

University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

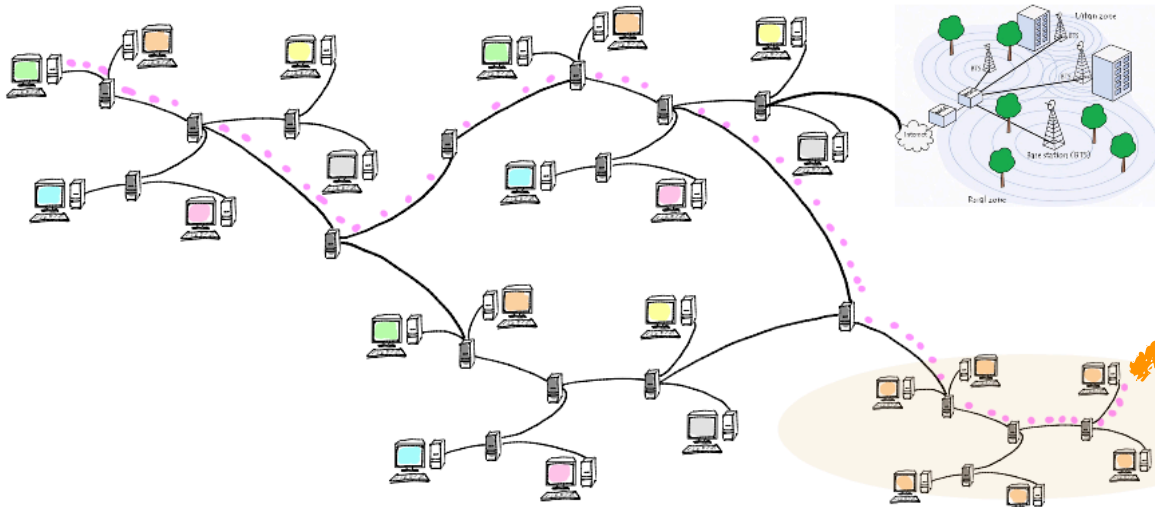
ECE 101: Exploring Digital Information Technologies

The Internet (part 2 of 3)

Review: Sending Data Packets Across the Internet!

Once it's all connected,

- you can **send packets of bits**
- **to any machine on the Internet!**



How?

You **just need an IP address!**

Here's one for
`ece.illinois.edu`



Every Computer Has a Unique IP Address

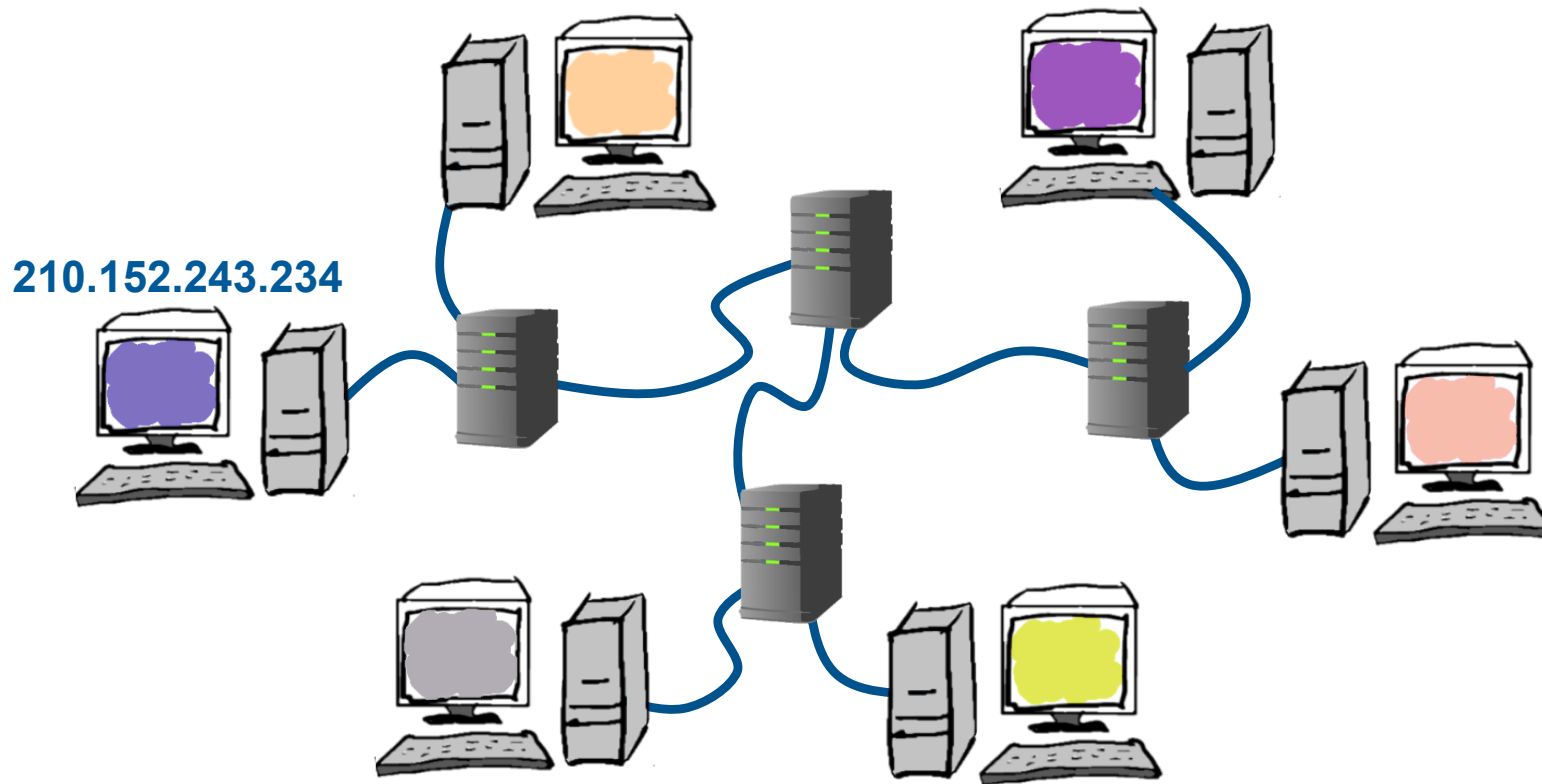
An Internet Protocol (**IP**) **address**

- **is 4 Bytes** (32 bits) ... i.e. IPV4
- Humans write **130.126.151.19**,
- but in the computer, it's
- **10000010 01111110 10010111 00010011**
- (without the spaces).

Every computer in the Internet **has an IP address**.*

*Sort of. Today, a household usually shares one public address.

Hierarchical Network



TO: 210.152.243.234



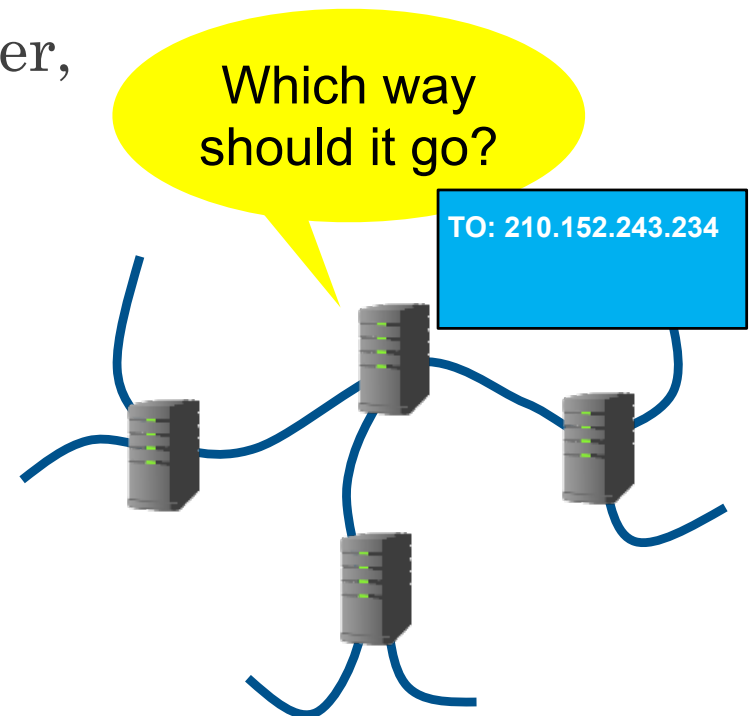
How do the Routers know where to forward Data Packets

When a **data packet arrives** at a router,

- the **router must decide**
- **on which** outgoing link/network interface
- **to forward** the packet.

The router looks at two things:

1. **destination IP address** in packet,
2. and its **forwarding table**
(also called a **routing table**).



Routers keep updated Routing Tables

Routers periodically

- **advertise IP address ranges that they can reach**
- to their other links,
- as if they are gossiping, who knows whom!!

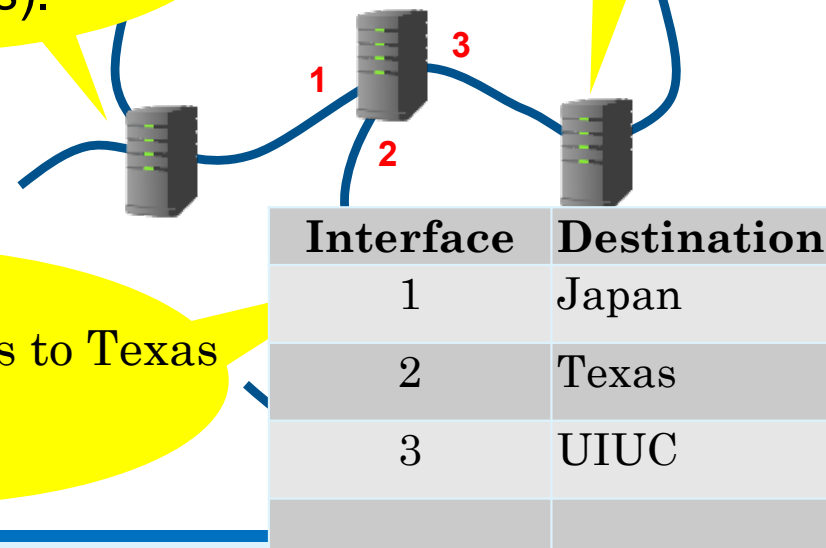
A **forwarding table** maps

- **IP addresses into**
- **interface numbers**
- (1, 2, or 3, for example).

I can get packets to Japan (IP addresses).

I can get packets to anyone in UIUC (IP addresses).

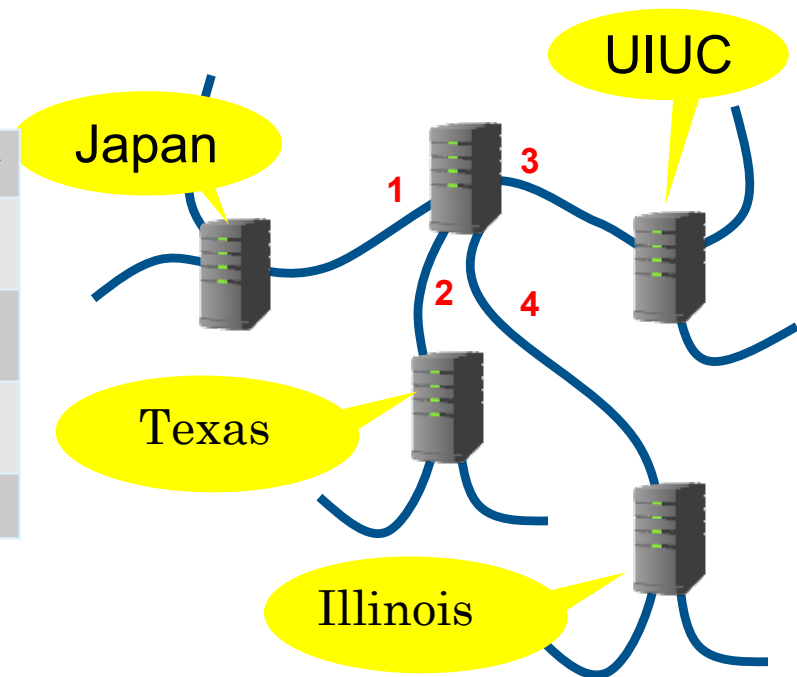
I can send packets to Texas (IP addresses)



Let's Route

- A packet arrives for Northwestern University (in Chicago, IL, USA).
- **Which interface?**
- A packet arrives for Tokyo University (in Tokyo, Japan).
- **Which interface?**
- A packet arrives for UIUC.
- **Which interface?**
- Usually, the **more specific answer is chosen**.

Interface	Destination
1	Japan
2	Texas
3	UIUC
4	Illinois

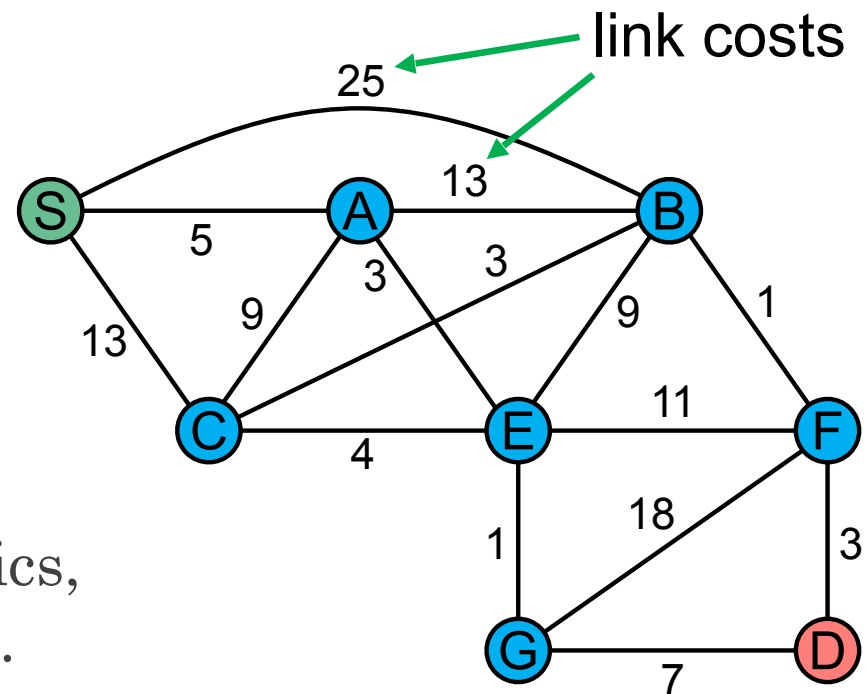


Network Graph for Routing

How to get from S to D?

Link cost may reflect

- length (delay),
- link capacity,
- congestion (queueing delay),
- any combination of those metrics,
- or just the router's preferences.



Computing Cost of a Path

How to get from S to D?

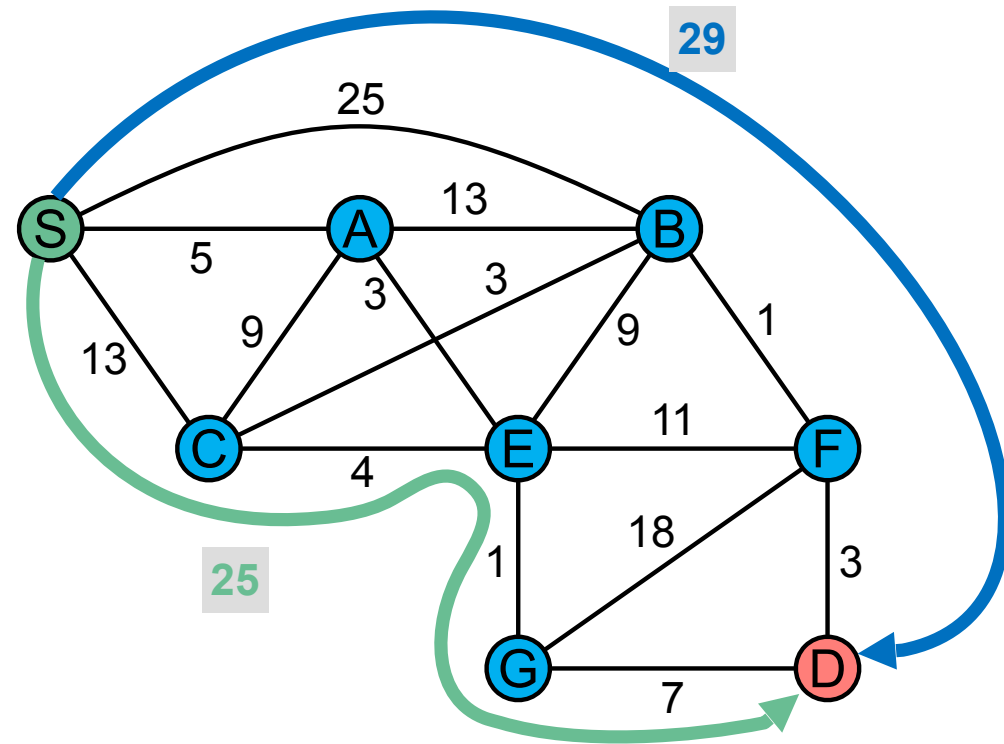
The **cost of a path** is

- the **sum of the costs**
- of the **links** in the path.

And the **desired path** is the path with the **smallest cost**.

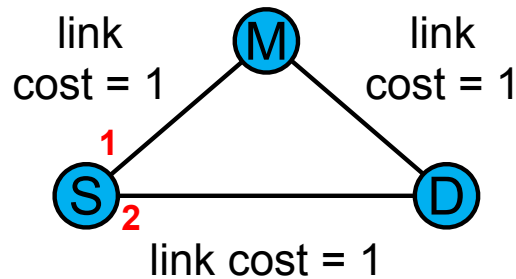
To construct a forwarding table,

- a router must **decide**
- **which path to follow**
- to reach each other node.



Let's Find Routes Together!

Complete the forwarding table for the node S.

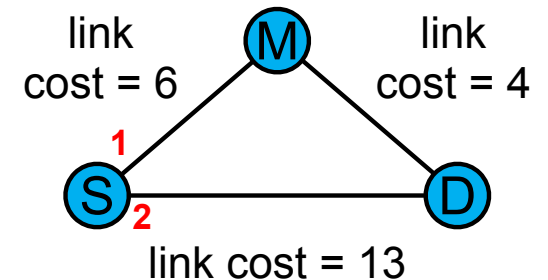
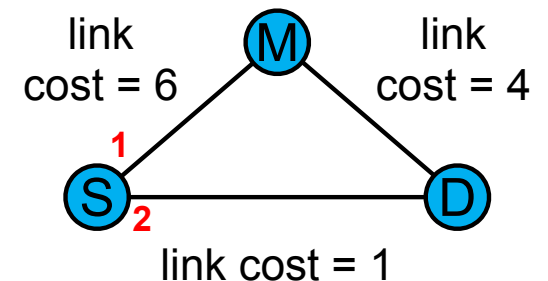


dest	interface
D	2
M	1

(Compute cost by “hops”:
all link costs are 1)

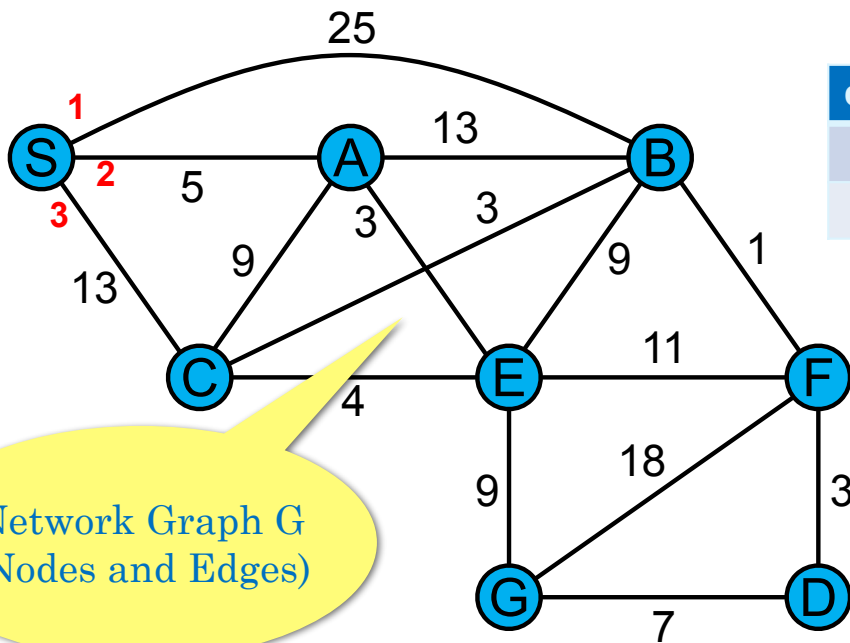
dest	interface
D	2
M	2

dest	interface
D	1
M	1



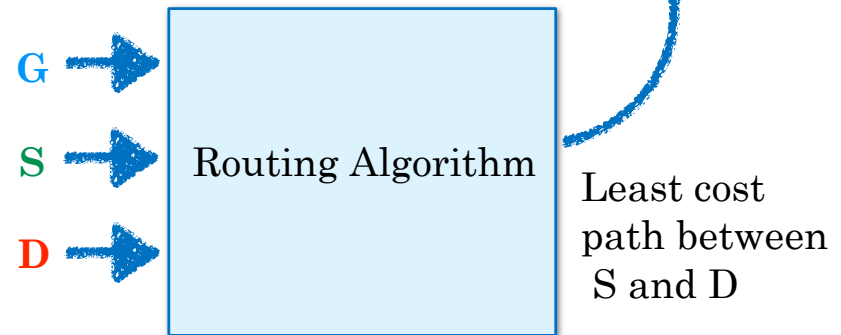
Here's a challenge ...

Complete the forwarding table for **destination node D** from the **source node S**.



dest	interface
D	?
...	...

The computer will do it for you!!



Data and Control Planes

Routers forward **packets**.

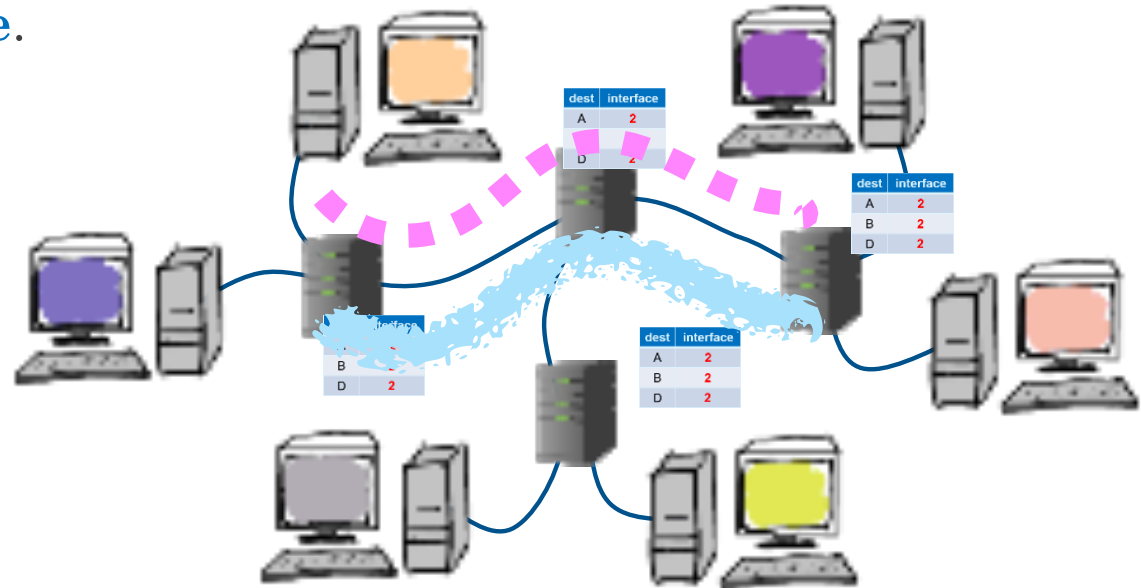
- Packets containing data are said to pass **through the data plane**.

Routers exchange **information about routes** available to them.

- These coordination messages travel **through the control plane**.

The two planes

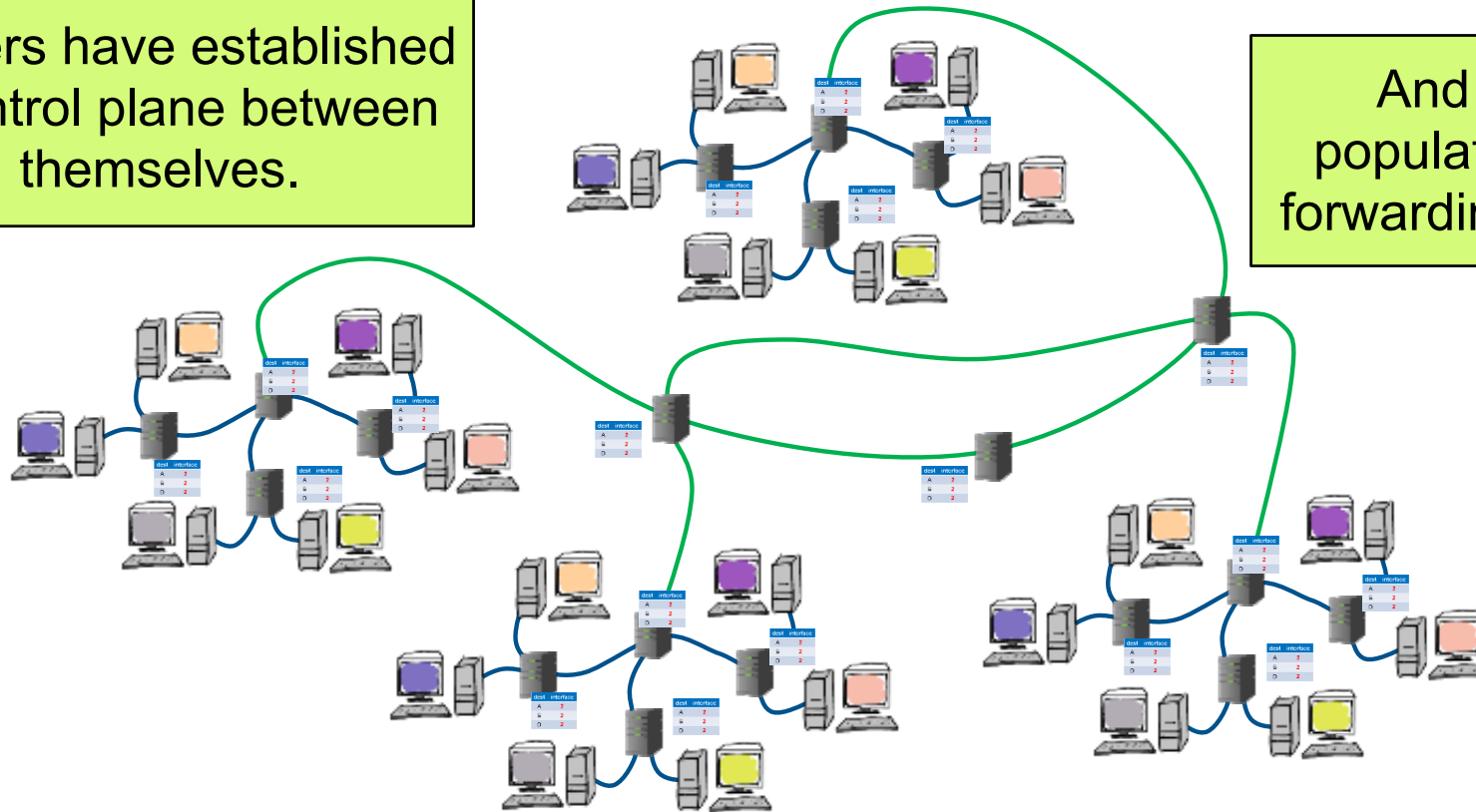
- use the same physical network
- but are logically separate.



Internet Plumbing's Ready: Routing and Forwarding

Routers have established
a control plane between
themselves.

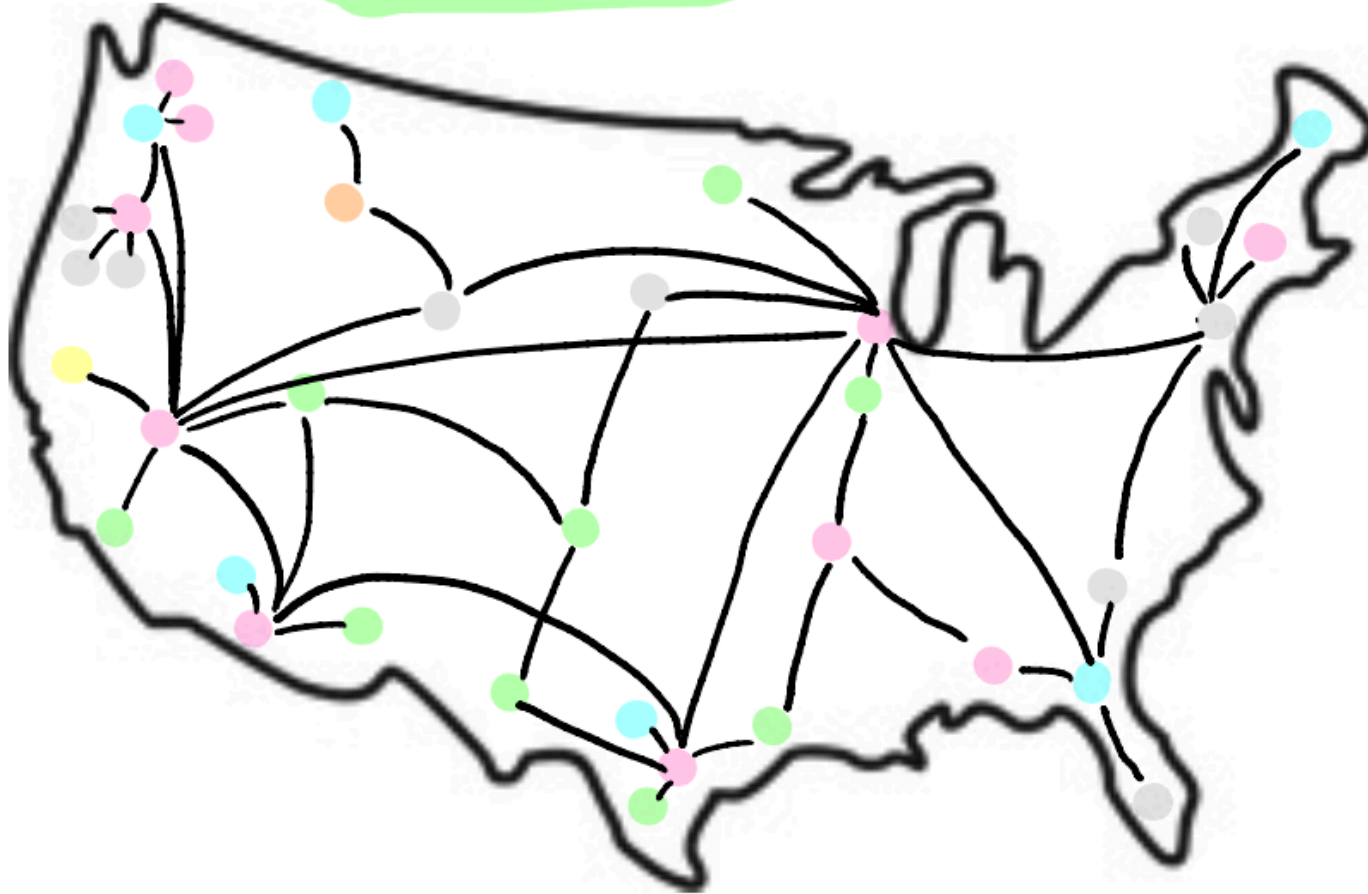
And have
populated their
forwarding tables.



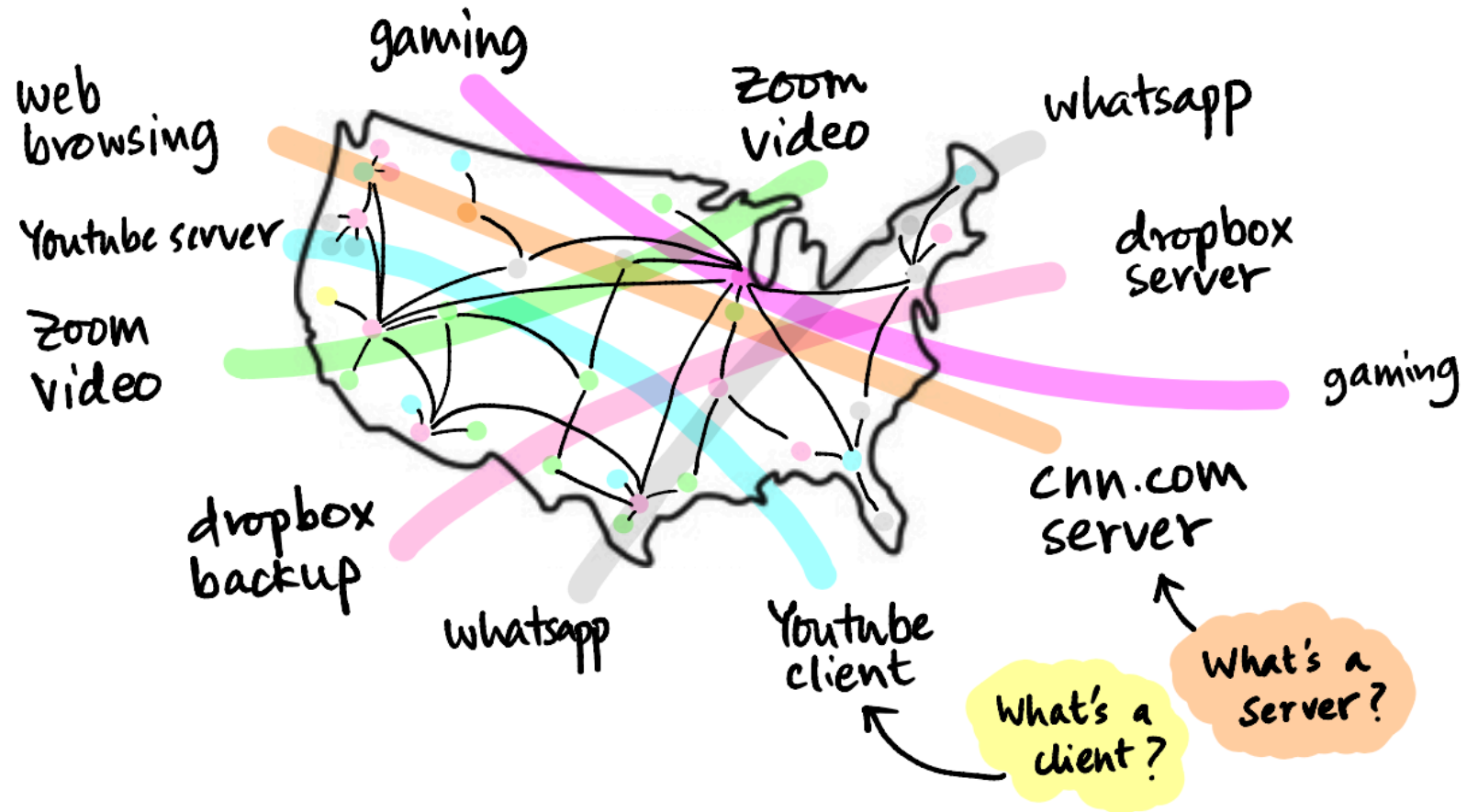
Class Participation Question: Answer on Canvas

What are the inputs to the routing algorithm and what does it output?

The airports are ready, and flights can link them
Now what?



Applications running on the "edge"



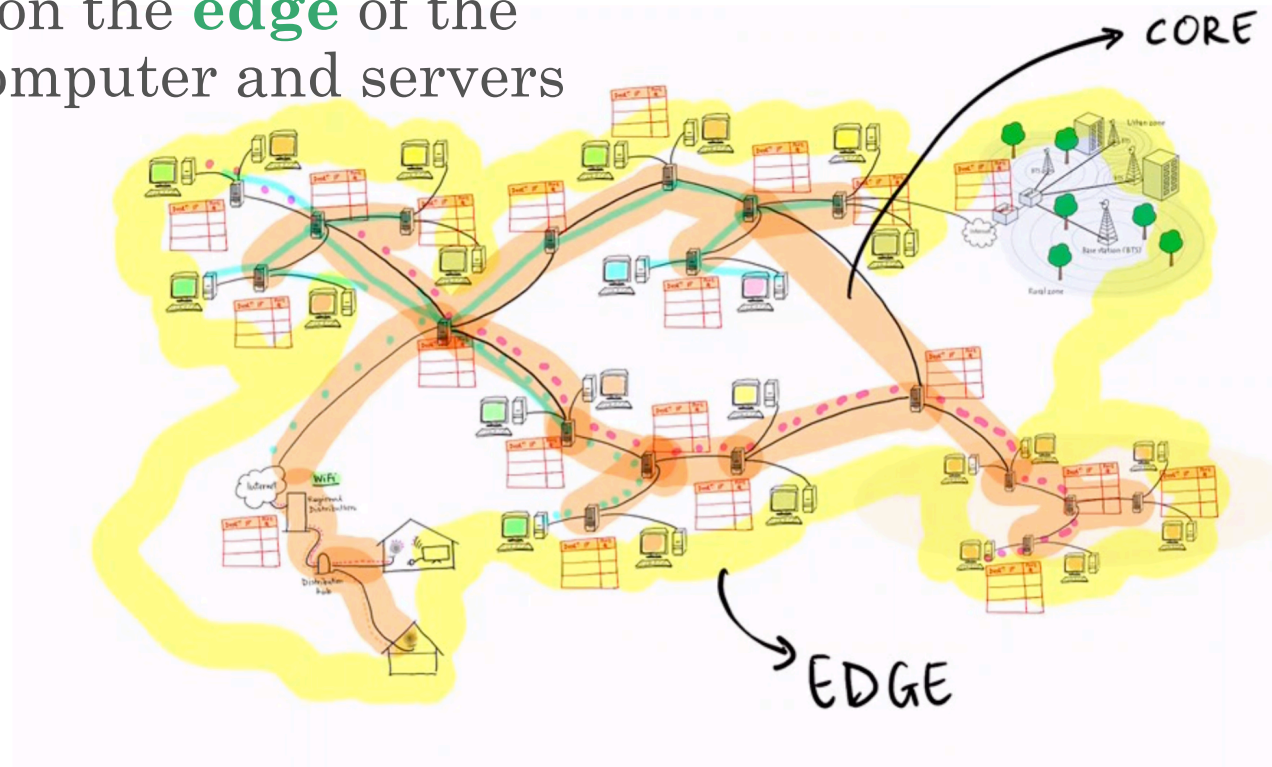
The **Edge** and the **Core** of the Internet

Applications operate on the **edge** of the network, where your computer and servers are connected.

Everything else

- all of the routers,
- all of the rest of the hierarchy,
- the plumbing of the Internet,

...these form the **core of the Internet**.



Application: Web Browser

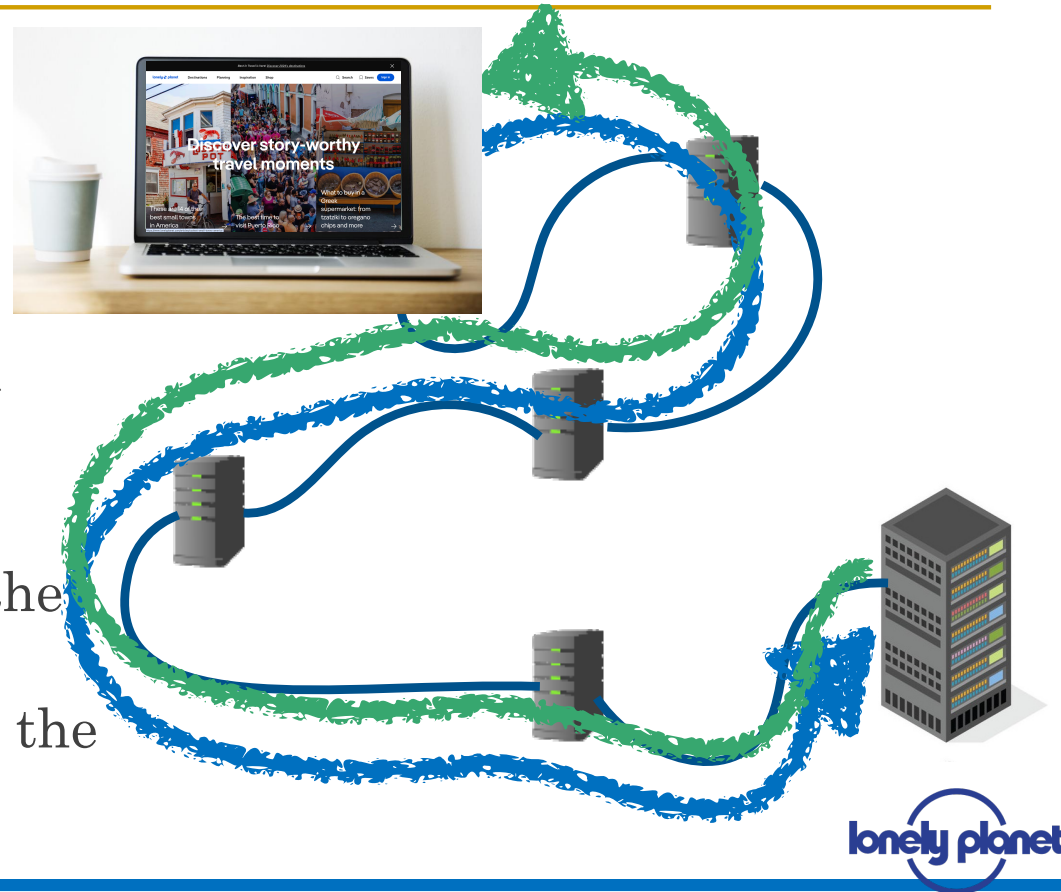
You are planning a vacation.

So you open your browser and type “lonelyplanet.com” (for example).

Somewhere across the Internet is a **server**—maybe in Dublin—that has the Lonely Planet homepage.

Your browser sends a **request** to the Lonely Planet server.

The server sends a **response** with the page back to your browser.



HTTP: An Application Protocol for Browsing the Web

The computers participating in the exchange

- your browser on **your laptop** and
- the lonely planet **server**,
- must use the same **protocol**.

The protocol **used for web browsing** is
Hyper-Text Transfer Protocol (**HTTP**)

HTTP Requests and Responses Use Human-Readable Text

So what does your computer actually send?

English sentences?

“Please give me the web page at lonelyplanet.com.”

No!

Like most computer protocols,

- **HTTP has rules**
- about how to request
- and receive information.

HTTP is more friendly, though—it does **use human-readable sequences of bytes** (text).

Contents of an HTTP Request

A HTTP request

- (like your browser sends) has
- a command (GET),
- a web page name (/index.html)
- a protocol specification (HTTP/1.0),
- and some options.

```
GET /index.html HTTP/1.0
User-Agent: Mozilla/4.0
Host: lonelyplanet.com
(more options, ending blank line)
```

Contents of an HTTP Response (Successful)

A HTTP response

- (returned from a web server to your browser) has
- a protocol specification (HTTP/1.0),
- an error code (200 OK),
- some options,
- and the web page itself (in [HyperText Markup Language](#), HTML).

```
HTTP/1.0 200 OK
Date: Fri, 9 Sep 2022 16:20:23 GMT
Server: Apache/2.2.15
(more info)

Bits for the web page (in HTML) !
```

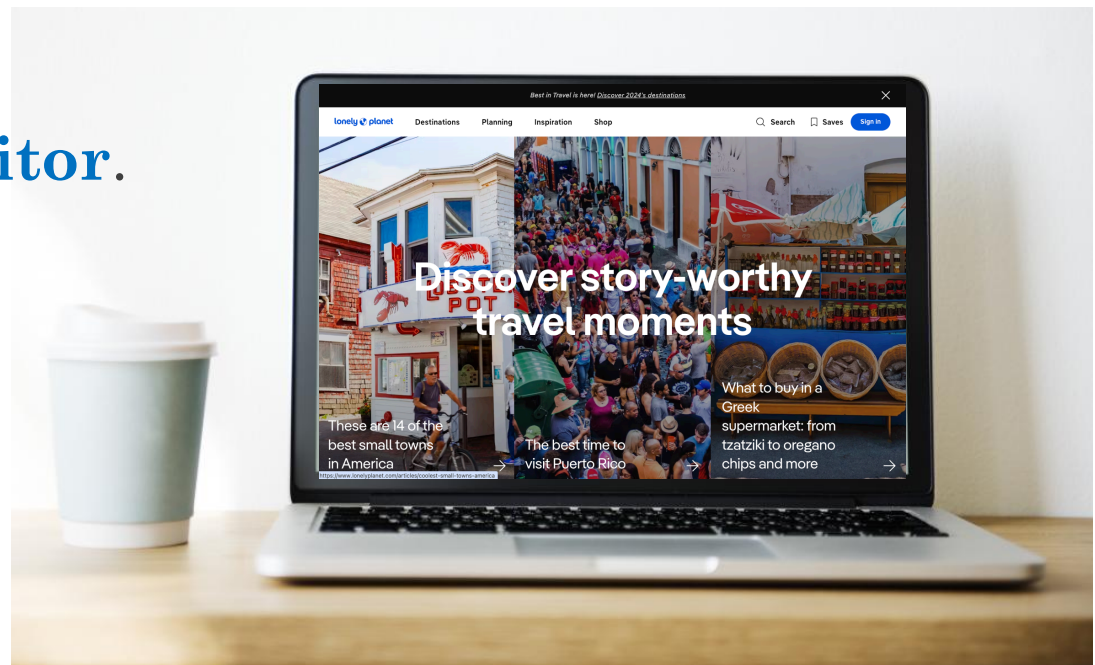
When Browser Receives HTTP Response, It Renders Page

When your **browser**

- receives the web page,
- it **renders** it **on your monitor**.

Usually, the browser

- **simultaneously makes requests**
- **for images, videos, and other things**
- embedded in the page.



Data Packets

really just
a sequence
of bits.

The request and response are sent as **data packets**

Request

Control Bits/Packet Header

Data Payload

Protocol Name	Version	Browser	...	Name of webpage
1010...01	10	10011		10001010 10001000 10000101 10001011 10000111

Response

Protocol Name	Version	Server	...	Contents of the webpage
1010...01	10	11001		1011010 11001000 10100001 11101011 10111111 1000010 10001000 10000101 10001011 10000111 ... 1011011 10101000 10000101 11111011 11000111

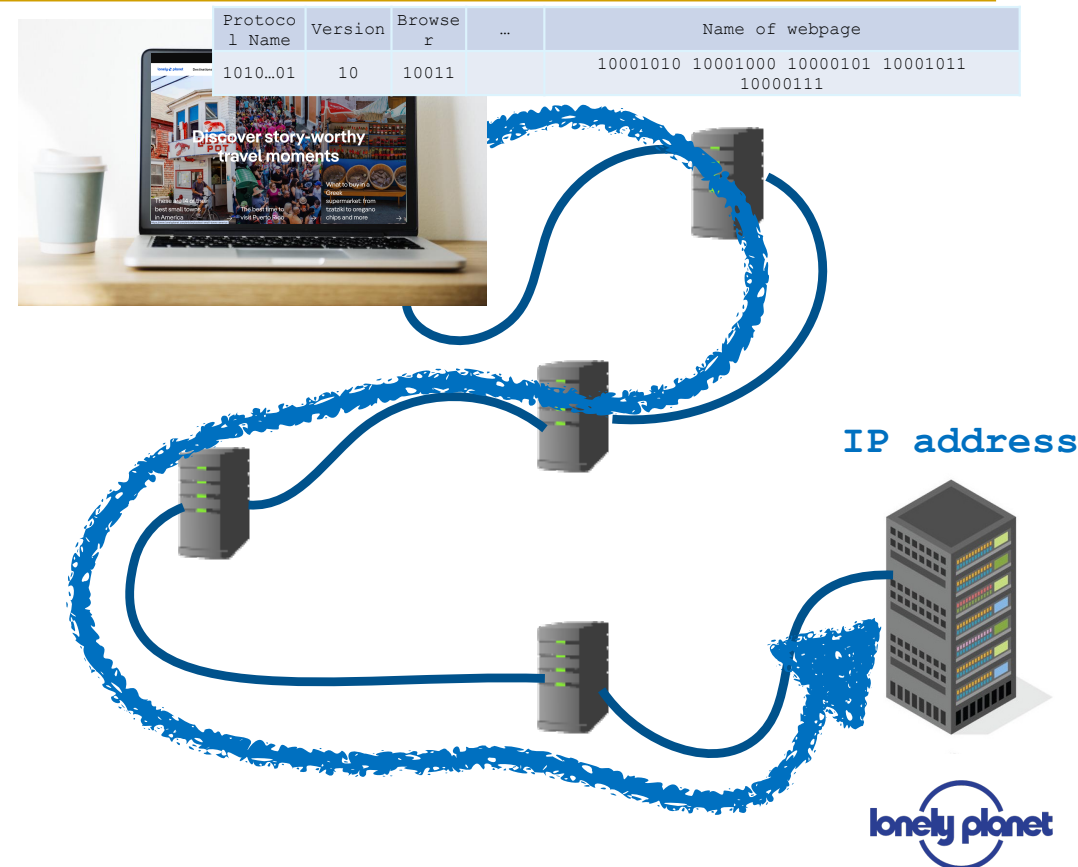
Where to send the packet?

Your browser prepares the **request** data packet to be sent.

Where do you send it to? What do you need to send it?

Do you have to know the addresses of all the servers everywhere?

No. You use the **DNS**



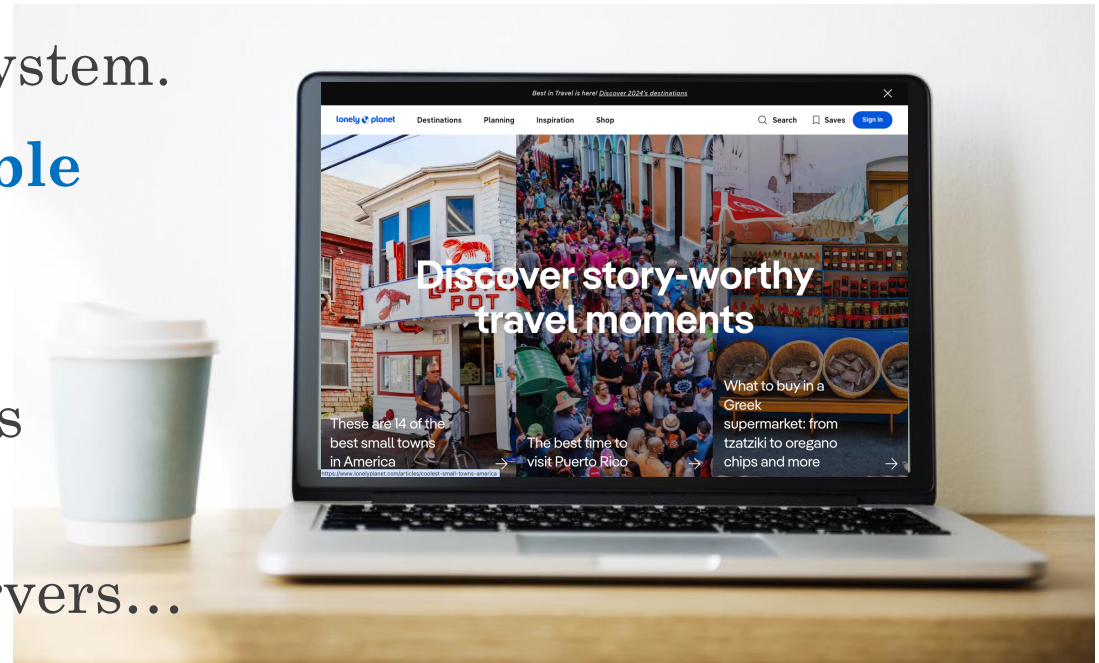
DNS Translates Human-Readable Names to IP Addresses

DNS, or the Domain Name System.

It **translates human-readable names** like lonelyplanet.com **into IP addresses**.

There are 13 DNS root servers around the world.

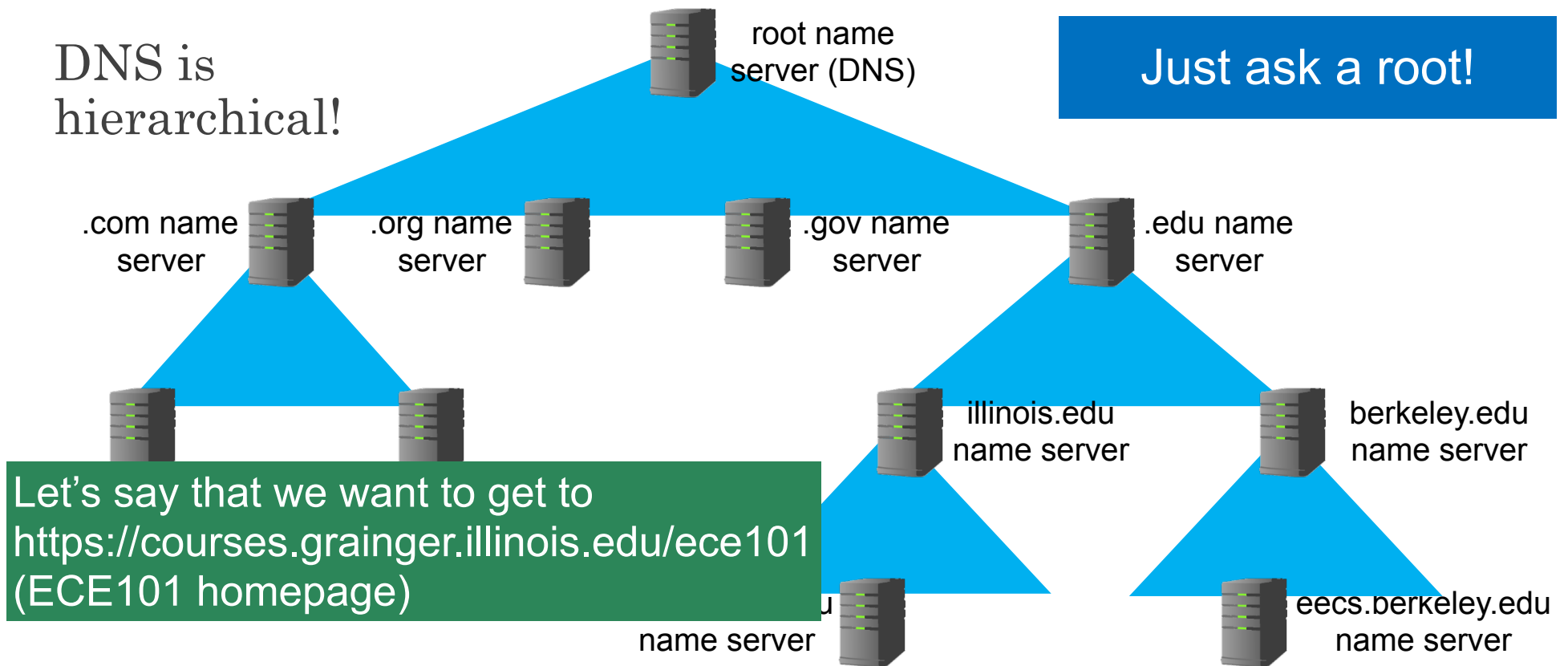
Billions of people using 13 servers...



Domain Name System (DNS) Server Hierarchy

DNS is hierarchical!

Just ask a root!



DNS Lookups Start Logically at the Root

DNS resolve
“https://courses.grainger.illinois.edu/”

root name
server (DNS)

**DNS Try server at
IP: W.X.Y.Z**

this
one

.com name
server

.org name
server

.gov name
server

.edu name
server

cnn.com
name server

google.com
name server

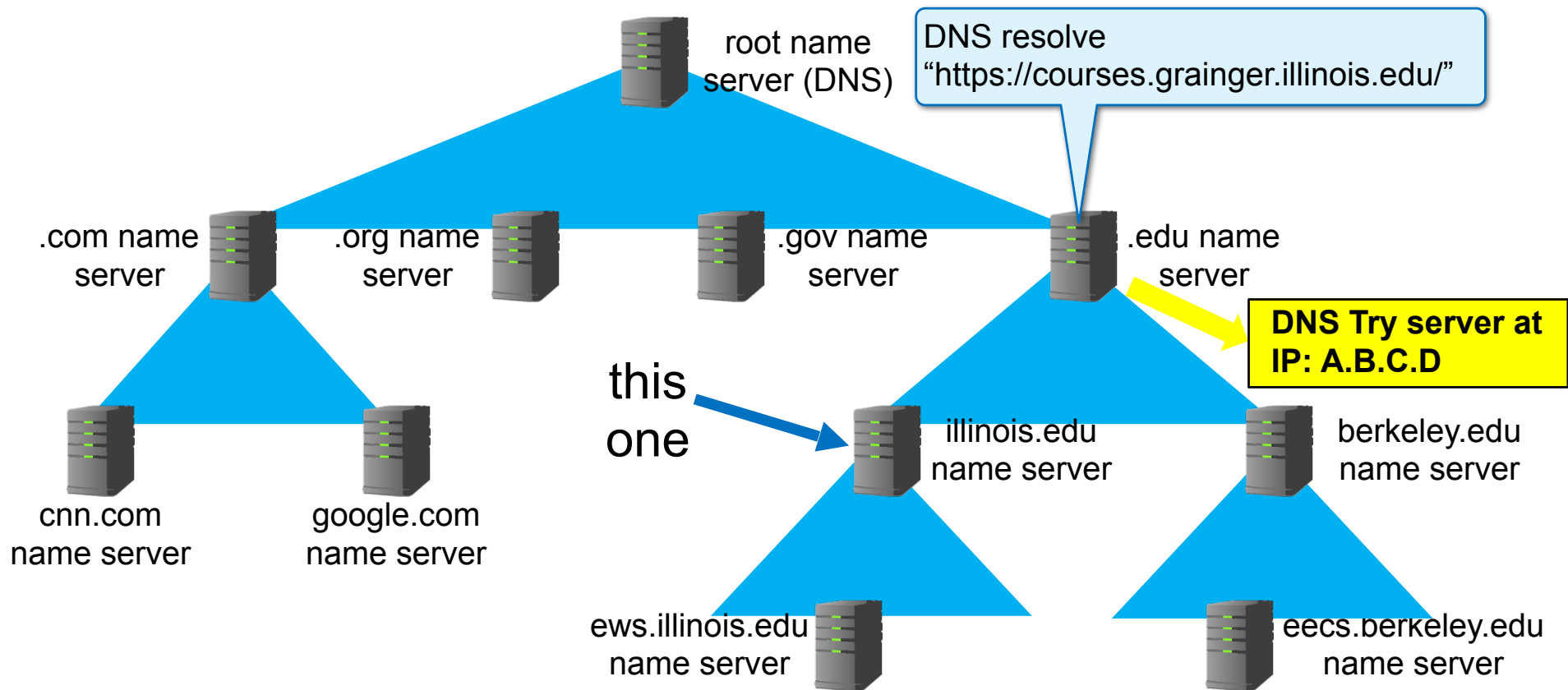
illinois.edu
name server

berkeley.edu
name server

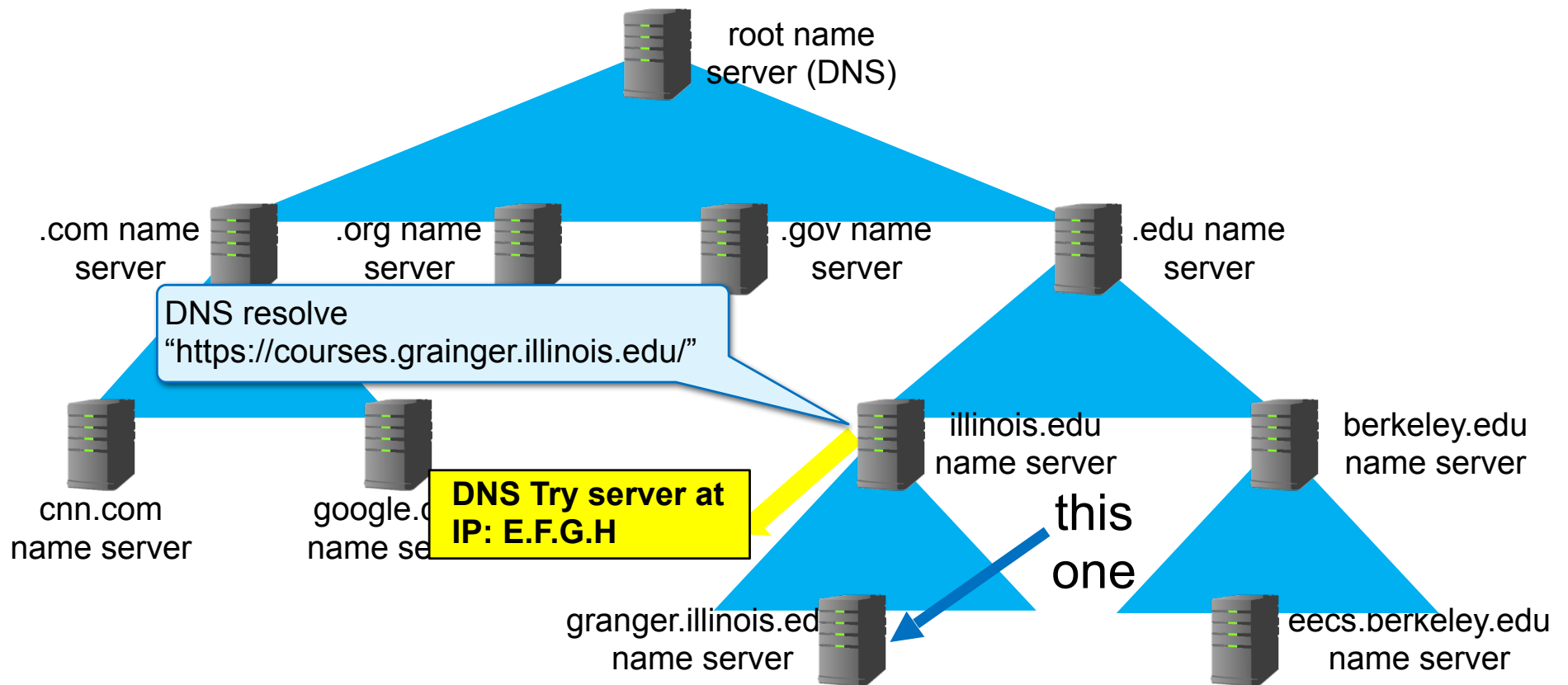
ews.illinois.edu
name server

eeecs.berkeley.edu
name server

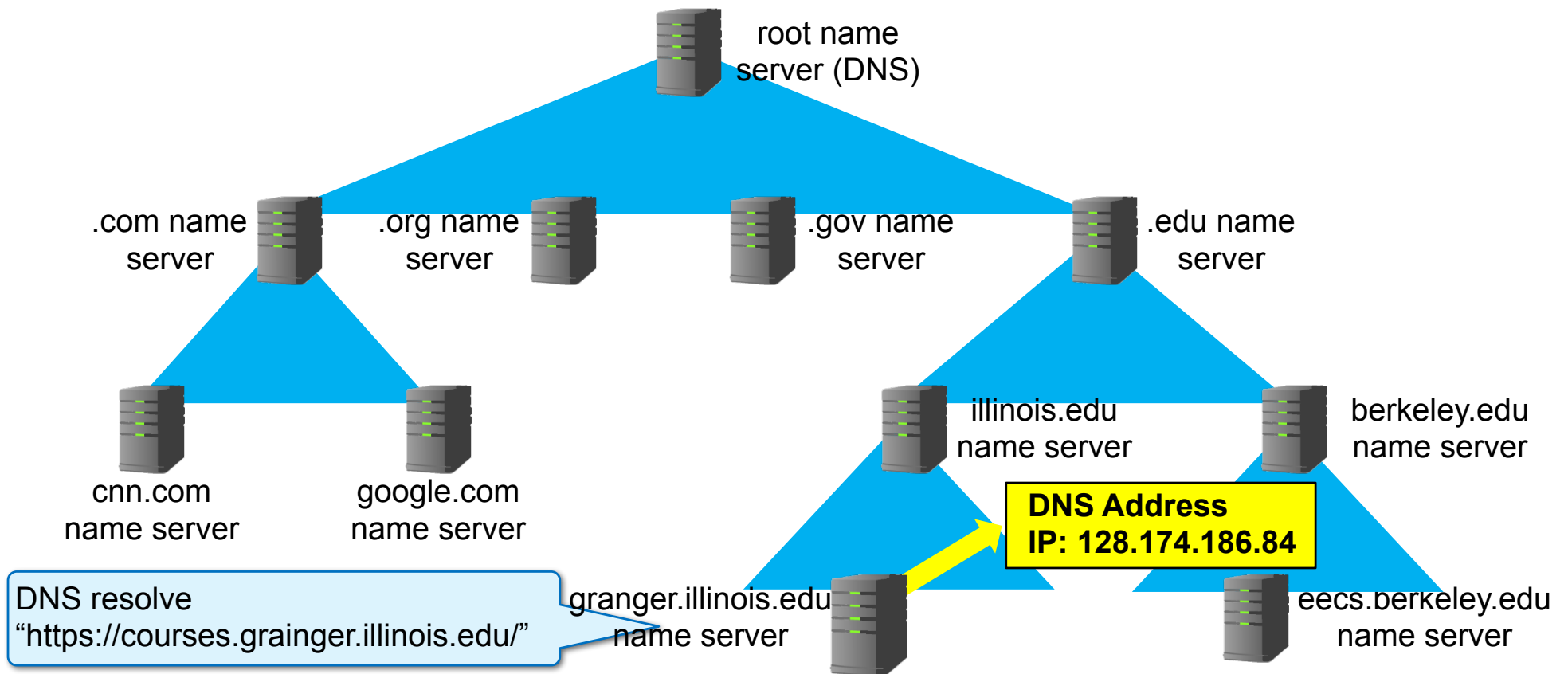
DNS Lookups Iterate Until an Answer is Found



DNS Keeps Chugging Along



Until, Finally, an Answer!

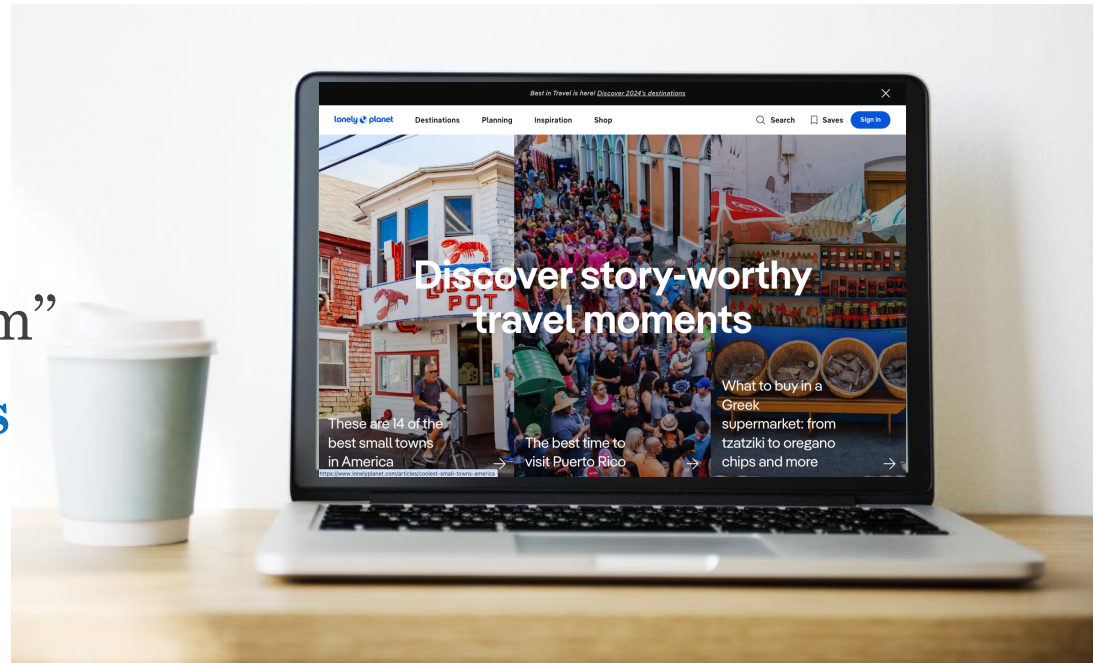


DNS Provides the IP Address for lonelyplanet.com

Back to our example...

Your web browser wants
to connect to “lonelyplanet.com”

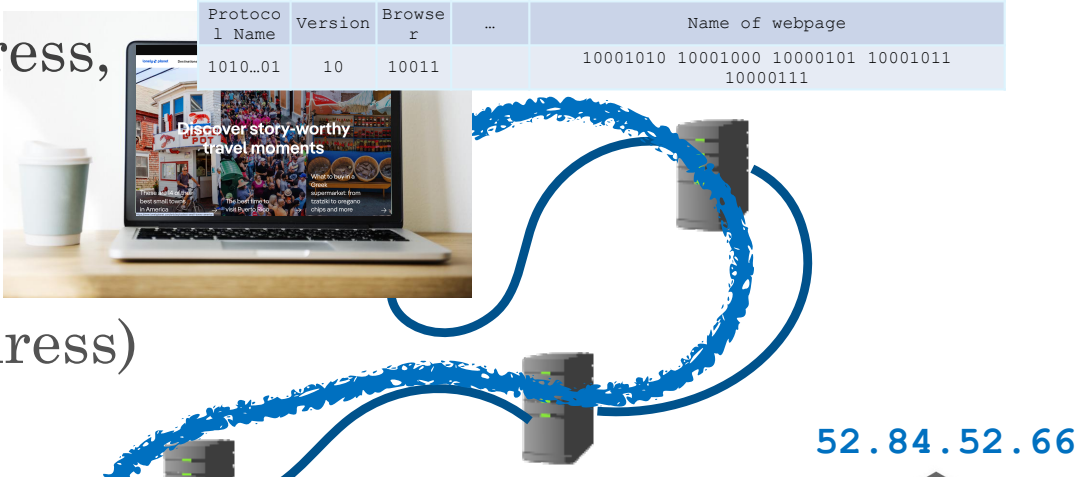
It **queries DNS and obtains
an IP address: 52.84.52.66.**



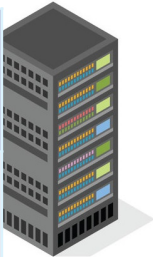
Send the request off ...

Once the browser has the IP address,

- it can add it to the data packet
- as **destination IP**
- in packet header **control bits**,
- also add **source IP** (return address)
- and **send**



Destination IP	Source IP	Protocol Name	Version	Name of webpage
00110100	10000010	1010...01	10	10001010 10001000 10000101 10001011 10000111
01010100	01111110			
00110100	11111111			
01000010	11100100			

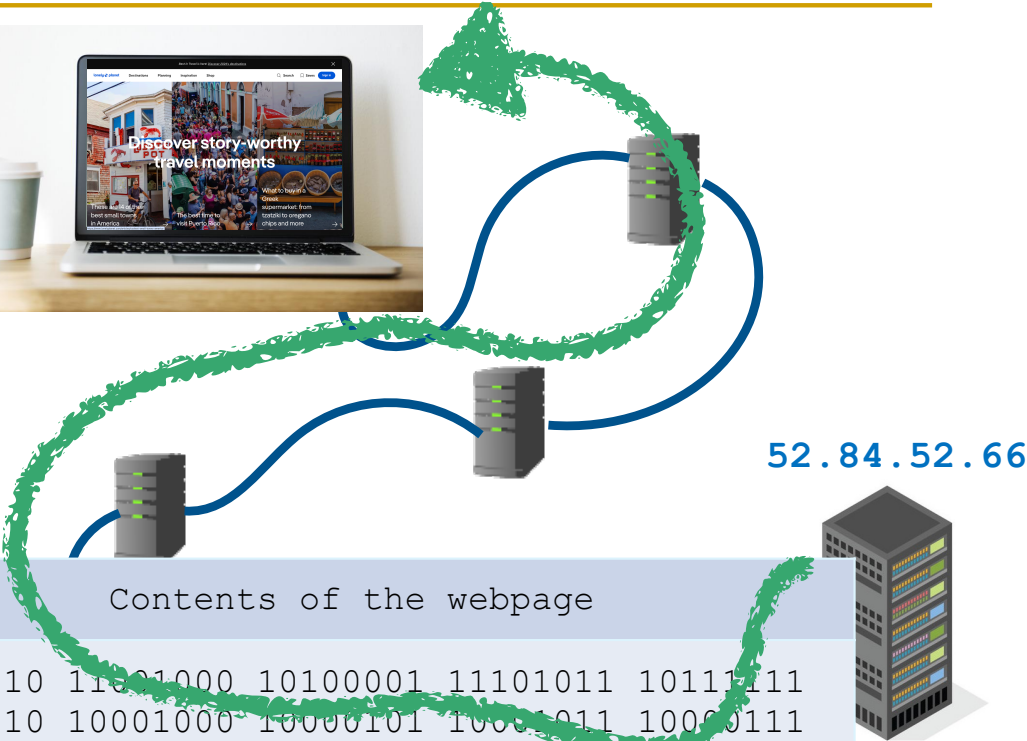
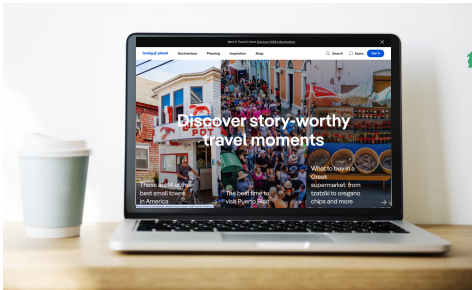


Get back a response ...

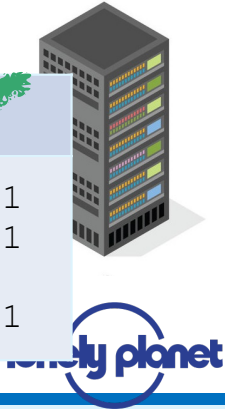
130.126.255.228

Once the server receives the request,

- it puts the webpage
- in a data packet
- and sends it back to your browser



Destination IP	Source IP	Protocol Name	Version	...	Contents of the webpage
10000010	00110100	1010...01	10		1011010 11001000 10100001 11101011 10111111
01111110	01010100				1000010 10001000 10000101 1000011 10000111
11111111	00110100				...
11100100	01000010				1011011 10101000 10000101 11111011 11000111



Terminology You Should Know from These Slides

- Forwarding
- Routing table
- Routing algorithm
- Route advertisements
- Routing
(building routing table)
- edge (of Internet)
- core (of Internet)
- Applications
- HyperText Transfer Protocol (HTTP)
- HTTP request and response
- Data packet
- Control bits/packet header
- Data Payload
- Domain Name System (DNS)

Concepts You Should Know from These Slides

- how a router uses its forwarding table
- routers forward packets in data plane and exchange routing information in control plane
- how to find shortest paths in small graphs
- how a web browser finds and renders a page for you
- why you need a protocol
- uses of packet header: IP addresses, protocol commands (example: HTTP GET)
- what service DNS provides and how it works