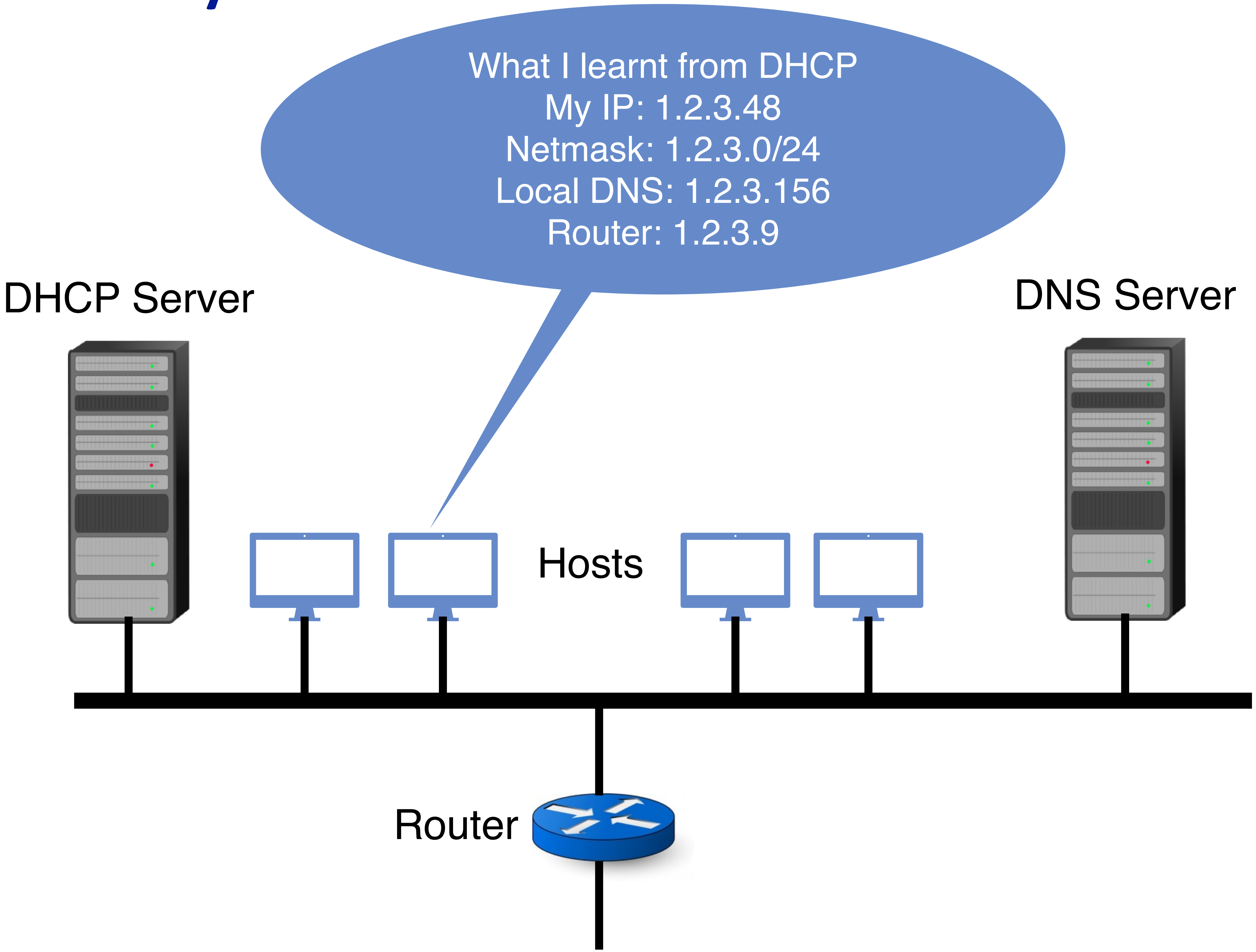
DHCP: operation

1. One or more local DHCP servers maintain required information
2. Client broadcasts a DHCP discovery message
3. One or more DHCP servers respond with a DHCP “offer” message
4. Client broadcasts a DHCP request message
    - Specifies which offer it wants
    - Echoes accepted parameters
    - Other DHCP servers learn they were not chosen

DHCP: operation

1. One or more local DHCP servers maintain required information
2. Client broadcasts a DHCP discovery message
3. One or more DHCP servers respond with a DHCP “offer” message
4. Client broadcasts a DHCP request message
5. Selected DHCP server responds with an ACK

Are we there yet?



ARP: Address Resolution Protocol

- Every host maintains an ARP table
  - List of (IP address — MAC address) pairs
  - For IP addresses within the same LAN
- Consult the table when sending a packet
  - Map destination IP address to destination MAC address
- But: what if IP address not in the table?
  - Either its not local (detected using DHCP)
  - If its local:
    - Sender broadcasts: “Who has IP address 1.2.3.156?”
    - Caches the answer in ARP table

Key Ideas in Both ARP and DHCP

- Broadcasting: can use broadcast to make contact
  - Scalable because of limited size
- Caching: remember the past for a while
  - Store the information you learn to reduce overhead

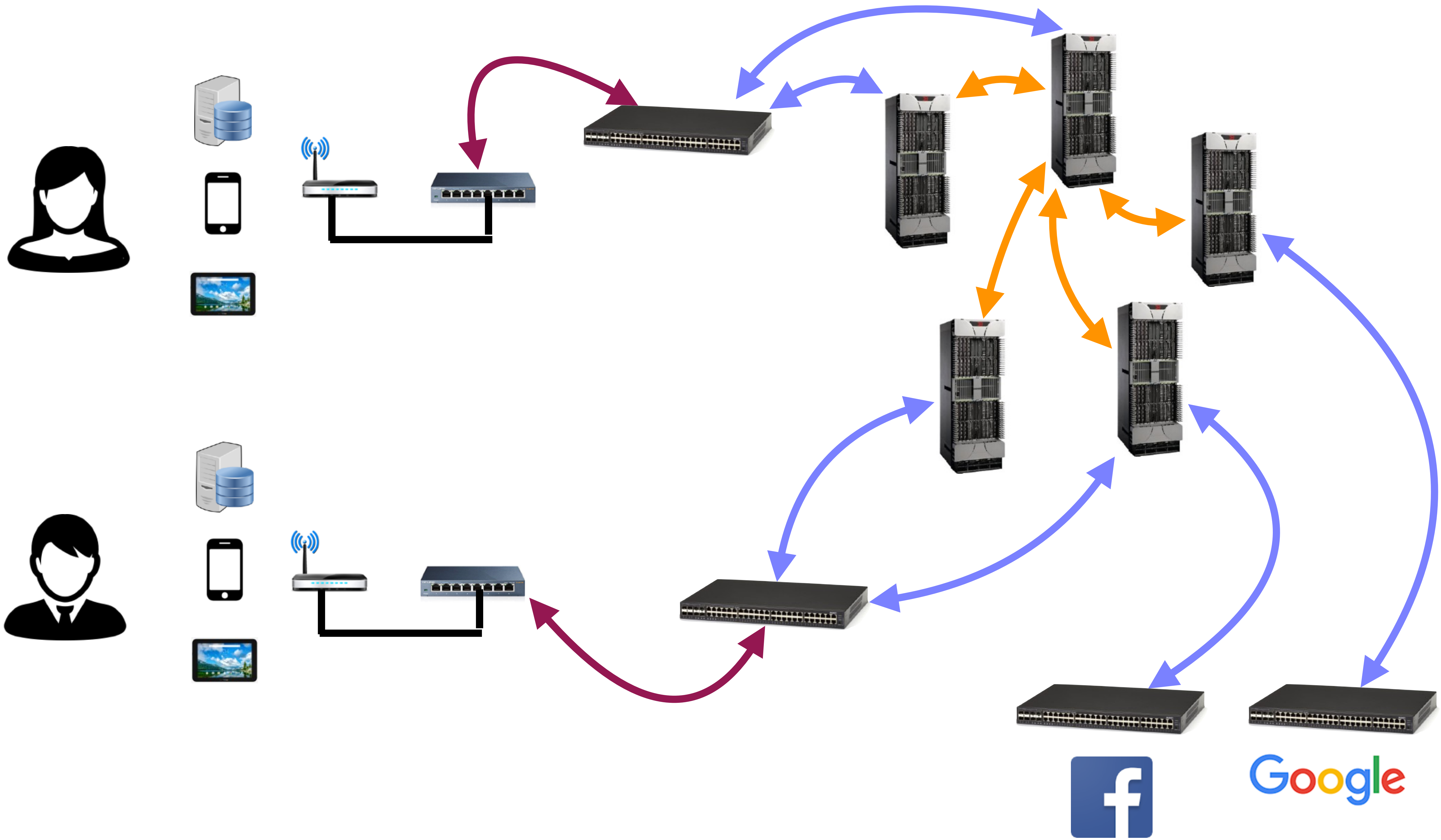
Taking Stock: Discovery

Layer	Examples	Structure	Configuration	Resolution Service
App Layer	www.cs.cornell.edu	Organizational hierarchy	~ manual	
				DNS
Network Layer	123.45.6.78	Topological hierarchy	DHCP	
				ARP
Link Layer	45-CC-4E-12-F0-97	Vendor(flat)	Hard-coded	

Putting all the pieces together

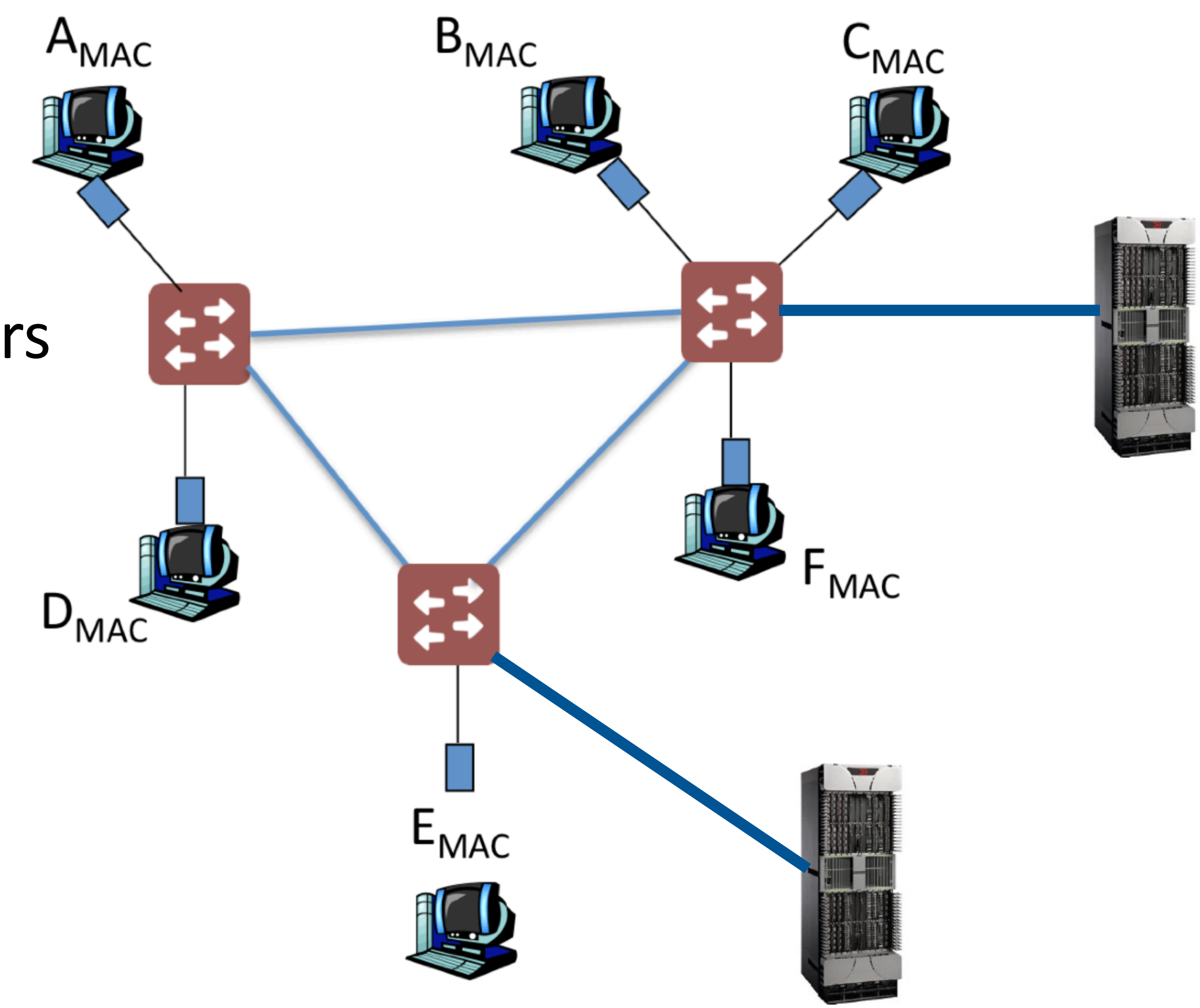
What is a computer network?

A set of network elements connected together, that implement a set of protocols for the purpose of sharing resources at the end hosts

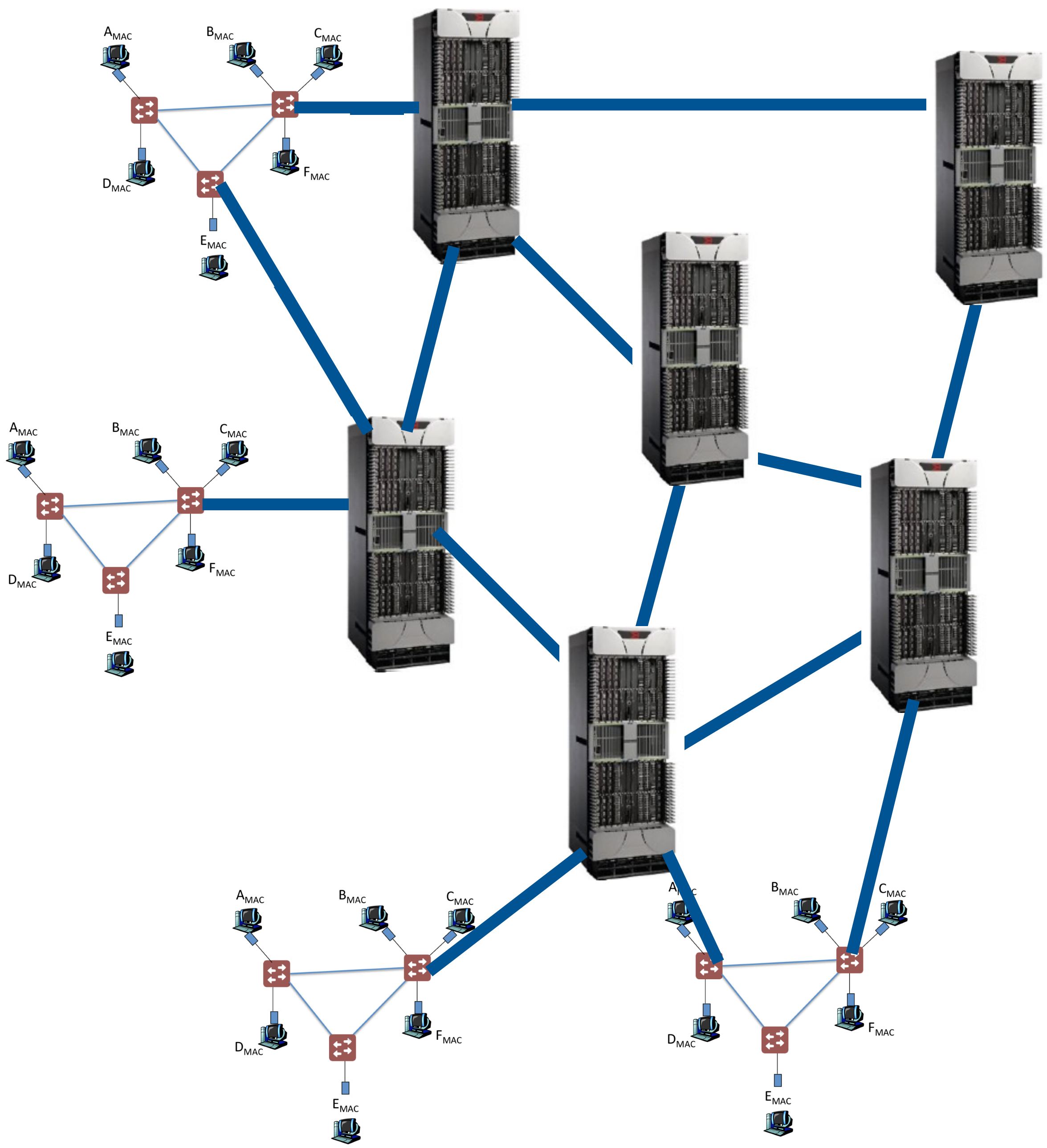


What does Internet actually look like?

- **The smallest component:**
  - A Network Interface Card (NIC), or a machine, or a server
  - Has a Link Layer MAC name/address
- **Multiple NICs connected in a Local Area Network (LAN) via**
  - Broadcast Ethernet,
  - Or, Switched Ethernet
- **Switches in LAN**
  - Connected to larger routers

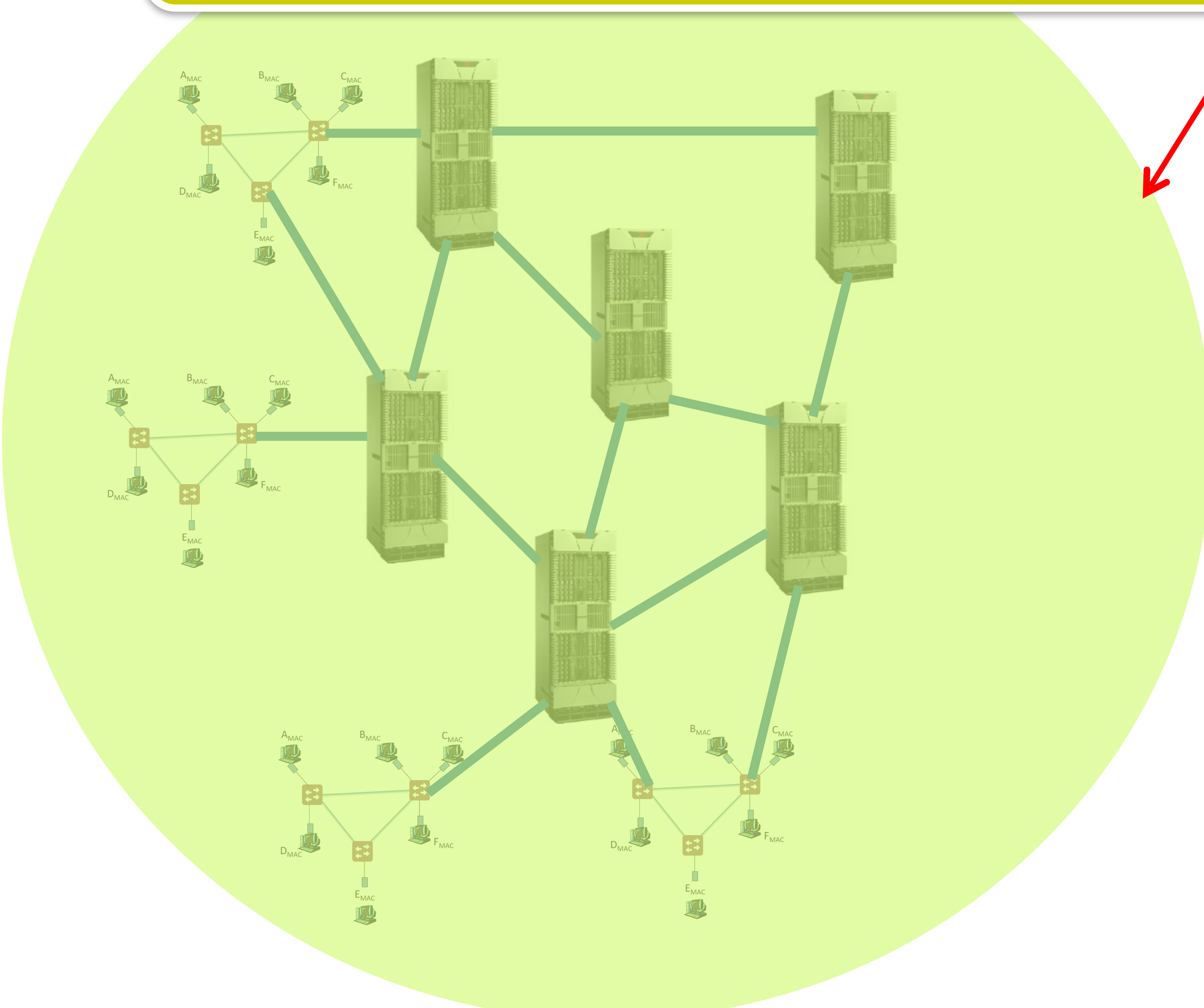


What does Internet actually look like?



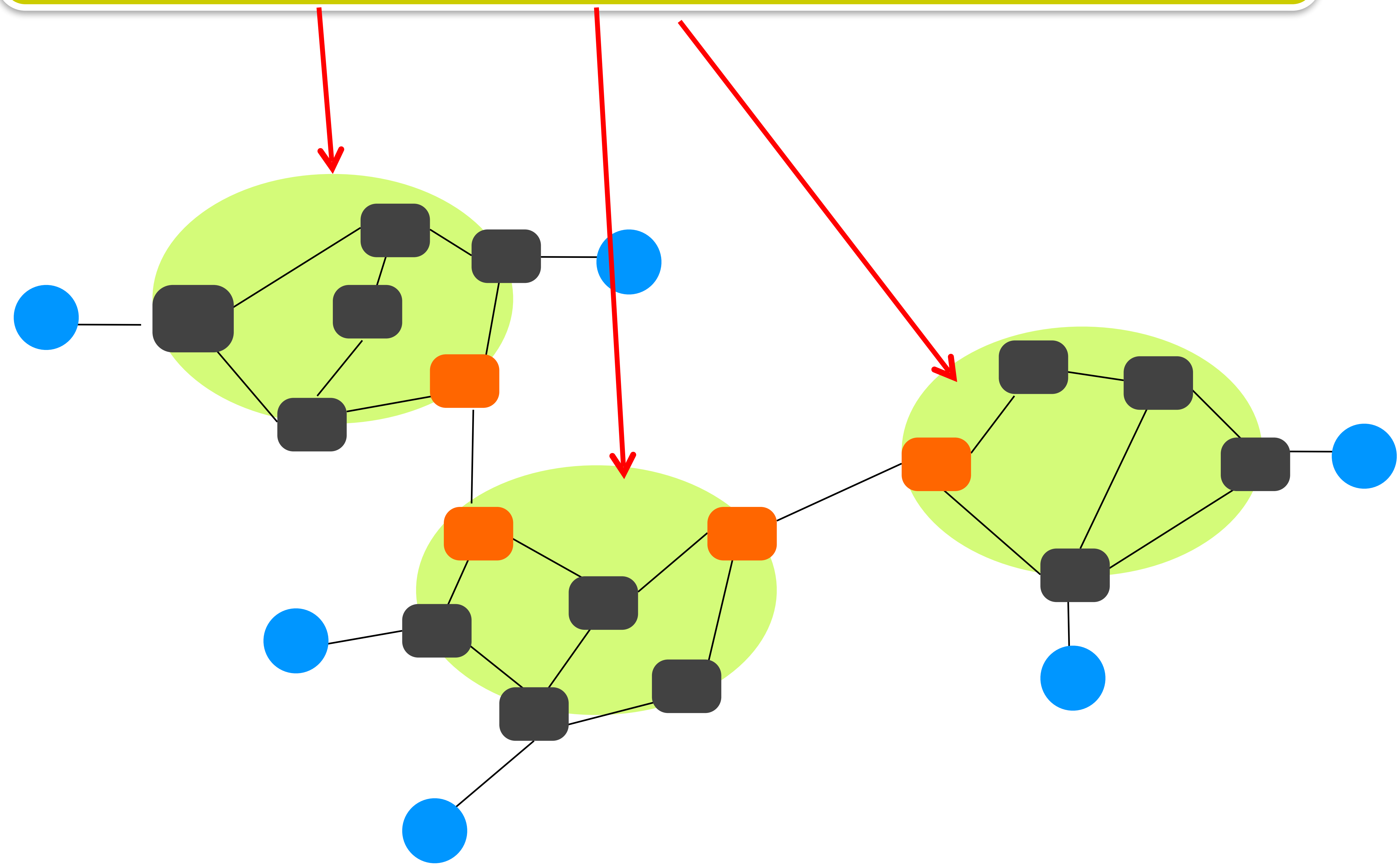
What does Internet actually look like?

"Autonomous System (AS)" or "Domain"
Region of a network under a single administrative entity

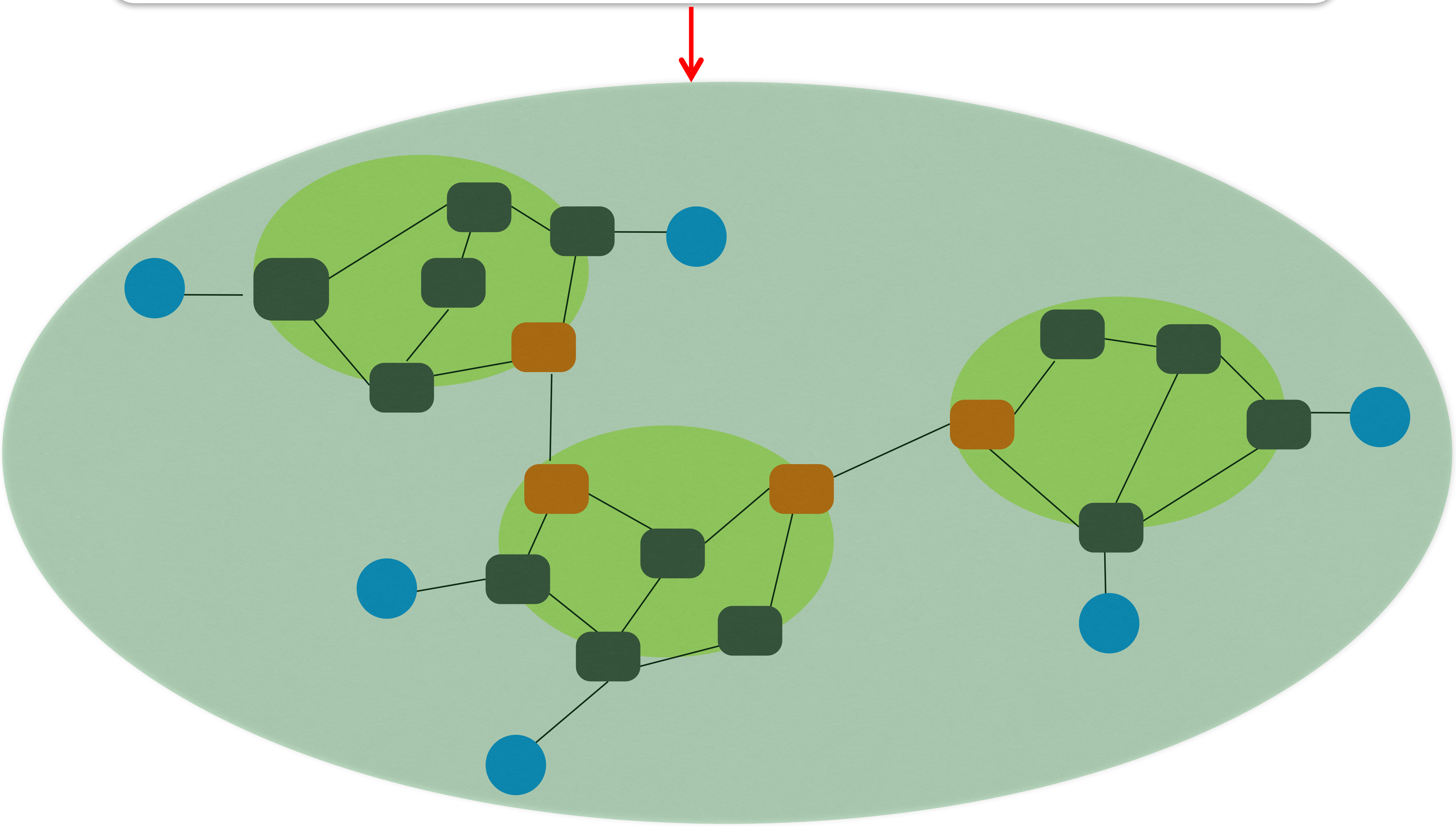


What does Internet actually look like?

Multiple “Autonomous Systems (AS)” or “Domains”
connect together using Border Routers



This entire infrastructure is a part of the INTERNET :-)

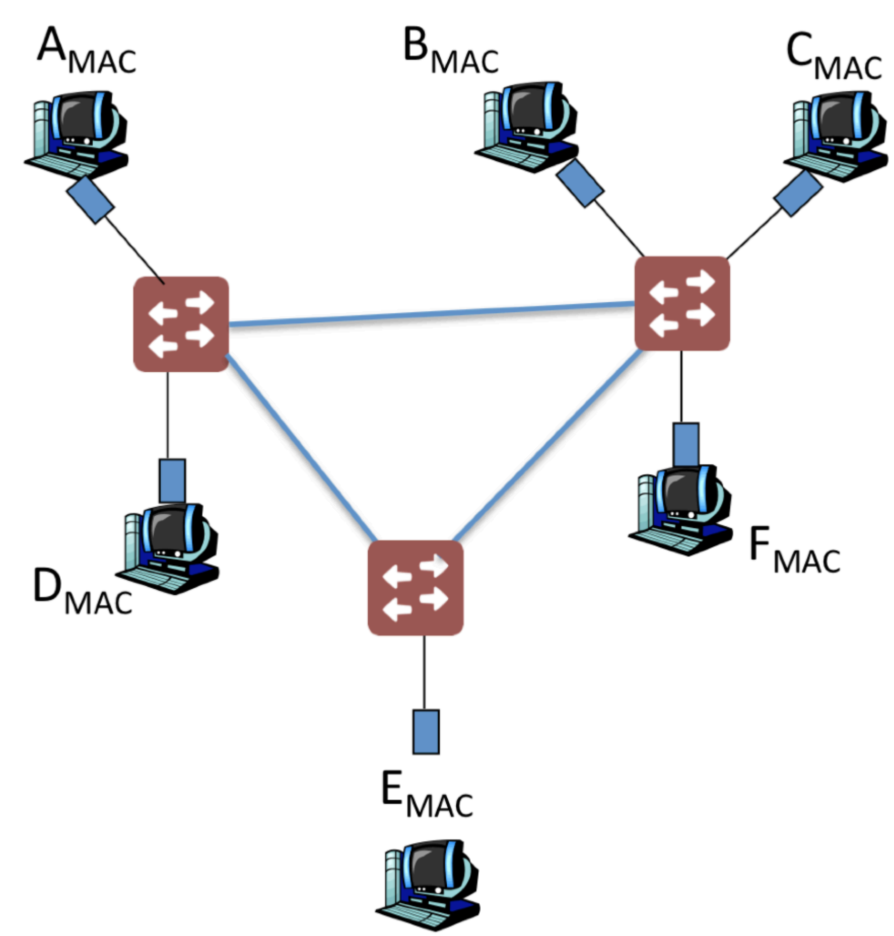


What is the other part of the Internet?

Protocols!

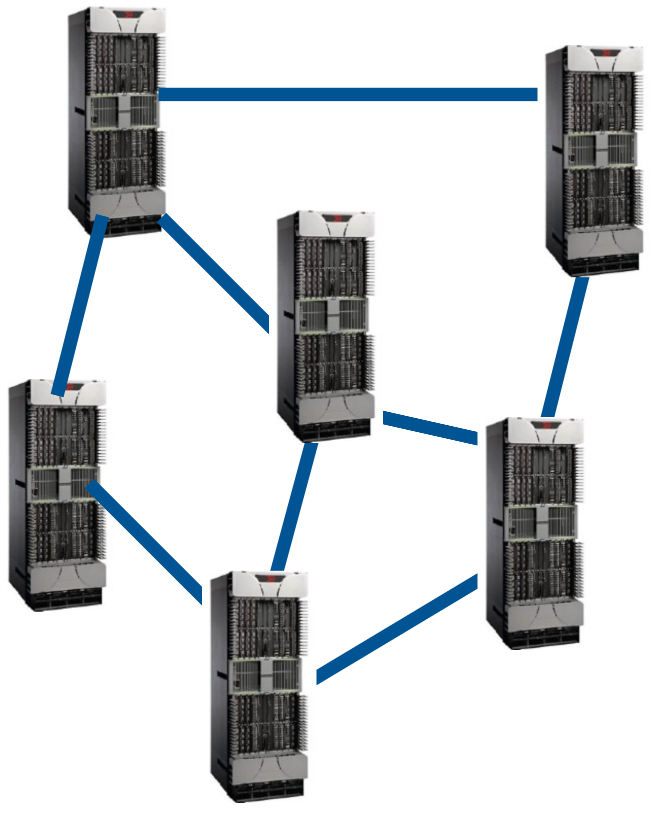
What protocols have we learnt on LAN?

- **Addresses**
  - Link Layer MAC names/addresses: come with the hardware
- **CSMA/CD Protocol:**
  - For transmitting frames on broadcast Ethernet
- **Spanning Tree Protocol:**
  - For transmitting frames on switched Ethernet



What have we learnt beyond LAN?

- **Link-state and Distance-vector Protocols:**
  - For finding routes (and a next-hop) to an IP address within an ISP
- **Border Gateway Protocol:**
  - For finding routes to an IP address range
- **Forwarding at routers**
  - Store **routing tables** (map **destination prefixes** to outgoing port)
  - Longest prefix match for destination address lookup



How does the Internet work?

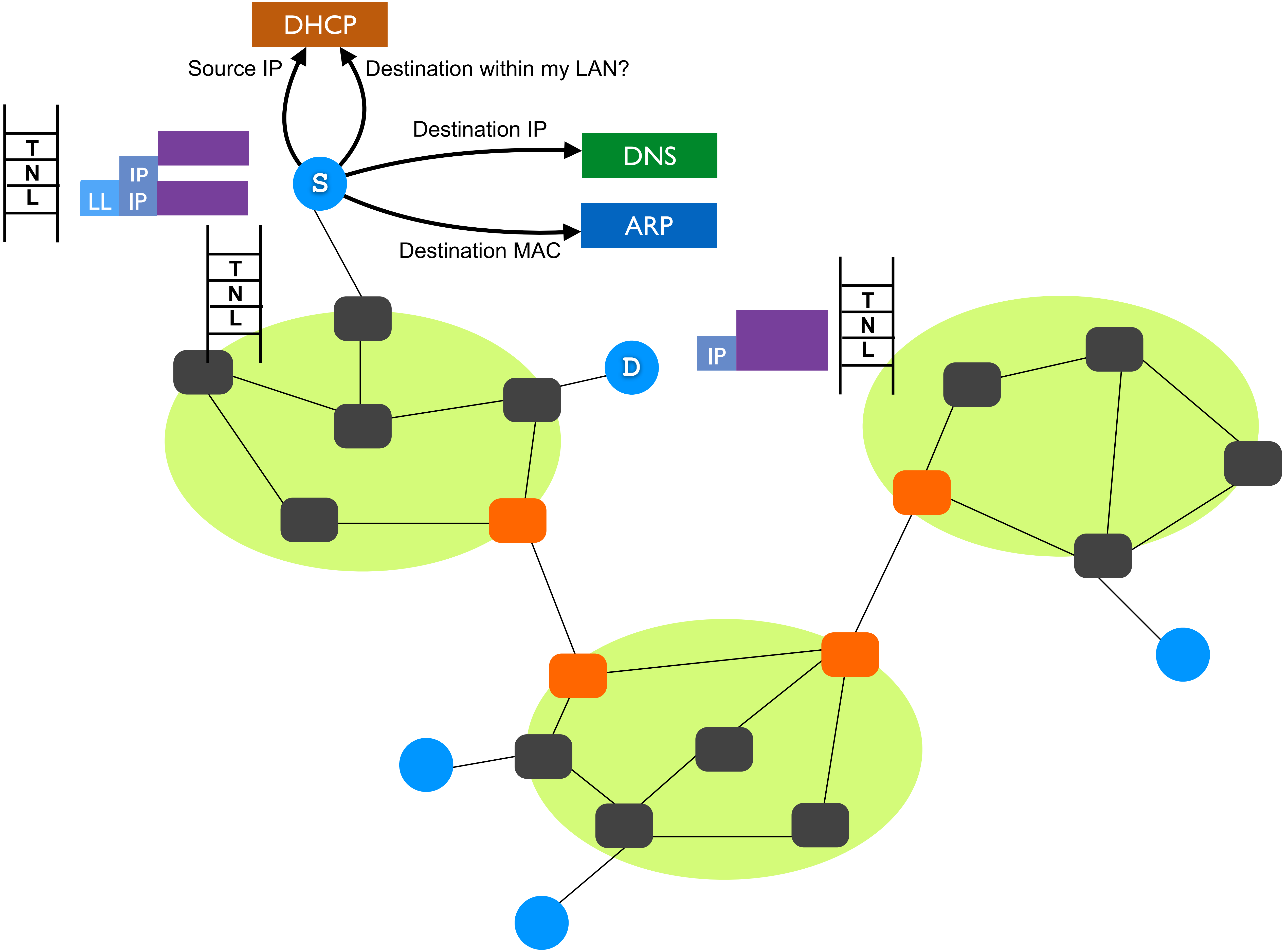
Are you ready?

(Count the number of protocols used for each packet)

How does Internet work — end-to-end?

- Network stack receives the packet from the application (roughly speaking)
- What is my IP address? (using DHCP)
- What is the destination IP address? (using DNS)
- Is destination IP address within my LAN? (using DHCP)
- **If destination IP address local:**
  - What is destination MAC address (using ARP)?
  - Convert packet into frames with correct source/destination address
  - Convert frames into bits
  - Forward the bits to the wire …
- **Each switch:**
  - Forwards to destination (using STP/CSMA/CD)

End-to-End I

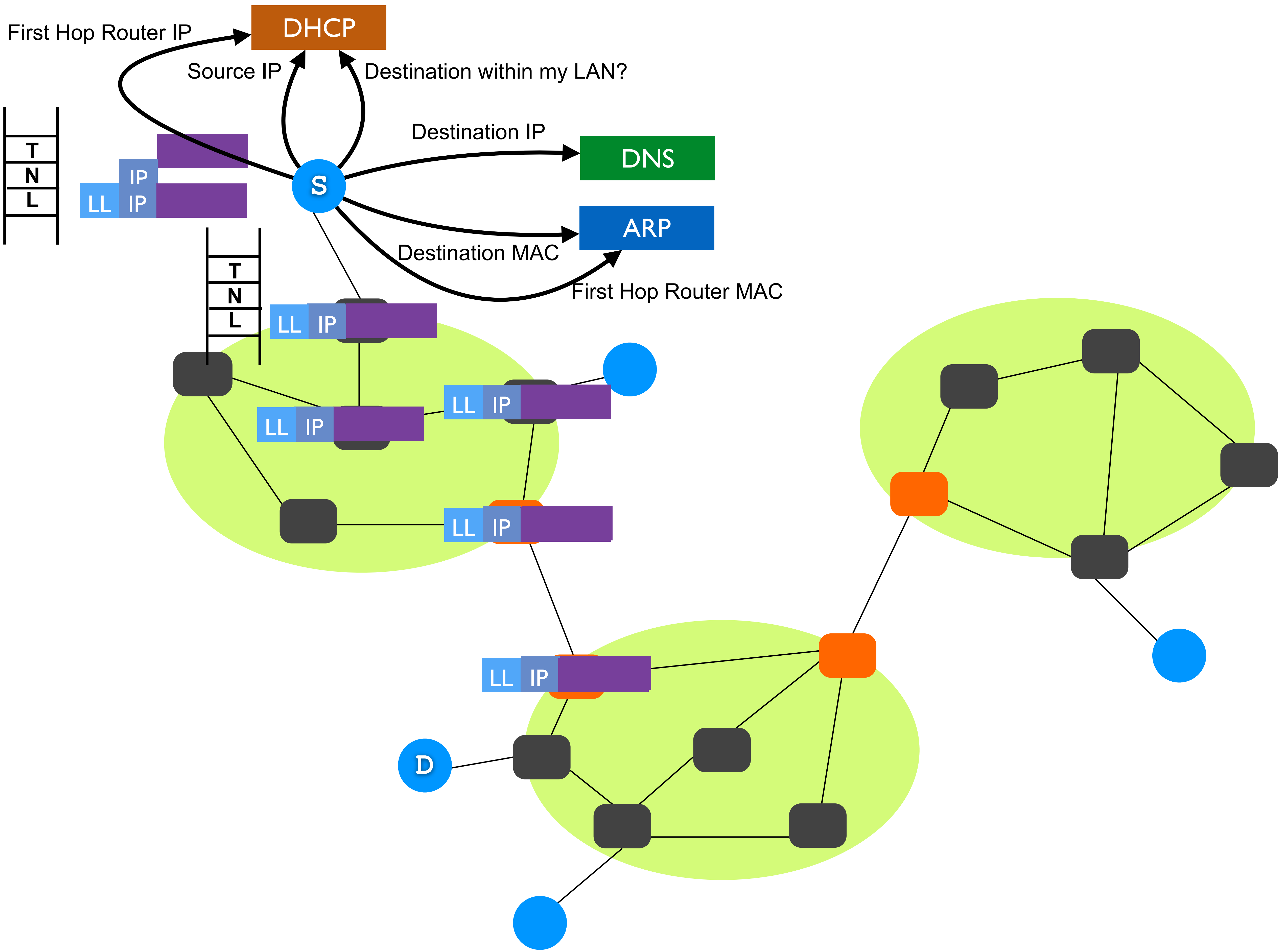


DHCP
Source IP
Destination within my LAN?
Destination IP
DNS
ARP
Destination MAC
T
N
L
IP
LL
IP
S
T
N
L
D
IP
T
N
L

How does Internet work — end-to-end?

- Network stack receives the packet from the application (roughly speaking)
- What is my IP address? (using DHCP)
- What is the destination IP address? (using DNS)
- Is destination IP address within my LAN? (using DHCP)
- **If destination IP address remote:**
    - **What is my next-hop router IP address? (using DHCP)**
    - **What is my next-hop router MAC address? (using ARP)**
    - Convert packet into frames with correct source/destination address
    - Convert frames into bits
    - Forward the bits to the wire …
- **Each router ….**

End-to-End II



DHCP
First Hop Router IP
Source IP
Destination within my LAN?
Destination IP
DNS
ARP
Destination MAC
First Hop Router MAC
S
T
N
L
IP
LL
IP
T
N
L
LL
IP
LL
IP
LL
IP
LL
IP
LL
IP
D

How does Internet work — end-to-end?

A router upon receiving a packet (implicit questions)

- **Is the destination in a LAN connected to me?**
    - Forward the packet to the destination
    - Using STP/CSMA/CD
- **Is the destination not in my LAN but in my ISP?**
    - Forward the packet to the next-hop router towards the destination
    - Using routing table entries via distance-vector routing algorithm
- **Is the destination in a different ISP?**
    - Forward the packet to the next-hop router towards the destination
    - Using routing table entries via BGP routing algorithm

Are We There Yet?

- Yes!
- How can we be sure?
- Lets go back to where we started ….

Recall the end-to-end story from our fifth lecture :-)

- Application opens a **socket** that allows it to connect to the **network stack**
- Maps **name** of the web site to its **address** using **DNS**
- The network stack at the source embeds the address and **port** for both the source and the destination in **packet header**
- Each **router** constructs a **routing table** using a distributed algorithm
- Each router uses destination address in the packet header to look up the **outgoing link** in the routing table
    - And when the link is free, forwards the packet
- When a packet arrives the destination:
    - The network stack at the destination uses the port to forward the packet to the right application

You now know how the Internet works!!!!

All that is remaining:

Reliability.