

CSE190/291

How The Web Tracks You / Targeted Advertising

The Internet, the Web, and You

UC San Diego

Housekeeping

General course things to know

- PA1, parts 1 + 2 are released!
 - **Part 1 due Thursday, 10/1!**
 - Part 2 due Tuesday, 10/13
- Part 1 – an introduction to web servers and deploying a web server to the public Internet
- Part 2 – implementing a full-scale “web application” that does some stateful tracking, using cookies, to track your behavior on a Google search competitor (Boogle)

Some questions from last time

Some questions from last time

- **Question:** Can I be in a group with someone from the other section?
- Yes! But send us a note on Piazza if so, and we need two final Gradescope submissions, one for each class (so we know not to flag the submissions as cheating)

Some questions from last time

- **Question:** Can I be in a group with someone from the other section?
 - Yes! But send us a note on Piazza if so, and we need two final Gradescope submissions, one for each class (so we know not to flag the submissions as cheating)
- **Question:** Are the exams really hard and am I going to be sad I took this class?
 - I can't speak to exam hardness (it's not made yet), but our intention is to write a very fair exam

Some questions from last time

- **Question:** Can I be in a group with someone from the other section?
 - Yes! But send us a note on Piazza if so, and we need two final Gradescope submissions, one for each class (so we know not to flag the submissions as cheating)
- **Question:** Are the exams really hard and am I going to be sad I took this class?
 - I can't speak to exam hardness (it's not made yet), but our intention is to write a very fair exam
- **Question:** I am really scared about these PAs. Should I drop the class?
 - No! We will help you! It will be great!

Previously on CSE190/CSE291...

- We talked about **privacy**, why it matters, and at a high level, what **web tracking** is and why it enables our existing online advertising ecosystem
- We discussed the Solove framework of privacy — and why defining privacy is somewhat tricky

Today's lecture — The Internet, The Web, and You

Learning Objectives

- Understand how the Internet works (at a 10,000 foot view), the players involved, and the general duct tape tape that keeps the pipes working
- Learn what an IP address is, what ports are, and how the Internet uses these simple abstractions to support big applications
- Understand (basically) how Web browsing works and the fundamental architecture that underpins the Web
- Discuss cookies and how **state** is applied on the web, and how that in and of itself can enable widespread tracking

The Internet

Understanding the basics

- What do you know about the Internet?

Understanding the basics

- What is the Internet?
 - “A global computer network consisting of interconnected networks using standard communication protocols.”
- My version: **a network of networks that lets computers talk to one another.**

Understanding the basics

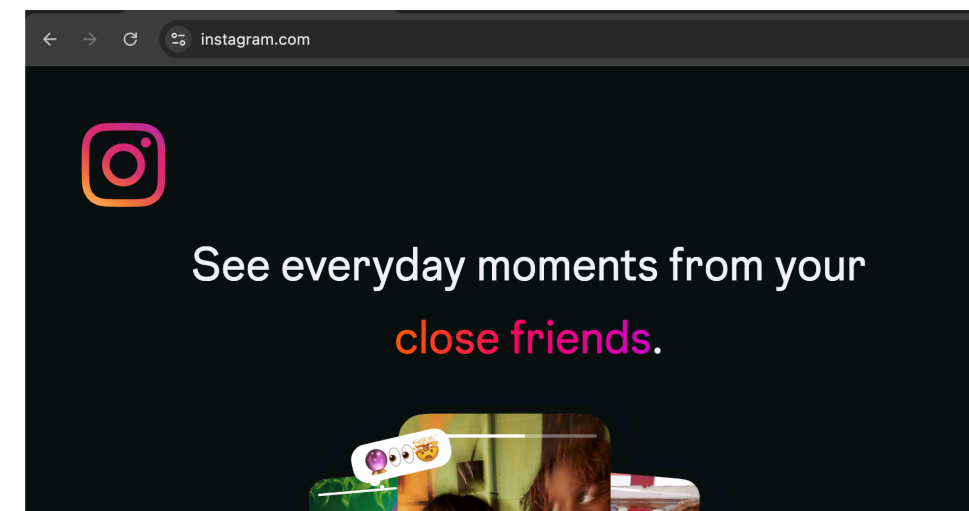
- What is the Internet?
 - “A global computer network consisting of interconnected networks using standard communication protocols.”
- My version: **a network of networks that lets computers talk to one another.**
- My other version: **magic**

What is a network of networks?

- Let's say my computer wants to talk to instagram.com. How does it do that?



my laptop



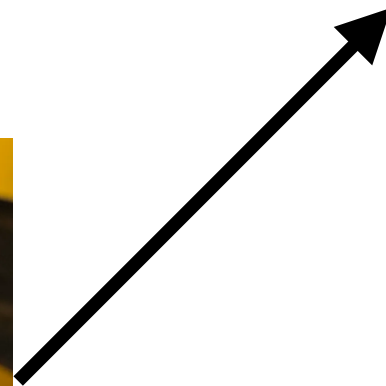
instagram.com

What is a network of networks?

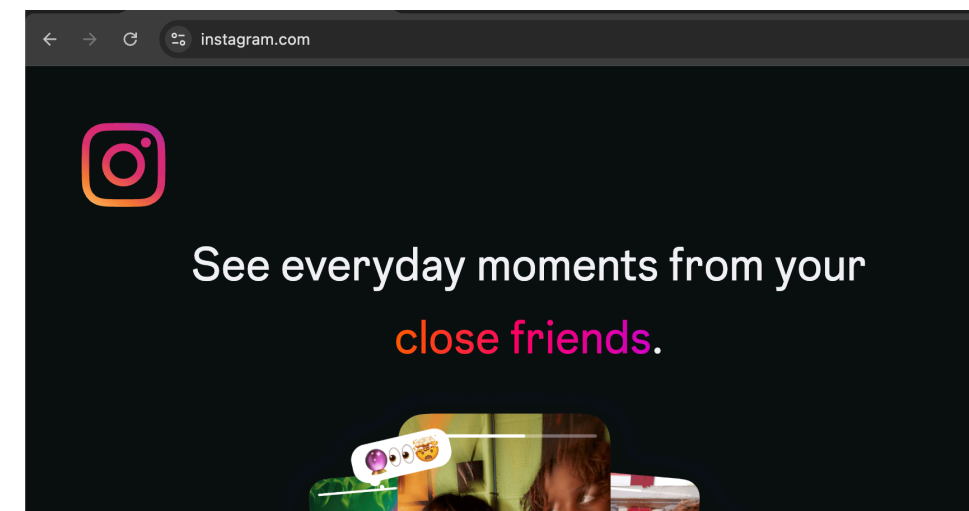
- Let's say my computer wants to talk to instagram.com. How does it do that?



my laptop



router, via
WiFi



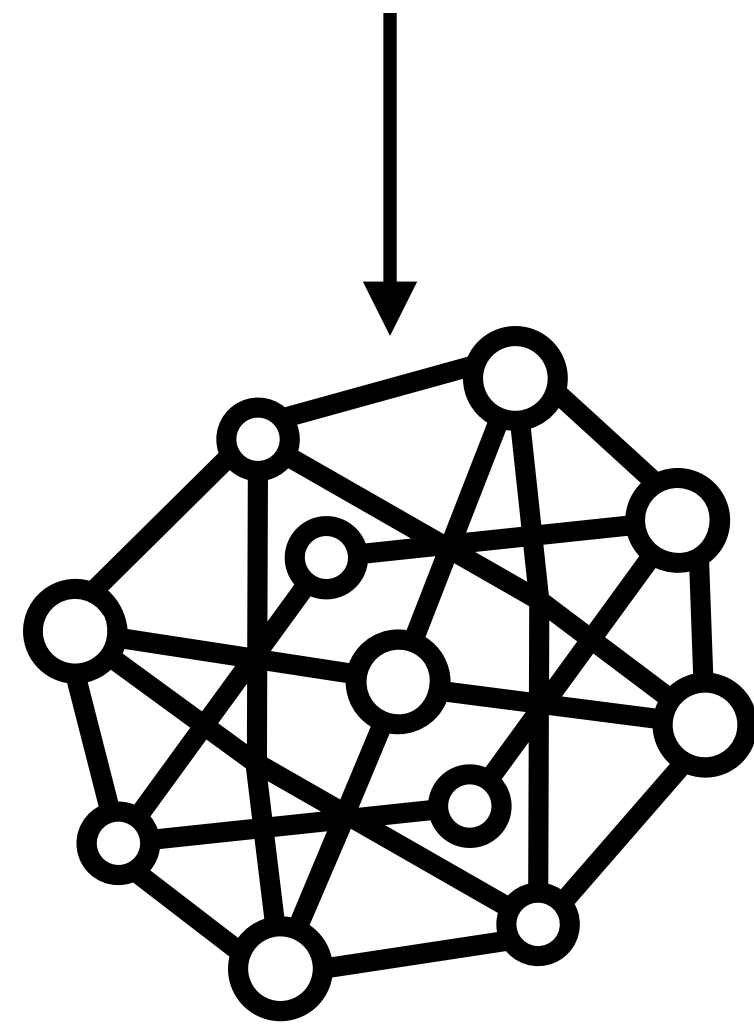
instagram.com

What is a network of networks?

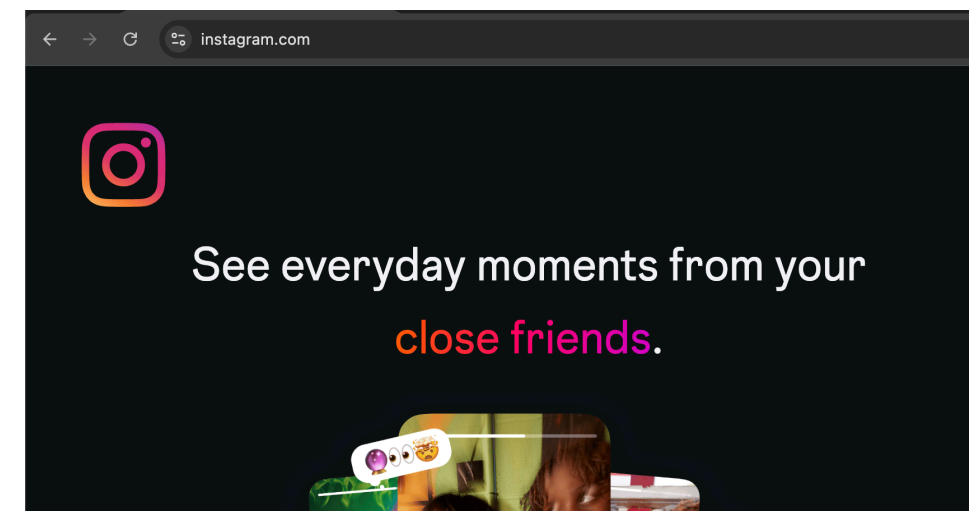
- Let's say my computer wants to talk to instagram.com. How does it do that?



my laptop



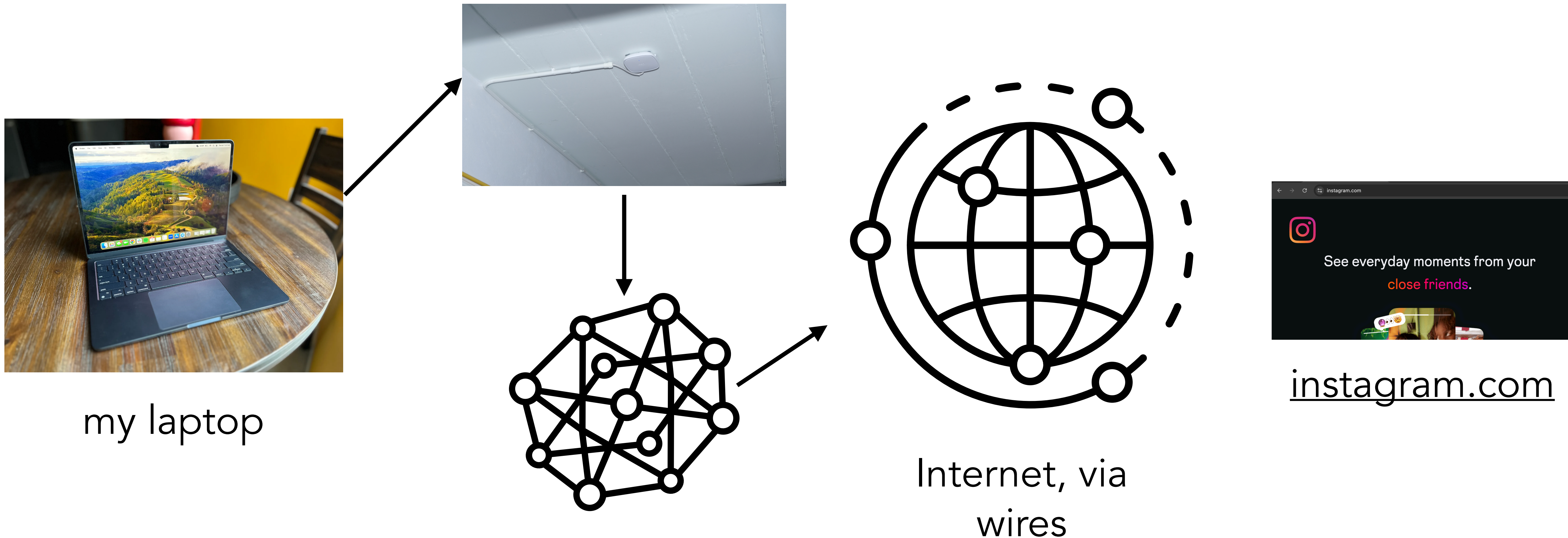
UCSD network, via
wires



instagram.com

What is a network of networks?

- Let's say my computer wants to talk to `instagram.com`. How does it do that?



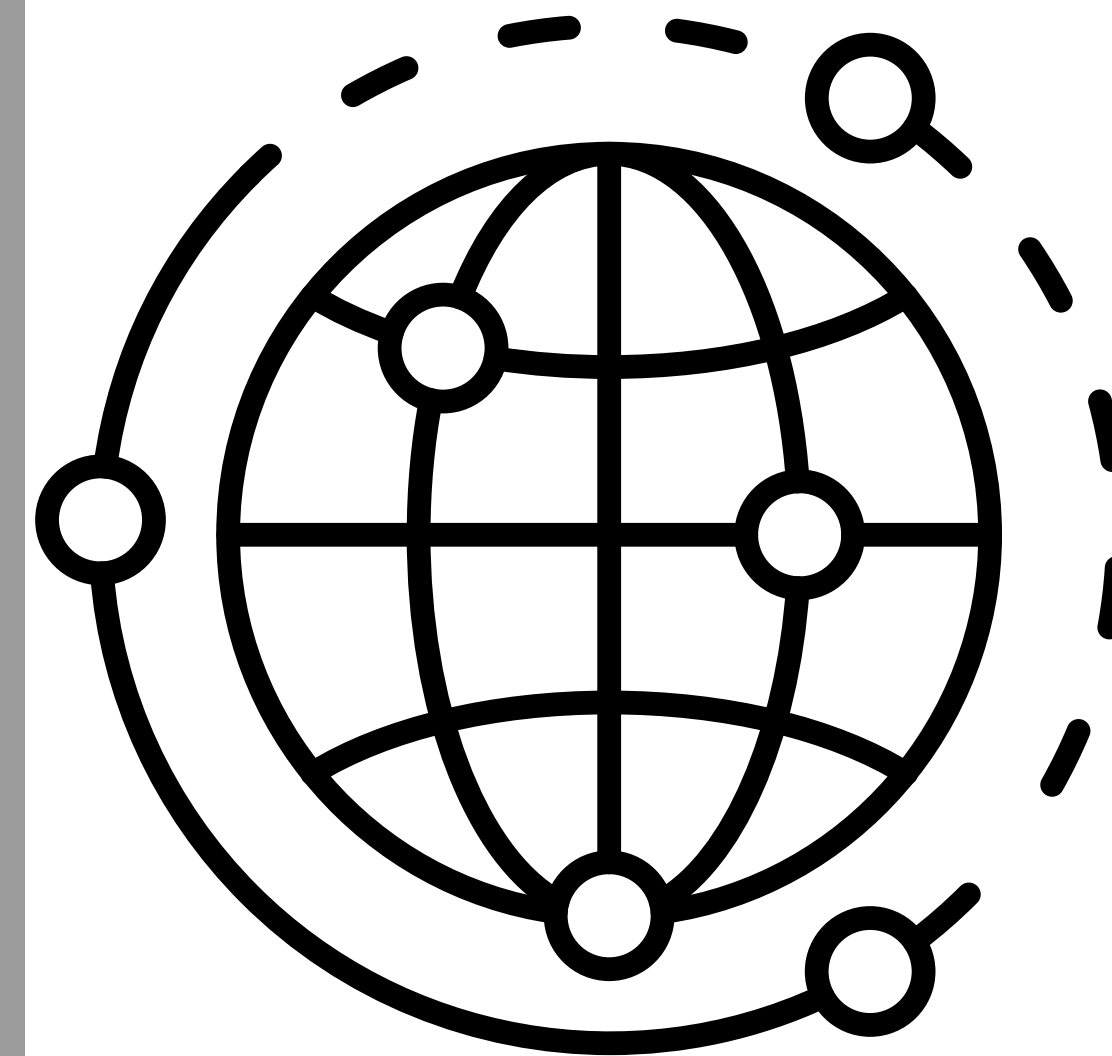
Internet Service Provider: An entity which enables computers access to the Internet

networks?

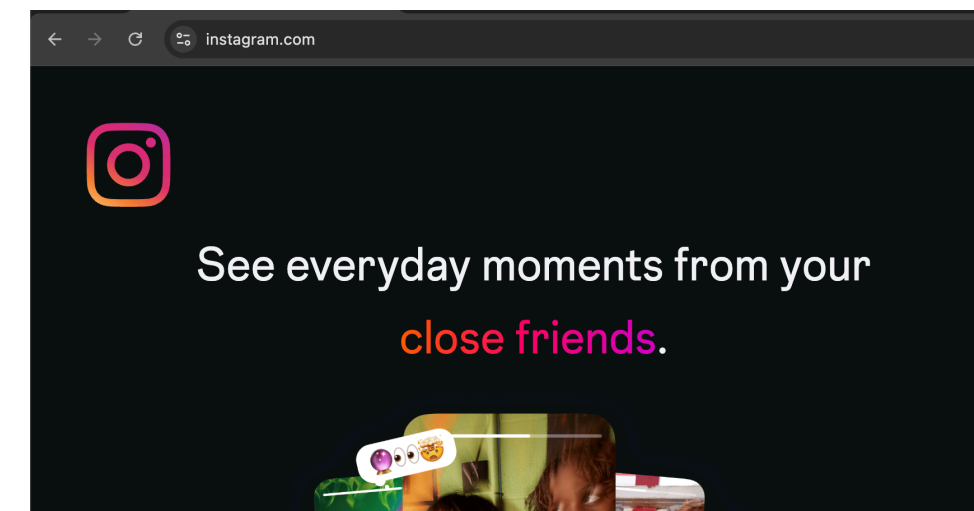
to instagram.com. How does it do that?



my laptop

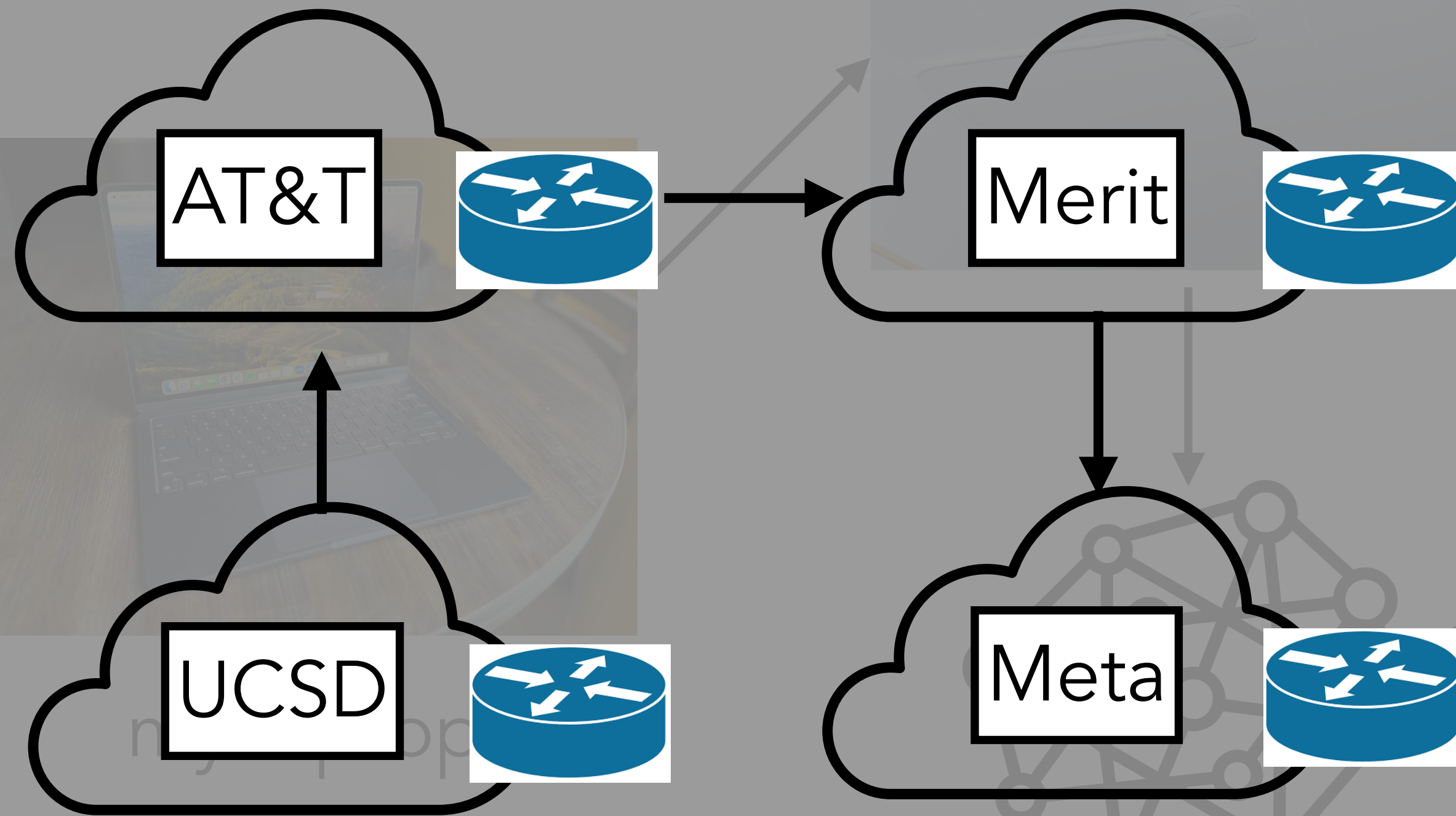


Internet, via wires



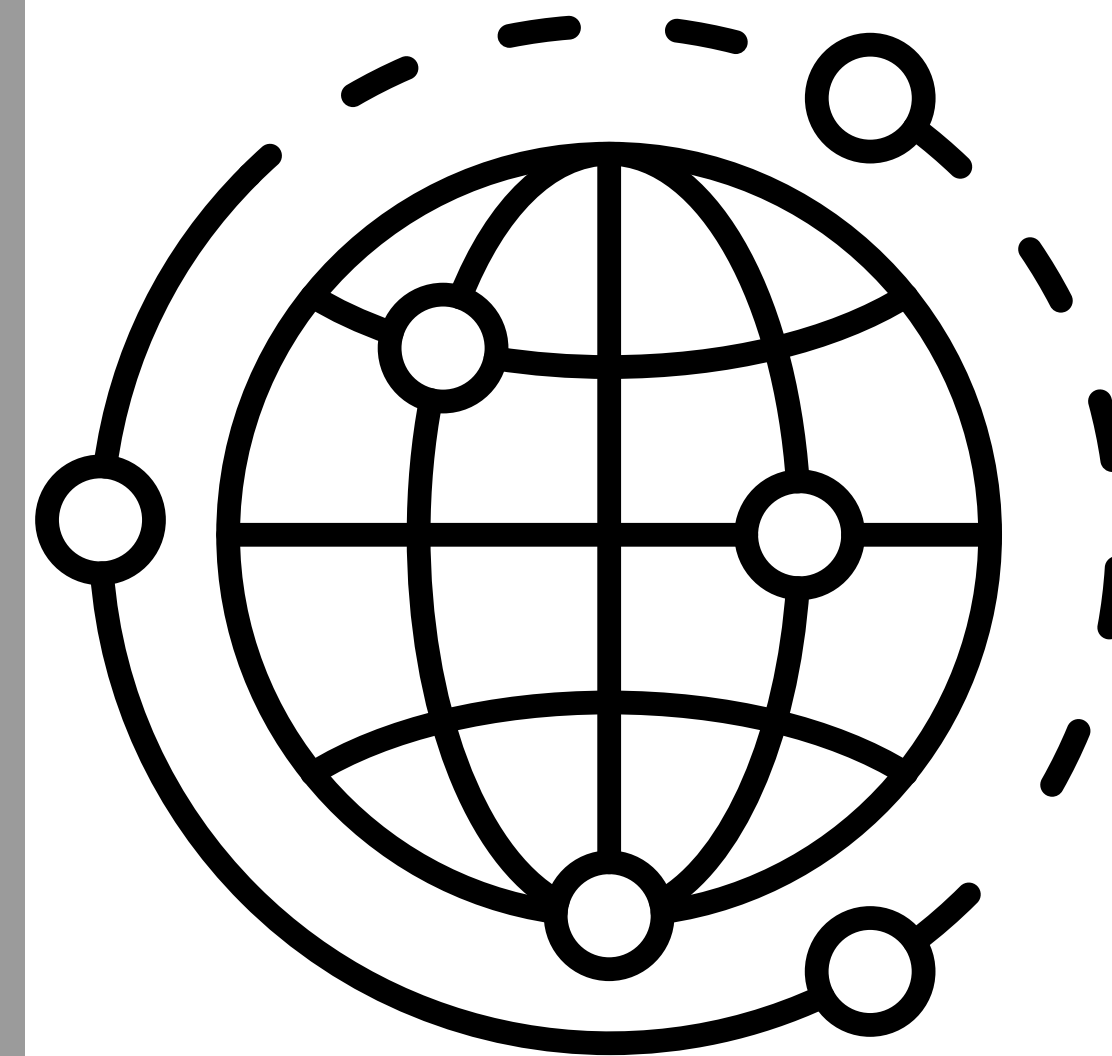
instagram.com

Internet Service Provider: An entity which enables computers access to the Internet

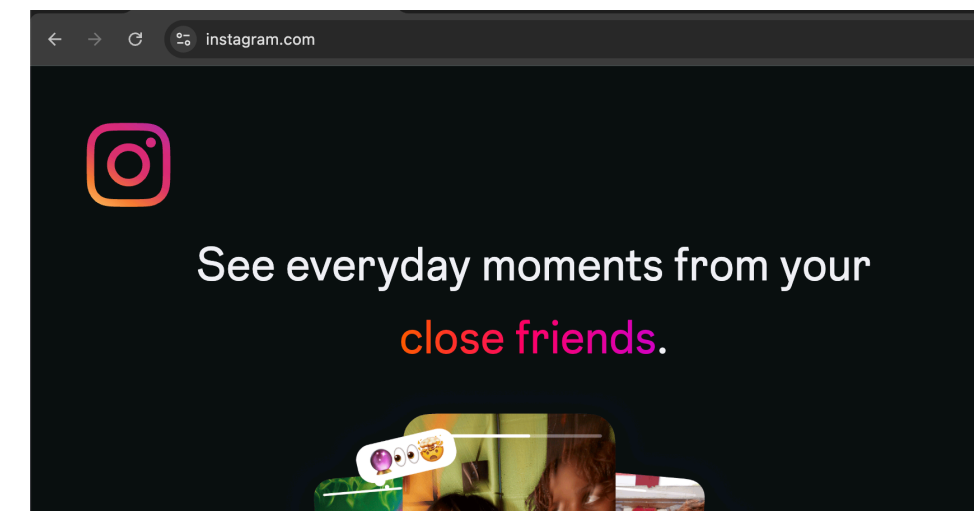


networks?

to instagram.com. How does it do that?



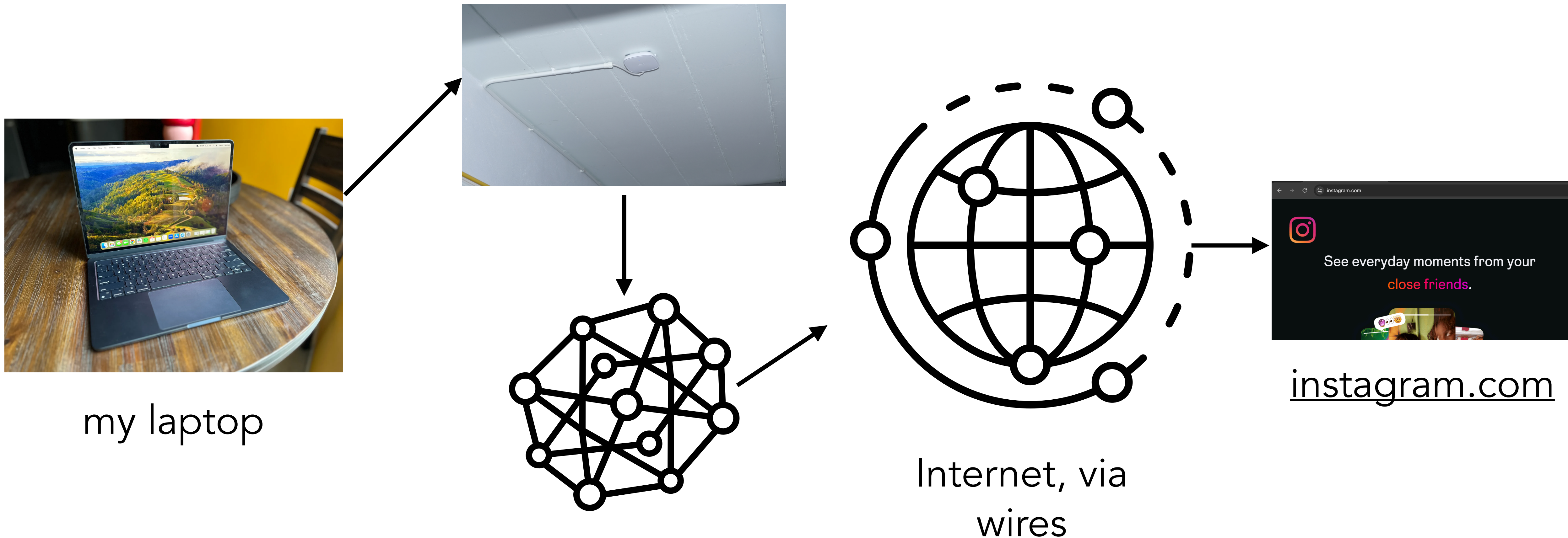
Internet, via
wires



instagram.com

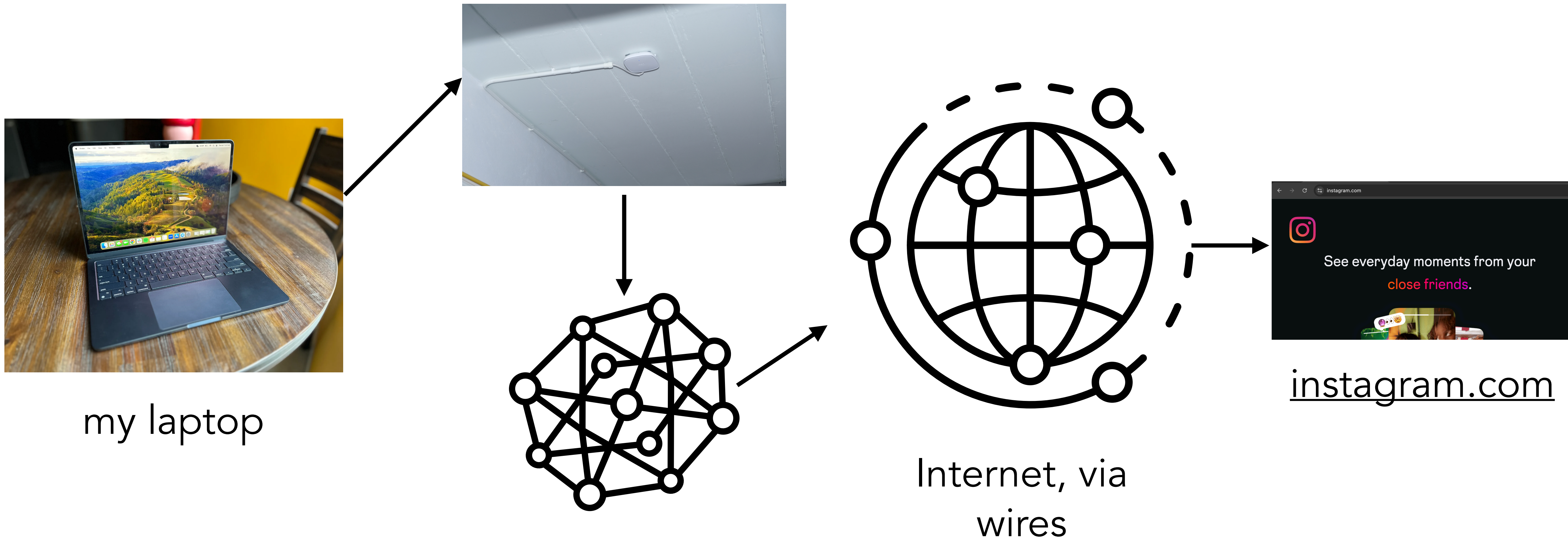
What is a network of networks?

- Let's say my computer wants to talk to `instagram.com`. How does it do that?



What is a network of networks?

- Let's say my computer wants to talk to `instagram.com`. How does it do that?



This all happens in $< 100\text{ms}$ for every request.

Wait, but, that's still magic...

- How does my computer know how to talk to instagram.com?
- How does the network of networks operate under the hood?
- What “data” is being sent from my computer to instagram.com?
- When I get data *back* from instagram.com, how does my computer know what to do with it?

Wait, but, that's still magic...

- How does my computer know how to talk to instagram.com?
- How does the network of networks operate under the hood?
- What “data” is being sent from my computer to instagram.com?
- When I get data *back* from instagram.com, how does my computer know what to do with it?

This is not a networking class, but we'll cover the 10,000 foot view enough for you to understand the pieces today.

How it starts: a packet

- Communication on the Internet is constructed of discrete, *self-addressed* chunks of data: **packets**
 - Networking uses a lot of terms of packets in different contexts... e.g., packet, frame, segment, datagram, cell... sorry
- Ultimately, the bits send on the wire are organized into packets, and there's lots of types of packets
 - UDP packets, TCP packets, HTTP packets... and much more!

The Internet, in short

- With a group, answer the following questions:
 - What is an IP address?
 - What is a protocol?
 - What is a port?
 - What is a domain name?

The Internet, in short

- With a group, answer the following questions:
 - **What is an IP address?**
 - What is a protocol?
 - What is a port?
 - What is a domain name?

IP addresses

- The most common unit of addressing on the modern Internet
 - IPv4 — 32 bits (~4.3 billion unique IP addresses)
 - IPv6 — 128 bits (340 undecillion unique IP addresses)
- Essentially, an IP address is just a number; you've probably seen them all the time
 - IPv4 e.g., 192.168.1.1, 127.0.0.1, 10.0.0.1...
 - IPv6 e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334
- Each "group" (octet or hextet) tells you the numeric value (so, ultimately, it's really just a **number**)

IP addresses

- Two network spaces
 - Public IP space — fixed number of IPs, and every IP is unique
 - **Your goal in PA1 is to deploy something to the public IPv4 Internet.**
 - Side note: Why do we have IPv6?

IP addresses

- Two network spaces
 - Public IP space — fixed number of IPs, and every IP is unique
 - **Your goal in PA1 is to deploy something to the public IPv4 Internet.**
 - Side note: Why do we have IPv6? We ran out of IPs!!!

IP addresses

- Two network spaces
 - Public IP space — fixed number of IPs, and every IP is unique
 - **Your goal in PA1 is to deploy something to the public IPv4 Internet.**
 - Side note: Why do we have IPv6? We ran out of IPs!!!
 - Private IP space — “infinite” number of IPs through network-address translation (e.g. **192.168.0.1**)

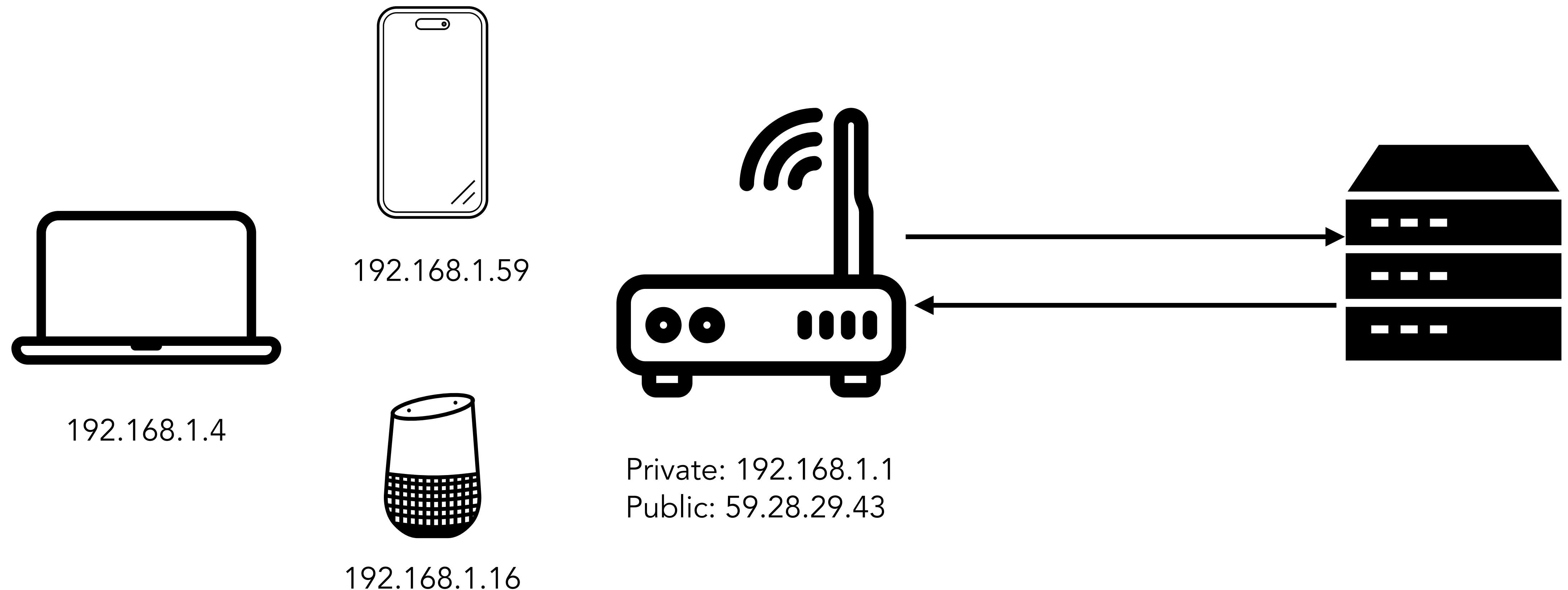
RFC 1918: BCP 5: Address Allocation for Private Internets

Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, E. Lear

BEST CURRENT PRACTICE

Network Address Translation

- Your home router will do what's called **network address translation**: handle packet flow from an infinite number of devices inside your private network to the public world



What does an IP address do for you?

- Routing
 - Your machine knows the location of the **local** router (gateway)
 - IP gateway knows routes to other networks to send packets (and packets usually take many hops to get where they're going)
- Error reporting
 - Send Internet Control Message Protocol (ICMP) packet back to source if there was a problem
- Fragmentation, reassembly, dropping packets
 - Sometimes packets are too big; they get split up; IP takes care of all this under the hood (take CSE 123 if you like this stuff)

IP addresss under the hood

IPv4 header format

<i>Offset</i>	<i>Octet</i>	0								1								2								3							
<i>Octet</i>	<i>Bit</i>	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	<i>Version</i> (4)				<i>IHL</i>				<i>DSCP</i>						<i>ECN</i>		<i>Total Length</i>															
4	32	<i>Identification</i>																<i>Flags</i>			<i>Fragment Offset</i>												
8	64	<i>Time to Live</i>								<i>Protocol</i>								<i>Header Checksum</i>															
12	96	<i>Source address</i>																															
16	128	<i>Destination address</i>																															
20	160																																
⋮	⋮																																
56	448																																

The Internet, in short

- With a group, answer the following questions:
 - What is an IP address?
 - **What is a protocol?**
 - What is a port?
 - What is a domain name?

Network Protocols

- How machines actually *speak* data to one another
- A spec that tells you exactly how to send, receive, and parse bytes of data you receive on the wire from an underlying network
- Usually encoded in documents called RFCs (request for comments) by the Internet Engineering Task Force (IETF)
- ***Hundreds of thousands*** of protocols exist.... not all are used (e.g., Telnet)

Network Working Group
Request for Comments: 2616
Obsoletes: [2068](#)
Category: Standards Track

R. Fielding
UC Irvine
J. Gettys
Compaq/W3C
J. Mogul
Compaq
H. Frystyk
W3C/MIT
L. Masinter
Xerox
P. Leach
Microsoft
T. Berners-Lee
W3C/MIT
June 1999

Hypertext Transfer Protocol -- HTTP/1.1

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

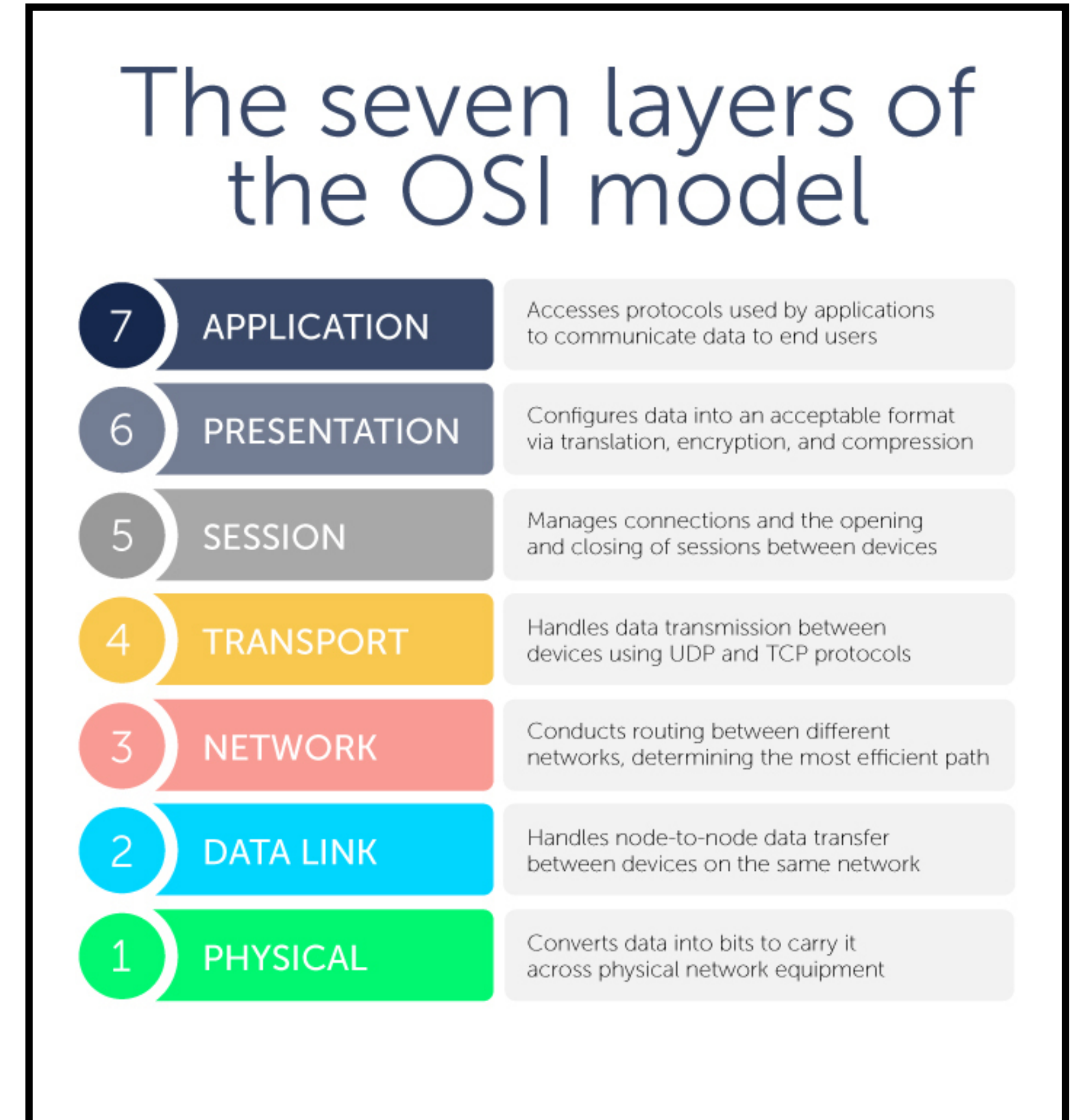
Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers [\[47\]](#). A feature of HTTP is

Anyone know any protocols?

Some example protocols by the layers...

- L7 — HTTP, FTP, SSH
- L4 — UDP, TCP, QUIC
 - User Datagram Protocol: single packet transmission with no reliability
 - Transmission Control Protocol: connection establishment, reliable transmission, flow-control
- L3 — Internet Protocol (IP), provides fragmentation, reassembly, and *routing*
- L2 — Ethernet; communication between wires



The Internet, in short

- With a group, answer the following questions:
 - What is an IP address?
 - What is a protocol?
 - **What is a port?**
 - What is a domain name?

Ports

- Most systems usually come with just one network interface card (NIC) — speaks packets in hardware
 - But so many applications need to use the NIC, so...
- Network protocols run on *ports*; an abstraction provided by operating systems to separate network traffic from one another
- Ports are used for inbound **and** outbound connections
- UDP: IP + ports

UDP header format^[7]

Offset	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source Port																Destination Port															
4	32	Length																Checksum															
8	64	Data																															
12	96																																
⋮	⋮																																

Examples of protocols and default ports

- Protocols are usually operated at a *default port* for communication
 - HTTP — port 80
 - HTTPS — port 443
 - Simple Mail Transfer Protocol (SMTP) — port 25
 - File Transfer Protocol — port 21
 - Secure SHell (SSH) — port 22
 - Telnet — port 23
- But you can run any networked service at any port, really!

The Internet, in short

- With a group, answer the following questions:
 - What is an IP address?
 - What is a protocol?
 - What is a port?
 - **What is a domain name?**

Domains and DNS at 10,000 feet

- Domain names are *human readable* names to index Internet machines
 - e.g., www.instagram.com, ucsd.edu, even *localhost*
- The Domain Name System (DNS) is our global, decentralized system for translating *names* to *IP addresses*
- Why might we need a system to do this?

Domains and DNS at 10,000 feet

- Domain names are *human readable* names to index Internet machines
 - e.g., www.instagram.com, ucsd.edu, even *localhost*
- The Domain Name System (DNS) is our global, decentralized system for translating *names* to *IP addresses*
- Why might we need a system to do this?
 - IP addresses are hard to remember, names are easier!
 - What's easier to remember? 75.2.44.127, or ucsd.edu?

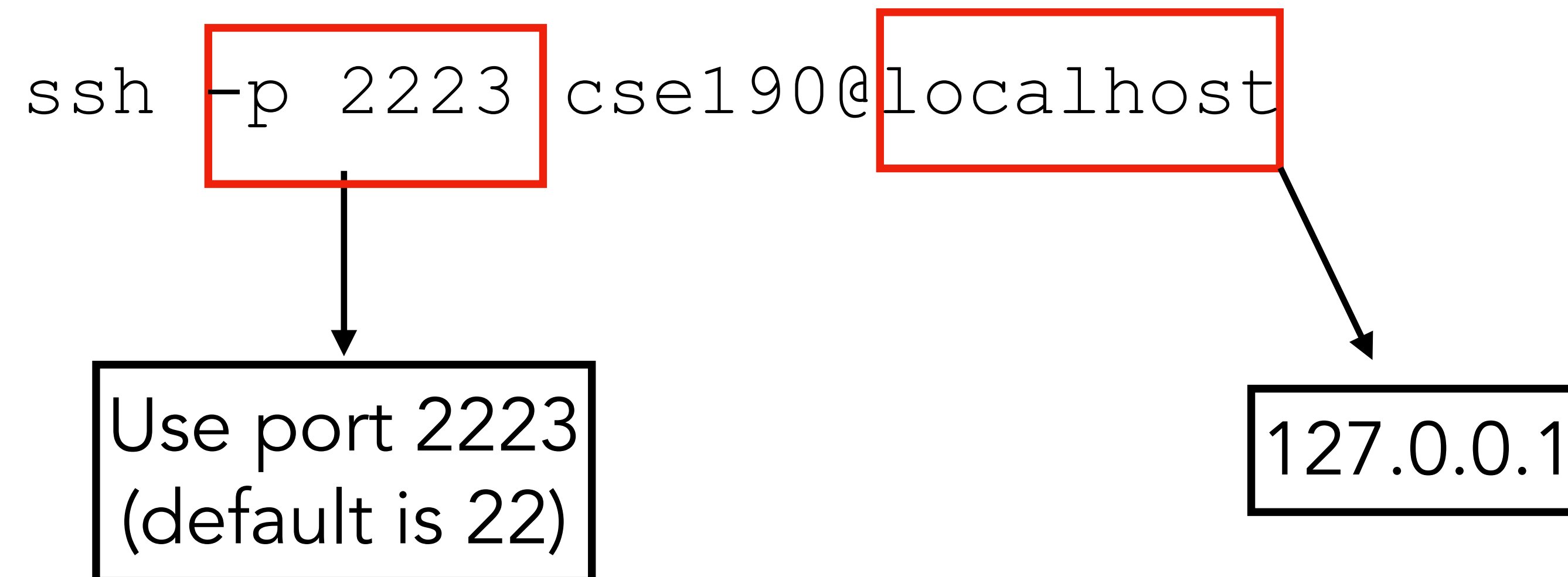
SSH explained

- When you SSH into a machine, you are speaking a *network protocol* that connects *one IP address* to *another IP address*

```
ssh -p 2223 cse190@localhost
```

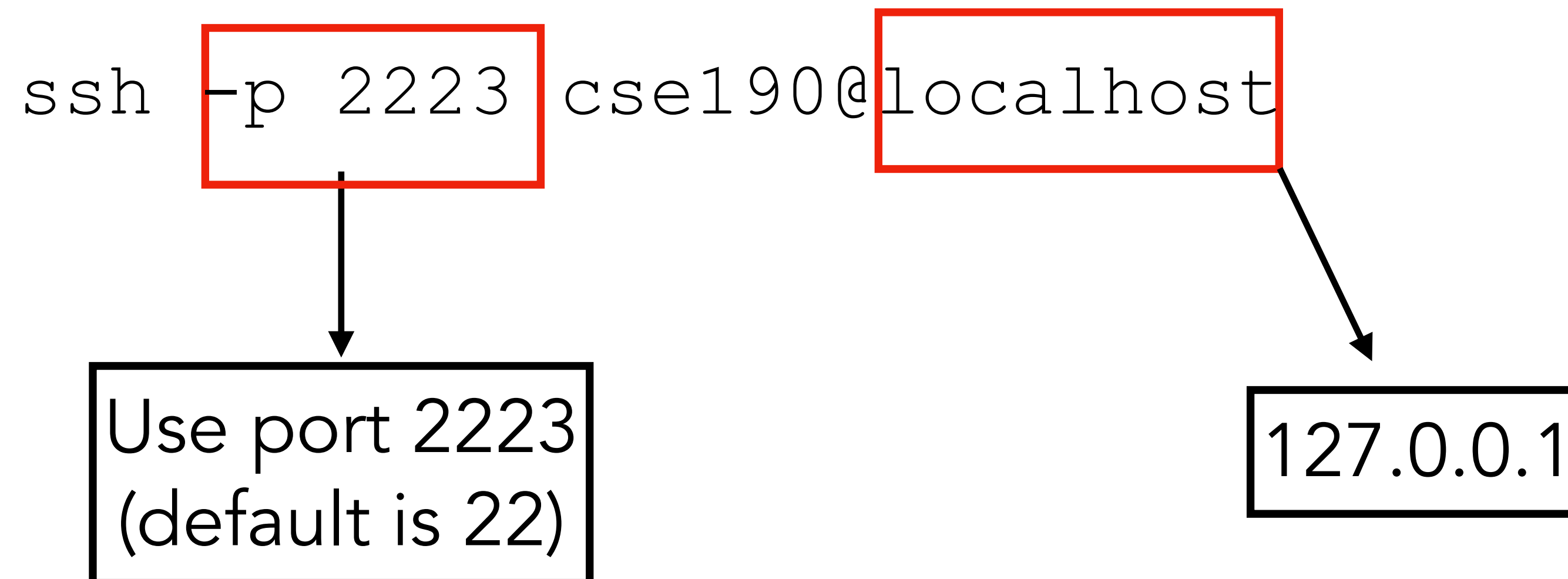

SSH explained

- When you SSH into a machine, you are speaking a *network protocol* that connects *one IP address* to *another IP address*



SSH explained

- When you SSH into a machine, you are speaking a *network protocol* that connects *one IP address* to *another IP address*



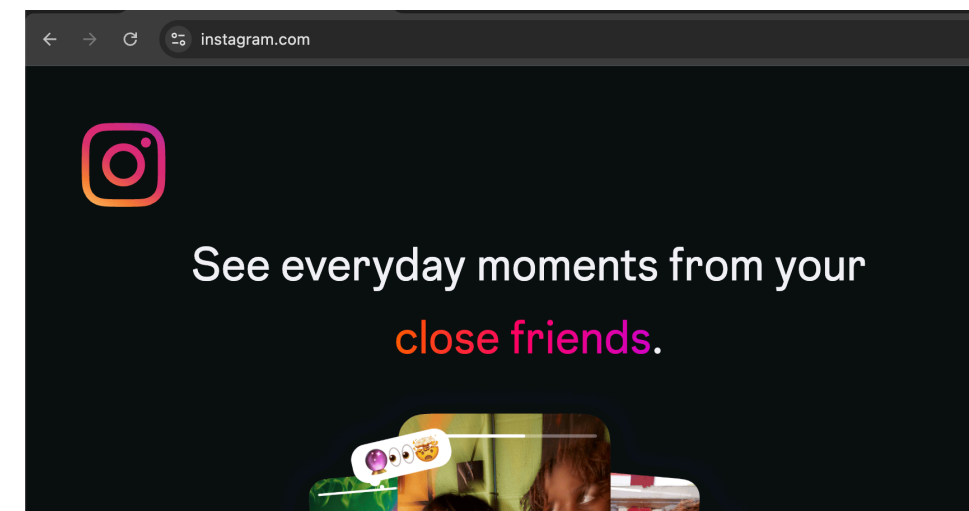
Network services can run on the same machine; your own computer has a **localhost** domain which you can always connect to by default!

The Internet, revisited

- Let's say my computer wants to talk to instagram.com. How does it do that?



my laptop



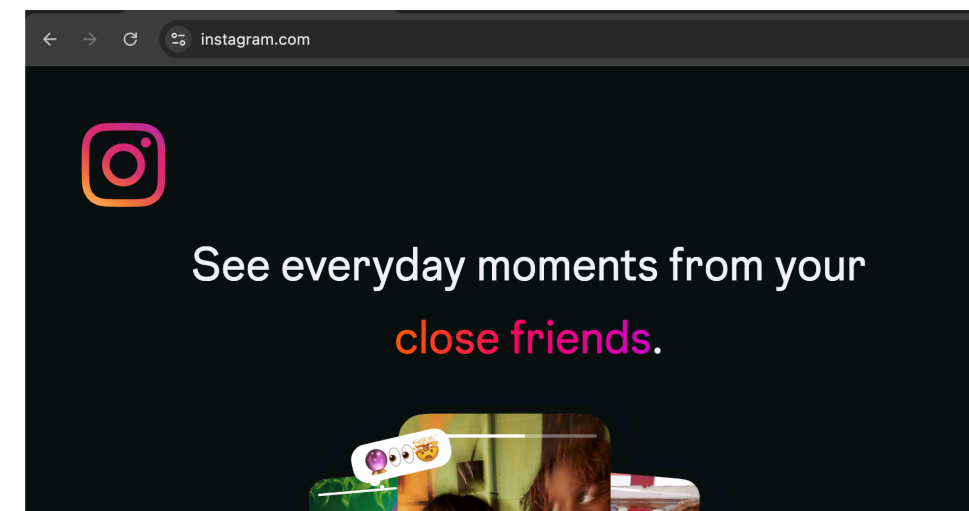
instagram.com

The Internet, revisited

- Let's say my computer wants to talk to `instagram.com`. How does it do that?
 - Laptop makes a DNS request for `instagram.com` (where does the request go?)



my laptop



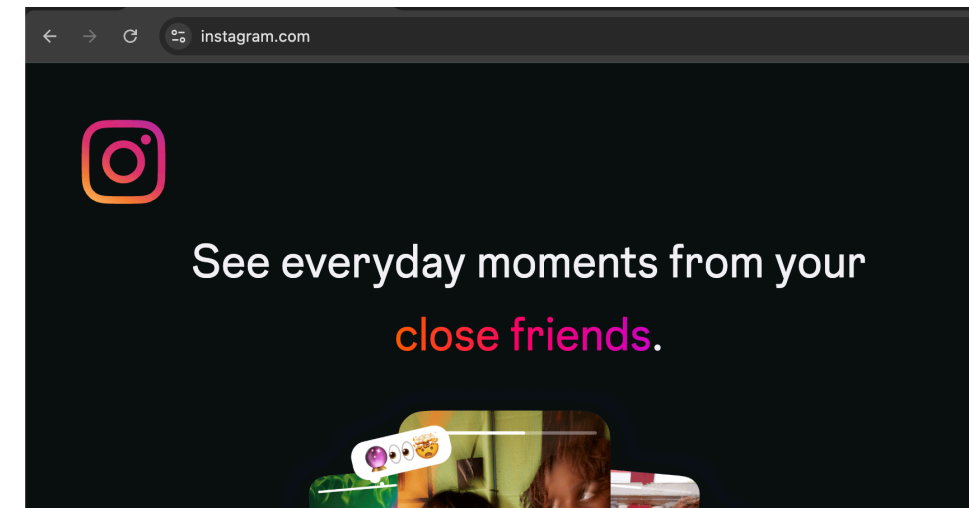
`instagram.com`

The Internet, revisited

- Let's say my computer wants to talk to [instagram.com](https://www.instagram.com). How does it do that?
 - Laptop makes a DNS request for [instagram.com](https://www.instagram.com) (where does the request go?)
 - Computer creates an **HTTP** packet (and all the lower layers) in the right format to send to [instagram.com](https://www.instagram.com) (what software?)



my laptop



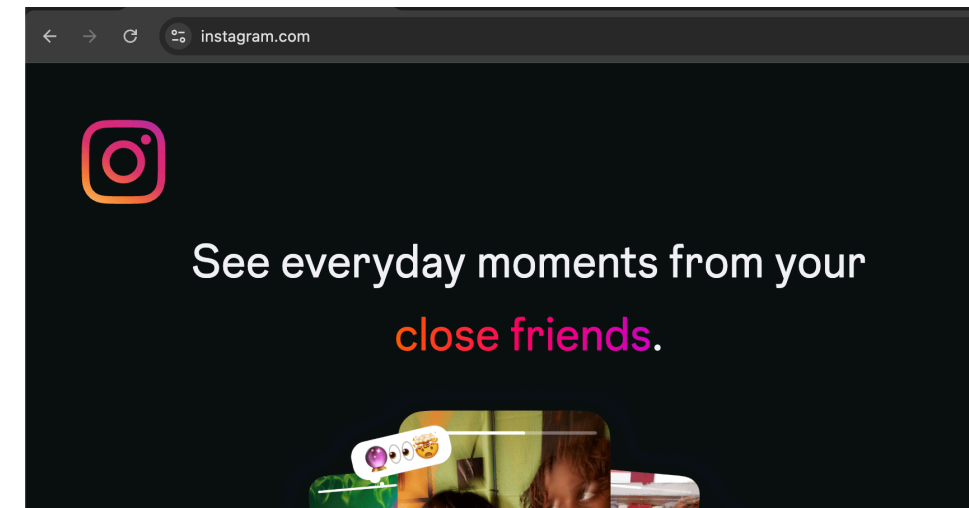
[instagram.com](https://www.instagram.com)

The Internet, revisited

- Let's say my computer wants to talk to [instagram.com](https://www.instagram.com). How does it do that?
 - Laptop makes a DNS request for [instagram.com](https://www.instagram.com) (where does the request go?)
 - Computer creates an **HTTP** packet (and all the lower layers) in the right format to send to [instagram.com](https://www.instagram.com) (what software?)
 - Computer sends the packet to the IP address for [instagram.com](https://www.instagram.com) (what IP address does [instagram.com](https://www.instagram.com) see?)



my laptop



[instagram.com](https://www.instagram.com)

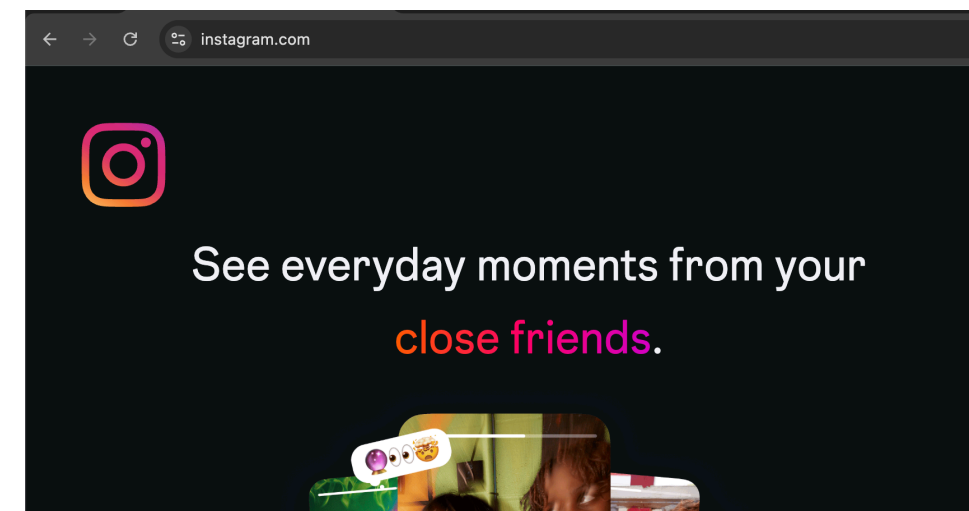
The Internet, revisited

- Let's say my computer wants to talk to [instagram.com](https://www.instagram.com). How does it do that?



my laptop

- Laptop makes a DNS request for [instagram.com](https://www.instagram.com) (where does the request go?)
- Computer creates an **HTTP** packet (and all the lower layers) in the right format to send to [instagram.com](https://www.instagram.com) (what software?)
- Computer sends the packet to the IP address for [instagram.com](https://www.instagram.com) (what IP address does [instagram.com](https://www.instagram.com) see?)
- And... through more engineering magic, it is received on the other end!



[instagram.com](https://www.instagram.com)

Break Time + Attendance



Codeword:
Internet4All!

<https://tinyurl.com/webtracking-attendance>

The Web

Polling the room

- How many people have built a website before?
- How many people have built a web app before?
- How many people have *deployed* a web app before?
 - Where?

What is the web, and how is it different from the Internet?

What is the web, and how is it different from the Internet?

Information system that runs on the Internet that allows *documents* to be connected to other *documents*, increasingly enabled through *scripting* and *server-side logic*

Web Fundamentals

- Take three minutes to answer these questions with folks around you.
 - What is a web server? Examples?
 - What is a web client? Examples?
 - What is HTTP?
 - What is a web application?



Web Fundamentals

- Take three minutes to answer these questions with folks around you.
 - **What is a web server? Examples?**
 - **What is a web client? Examples?**
 - **What is HTTP?**
 - **What is a web application?**



Interfacing with the Web

Client / Server Model



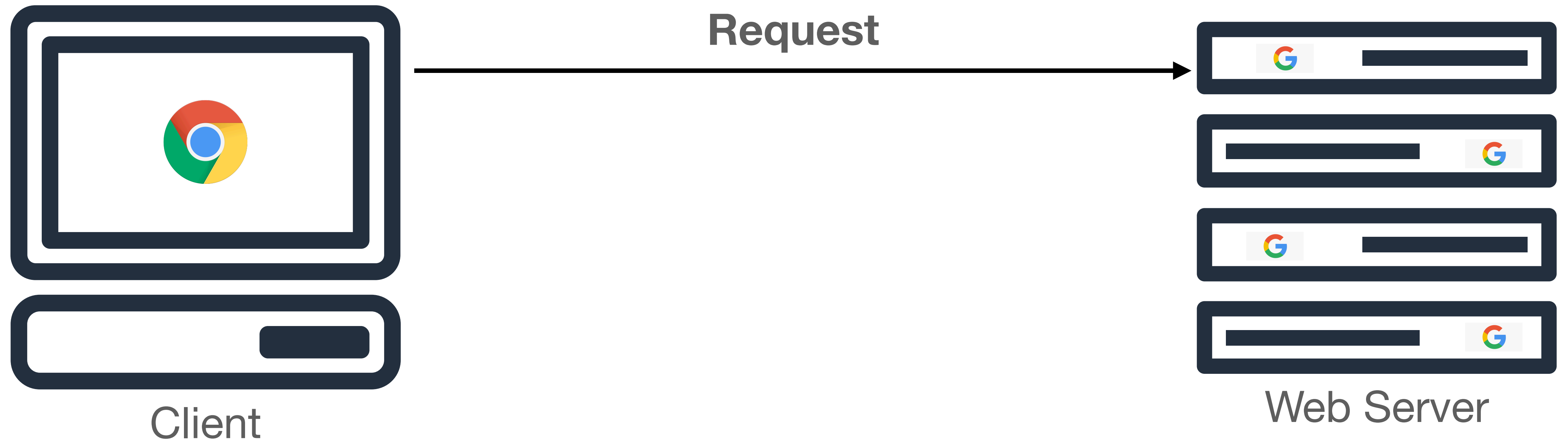
Client



Web Server

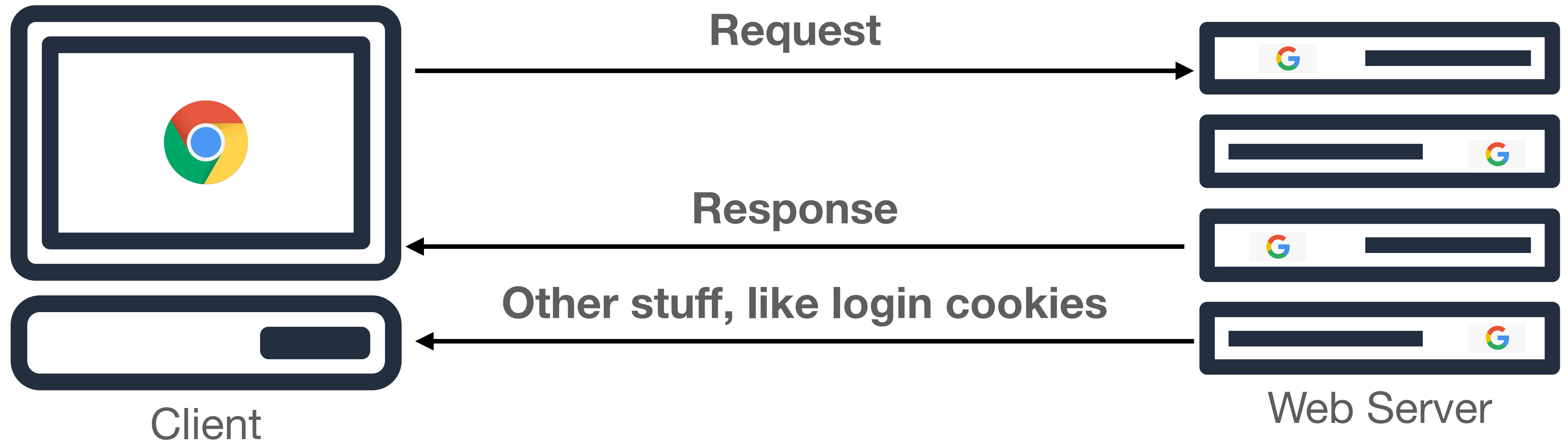
Interfacing with the Web

Client / Server Model



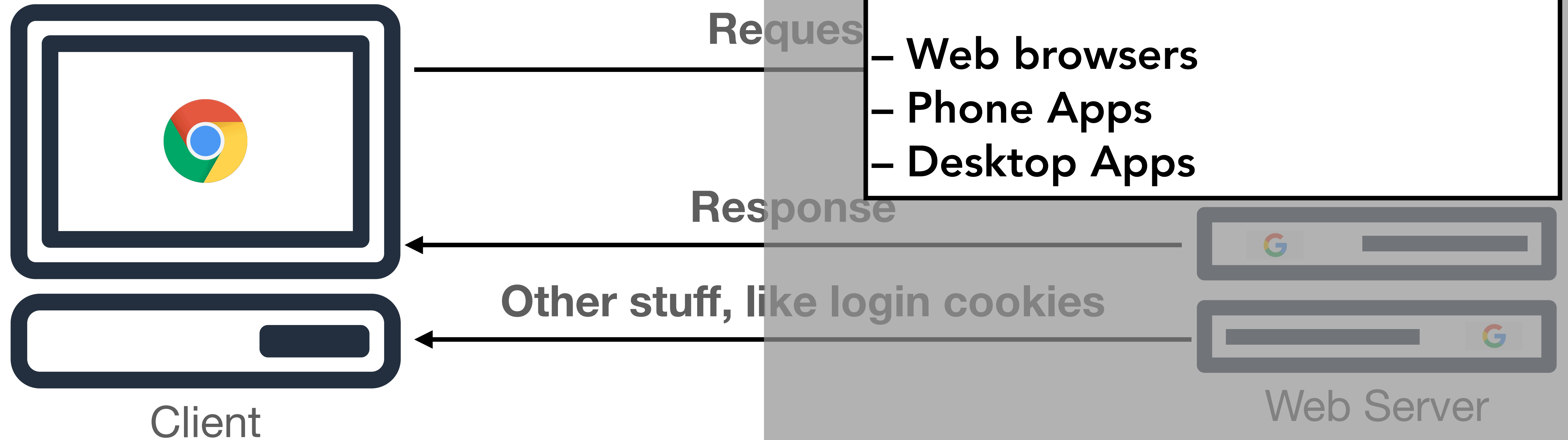
Interfacing with the Web

Client / Server Model



Interfacing with the Web

Client / Server Model



Modern web architecture

- Web client (typically a browser) issues requests
- Web server responds
- Web client will handle the response for the user

Web clients

- How do web browsers make requests to web servers?

Web clients

- How do web browsers make requests to web servers?
 - User types in a URL into the browser bar
 - User re-loads a page
 - User clicks on a link
 - Web server responds with a redirect (aka, browser should request a new page)
 - Web page could embed another page (independent of user interaction)
 - Script within a webpage can issue a request

Web servers

- What are the ways in which web servers can respond to client requests?

Web servers

- What are the ways in which web servers can respond to client requests?
 - Return a document (HTML) for the page
 - Return any other resource (e.g., images, scripts, CSS, etc.)
 - Run some custom logic or code and return outputs (e.g., APIs)
 - Talk to some external system and return output (e.g., database)
 - Do nothing and waste your time

Web client response handling

- What are some ways that web clients handle server responses?

Web client response handling

- What are some ways that web clients handle server responses?
 - Render HTML + CSS
 - Execute embedded JavaScript
 - Invoke some external plugin (e.g., PDF reader)

Web Fundamentals

- Take three minutes to answer these questions with folks around you.
 - What is a web server? Examples?
 - What is a web client? Examples?
 - **What is HTTP?**
 - What is a web application?



Hypertext Transfer Protocol (HTTP)

- HTTP — a protocol invented in 1989 that still power the web today
 - Basic concept is to allow fetching of resources (e.g., HTML documents, nowadays, basically anything)
- How do we identify where a resource is on the web?

Hypertext Transfer Protocol (HTTP)

- HTTP — a protocol invented in 1989 that still power the web today
 - Basic concept is to allow fetching of resources (e.g., HTML documents, nowadays, basically anything)
- How do we identify where a resource is on the web?
 - We use what's called a Uniform Resource Locator (URL)



How the Web Tracks You / Targeted Advertising

UCSD CSE 190/291, Fall 2026

The goal of the course is to introduce students to the vast tracking ecosystem that underlies the web. Everyone is likely aware that the price users pay for "free" information online -- from the YouTube video you use to study to your favorite niche subreddit -- is seeing dozens of targeted ads. But fewer are fully aware of the immense volumes of data that are stored and sold in order to place those advertisements. And fewer still understand the technologies used to power this economy. This course will show why the web evolved this way, how those targeted ads are placed, and why companies are willing to pay for this advertising.

Specific topics will include technical introductions to fundamentals of the web (i.e., the protocols and infrastructure that support the web today), tracking techniques including cookies and browser fingerprinting, the mechanics of ad auctions, as well as marketing databases and targeting inferences. Students will also learn about the history of advertising, tactics companies use to persuade consumers, and the ethical implications of a surveillance capitalism economy.

URLs demystified

- All resources on the web have a URL, which is broken into several parts:

`https://www.kumarde.com:443/classes/fa26/cse190/lectures/web_lec.pdf?lang=en`

URLs demystified

- All resources on the web have a URL, which is broken into several parts:
 - Scheme: Tells you what protocol you're using to communicate. On web, typically HTTP or HTTPS.

`https://www.kumarde.com:443/classes/fa26/cse190/lectures/web_lec.pdf?lang=en`

URLs demystified

- All resources on the web have a URL, which is broken into several parts:
 - Domain: The human-readable name that describes the server you're trying to talk to. kumarde.com and www.kumarde.com are *different domains*.

`https://www.kumarde.com:443/classes/fa26/cse190/lectures/web_lec.pdf?lang=en`

URLs demystified

- All resources on the web have a URL, which is broken into several parts:
 - Port: The network port to communicate with the server. HTTP default runs on 80, HTTPS on 443. Lots of default ports (22 for SSH, 23 for Telnet, etc.)

`https://www.kumarde.com:443/classes/fa26/cse190/lectures/web_lec.pdf?lang=en`

URLs demystified

- All resources on the web have a URL, which is broken into several parts:
 - Path: The full “file path” to the resource you’re looking for. Mirrors file path structure on most servers. APIs leverage paths to designate resources.

`https://www.kumarde.com:443/classes/fa26/cse190/lectures/web_lec.pdf?lang=en`

URLs demystified

- All resources on the web have a URL, which is broken into several parts:
 - Query string: Typically includes parameters delimited by "&" that contain useful information, e.g., language, etc.

https://www.kumarde.com:443/classes/fa26/cse190/lectures/web_lec.pdf?lang=en

HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



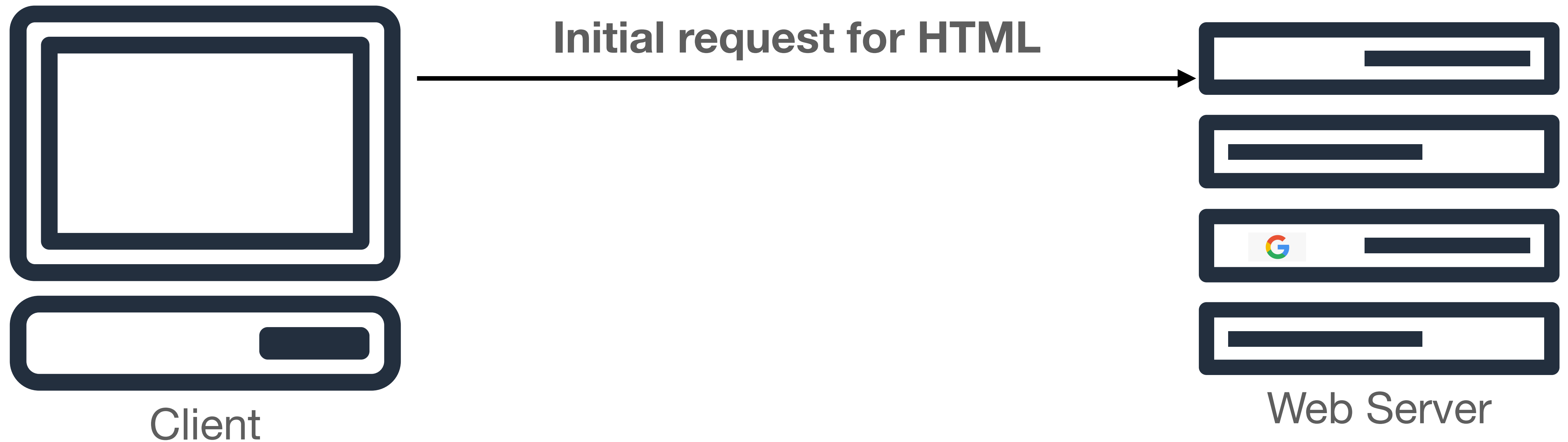
Client



Web Server

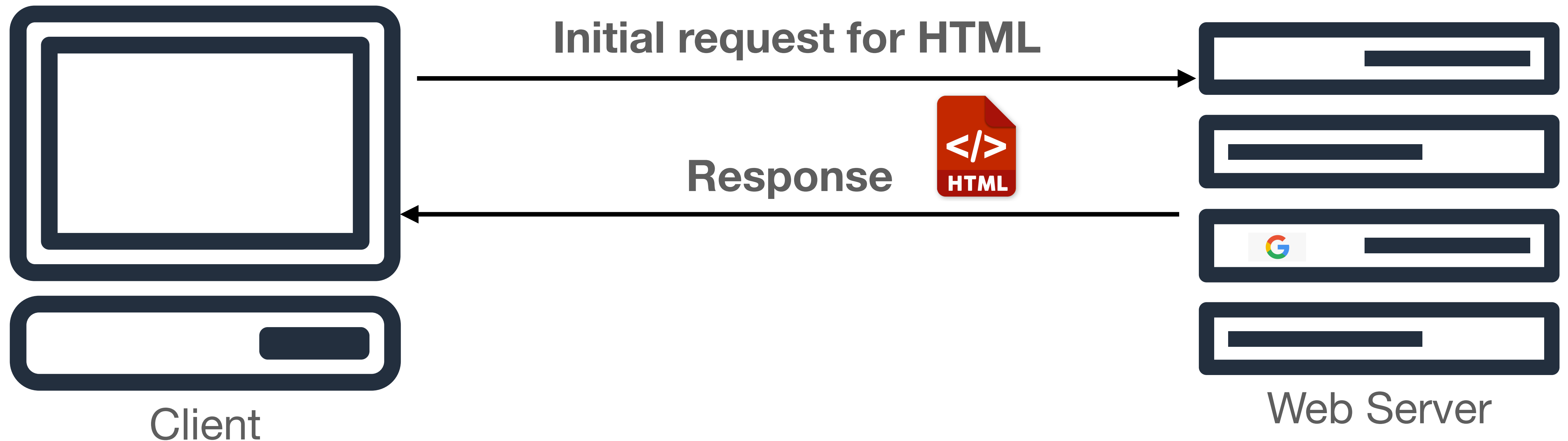
HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



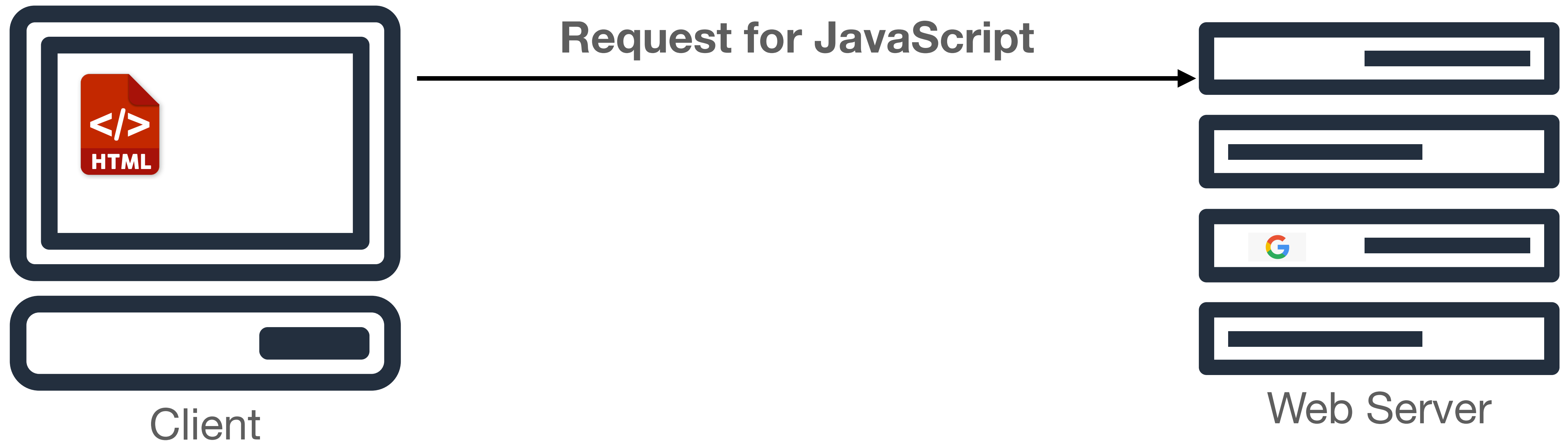
HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



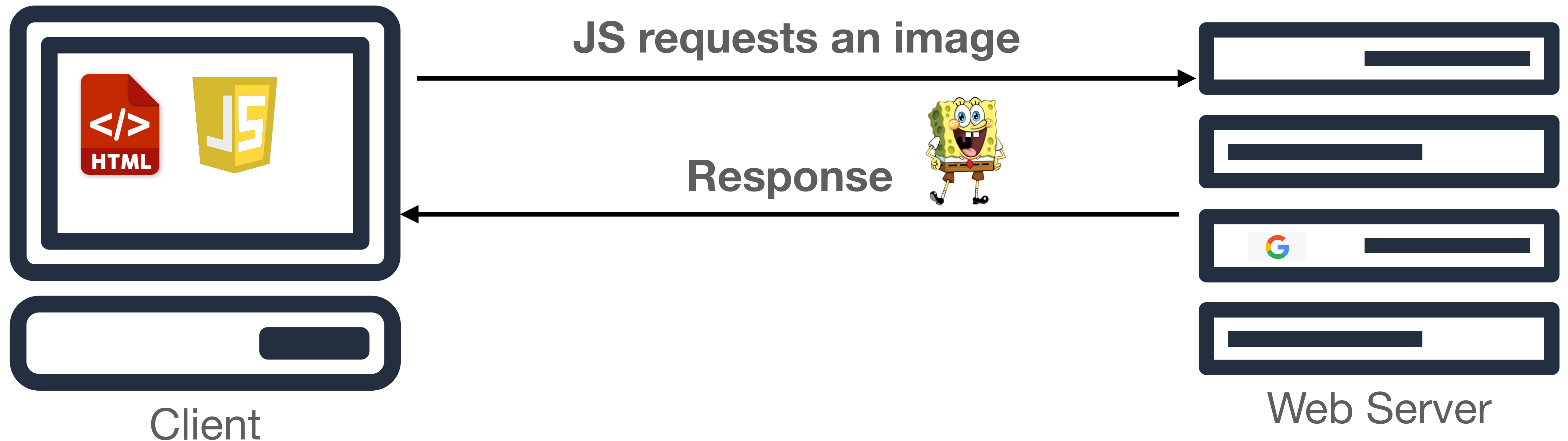
HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



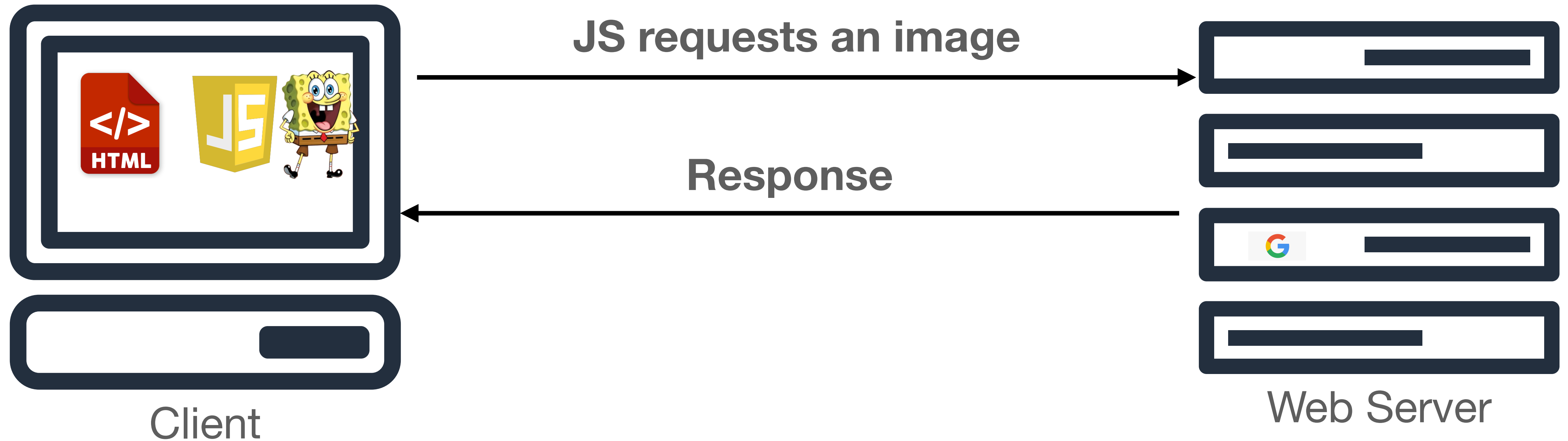
HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



HTTP is request-response

- Unlike other network protocols, HTTP is strictly **request-response**; meaning clients and servers exchange individual messages (as opposed to a stream of data)



HTTP Request Anatomy

- What does an HTTP Request look like?

```
GET /index.html HTTP/1.1
```

```
Accept: image/gif, image/x-bitmap, image/jpeg, */*
```

```
Accept-Language: en
```

```
Connection: Keep-Alive
```

```
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
```

```
Host: www.example.com
```

```
Referer: http://www.google.com?q=dingbats
```

HTTP Request Anatomy

- What does an HTTP Request look like?

Method Path Protocol Version

```
GET /index.html HTTP/1.1
```

```
Accept: image/gif, image/x-bitmap, image/jpeg, */*
```

```
Accept-Language: en
```

```
Connection: Keep-Alive
```

```
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
```

```
Host: www.example.com
```

```
Referer: http://www.google.com?q=dingbats
```


HTTP Request Anatomy

- What does an HTTP Request look like?

```
GET /index.html HTTP/1.1
```

```
Accept: image/gif, image/x-bitmap, image/jpeg, */*
```

```
Accept-Language: en
```

```
Connection: Keep-Alive
```

```
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
```

```
Host: www.example.com
```

```
Referer: http://www.google.com?q=dingbats
```

HTTP Headers

HTTP Request Anatomy

- What does an HTTP Request look like?

```
GET /index.html HTTP/1.1
```

```
Accept: image/gif, image/x-bitmap, image/jpeg, */*
```

```
Accept-Language: en
```

```
Connection: Keep-Alive
```

```
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
```

```
Host: www.example.com
```

```
Referer: http://www.google.com?q=dingbats
```

Body (usually empty for requests)

Aside: HTTP Versions

- Today's servers generally support a mix of HTTP 1.1, 2, and 3
 - Differences are *mainly* about efficiency; solving head-of-line blocking problems, relying on different transport protocols, etc.
 - Modern browsers can speak everything
- HTTP/2 used by 34.1% of all websites (per w3techs)... actually **declining over time**
 - Pipelining requests for multiple objects, header compression, server push
- HTTP/3 used by 37% of all websites (per w3techs)
 - Uses QUIC instead of TCP

HTTP Response Anatomy

- What does an HTTP Response look like?

```
HTTP/1.0 200 OK
Date: Sun, 21 Apr 1996 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Thu, 18 Apr 1996 17:39:05 GMT
Set-Cookie: ...
Content-Length: 2543
```


HTTP Response Anatomy

- What does an HTTP Response look like?

Status Code

```
HTTP/1.0 200 OK
Date: Sun, 21 Apr 1996 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Thu, 18 Apr 1996 17:39:05 GMT
Set-Cookie: ...
Content-Length: 2543
```

HTTP Response Anatomy

- What does an HTTP Response look like?

Status Code

```
HTTP/1.0 200 OK
Date: Sun, 21 Apr 1996 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Thu, 18 Apr 1996 17:39:05 GMT
Set-Cookie: ...
Content-Length: 2543
```

HTTP Headers

HTTP Response Anatomy

- What does an HTTP Response look like?

Status Code

```
HTTP/1.0 200 OK
```

```
Date: Sun, 21 Apr 1996 02:20:42 GMT
```

```
Server: Microsoft-Internet-Information-Server/5.0
```

```
Connection: keep-alive
```

```
Content-Type: text/html
```

```
Last-Modified: Thu, 18 Apr 1996 17:39:05 GMT
```

```
Set-Cookie: ...
```

```
Content-Length: 2543
```

HTTP Headers

```
<html> Hello there! </html>
```


HTTP Basics

- Client sends HTTP requests, each come with a **method**
 - GET: retrieve a resource
 - POST: update a resource (submit a form, publish a post, etc.)
 - Anyone have a guess as to what HEAD does?
 - There are a few others (PUT, PATCH, DELETE); older browsers don't use them
- Servers send HTTP responses, each come with a **status code**
 - 200: OK
 - 302: Found (redirect)
 - 404: Not found

HTTP Basics

- Websites have lots of resources: you should “Inspect Element” in your browser to see them

Deepak Kumar

Assistant Professor
University of California San Diego
kumarde@ucsd.edu
[publications](#) | [cv](#) | [scholar](#)

I study the security and digital safety threats that emerge when people interact with sociotechnical systems at scale. These days, I've been calling this subfield *sociotechnical cybersecurity*. I blend large-scale, data-driven measurements with human-centered studies to understand how such threats, which range from misinformation to online abuse, are operationalized in practice and how people experience them. Ultimately, my work contributes insights, data, defenses, and systems that seek to make the Internet a safer place for all people.

If you've come looking for my creative work, go [here](#).

Students

I will be recruiting PhD students for Fall 2026. Please mark me in your application to UCSD if you are excited about joining our lab.

I am always interested in working with motivated students on research. You can find a page that details my interests and [what you can expect from me as an advisor here](#).

- Arshia Arya, PhD Student
- Haodi Zou, PhD Student
- Seoyoung Kweon, PhD Student (with Stefan Savage, Geoff Voelker)
- Paul Chung, PhD Student (with Stefan Savage, Geoff Voelker)

Teaching

- Winter 2026: CSE127 Computer Security (undergrad)
- Winter 2025: CSE227 Computer Security (grad)
- Fall 2024: CSE291 Sociotechnical Cybersecurity (grad)

Select Publications

Characterizing the MrDeepFakes Sexual Deepfake Marketplace
Best paper honorable mention!
Catherine Han, Anne Li, [Deepak Kumar](#), Zakir Durumeric. USENIX Security Symposium (USENIX), August 2025. [\[pdf\]](#)

Deepak Kumar

Inspect Element

Network

Name	Status	Type	Initiator	Size	Time
kumarde.com	304	docum...	Other	0.1 kB	24 ms
js?id=UA-77723744-1	307	script / ...	(index):5	0.0 kB	63 ms
index.css	200	stylesh...	(index):15	(memo...	0 ms
css?family=Lato:600,400,300...	200	stylesh...	(index):16	(memo...	0 ms
css?family=Montserrat&displ...	200	stylesh...	(index):17	(memo...	0 ms
jquery.min.js	200	script	(index):18	(memo...	0 ms
2f9be15f0a.js	200	script	(index):19	(memo...	0 ms
jquery-3.4.1.slim.min.js	200	script	(index):20	(memo...	0 ms
popper.min.js	200	script	(index):21	(memo...	0 ms
bootstrap.min.js	200	script	(index):22	(memo...	0 ms
ucsd-headshot.jpg	200	jpeg	(index):59	(memo...	0 ms
bootstrap.min.css	200	stylesh...	(index):14	(disk c...	4 ms
css?family=Fira+Sans:300,70...	200	stylesh...	index.css:1	(memo...	0 ms
google-analytics_analytics.js	Finished	script / ...	js	0.0 kB	13 ms
google-analytics_analytics.js	200	script	google-analytics	3.7 kB	1 ms
va9B4kDNxMZdWfMOD5VnP...	200	font	css?family=Fira+S	(memo...	0 ms
va9B4kDNxMZdWfMOD5VnL...	200	font	css?family=Fira+S	(memo...	0 ms
free.min.css?tokens=2f9be15f...	200	fetch	2f9be15f0a.js:2	(disk c...	2 ms
free-v4-shims.min.css?token...	200	fetch	2f9be15f0a.js:2	(disk c...	2 ms
free-v4-font-face.min.css?to...	200	fetch	2f9be15f0a.js:2	(disk c...	2 ms
content.js	200	script	import-content.js	119 kB	1 ms
actions-639cd34d.js	200	script	content.js:1	102 kB	2 ms
helpers-10be1fb3.js	200	script	content.js:2	4.1 kB	1 ms
globalStyles-22fbf6ab.js	200	script	content.js:3	150 kB	2 ms
favicon.ico	304	vnd.mi...	Other	0.1 kB	129 ms

Web sessions

- HTTP is a **stateless protocol**. What does this mean?

Web sessions

- HTTP is a **stateless protocol**. What does this mean?
 - Requests are processed independently; no notion of a *session* by default
- But.... most web applications are session-based, so how?

Web sessions

- HTTP is a **stateless protocol**. What does this mean?
 - Requests are processed independently; no notion of a *session* by default
- But.... most web applications are session-based, so how?
- Cookies!



Web sessions

- HTTP is a **stateless protocol**. What does this mean?
 - Requests are processed independently; no notion of a *session* by default
- But.... most web applications are session-based, so how?
- Cookies! 
- Cookies are used for lots of things, good and bad, including
 - Sessions (e.g., login, shopping carts)
 - Personalization (e.g., user preferences, themes, etc.)
 - **Tracking** (e.g., online advertising)

How do web cookies work?

- The web server sets cookies in a response to the web browser (using an HTTP header)
 - Set-Cookie: <cookie-name>=<cookie-value>;
 - Also defines “properties” for each cookie
 - e.g., when they expire, what domains they’re for, what protocols to use with, etc.
- Browser automatically attaches cookies to **every subsequent request to that web server**
 - Unless you disable them, but... you’ll be annoyed

Web Cookie Typology

- Session cookies
 - Expiration property is not set
 - Exist *only* during current browser session
 - Deleted when your browser is shut down
 - Examples?



Web Cookie Typology

- Session cookies
 - Expiration property is not set
 - Exist *only* during current browser session
 - Deleted when your browser is shut down
 - Examples?
- Persistent cookies
 - Saved until server-defined expiration time
 - Examples?



Server sets cookies in headers



```
HTTP/1.0 200 OK
Date: Sun, 21 Apr 1996 02:20:42 GMT
Server: Microsoft-Internet-Information-Server/5.0
Connection: keep-alive
Content-Type: text/html
Last-Modified: Thu, 18 Apr 1996 17:39:05 GMT
Set-Cookie: trackingID=253407892435087
Content-Length: 2543
```

Client sends cookies every request

```
GET /index.html HTTP/1.1

Accept: image/gif, image/x-bitmap, image/jpeg, */*
Accept-Language: en
Connection: Keep-Alive
User-Agent: Mozilla/1.22 (compatible; MSIE 2.0; Windows 95)
Host: www.example.com
Cookie: trackingID=253407892435087
Referer: http://www.google.com?q=dingbats
```


Web Fundamentals

- Take three minutes to answer these questions with folks around you.
 - What is a web server? Examples?
 - What is a web client? Examples?
 - What is HTTP?
 - **What is a web application?**



Putting it all together: web applications

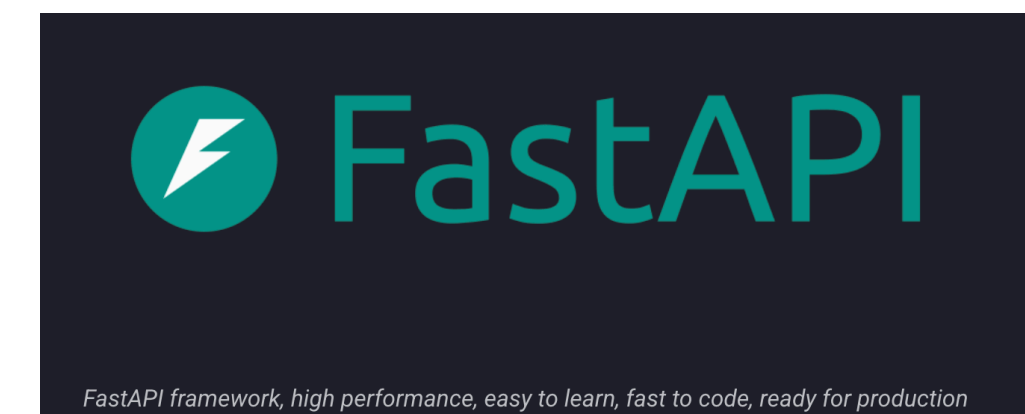
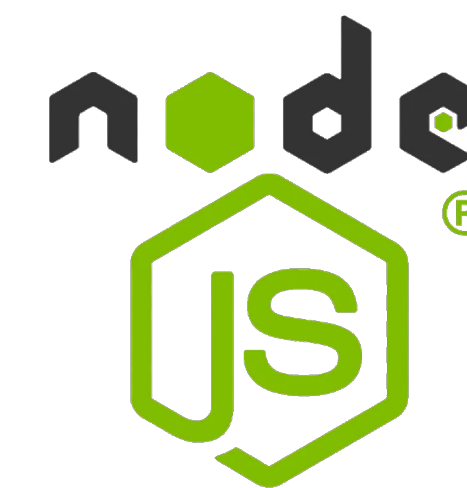
A software system that runs on a remote server and is interfaced with via a browser



Client



frontend



(example) backends

What's behind the backend? (Next time)

What's behind the backend? (Next time)



- This specific database monster, who you will meet next time!
- ... Brought to you by this prompt: "Make a database monster. Interpret as you will."
- Next time is all about databases and advertising! Kristen will be lecturing, so I'll see you next Tuesday, where we'll talk about JavaScript, cookies, and more web