CS 43: Computer Networks

01: Course Administration & Introduction

September 8, 2020

Slides adapted from Kurose & Ross, Kevin Webb

Today

- What is this course about?
- Introduction
  - What does it take to transmit a packet over the Internet?
- Course Administration
  - Structure & Grading
  - Academic Honesty
  - How does this class work?

What This Class is about

How networks (focus on the Internet) work

Mobile phone

www.google.com

Google Server

The Internet is Exciting!

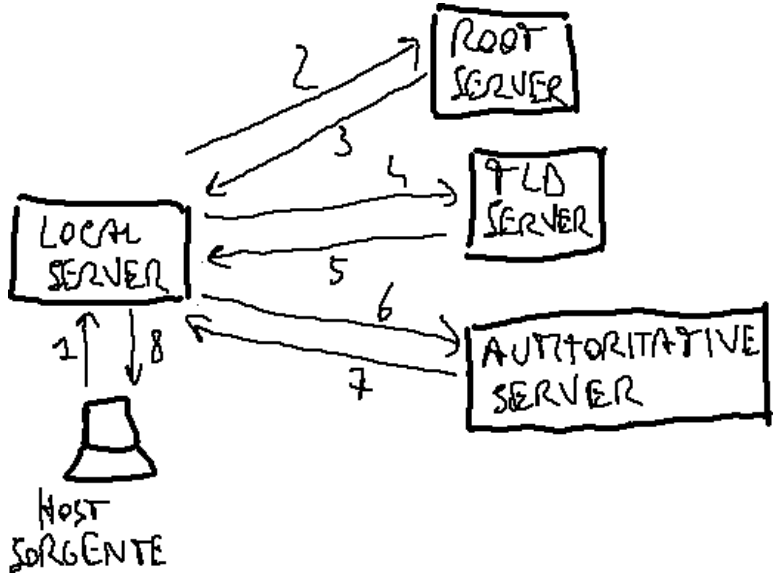
- Rapid growth and success.
  - 1977: 111 machines on Internet
  - 1981: 213
  - 1983: 562
  - 1986: 5000
  - 1989: 10,000
  - 1992: 1,000,000
  - 2001: 150 – 175 million
  - 2002: > 200 million
  - 2024: ~ 5.5 billion (>4B are phones/tablets)

What This Class is about

How applications that use networks work:

HTTP 404 Error Message

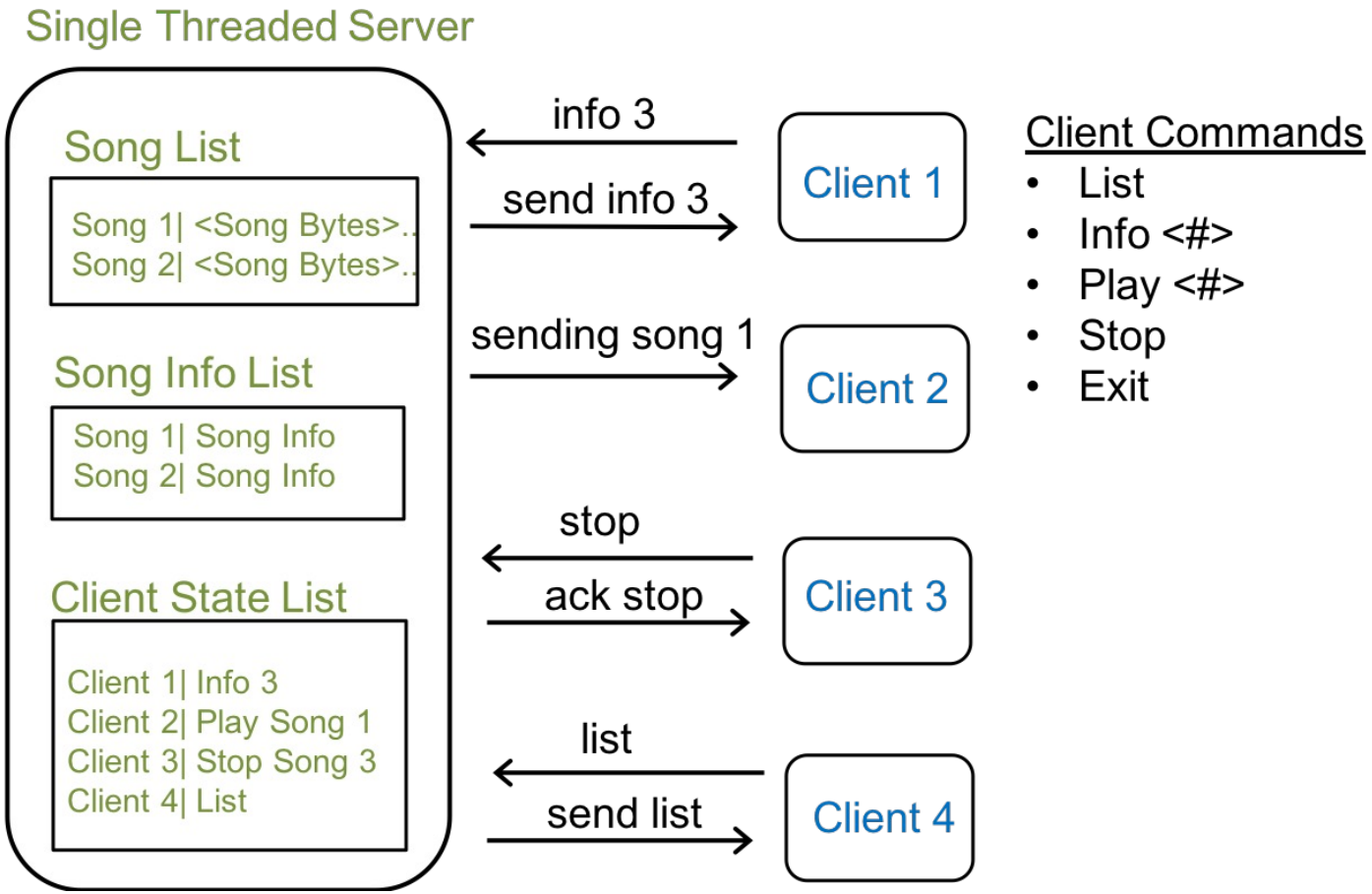
Email



DNS

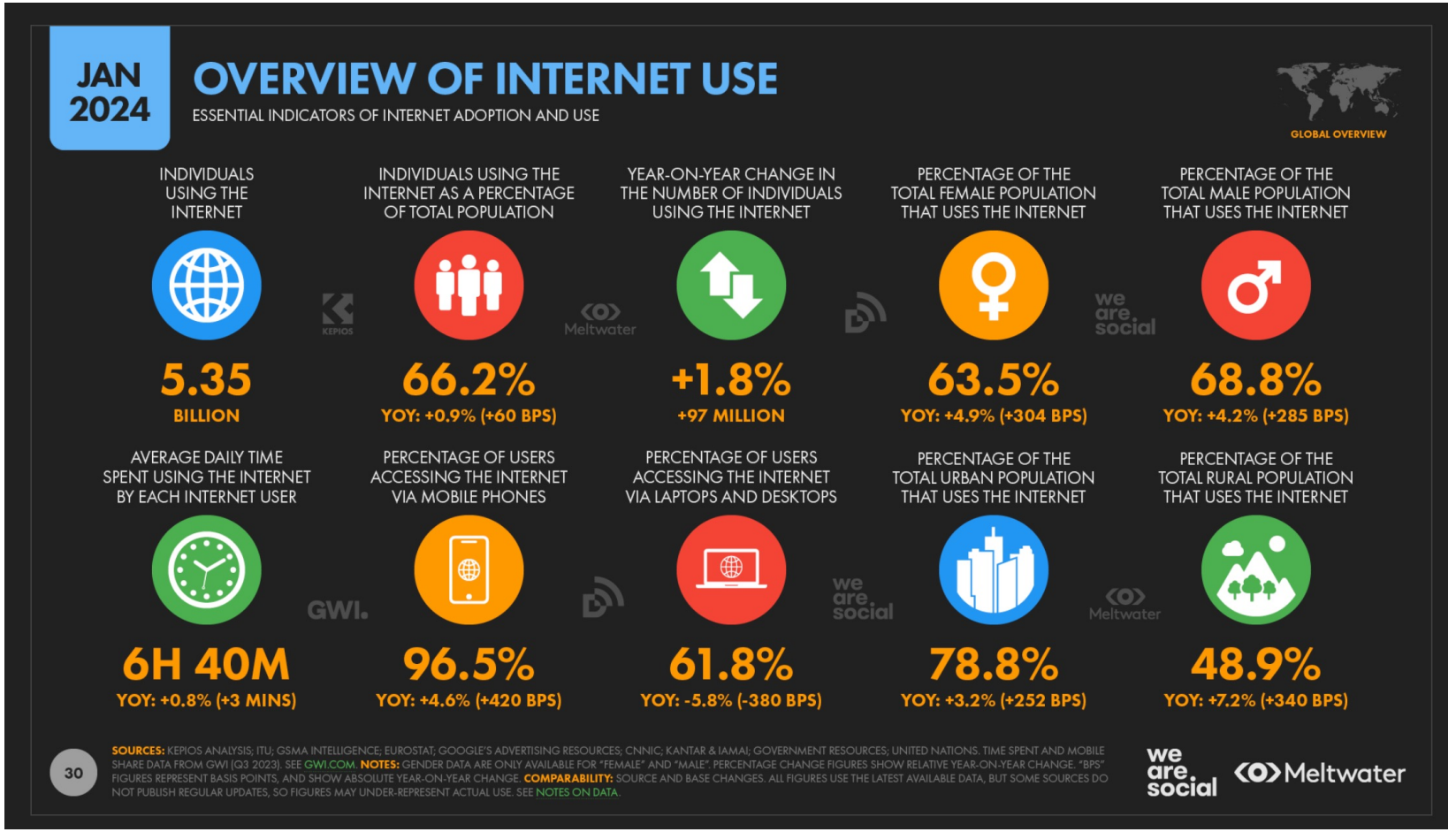
What This Class is about

How to write programs that communicate over networks



What This Class is about

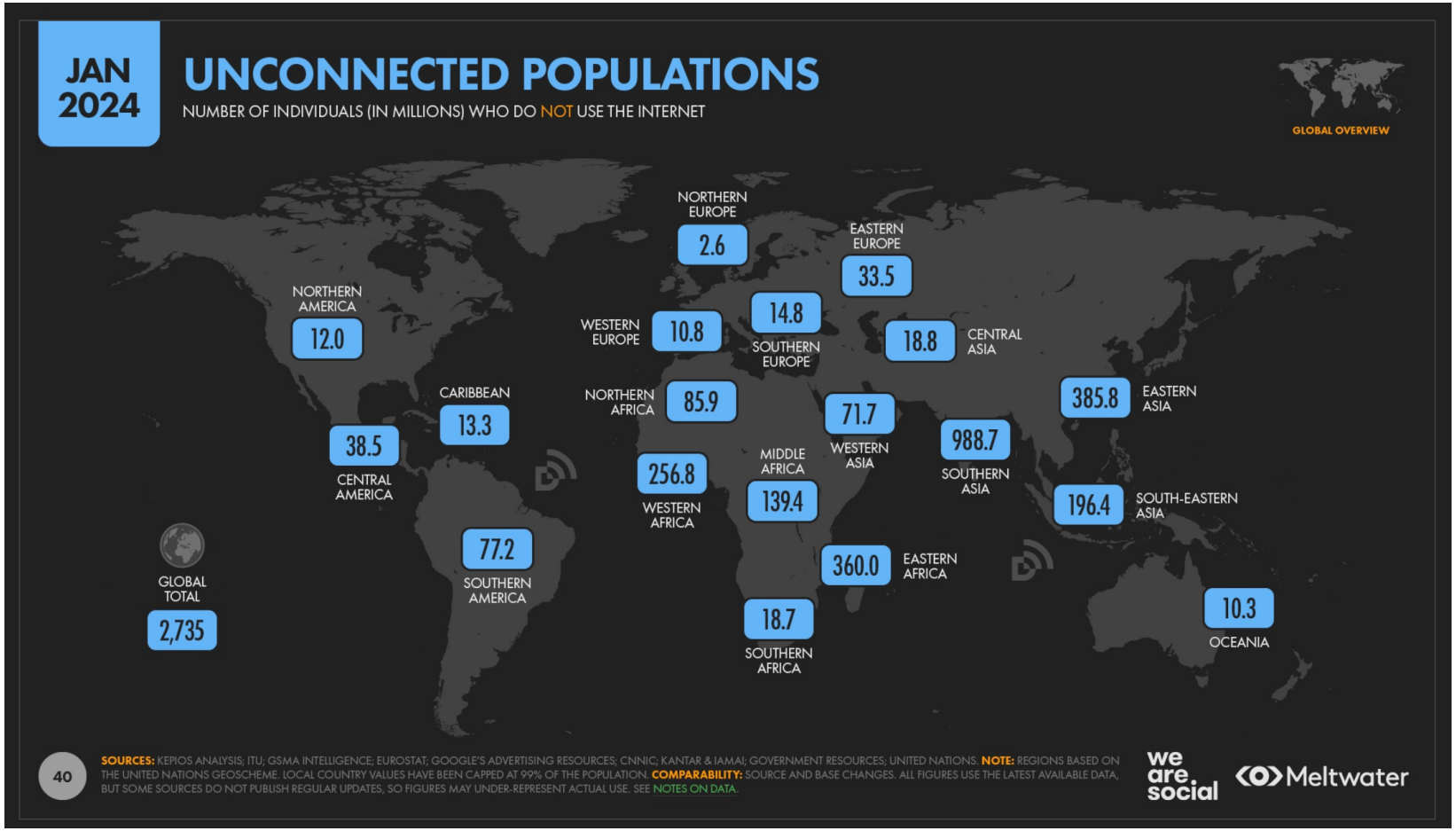
How different protocols, policies, and mechanisms interact.



https://datareportal.com/reports/digital-2024-global-overview-report

What This Class is about

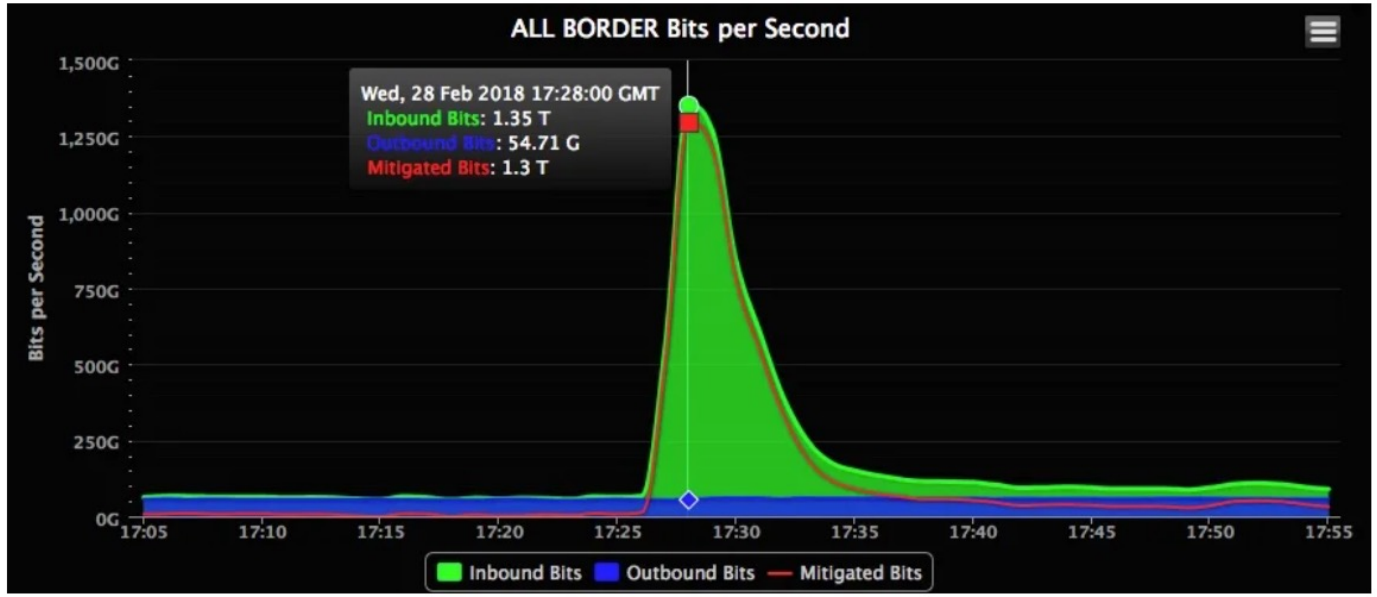
How different protocols, policies, and mechanisms interact.



https://datareportal.com/reports/digital-2024-global-overview-report

What This Class is about

How different protocols, policies, and mechanisms interact.



Real-time traffic from the DDoS attack. AKAMAI

Github, 2018: One of the largest DDoS Attacks

- *1.3 Tbps, packet sending rate: 126.9 million per second.*

https://www.cloudflare.com/learning/ddos/memcached-ddos-attack/

What This Class is about

How different protocols, policies, and mechanisms interact.

Enlarge / Netflix took an active role in fighting for net neutrality rules in 2014.

Image Courtesy: Ars Technica

Net Neutrality: ISPs vs Content Providers

https://arstechnica.com/tech-policy/2017/07/facebook-alphabet-amazon-and-netflix-called-to-testify-on-net-neutrality/

What This Class is about

How different protocols, policies, and mechanisms interact.

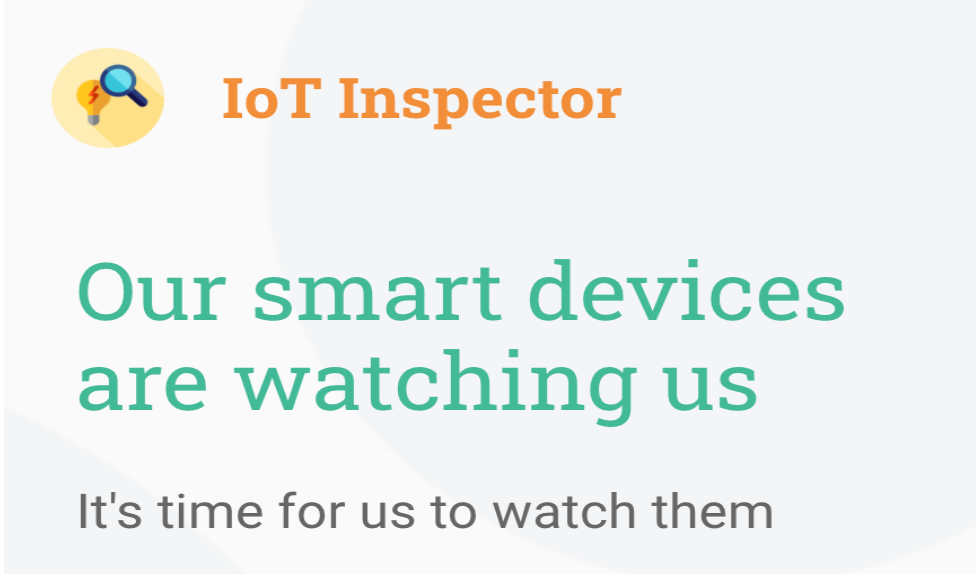


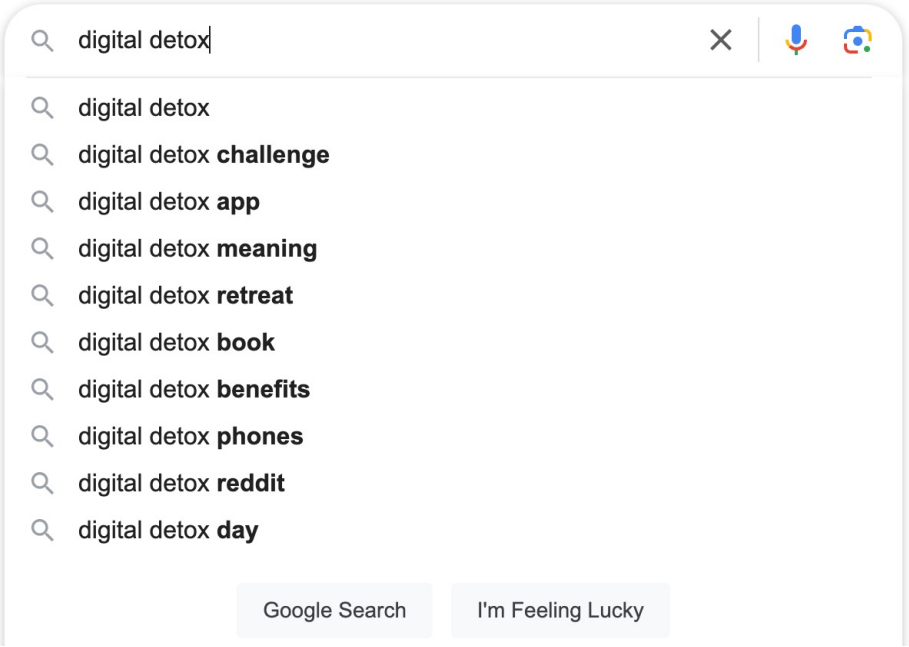
Image Courtesy: https://iotinspector.org/

Network Privacy

https://www.nytimes.com/2020/01/07/opinion/location-tracking-privacy.html

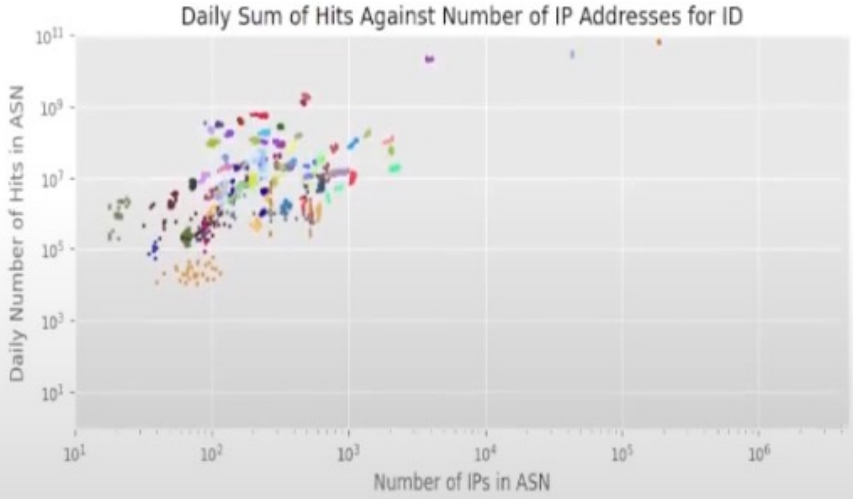
Why should you care?

When was the last time you went 24 hours without going online?



SPARK Lab

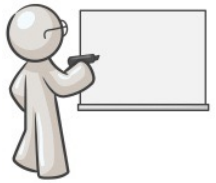
Differential Privacy in the Wild: Privatizing Queries over a Large Corpus of Real-World Network Traces



Content distribution networks (CDNs) are groups of internet servers placed all over the world, to allow content to be pushed closer to the end user. In doing this, it is necessary to geo-locate the IP address of the user's device. Network traces need to be released in order to improve network performance and better understand how users interact with the network. However, these large-scale datasets can be used to identify personal information about the users of the networks, which can compromise user privacy. Differential Privacy is our proposed solution for this problem. Differential privacy provides a quantifiable guarantee of privacy by inserting a controlled amount of noise into the data. The goal of differential privacy is to ensure privacy, for a controlled and reasonable tradeoff in accuracy.

In our research, we investigate the goodness-of-fit of queries, such as counts and range queries, for data analysis. Our first phase of research involves thorough data exploration, to determine our unit of privacy whether at the event-level or group-level. Furthermore, we compute a large range of queries to locate important global trends. This is toe ensure that our privacy parameters maintain a strong privacy guarantee without compromising trends important for understanding network-user interactions.

Classes: Interactive Classes with Peer Instruction

Readings	Reading Quiz	Instruction	Labs/Exam
			
First Exposure	Gauge understanding	Explore, add context, Provide feedback	Show Knowledge Mastery

- You do the “easy” part before class
- Class is reserved for interactive, customized experiences
- To learn, YOU must actively work with a problem and construct your own understanding of it

Clickers!

- Lets you vote on questions in real time.
- Like pub trivia, except the subject is always security ☺

Clicker Registration

https://forms.gle/LibhFpehK6AM5jGaA

If you don't register your clicker, I can't give you credit for quizzes / participation!

Participation scores count from week 2 (via paper hand-ins or clickers)

Locating your Clicker ID

iClicker with Hexadecimal ID

Hexadecimal number: numbers 0-9 and letters A – F

ID is also visible when you turn your clicker on.

Schedule

- Tentative Dates:
  - Midterm 1: Oct 21st. [SELECT TIMES BELOW]
  - Midterm 2: Dec 3rd, In Class
- Midterm Polls:
  - Select all times that work for you: https://forms.gle/DTeiz8vXxpL6a4qS7
  - Respond by the end of week-1!
  - Let me know if these dates are problematic this week!
- Final Project: Presentations during exam week
- Labs @ SCI 254
  - Labs are held on Wednesday: 1:15-2:45PM | 3:00-4:30PM
  - Prev. Lab due on Tuesdays via Github: https://github.swarthmore.edu

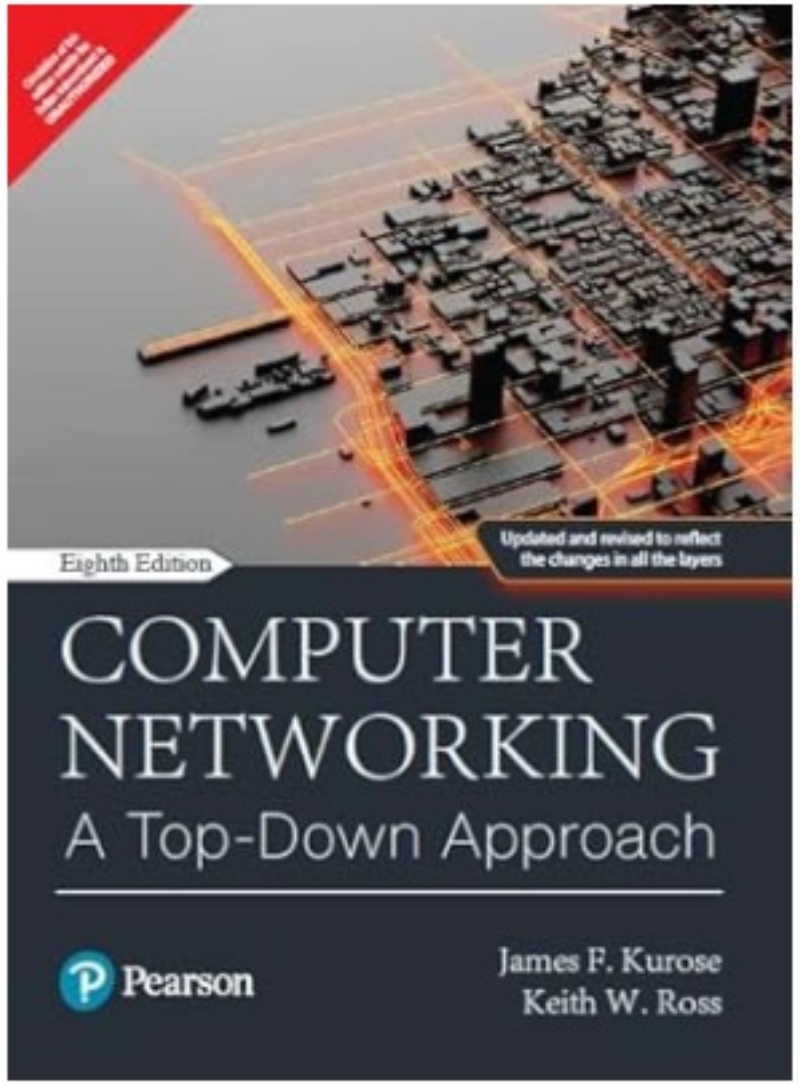
Resources: EdStem

- Edstem Q&A Forum: https://edstem.org/us/join/frrgh4
- All announcements will be on EdStem
- Use Edstem! (counts towards your grade)
  - asking questions (not asked previously)
  - answering questions (you've worked through)
  - when in doubt (e.g., posting code)– leave a private message
  - Response within a day

- Email ***doesn't scale***: course related questions/comments edstem/office hours

Resource: Readings

- Course readings posted on website
- Textbook on Moodle:
  - Computer Networking: A Top-Down Approach by Kurose and Ross. ISBN-13: 978-9356061316



Course Grade Distribution

- 5% Readings Quizzes (based on assigned readings/videos)
- 5% Class and Lab Attendance
- 10% Final Project
- 40% Midterm-1 (20%) and Midterm-2 (20%)
- 40% Lab assignments (3%, 5%, 8%, 8%, 8%, 5%, 3%)

Course Grade Distribution

- 5% Readings Quizzes (based on assigned readings/videos)
- 5% Class and Lab Attendance
- 10% Final Project
- 40% Midterm-1 (20%) and Midterm-2 (20%)
- 40% Lab assignments (3%, 5%, 8%, 8%, 8%, 5%, 3%)

I will drop your three lowest quizzes/no-shows.

How to succeed in an Upper Level Class

- Reading comprehension!
- Pre-Reqs: 31 & 35 and ACTUALLY applying material you learnt from those classes
  - remember valgrind and gdb? they'll be your best friends … again!
- Working through code, problem sets, and reading material like you would in the "real-world"
  - making sure you read and understand required readings/videos
  - try/brainstorm different approaches...
  - growth mindset

- It’s been a weird couple of years …and it’s okay to not be on top of everything
- Please reach out to:
  - Me (Vasanta)
  - Your Academic Advisors
  - Student Deans
  - Counseling & Psychological Services

by KC Green

Policies: Late Submissions

Genie (as William F. Buckley Jr)“ There are a few,..provisos, a, a couple of quid pro quos.” - in Aladdin

- Lab Lateness
  - 2 days of extra time for the semester (granularity of days)
  - Email AFTER you are done!
  - No Email: Grade whatever is present at the deadline.

Policies: Academic Dishonesty

- Collaboration
  - **You may discuss approaches, not solutions**
  - You must submit your own work
  - Exams may include questions on programming
- Cheating
  - We take this very seriously. It can have a negative impact on your course grade, your GPA and your record at Swarthmore and beyond.
  - **Don't do it!**

Policies: Academic Dishonesty

- Few examples of cheating on labs
  - Screen sharing with folks not in your lab partnership
  - “Let me read my code out to you, or share the exact API for a particular function”
  - Share in words the content in your code: “I first used strncpy to copy the string up to n bytes, and then appended a null character at the end”
  - I’m applying a “security mindset” to “think like an attacker” on course assessment infrastructure

Policies: Academic Dishonesty

- Examples of how not to cheat:
  - Behave as though you are a CS ninja
  - “What approaches did you try so far?”, “Looks like you have gotten more of the string than you need to, use man pages to look at other string functions”
  - Don’t know how to help your friend? Ask them to post to Edstem to the class or send a post privately to me.

Administrative Questions?

- All of this info is on the class website
- Feel free to ask Q&A on the Edstem discussion board
- I am running this course after the pandemic … so please anticipate
  - changes to the topics we cover
  - scope of lab assignments
  - possible issues with code/VM etc.
- Would be great to get (constructive) feedback!

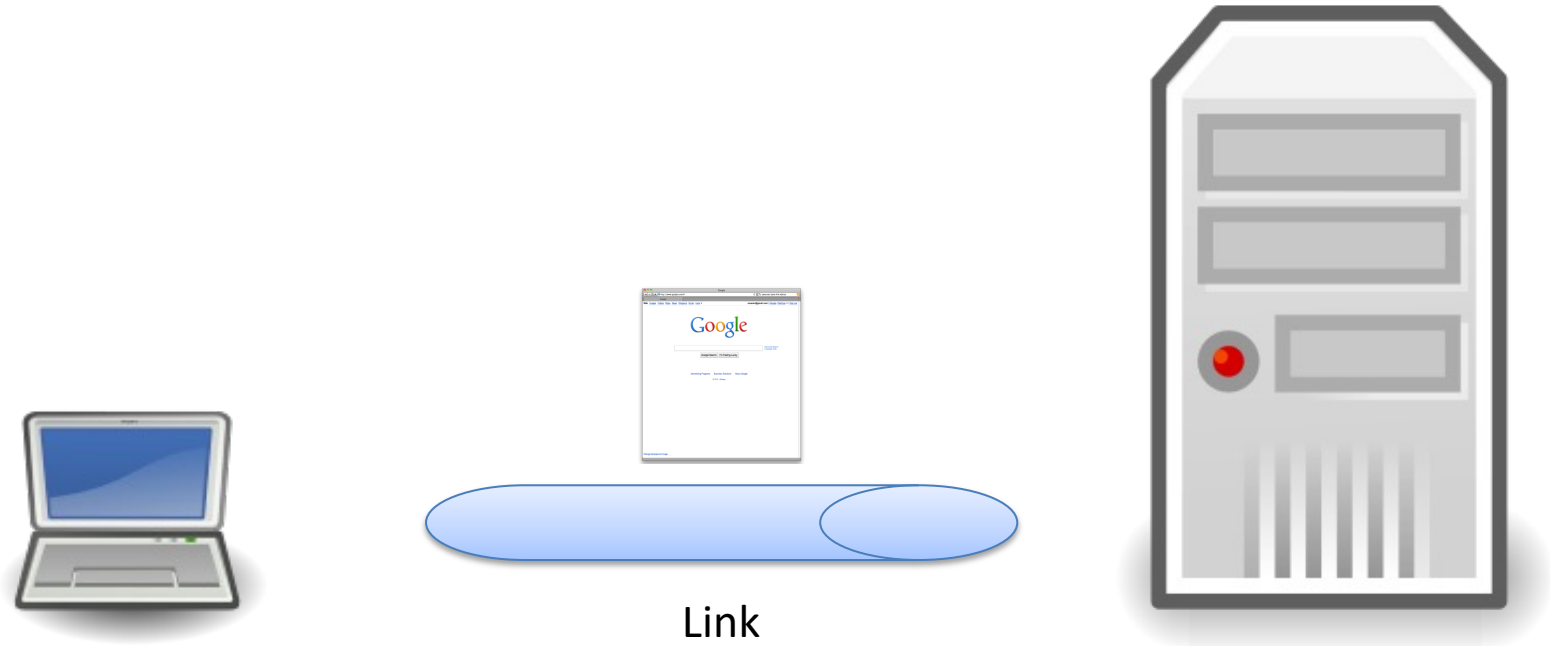
What is the goal of a network?

- Allow devices communicate with one another and coordinate their actions to work together.

- Piece of cake, right?

A "Simple" Task

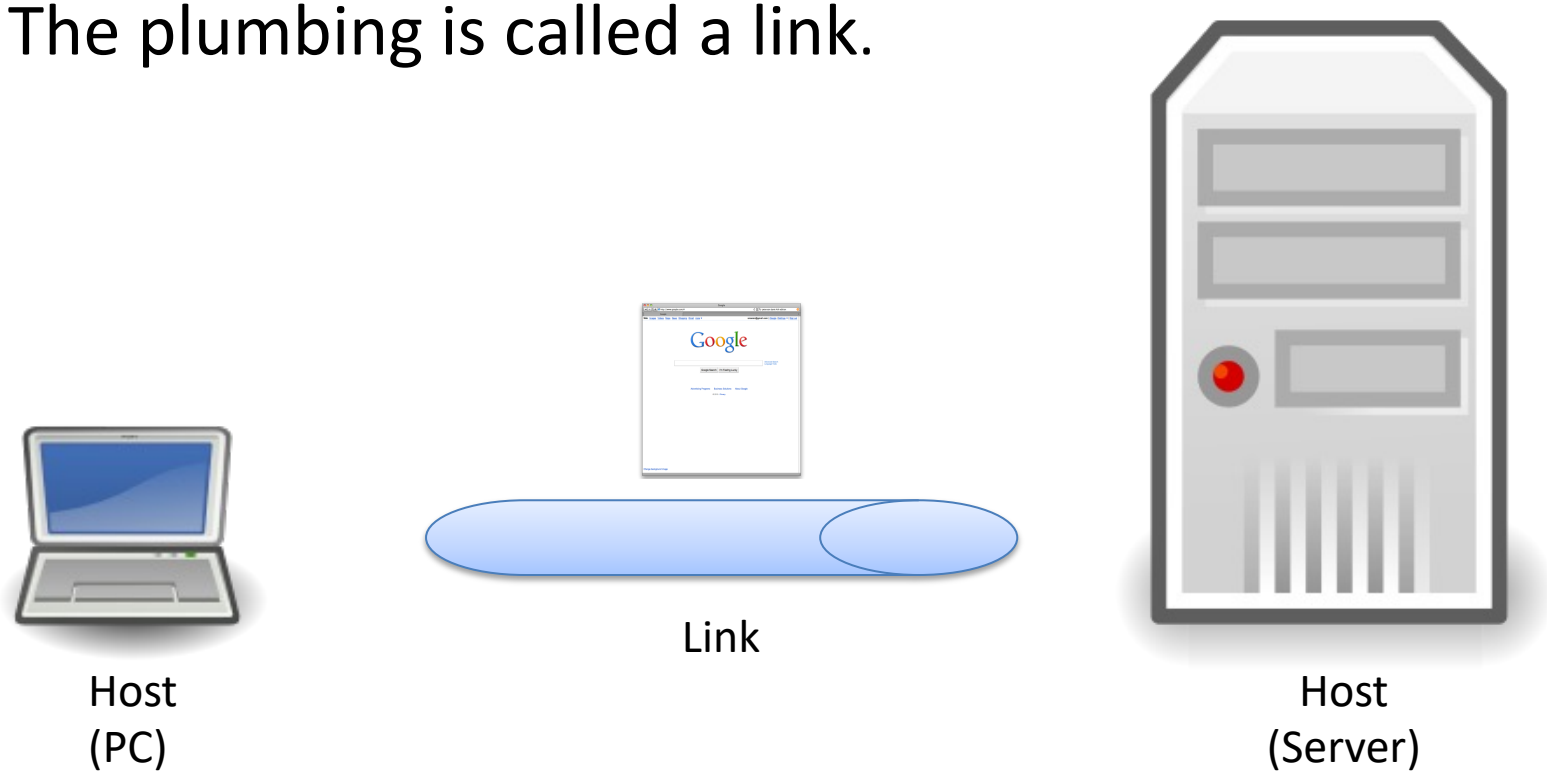
Send information from one computer to another



A "Simple" Task

Send information from one computer to another

- hosts: endpoints of a network
- The plumbing is called a link.

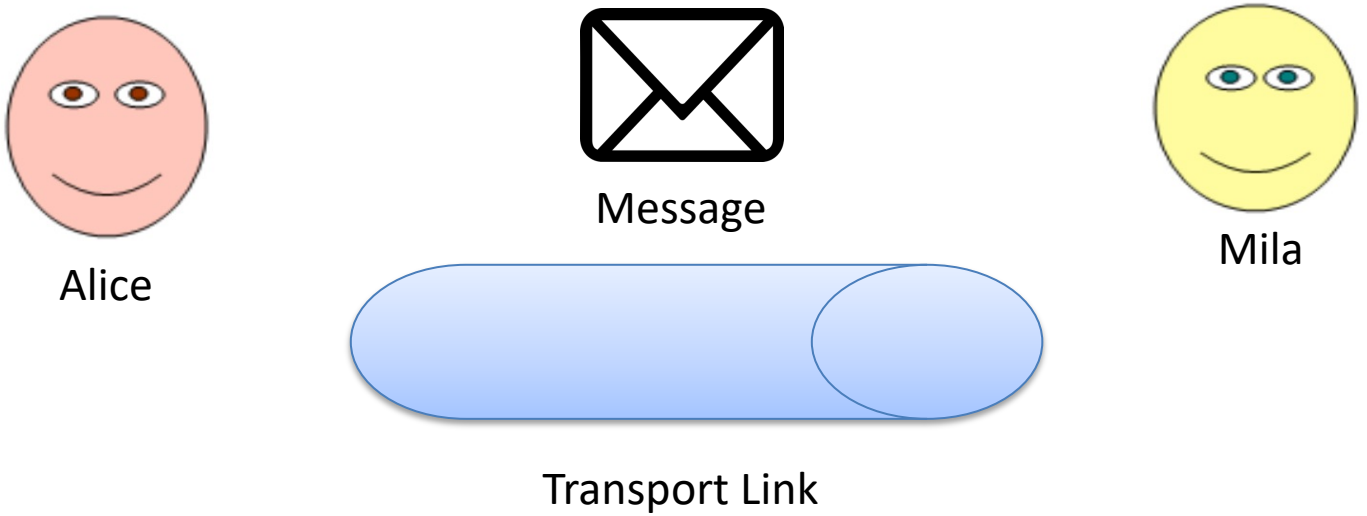


A "Simple" Task: Sending a message from host to destination

But first... let's try the postal system, something we are all (still!) familiar with and address a couple of key challenges..

A "Simple" analogous task: Post-it Note

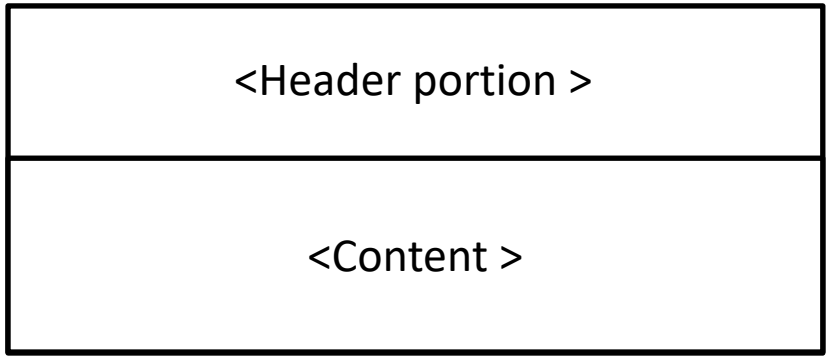
Alice and Mila are Swatties starting out their semester and are roommates. Alice wants to give Mila a reminder to get milk.



A "Simple" analogous task: Post-it Note

Alice and Mila are roommates, Alice wants to give Mila a reminder to get milk. Figure out some key tasks:

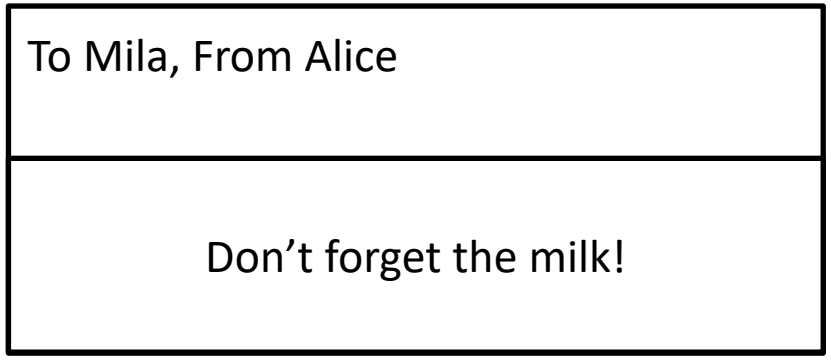
1. **Structure of the message:**
   - Construct the message that Alice posts to Mila.
2. **Organizing a drop-off point.**
   - Who chooses the drop-off point?
3. **Write a protocol to write a note /post—it to your housemate**



A "Simple" analogous task: Post-it Note

Alice and Mila are roommates, Alice wants to give Mila a reminder to get milk.

1. **Structure of the message: (Alice to Mila)**

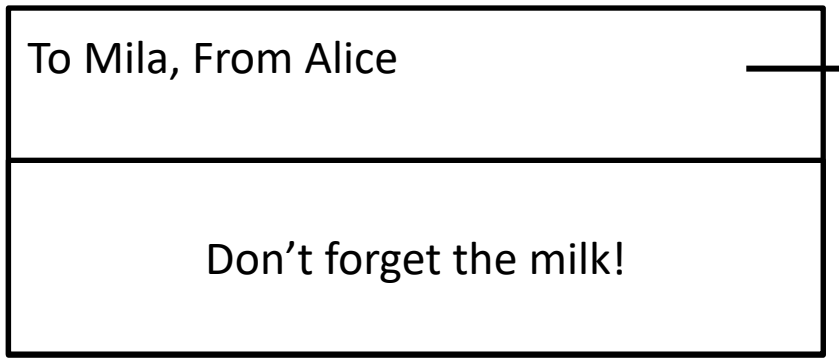


Irrespective of the source and destination, the format of the message stays the same.

A "Simple" analogous task: Post-it Note

Alice and Mila are roommates, Alice wants to give Mila a reminder to get milk.

1. **Structure of the message: (Alice to Mila)**



Header: contains sender/receiver information.

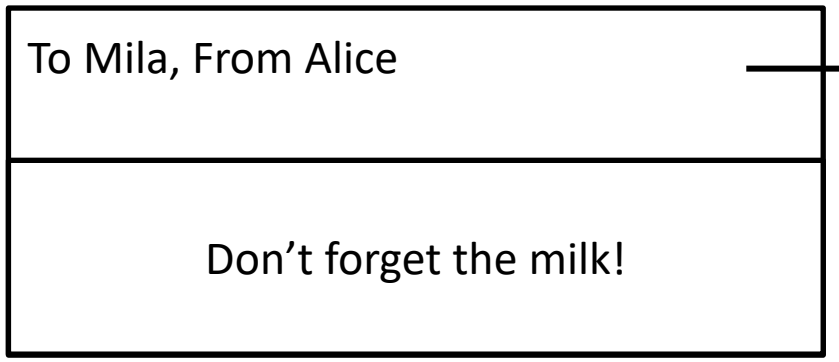
- metadata about the message
- what other metadata might you add?

Irrespective of the source and destination, the format of the message stays the same.

A "Simple" analogous task: Post-it Note

Alice and Mila are roommates, Alice wants to give Mila a reminder to get milk.

1. **Structure of the message: (Alice to Mila)**

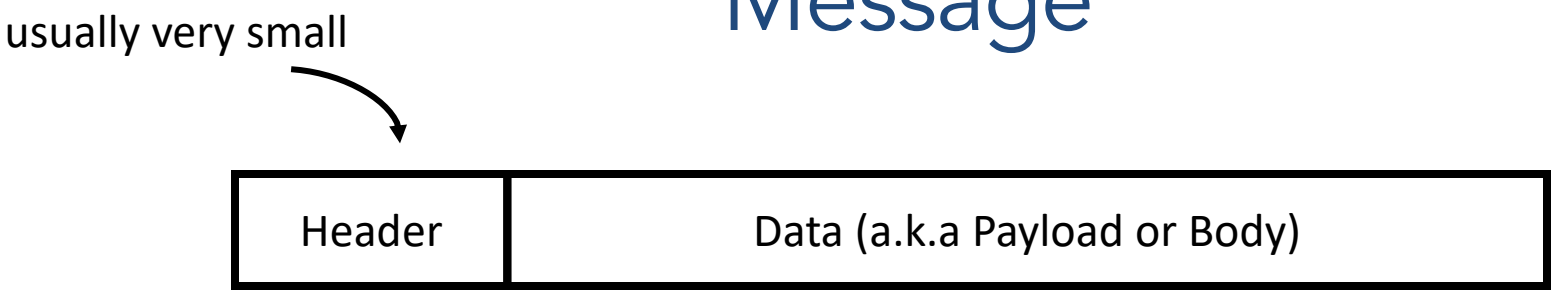


Header: contains sender/receiver information + additional state

- timestamp
- urgent! (priority)
- ordering of messages (1 of 10..)
- error control..

Irrespective of the source and destination, the format of the message stays the same.

Message



- Message: Header + Data
- Data: what sender wants the receiver to know
- Header: information to support protocol
  - Source and destination addresses
  - State of protocol operation
  - Error control (to check integrity of received data)

A "Simple" analogous task: Post-it Note

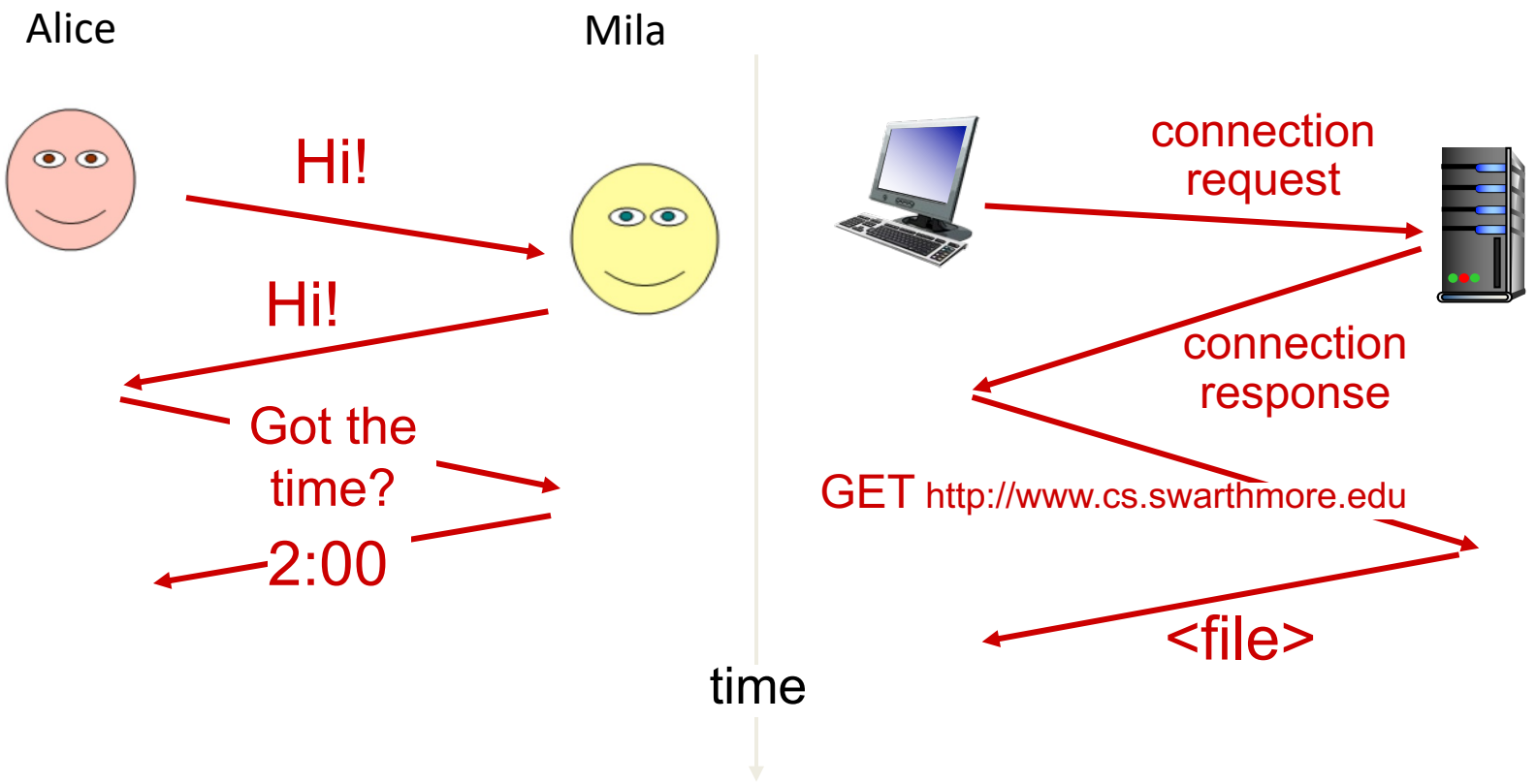
Alice and Mila are roommates, Alice wants to give Mila a reminder to get milk.

2. **<u>Organizing a drop-off point.</u>**
    - Who decides?
    - Generally by mutual consensus – previously agreed upon location.

<u>Everyone agrees to place messages on refrigerator to relay messages to housemates</u>

What is a protocol?

Protocol: message format + transfer procedure



Human Protocol

Network Protocols (defined in RFCs)

What is a protocol?

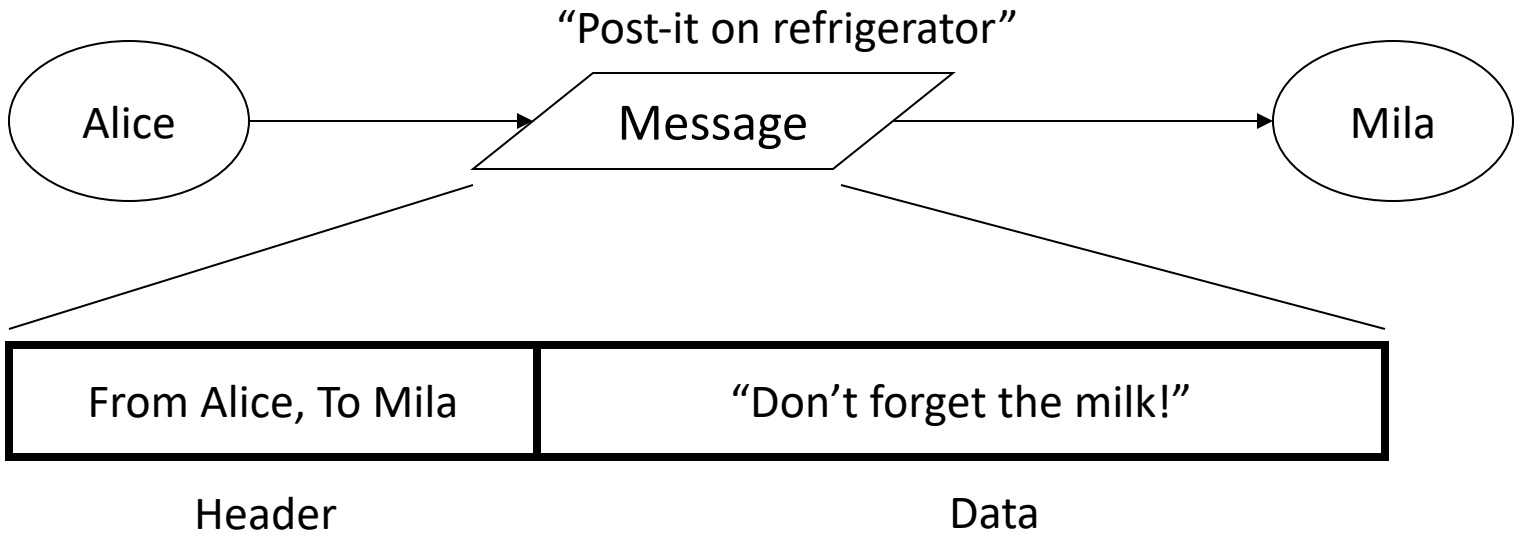
Goal: get message from sender to receiver

Protocol: message format + transfer procedure

- Expectations of operation
  - first you do x, then I do y, then you do z, ...

- Multiparty! so no central control
  - sender and receiver are separate processes

A "Simple" analogous task: Post-it Note



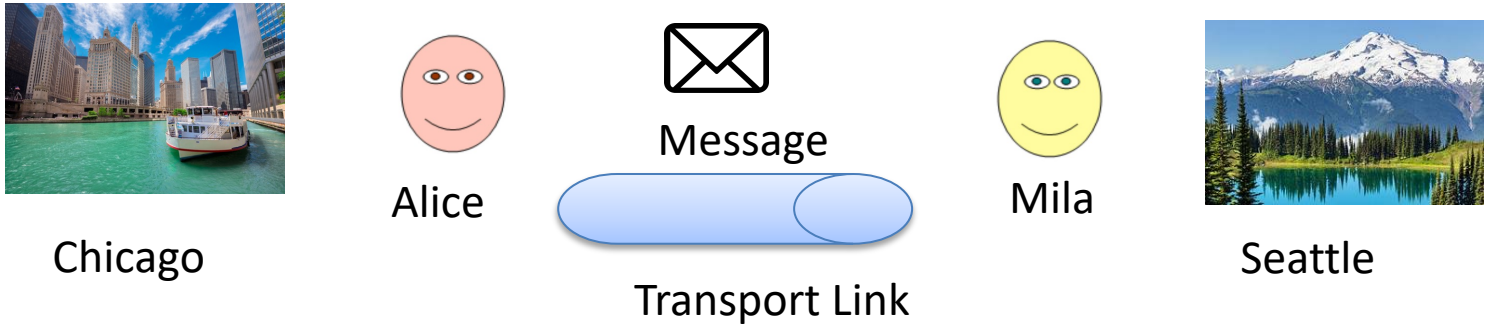
Write a protocol to write a note /post—it to your housemate

Protocol: message format + transfer procedure

- Message format: (from, to), message contents
- Transfer procedure: post on refrigerator

A "Simple" analogous task: Postal Mail

Alice moves to Chicago and Mila to Seattle for summer internships. Alice would like to send Mila a birthday card. Think of this as filling two different pieces of information (1. the birthday card, 2. the mailing envelope).



A "Simple" analogous task: Postal Mail

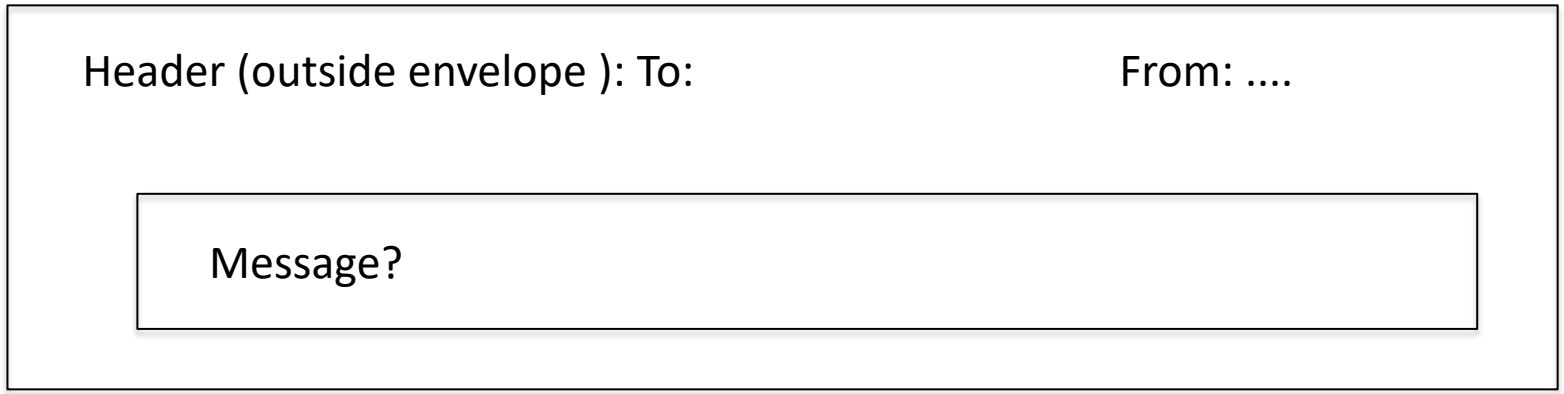
Alice would like to send Mila a birthday card.

1. **Construct the message and header. Have the header and message portions changed from the previous scenario?**
2. **List the message format and transfer procedure of the "mail sending protocol" that Alice uses.**
   - Who chooses the drop-off point?
   - Is this the only protocol in use?
3. **Message transportation and delivery**
   - Whose job is it to:
     - choose the carrier?
     - plan the route?
     - deliver the message?
     - ensure the message is not lost?

A "Simple" analogous task: Postal Mail

Alice would like to send Mila a birthday card.

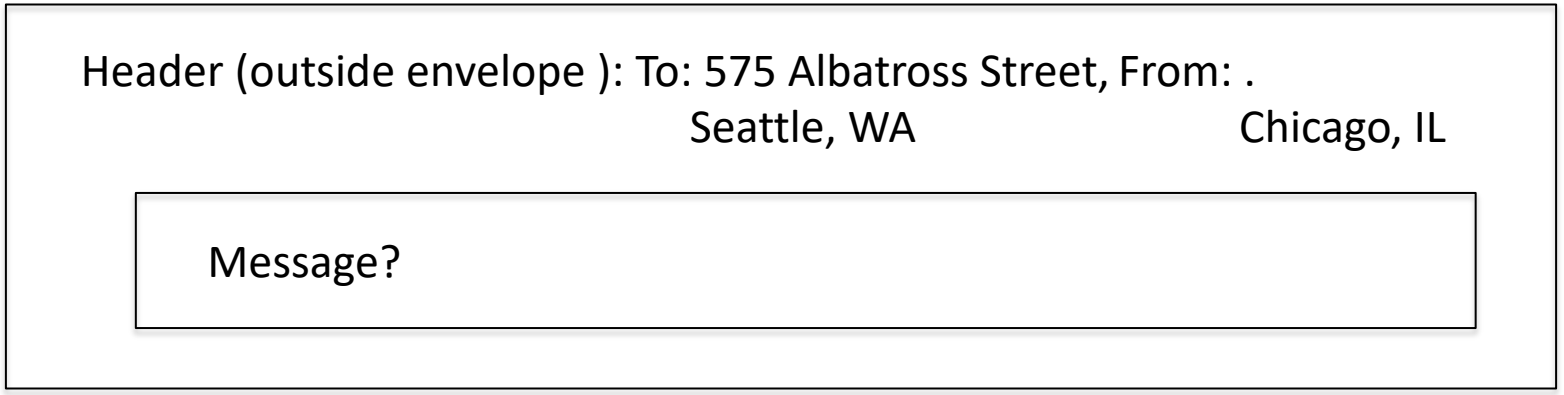
1. **<u>Construct the message and the header. Have the header and message portions changed from the previous scenario?</u>**



A "Simple" analogous task: Postal Mail

Alice would like to send Mila a birthday card.

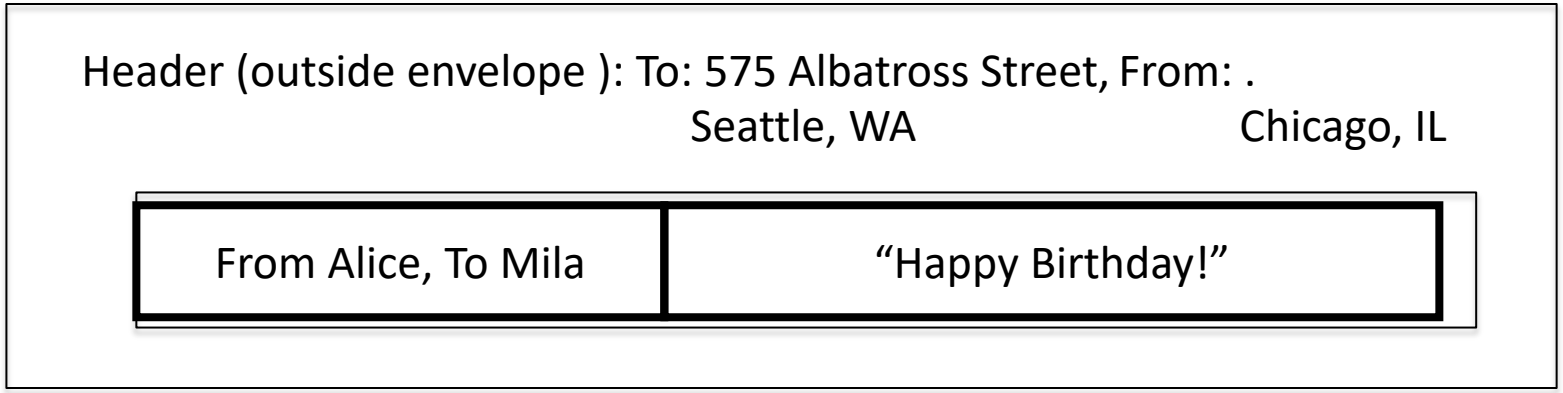
Header portion of the envelope



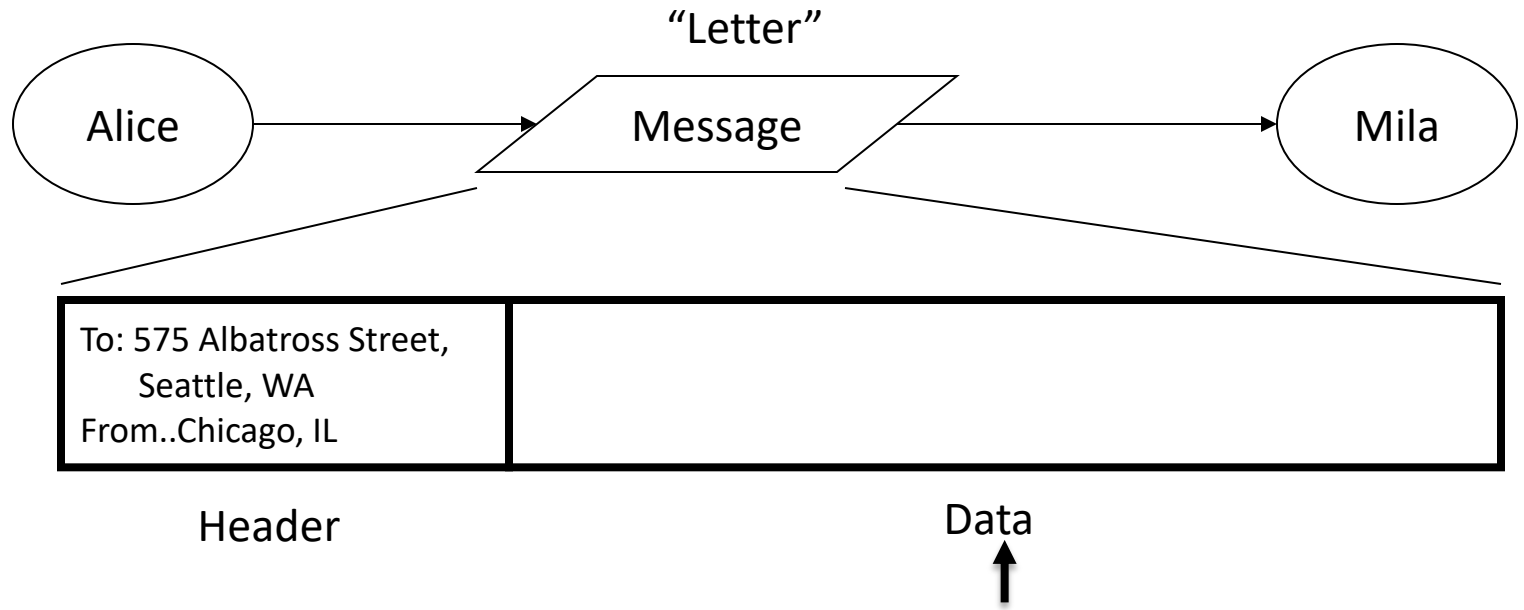
A "Simple" analogous task: Postal Mail

Alice would like to send Mila a birthday card.

Message portion of the envelope

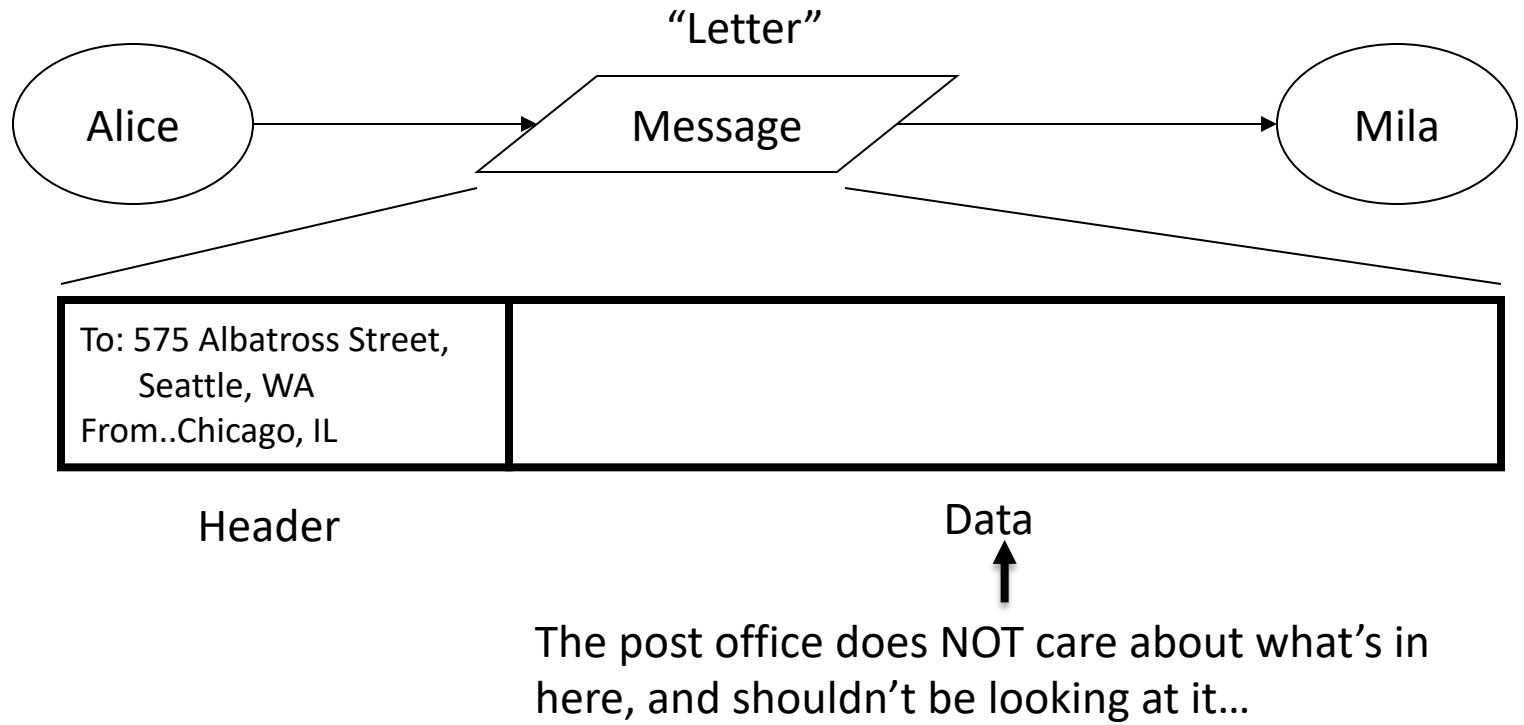


A "Simple" analogous task: Postal Mail



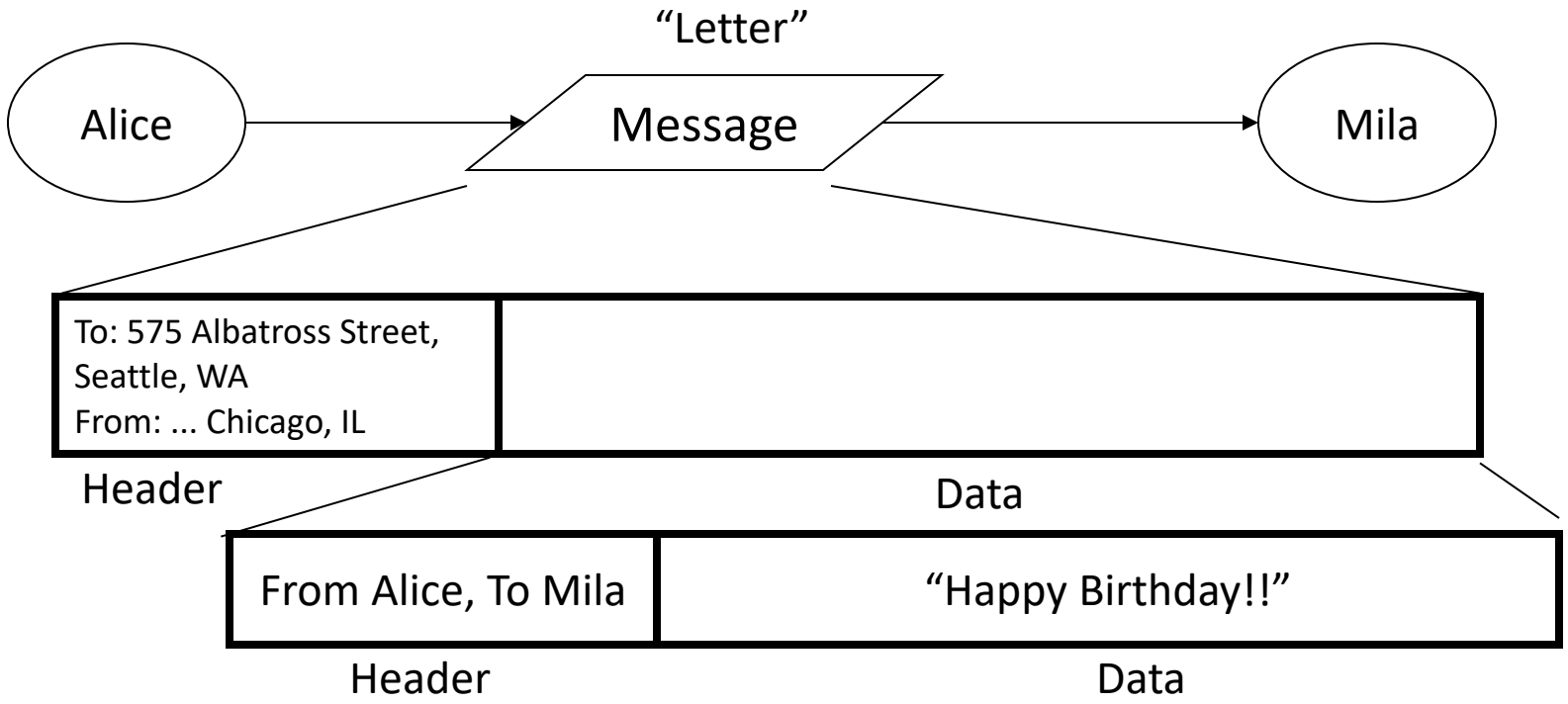
The post office does NOT care about what's in here, and shouldn't be looking at it…

A "Simple" analogous task: Postal Mail



- **Mail Sending Protocol**
  - Message format: (from, to), message contents
  - Transfer procedure: post mail in mailbox (agreed upon convention)

A "Simple" analogous task: Postal Mail: other protocols in use?



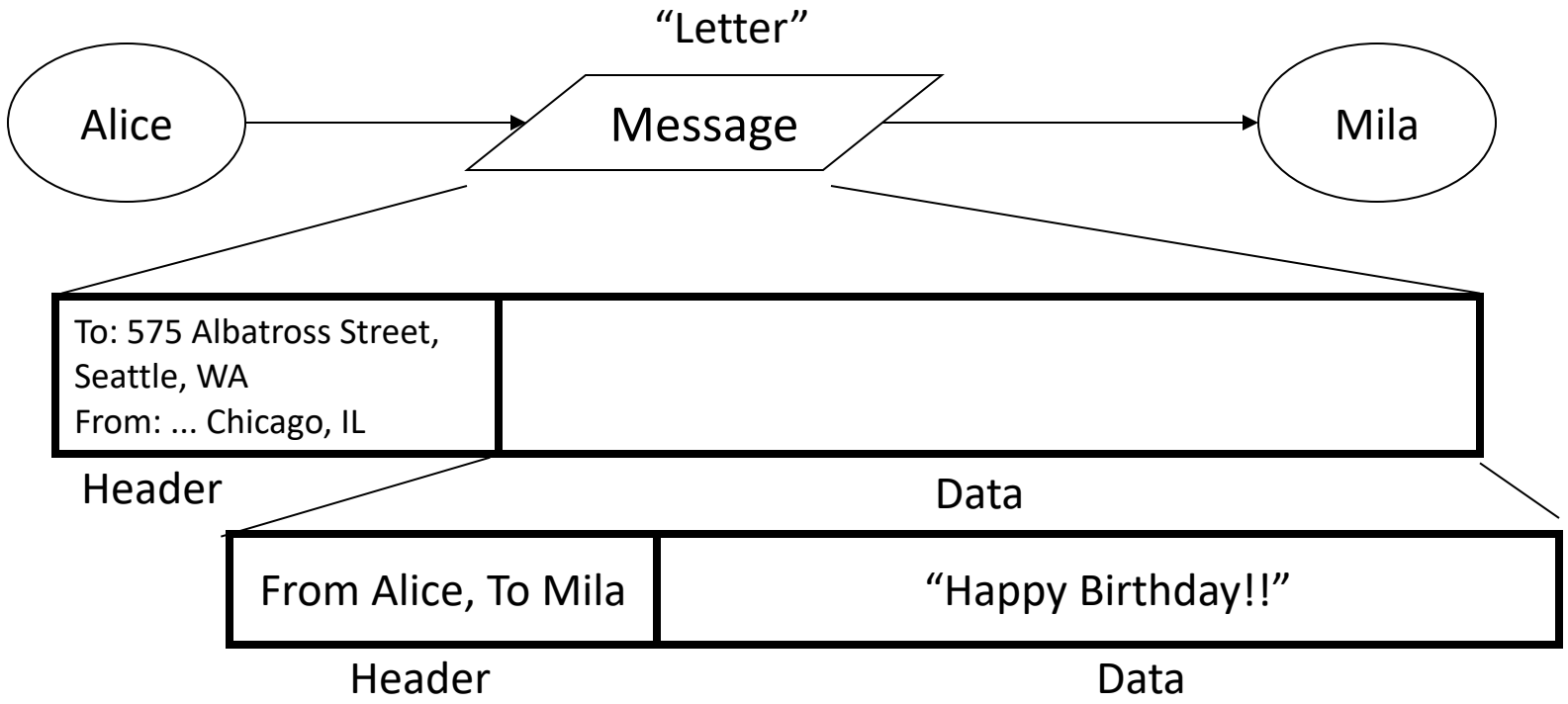
Mail Protocol

- Message format: (from, to), message contents
- Transfer procedure: post mail in mailbox (agreed upon convention)

Card Protocol (within the mail protocol!)

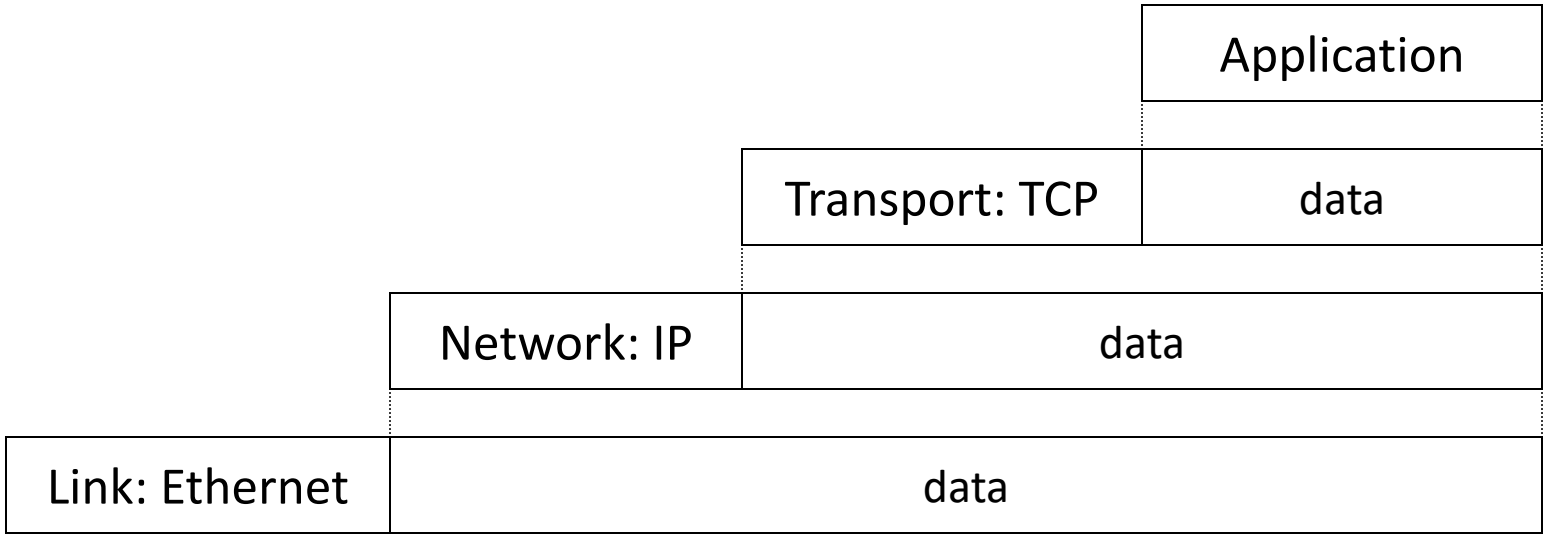
- Message format: (from, to), message contents

Message Encapsulation



- Card protocol: (message + header) treated as payload
- Put it in another protocol: append an additional header

Message Encapsulation



- Higher layer within lower layer
- Each layer has different concerns, provides abstract services to those above

A "Simple" analogous task: Postal Mail

- **Message transportation and delivery**
- Who's job is it to:
  1. provide the sender and receiver addresses?
  2. choose the carrier?
  3. plan the route?
  4. deliver the message?
  5. ensure the message is not lost?

A "Simple" analogous task: Postal Mail

- **Message transportation and delivery**
- Who's job is it to:

1. provide the sender and receiver addresses?
2. choose the carrier?

(1, 2): Alice decides as the "end host"

3. plan the route?
4. transport vehicles?

(3, 4): Postal Department decides as the service that provides message transfer

5. ensure the message is not lost? (reliability)

Reliability? Open question – stay tuned!

Layering: Separation of Functions

Letter: written/sent by Alice, received/read by Mila
Postal System: Mail delivery of letter in envelope

- Alice and Mila
  - Don’t have to know about delivery
  - However, aid postal system by providing addresses
- Postal System
  - Only has to know addresses and how to deliver
  - Doesn’t care about “data”: Alice, Mila, letter

Abstraction!

- Hides the complex details of a process
- Use abstract representation of relevant properties make reasoning simpler
- Ex: Alice and Mila knowledge of postal system:
  - Letters with addresses go in, come out other side

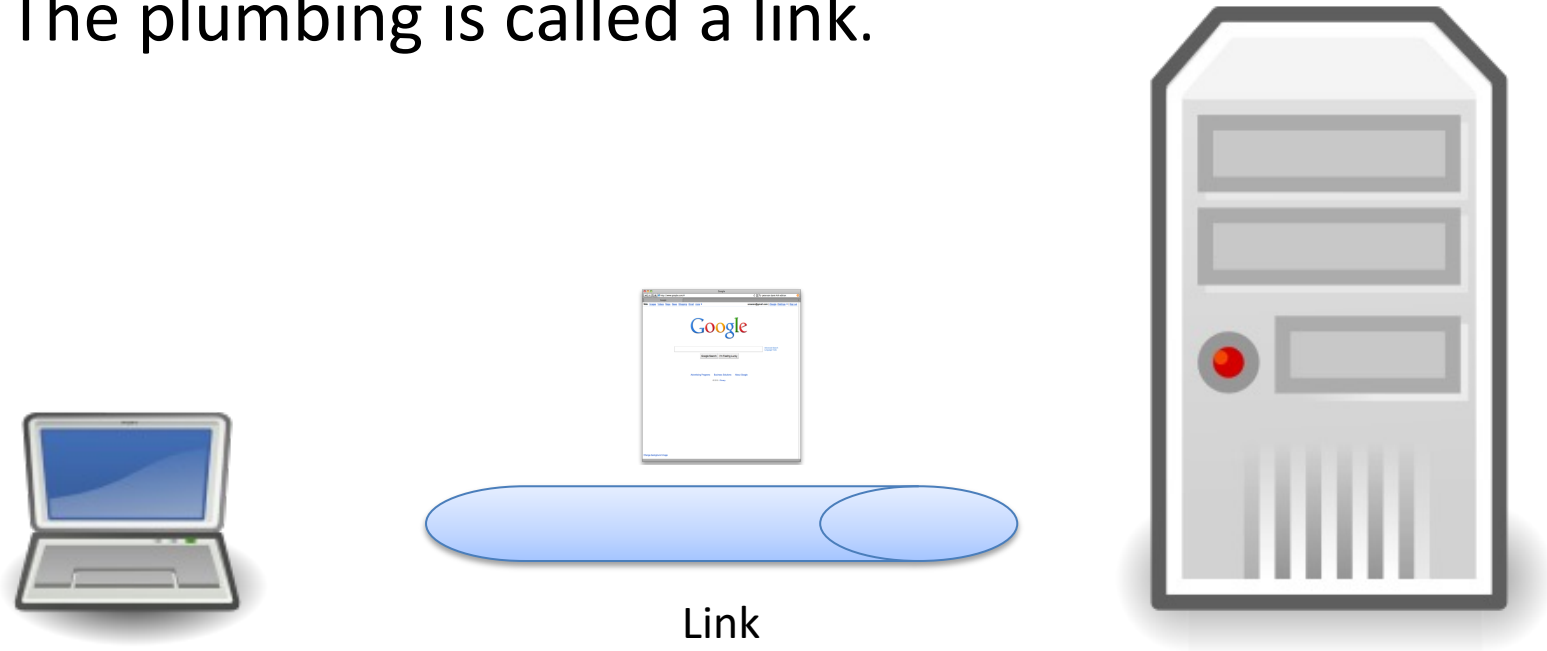
A “Simple” analogous task: Postal Mail

- Many more considerations..
  - Who decides the the sender and receiver addresses? Does someone maintain a mapping peoples’ names to addresses?
  - Can Mila always be guaranteed of this delivery date? What factors influence delivery ?
  - What if the mail gets lost – who’s responsibility is it? Alice, Mila or someone else?
  - What about security? privacy?

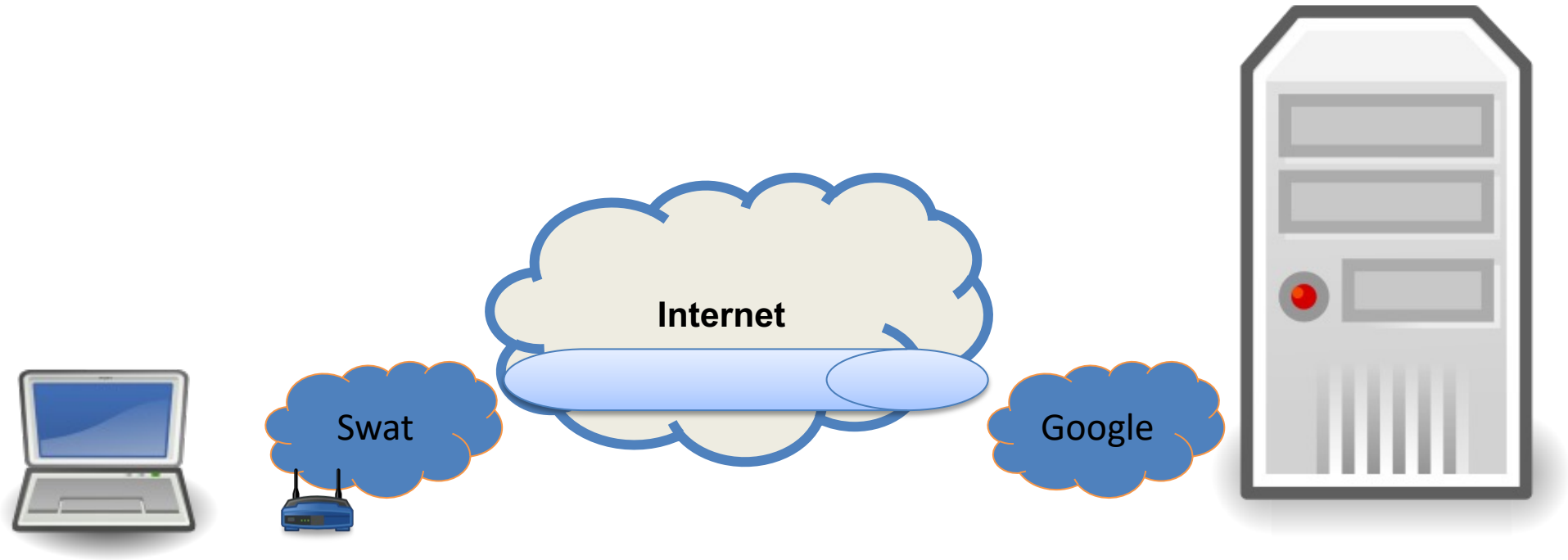
A "Simple" Task

Send information from one computer to another

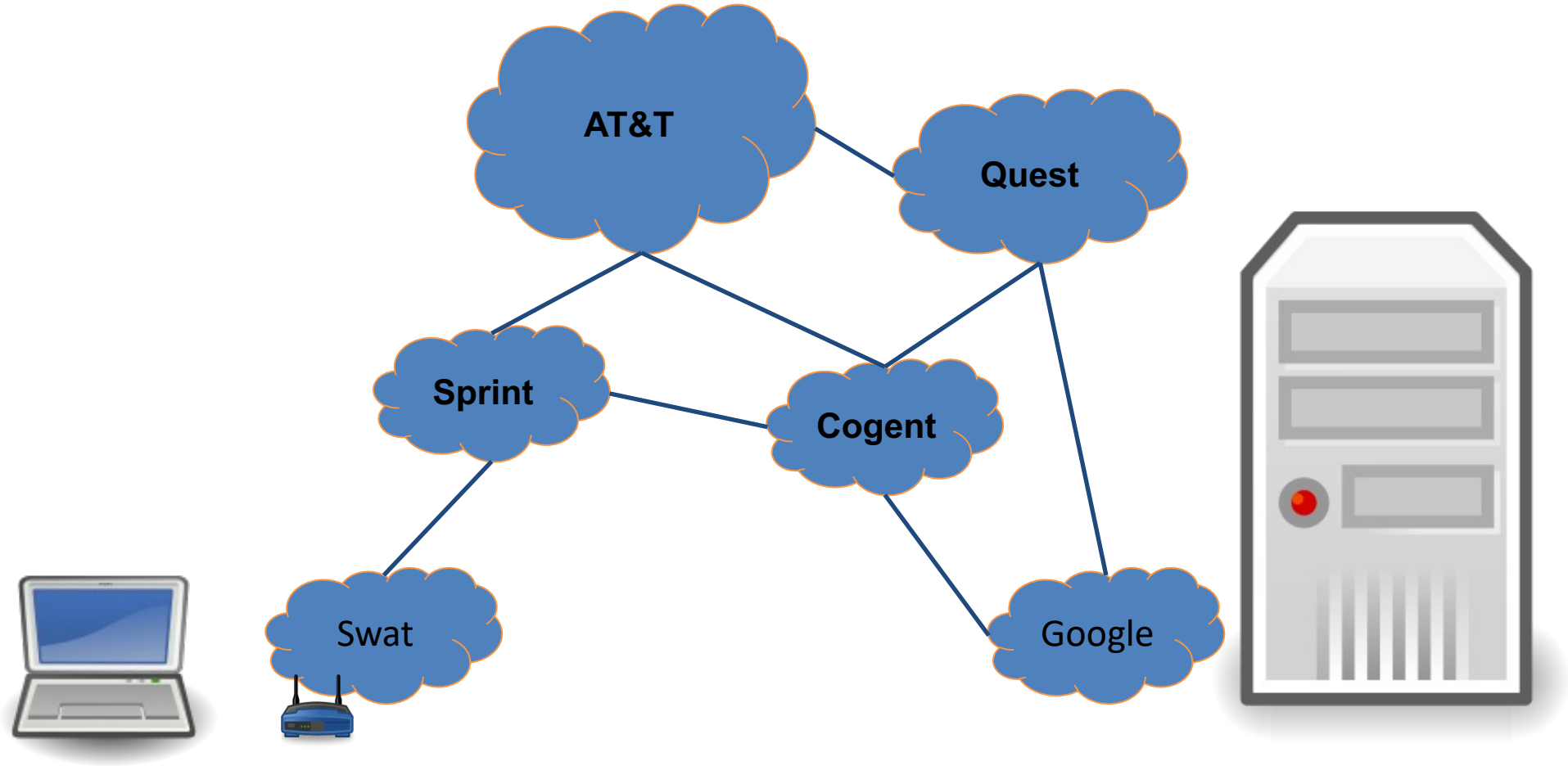
- hosts: endpoints of a network
- The plumbing is called a link.



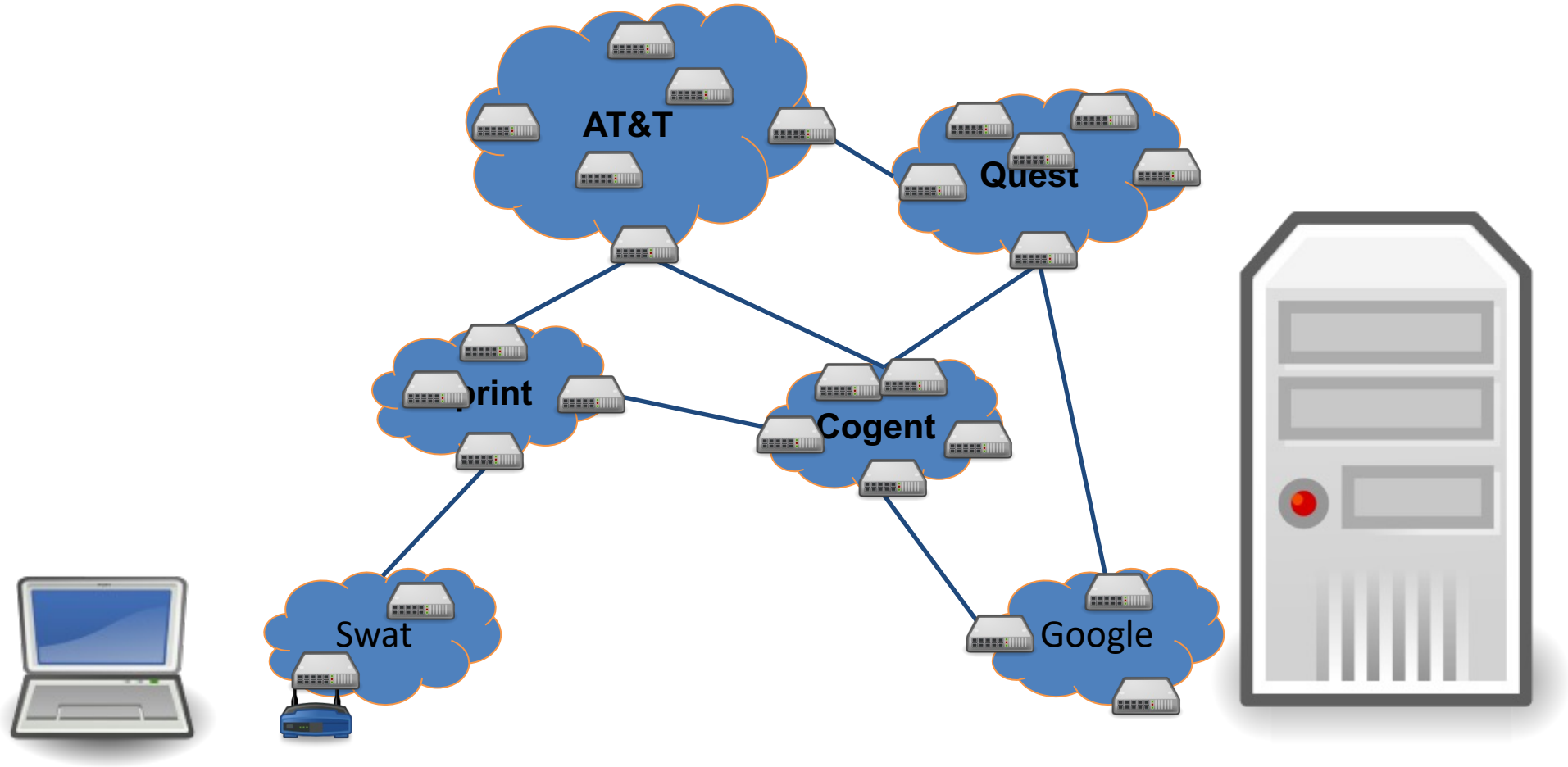
Not Really So Simple…



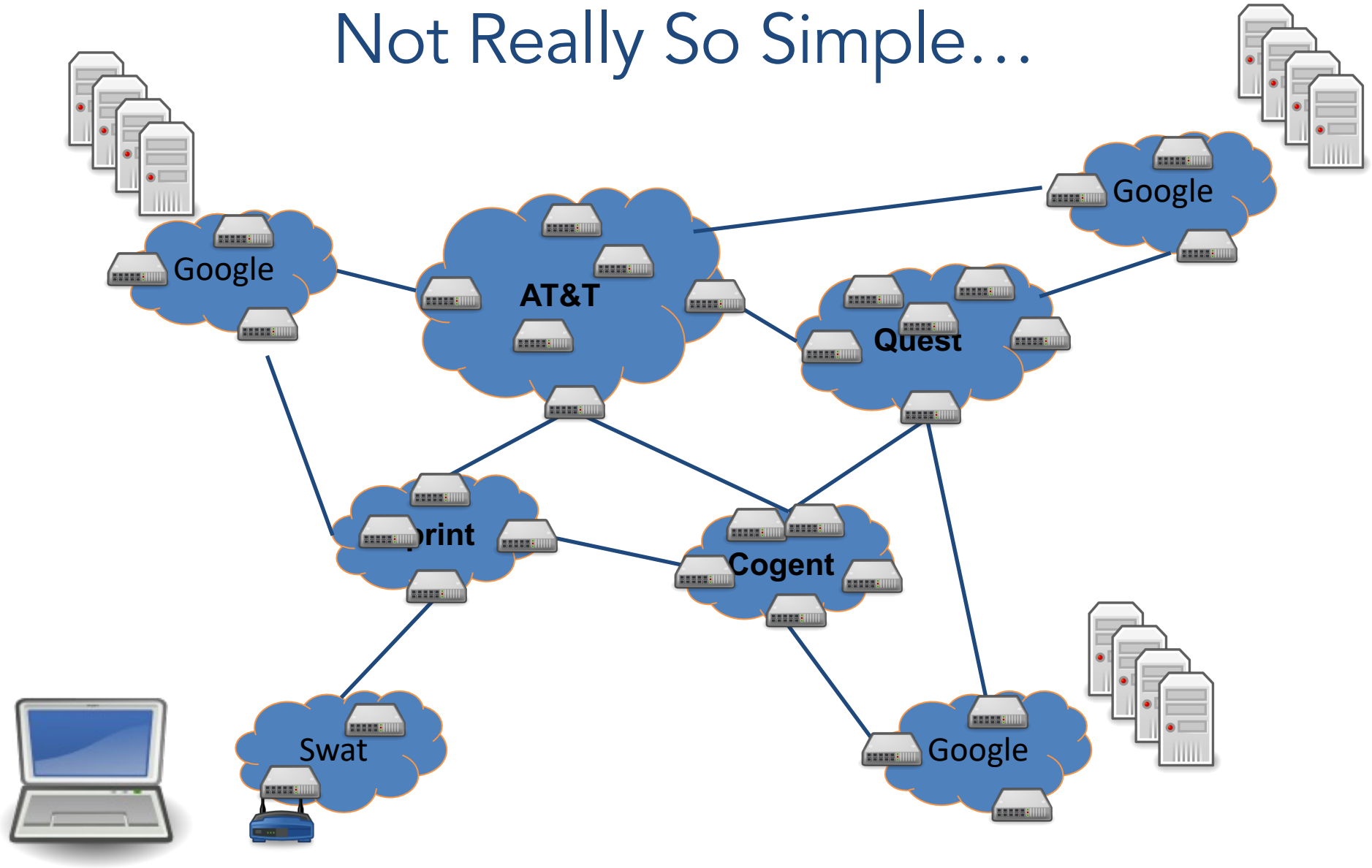
Not Really So Simple…



Not Really So Simple…



Not Really So Simple…



We only need…

- Manage complexity and scale up
  - Layering abstraction: divide responsibility
  - Protocols: standardize behavior for interoperability

We only need…

- Manage complexity and scale up
- Naming and addressing
  - Agreeing on how to describe/express a host, application, network, etc.

We only need…

- Manage complexity and scale up
- Naming and addressing
- Moving data to the destination
  - Routing: deciding how to get it there
  - Forwarding: copying data across devices/links

We only need…

- Manage complexity and scale up
- Naming and addressing
- Moving data to the destination
- Reliability and fault tolerance
  - How can we guarantee that the data arrives?
  - How do we handle link or device failures?

We only need…

- Manage complexity and scale up
- Naming and addressing
- Moving data to the destination
- Reliability and fault tolerance
- Resource allocation, Security, Privacy..

We only need…

- Manage complexity and scale up
- Naming and addressing
- Moving data to the destination
- Reliability and fault tolerance
- Resource allocation, Security, Privacy..

(Lots of others too.)

Next Class

- Layering & division of responsibilities
- OSI Model
- End-to-end argument
- HTTP! An Application Layer Protocol

TODO List

- Reading: Protocols
  - Sections 1.1, 1.5
- Sign up on Edstem!
- Please let me know:
  - Your preferred name/pronouns, if different than roster information
  - Academic accommodations