

Lecture 1: Course overview and Introduction to computer networks

CS356: Computer Networks, Fall 2025

Instructor: Venkat Arun

(Adapted from Daehyeok Kim's slides)

Course staff

Instructor: Venkat Arun

- Assistant professor in UT computer science
- Research areas: Networked systems and formal methods
- Website: <https://www.cs.utexas.edu/~venkatar/>

TA: Ritul Satish

- PhD student
- Research Areas: Getting LLMs to do useful things

Communication channels

Course webpage: <https://www.cs.utexas.edu/~venkatar/f26/>

- Syllabus, programming assignments, lecture slides, ...

Canvas: <https://utexas.instructure.com/courses/1450929>

- Assignment submission, grade book, and attendance

Ed discussion: <https://edstem.org/us/courses/102831/discussion>

- Announcements, Q&A, and discussion

Let me know if you don't have access to any of them

Office hours

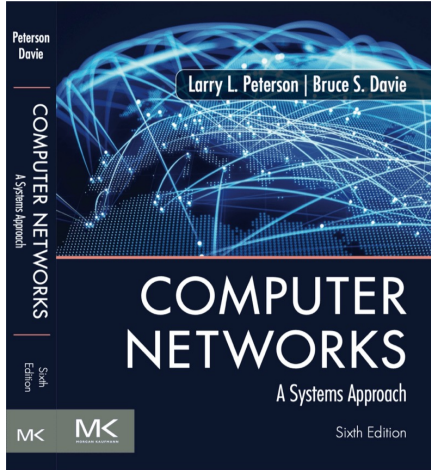
Instructor: Venkat Arun

- After class either outside class or in GDC 6.726

TA: Ritul Satish

- TBD

Textbook

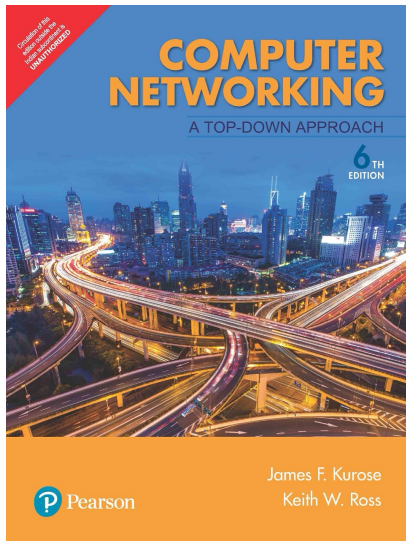


Book 1: Computer Networks: Systems Approach, 6th edition by Larry Peterson and Bruce Davie

Available online: <https://book.systemsapproach.org/>

Book 2: Computer networking: a top-down approach

Available through UT for a price. Opt out if you do not want it.



Lectures will not exactly follow the books

- Please attend lectures!

The internet is an impressive feat of engineering

- Can send data between almost any two locations in the world
- Through equipment operated by tens of thousands of different organizations
- Supporting almost any application
- The same basic concepts have survived over decades
- Yet it evolves rapidly, adapting to technological and societal changes and getting completely new abilities every 5-10 years
- Arguably, the largest human-built machine

The three parts of computer science

- Theory
 - Focuses on the fundamental limits of computation and knowledge. What problems can computers solve in principle, and how efficiently? How can information be communicated securely, even with untrusted parties? What does it mean to *prove* something, and how do proofs give us certainty?
- Artificial intelligence
 - Asks how we can make computers perceive, reason, and act in the real world. It teaches you to think in terms of uncertainty and approximation—reasoning in “fuzzy” ways—while grounding ideas in rigorous models and experimental evidence.
- Computer Systems (this course)
 - Examines how to build real-world computer systems that actually work. How do hardware and software interact? What are the tradeoffs in performance, reliability, and scalability when designing systems people rely on every day?

About this course

- To learn what computer networks are and how they work today
- To understand why they are designed the way they are and how they are likely to evolve
- General principles of systems design by studying one example: Computer Networks

Ed discussion

We will be using Ed discussion for all discussions

Please post class-related questions on Ed instead of emailing the instructor or TA

- You will get answers quicker, and it will benefit the whole class

If you have a more personal question, please send private message through Ed Discussion

Please use common sense when posting questions:

Grading

3 Quizzes (60%): in class quizzes

No Final exam

Programming assignments (40%): five programming assignments

Grades will be uploaded to Canvas

Grading bonuses

- Class participation (4 + 1 = 5% bonus)
 - 4%: $4 * (\text{fraction of classes attended as measured by instapoll})$
 - 1%: In-class participation
- Assignments will often have a bonus component

Grading philosophy

- Quizzes will be challenging, but you are not expected to get full points
- Assignments are non-trivial. We've changed things for the LLM era (details soon)
- Final letter grades will be easy

Class participation

Everyone is expected to attend lectures and participate

- Your attendance will be recorded on Canvas
- Lectures will be recorded on a “best effort” basis (i.e. if the recording equipment works) for later review.

Please ask questions!

- Will make class more fun for everyone
- Others may also benefit from your question

Quizzes

You are allowed to bring 3 pages of notes to the quizzes.

You can ask us questions: The course staff will be available during the quiz. You may ask us specific and factual questions during the quiz. If we decide that the question is regarding nitty-gritty details about network protocols, we will answer it (e.g. how a particular routing algorithm works or what port number is used for what protocol).

Programming assignments

Five programming assignments in groups of two. Group of 1 or 3 needs special permission from instructor.

Form a group as soon as possible

- Use Ed's "Finding Partners" category to find your partner

Submission: "Assignments" menu in Canvas

Late submission: 5 "free" late days for non-synchronous assignments after that you will be penalized 20% of the total points of the assignment per day. No late days for synchronous assignments

Assignment policy

~~All code and results you submit must be the original work of your group~~

~~Cheating and plagiarism will not be tolerated and will be dealt with in accordance with the University of Texas policies and procedures~~

~~If you fail to adhere to the policy, your case will be sent to the Dean of Students and placed on your file~~

You may post solutions online, in public GitHub repositories, consult LLMs or discuss amongst yourselves as much as you wish!

Use of LLMs (e.g. ChatGPT, Codex, Claude, ...)

They are an exciting new tool, and we encourage you to use them. Using them effectively will be a useful skill.

UT students have free access to Codex

A new kind of assignment

- We have developed a network simulator. It simulates network switches which are programmable. You write programs for those switches
- The programs operate the simulated network
- First three assignments guide you through setting up a functioning feature-rich network
- The next two are new. All your networks will co-exist together in a larger inter-network. There will be simulated customers who want to use the network. Your job is to attract business from them and compete with your peers to make the most money
- These assignments will happen in several rounds. You will submit code before each round. We will run the simulation. You can download logs from the simulation after it ends, learn from it and update your code for the next round. You can “buy” new (simulated) infrastructure using (simulated) money

Why are we ok with you using LLMs

- We will test your knowledge in the quizzes. In each quiz, there will be some questions about how *your* submission will behave. We will check it against how it actually behaves.
 - This means an LLM can write 100% of your code, but you still need to know what it does
- We are confident that LLMs do not know the answer to the final assignment because:
 - Nobody knows the answer
 - There is no answer

Why isn't there one perfect answer?

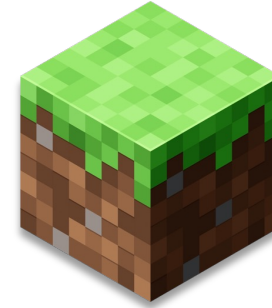
- The best answer depends on what other students are doing. E.g. if everyone follows the LLMs suggestions, one team can break away from the script and make a large profit
- The network will not be fully observable (just like in real life!). You have to implement any observability you want in order to understand how traffic behaves

Feedback

- This course is for *you*. You can help shape it. Please let me know what does and does not work for you during the course.
- Channels of communication:
 - Private (or non-private) messages on Ed Discussion
 - Email
 - Speak up after/during class

Let's get started!

What are the applications on the internet?



WWW, media streaming, real-time communication, gaming, ...

What's the **common requirement** of these applications?

- **Connectivity** – In principle, the network just needs to transport bytes. In practice, there are differences in what the apps need, and the internet is flexible enough to support that.

App example1: World Wide Web (WWW)

The most universal application accessed using a web browser

Interface: Uniform Resource Locator (URL)

- E.g., <https://cs.utexas.edu/~venkatar/f25>

[https](#) → Hypertext Transfer Protocol Secure (HTTPS) should be used to download the file

[utcs356.github.io](#) → machine name that serves the file (host)

[f25/index.html](#) → identifier of the particular page

It is called “the web” because this universal naming convention allows us to create a *web* of inter-connected pages.

These days web-pages resemble arbitrary application programs more than formatted text.

Retrieving a page via HTTP(S)

1. Retrieve an **address** for the host (cs.utexas.edu)
 1. E.g., 185.199.109.153
2. Set up a **connection** to the host
3. **Request object(s)** – many messages, possibly
4. **Receive object** – many messages, possibly

Key performance metric: **Page load time** (PLT)

- Time to load the entire page (including images, scripts, etc)
- Or time to display just the visible part of the page

App example 2: Media streaming

A widespread application class of the Internet

- E.g., Netflix is responsible for 15% of Internet traffic^[1]
- More broadly, video is responsible for ~70% of the bytes

Unlike web page loading, it demands a **continuous data flow**

Users want good **“quality of experience”** (QoE)

- Delivery start can be delayed but not too much
- Low rebuffering events, high bandwidth/quality

App example 3: Real-time communications

Popular application types that have grown rapidly in recent years

- E.g., Zoom, Teams, Skype, ...

Tight timing constraints

- When you speak, you want to be heard at the other end near simultaneously

It must provide consistent views across multiple participants

App example 4: Next generation real-time communication

Augmented/Virtual reality applications, networked robots

Much shorter timing constraints ($\sim 50\text{ms}$ instead of $200\text{-}300\text{ms}$)

Sensitive to violations at the tail

- Even if 99% of frames are displayed on time, user experiences a disruption once every 3 seconds at 30 frames/second

App example 5: Massively distributed computation in supercomputers/datacenters

- Massive scientific computations
- Web search! We take it for granted
- Massive neural networks (Large Language Models, computer vision...)

All these applications need a very large number of computers to coordinate very closely with each other

Particularly important since individual processors have stopped getting faster

Network design requirements

Network operators

- Scalable connectivity and cost-efficient resource sharing
- Manageability (for configuration, troubleshooting, accounting, ...)

World at large

- Flexible connectivity that can be extended by anyone and adapt to new technology

Application developers

- Support for common services via well-defined abstractions

Requirement 1: Scalable connectivity

A network must provide connectivity among a set of computers

- From small, private networks with limited connections to large-scale networks (e.g., Internet)

“Scalable” connectivity

- The network must provide connectivity that support an arbitrarily large number of computers
- We call such connectivity “scalable connectivity”

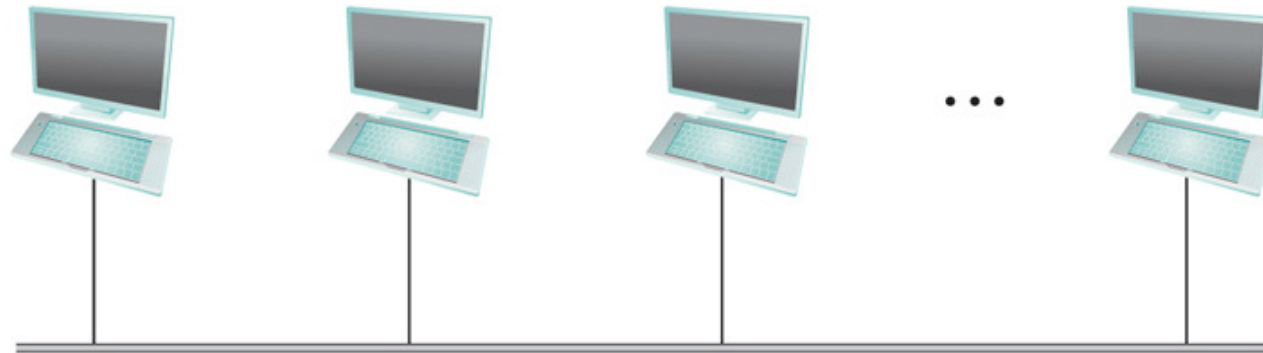
Two Key Principles

- Switched Networks
- Packet switching

Direct links



Point-to-point

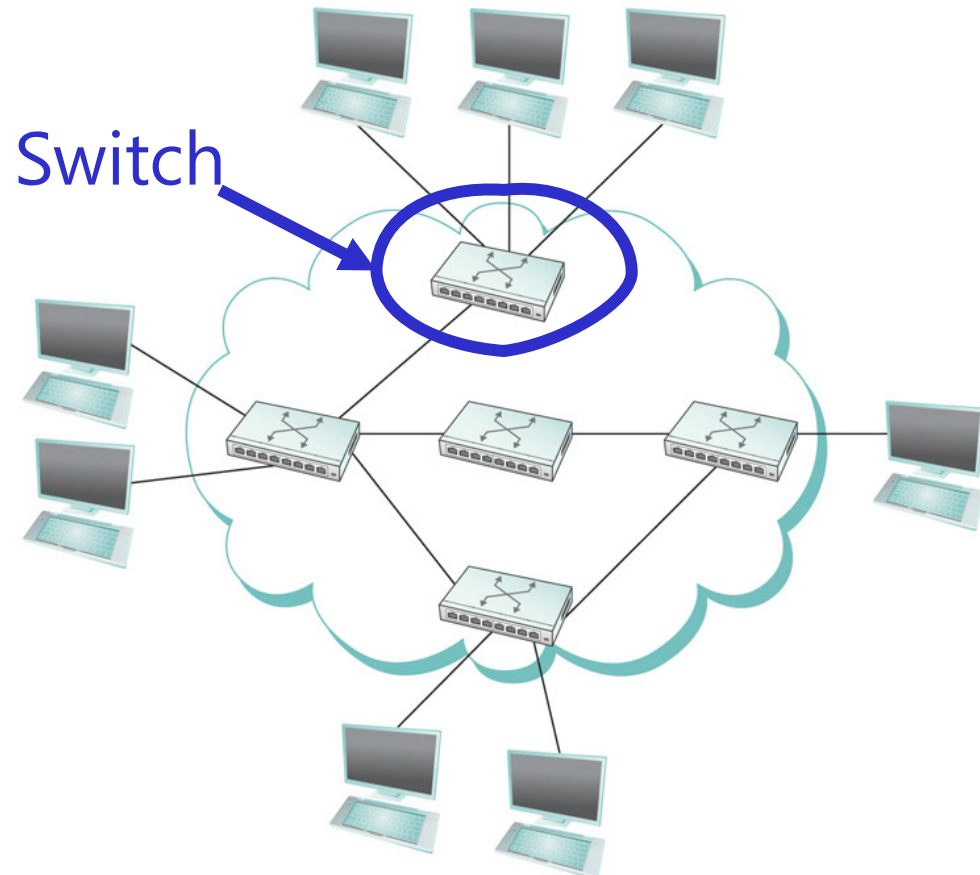


Multiple-access
(e.g., Wi-Fi or cellular networks
where everyone in the vicinity can
hear every transmission, and two
devices cannot "talk" at once)

What limits the size of networks only with direct links?

- Number of nodes they can connect
- Geographical distance they can cover (e.g., WiFi coverage)

Switched networks



Allowing a collection of computers can be “indirectly” connected. Switches have multiple ports (e.g. wires) and dynamically decides which wire connects to which other wire at each time instant.

Circuit-switched

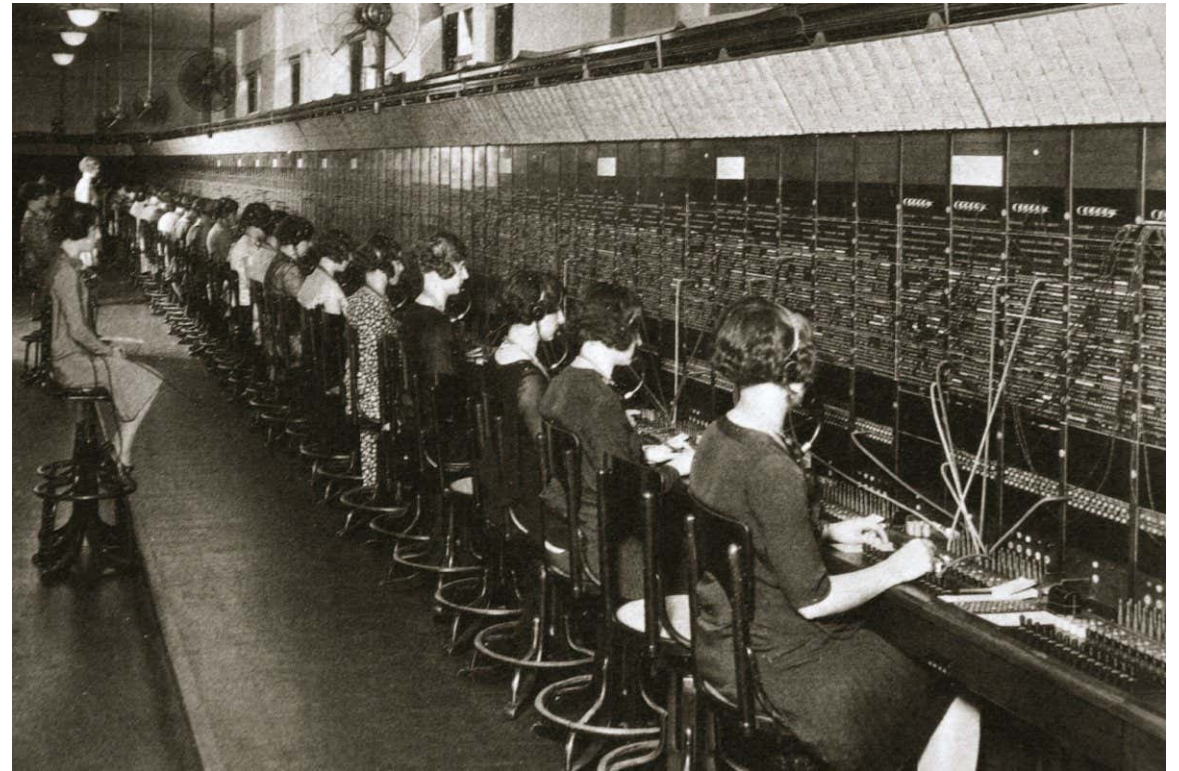
- Establish a connection. Use it for a while
- Used in telephone networks
- Recently used in optical networks

Packet-switched

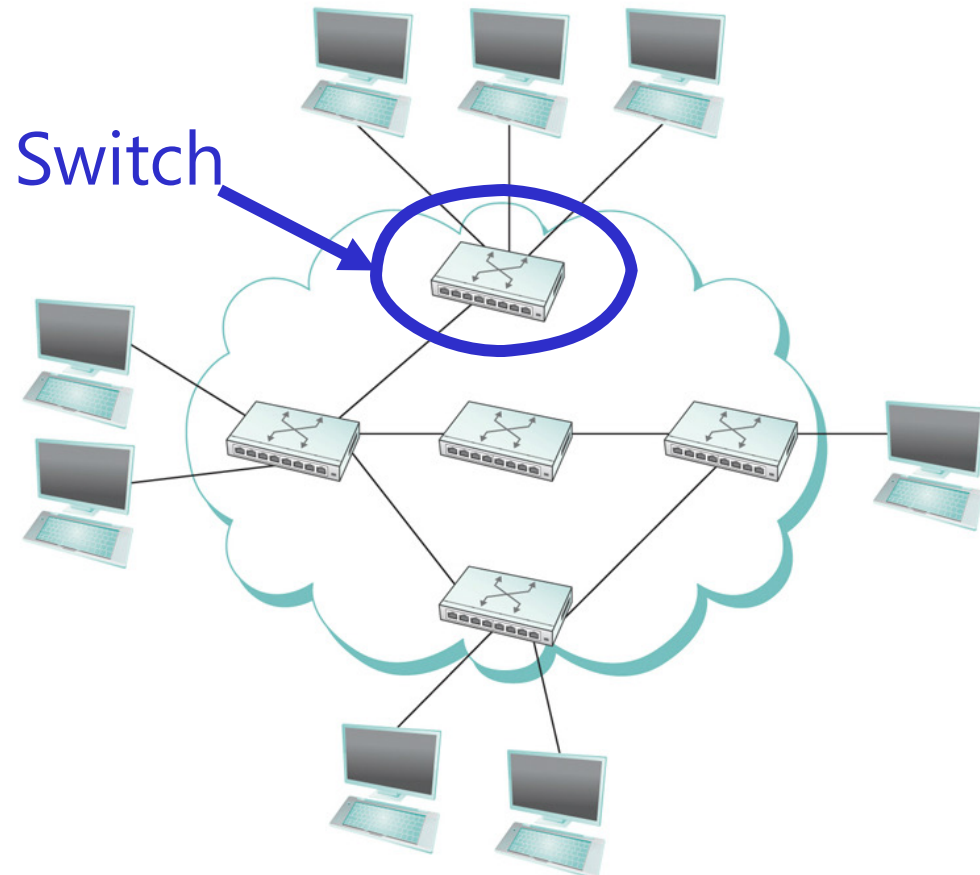
- For each packet of data, figure out which wire it should be sent on. Decide based on the packet’s final destination
- Used in majority of computer networks
- Focus of this course

Circuit switched network

- In the old days, people would call. The call would come in through a copper wire. A human operator would answer. People would say where they want to connect. Human would physically connect the copper wire to the appropriate destination.
- Long distance calls could involve multiple "hops" with multiple switches
- After you finish your call, the copper cable gets freed for someone else to use
- We still have automated versions of this today. There is a potential resurgence with optical networks



Packet-switched networks



Switches: forwarding messages from one node to another

- Every node has a unique address

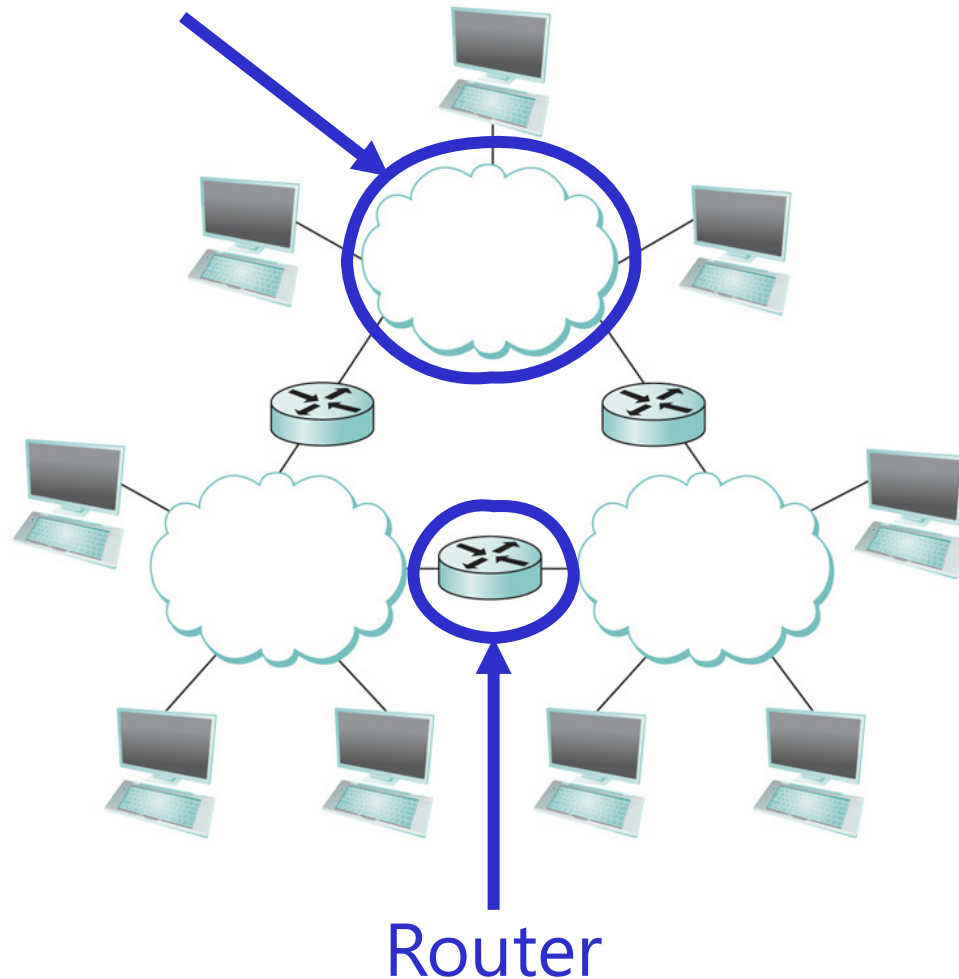
Forwarding table

Node address	Switch port number
Node 1's address	0
Node 2's address	1
...	...

- If a message destined for node 1 arrives, then send it out on port 0
- We will discuss the details later

Internetwork of switched networks

Switched network



Stitching switched networks to **scale** networks to larger size

Routers: forwarding messages **between networks**

- Using a different type of address type called **IP address**, so every router does not need every node's address
- We will discuss the details later

An internetwork is managed by a single administrative entity

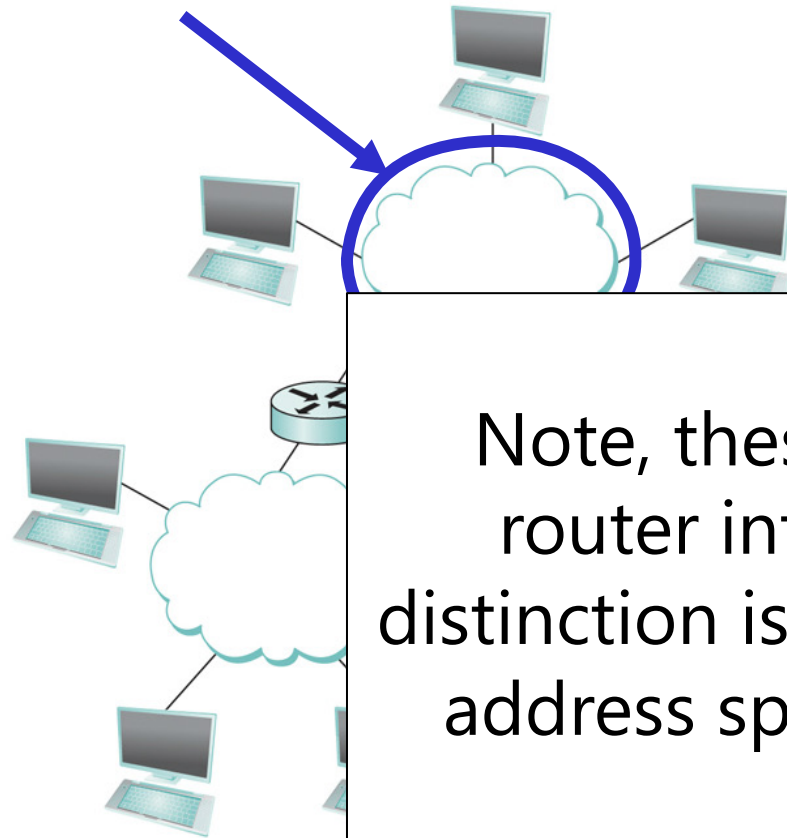
- UT Austin, AT&T, Google, ...

Internetwork of switched networks

Switched network

Stitching switched networks to **scale** networks to larger size

Routers: forwarding messages **between networks**



Note, these days people use switch and router interchangeably. The important distinction is whether there is hierarchy in the address space (we will discuss more later)

type called IP
not need every

single

Router

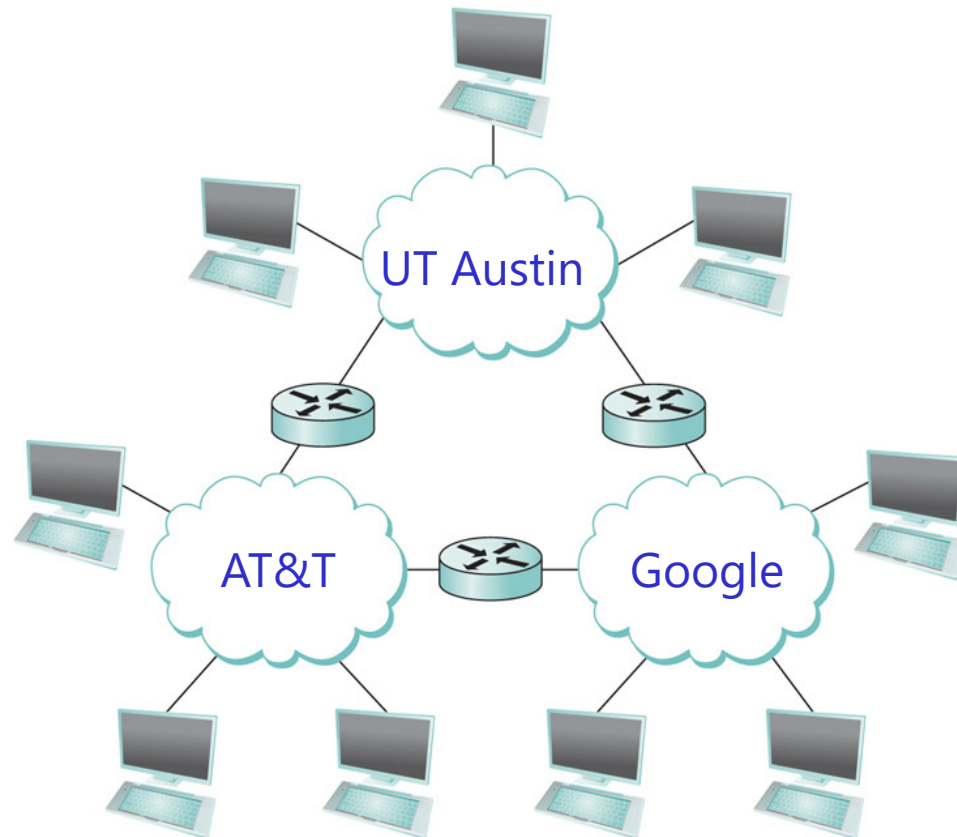
administrative entity

- UT Austin, AT&T, Google, ...

The “Internet”

Recursively grouped interconnection of internetworks

- The interconnections are again made via routers



Summary

Computer networks are an essential infrastructure for modern apps

- Must be **general** to support diverse applications' needs
- Must adapt to changing technology and societal needs

Scalable connectivity: support for an ever-increasing number of computers on a network

- **Packet-switched** networks via switches
- **Internetwork of switched networks** via routers
- **The "Internet"**: recursively grouped interconnection of internetworks

Next up: other requirements and network architecture