Advanced Computer Networks

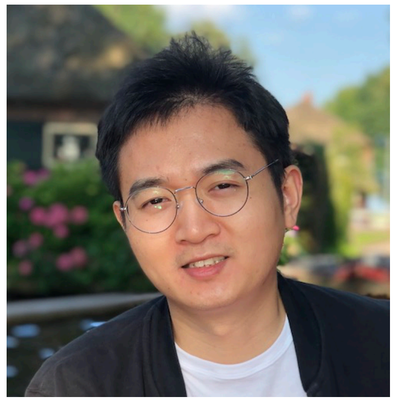
Introduction

Lin Wang

Period 2, Fall 2022

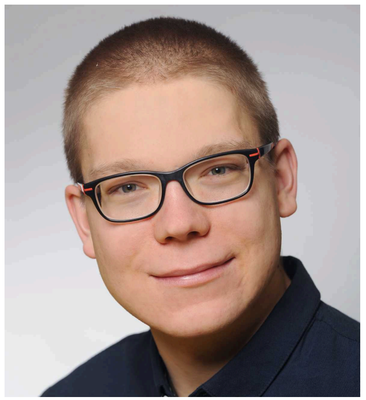
VU VRIJE UNIVERSITEIT AMSTERDAM

Teaching team



Lin Wang (Lecturer)
Assistant Professor
https://linwang.info

George Karlos (TA)
PhD researcher



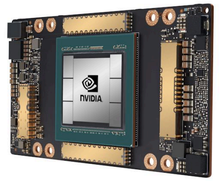
Florian Gerlinghoff (TA)
MSc student

Our research

We focus on high-performance distributed computing and networking

Programmable networks

In-network computing, network monitoring



Machine learning systems

Real-time inference serving, embedded machine learning

Thesis topics 2022: shorturl.at/ekEIX

What do you know about networking?

Goals of the course

To get familiar with the **state-of-the-art** of computer networking technologies

To be able to reason about the **designs/principles** in networks and networked systems

To gain **hands-on experience** with networked systems programming and outlook for research

To practice the **art of reading** research papers

It is a **big** field, so we can only focus on just **a few** topics.

Course logistics

All teaching activities will be in-person

- Check Rooster for locations
- Lectures recorded for offline studies
- Exams will be on-campus (no online alternatives)

Communication channels

- All announcements and all material on Canvas
- Discussions on Canvas encouraged
- To questions: please send an email to vu.acn.ta@gmail.com (emails to our personal accounts may not be processed)

Office hours

- Every Wednesday 9:30 - 10:30, NU-11A33

Course grading

Project: 50 points

Final exam: 50 points

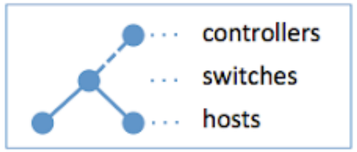
PASS condition

- If you obtain no less than 25/50 points in **both** components

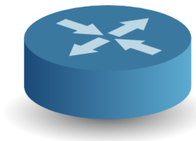
Final grade scaling

[95, 100] → 10.0	[68, 75) → 7.5
[90, 95) → 9.5	[62, 68) → 7.0
[85, 90) → 9.0	[56, 62) → 6.5
[80, 85) → 8.5	[50, 56) → 6.0
[75, 80) → 8.0	[0, 50) → FAIL

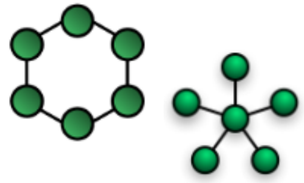
Project labs preview



Lab0: welcome and warm-up

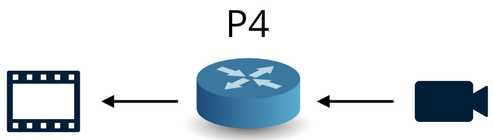


Lab1 (5 points): implement a learning switch

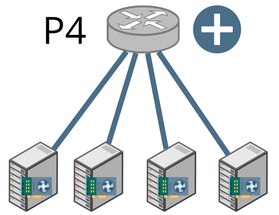


Lab2 (10 points): data center network topology

Lab3 (10 points): build your own data center in Mininet



Lab4 (10 points): video interception in P4



Lab5 (15 points): in-network aggregation in P4

Project labs organization

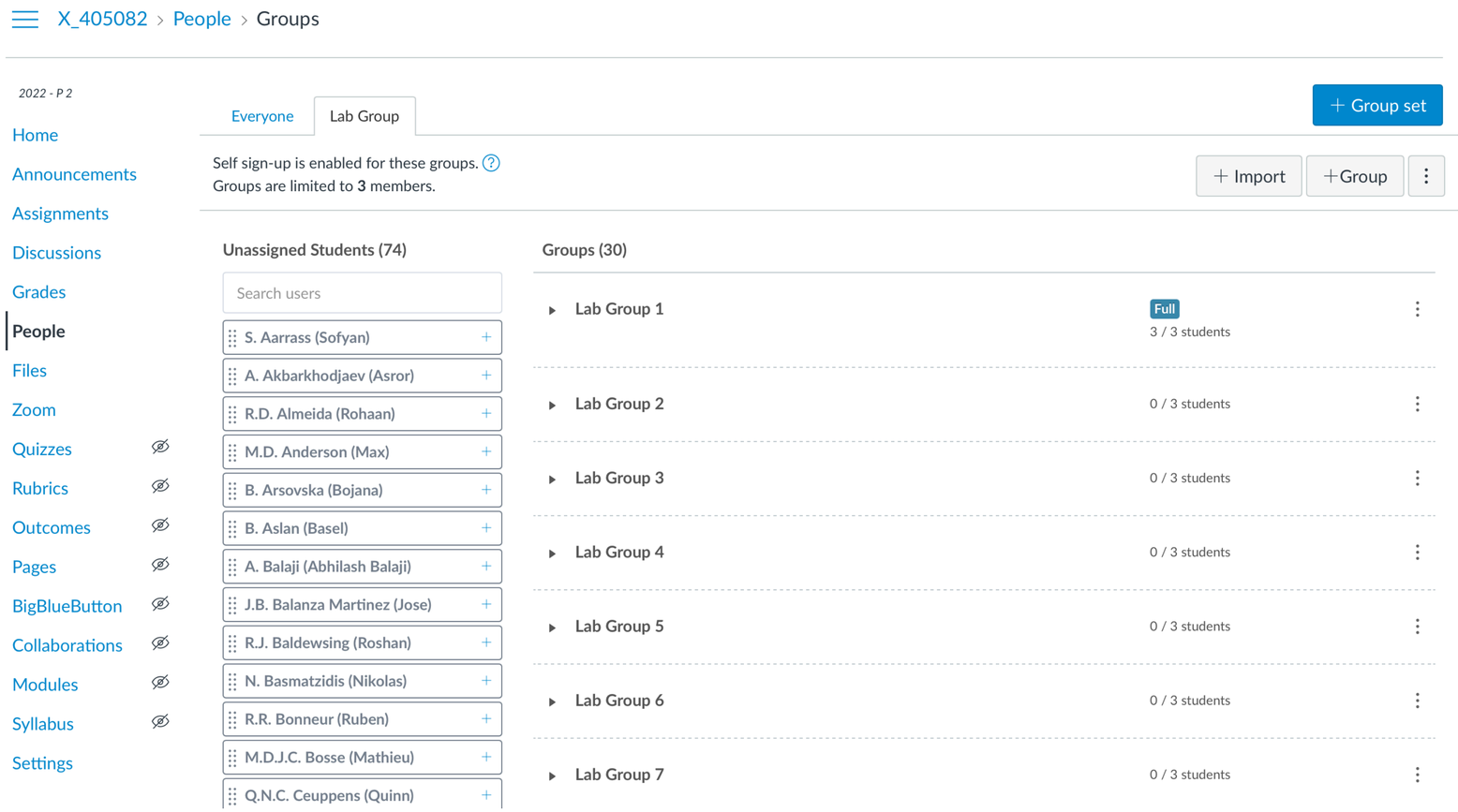
Individual assignments for lab0 and lab1

- Lab0 is a warm-up (with **important instructions**), no points and no submission needed, but **do not skip it**
- Lab1 will be assessed with a Canvas quiz, no code submission needed
    - Only one chance, going back to answered questions not allowed

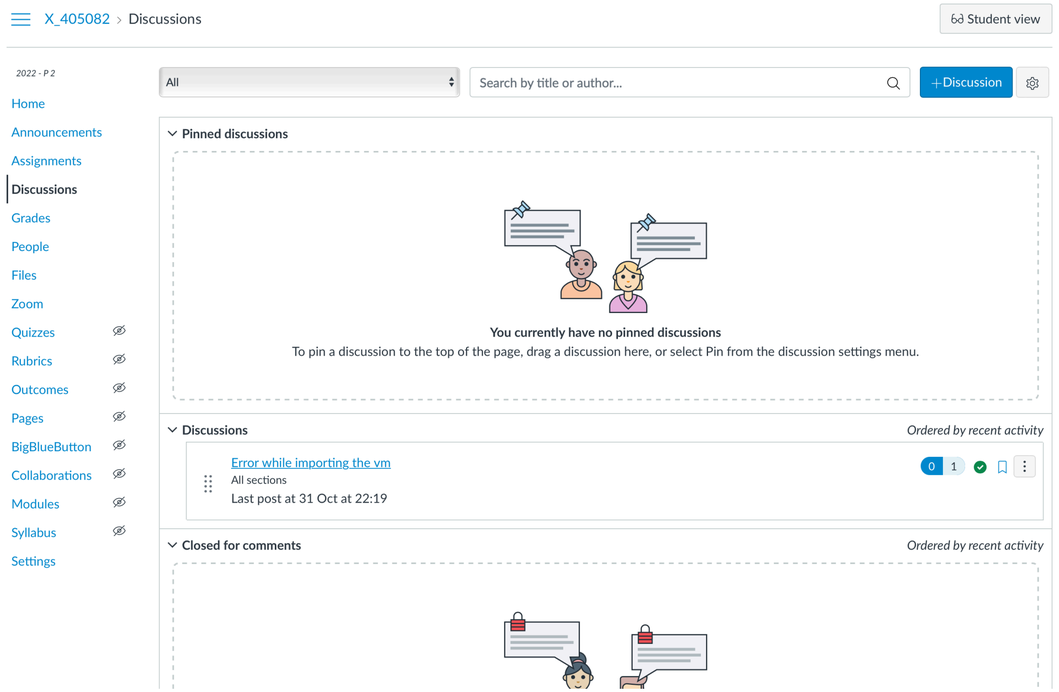
Group assignments for lab2 through lab5

- You work in a group of max. 3
- Choose your own group mates, deadline **Wednesday November 9, 2022**
- Split the work evenly, all of you need to understand the entire code
- Submission: code + report in PDF, all in one zip file

Enroll yourselves to a group on Canvas



Peer discussions are encouraged



You can post general questions/doubts in the discussion and get help from each other, but please do not post your code or spoil answers directly.

Integrity

Zero tolerance → You should **not plagiarize anything** in this course (and other courses)

The following are considered plagiarism

- Copy (part of) a solution from another team or from the Internet
- Buy a solution from any source
- Copy + make changes to any of the above

What happens if someone commits plagiarism

- The case will be reported to the exam committee
- It is up to them to decide on disciplinary actions

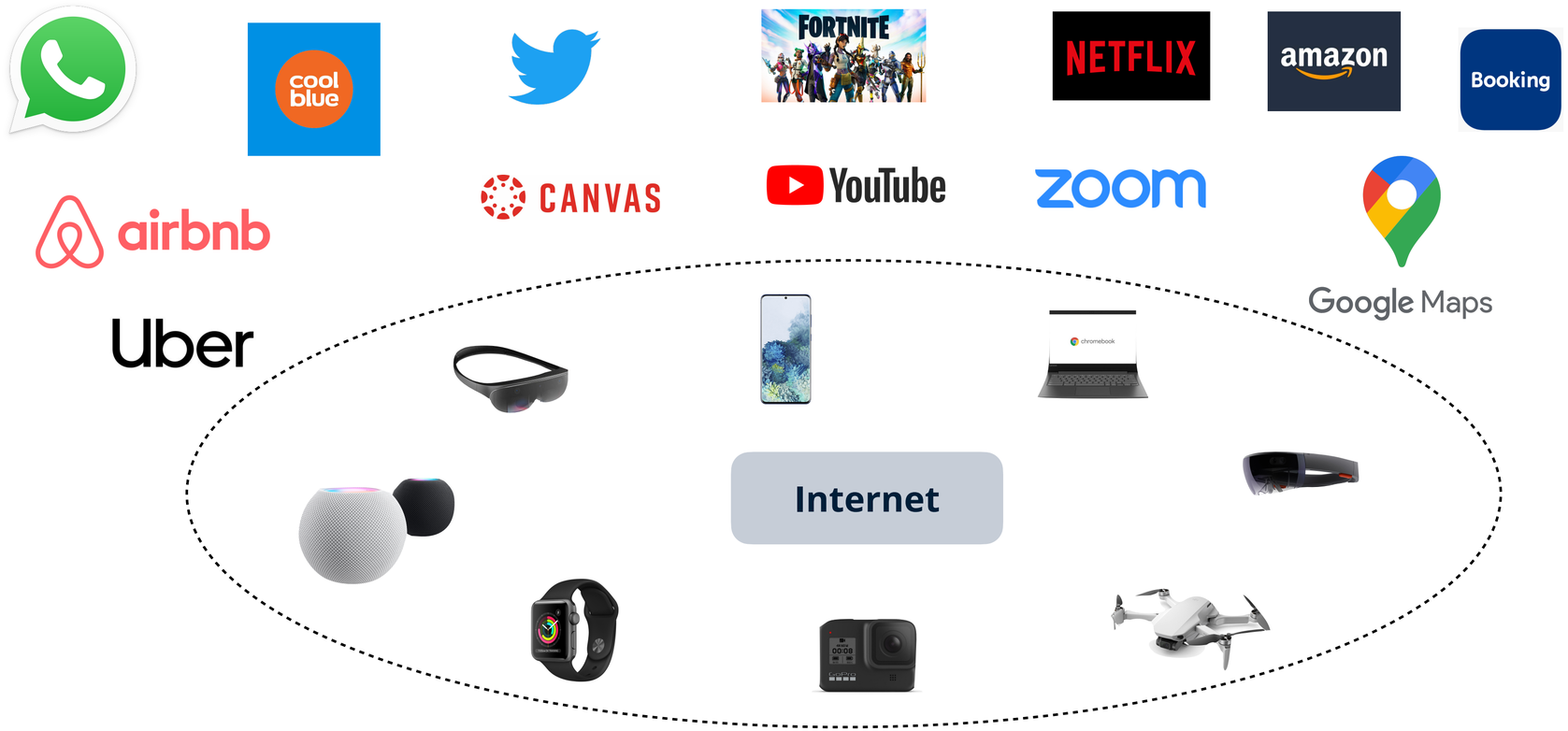
CS-VU diversity and support information

When in doubt: **please be respectful and courteous to everyone!**

- For class-related interactions: please come talk to the teacher(s)
- Talk to your study advisors to find resources and support information
- CS BETA diversity address to send your concerns/questions: diversity.cs.beta@vu.nl
- Department diversity office hours, ***every 3rd Monday of the month, 12-1pm (lunch hours)***
  - STORM Diversity Committee (DiversityCie), diversitycie@storm.vu.nl
    - Student-led committee for promoting diversity in our faculty
- Support information
  - Student well being: https://vu.nl/en/student/student-wellbeing
  - Safe social setting on campus: https://vu.nl/en/about-vu/more-about/safe-social-setting-on-campus
- https://3d.vu.nl/ for diversity and support related issues at the VU level

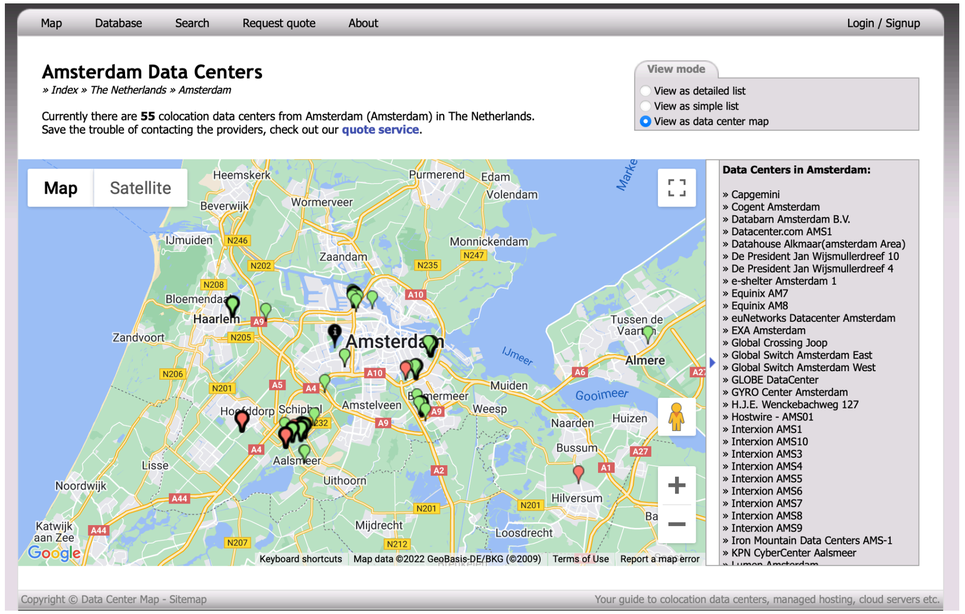
Questions?

Why this course?



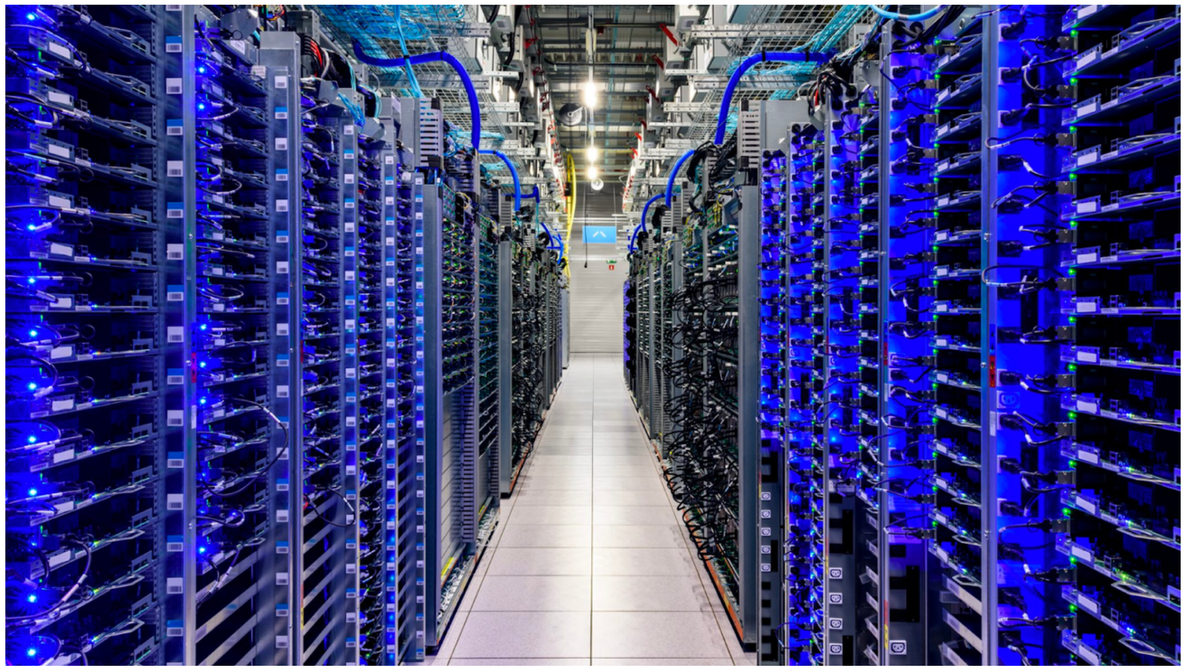
The Internet is behind most of our daily activities nowadays, and it has a huge impact on our society!

Data centers in Amsterdam



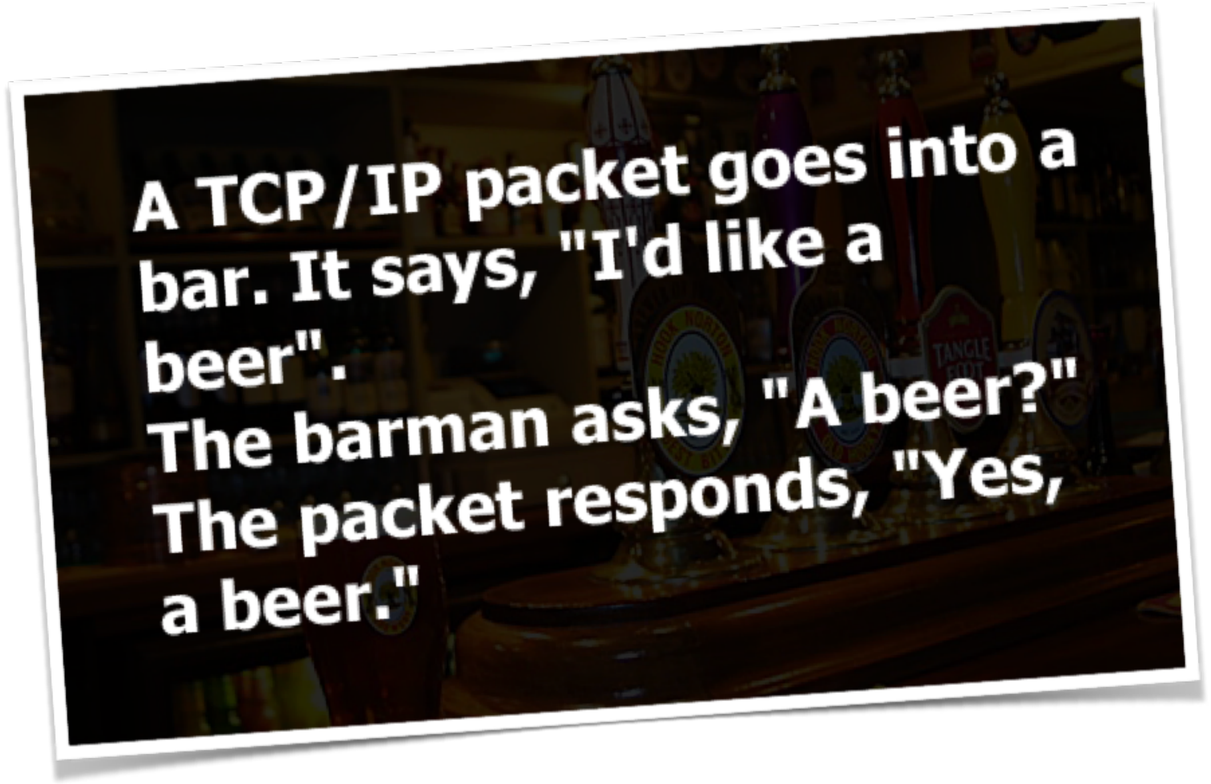
https://www.dutchdatacenters.nl/en/data-centers/what-is-the-economic-impact-of-data-centers/

Why this course?



What technologies are used to build a modern data center @Google or @Amazon?

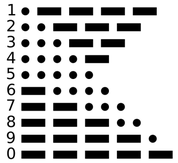
Why this course?



For the love of terrible jobs...

Internet: then and now

History of Internet: telephone network



1838: Morse code

1927: first telephone service (US-UK)

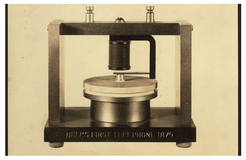
1962: satellite in telecommunications

1981: first commercial cellular network (1G)

2G, 3G, 4G, 5G...

1672: first acoustic telephone

1876: first telephone by Bell



1934: first radio telephone call

1973: first cellular mobile call

History of Internet: visions at that time

Memex

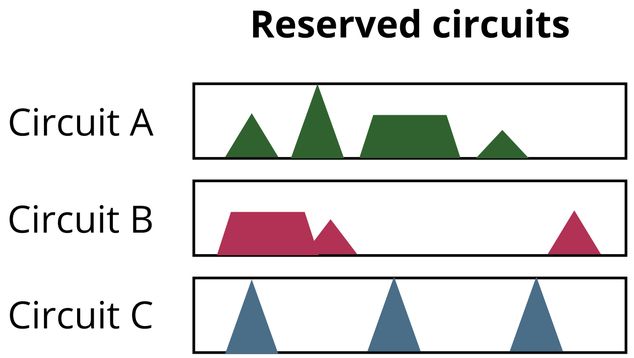
- Vannevar Bush, “As we may think”, 1945
- A hypothetical proto-hypertext system in which individuals would compress and store all of their books, records, and communications

Galactic network

- J.C.R. Licklider (MIT), “Galactic network”, 1962
- Concept of a global network of computers connecting people with data and programs
- First head of DARPA (Department of Defense Advanced Research Projects Agency) computer research, October 1962

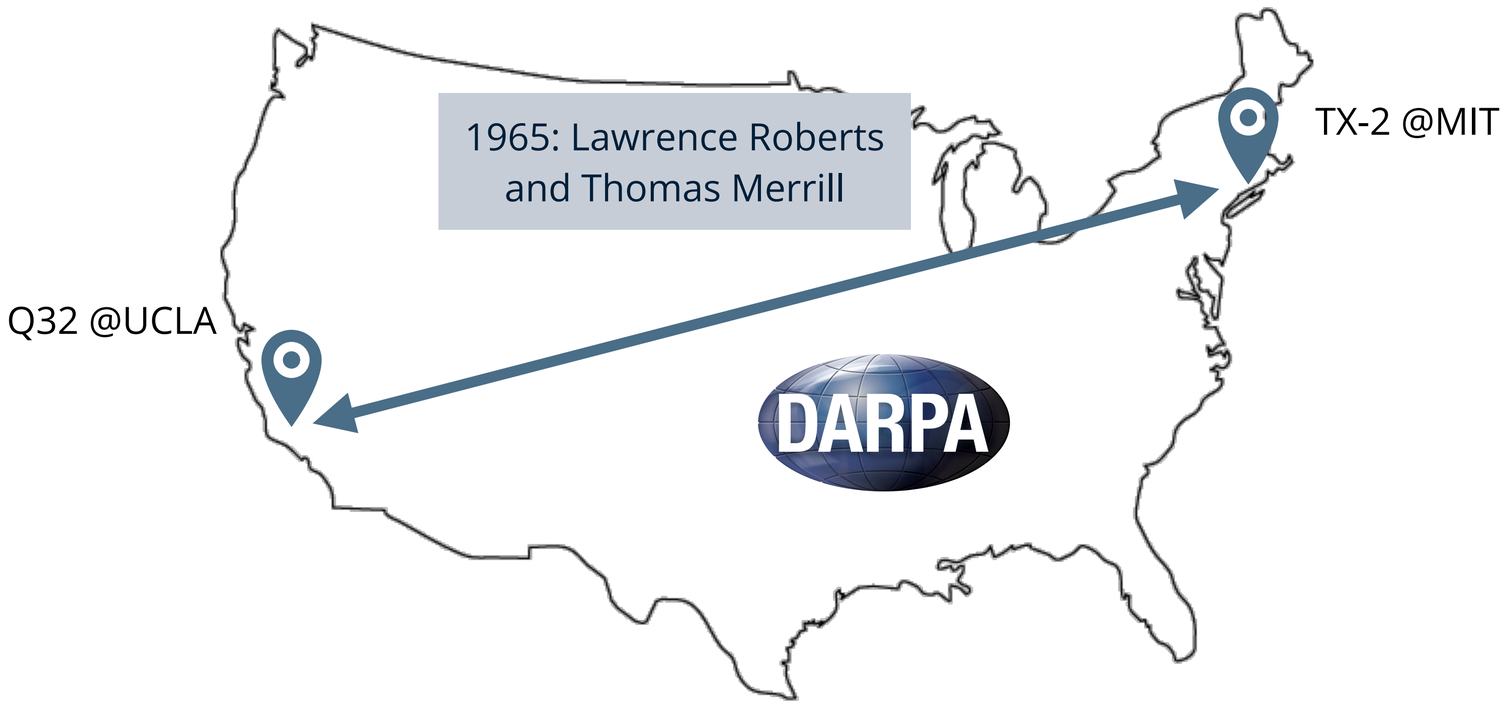
https://www.internetsociety.org/internet/history-internet/brief-history-internet/

Circuit switching



- Physical channel carrying stream of data from source to destination
- Three phases: setup, data transfer, tear-down
- Data transfer involves no routing

History of Internet: first wide area network



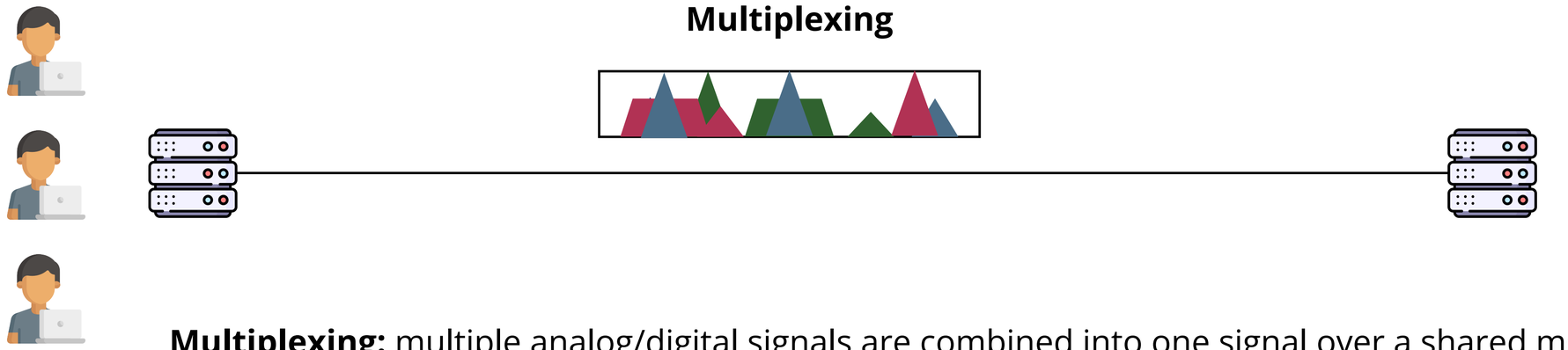
Connection is through the telephone line – it works, but it is inefficient and expensive – confirming the motivation for packet switching

Packet switching

1960s: Time-sharing operating systems (e.g., MULTICS) began to emerge

Leonard Kleinrock: queueing-theoretic analysis of packet switching in his MIT PhD thesis (1961-63) demonstrated **value of statistical multiplexing**

Concurrent work from Paul Baran (RAND), Donald Davies (National Physical Labratories, UK)



Multiplexing: multiple analog/digital signals are combined into one signal over a shared medium

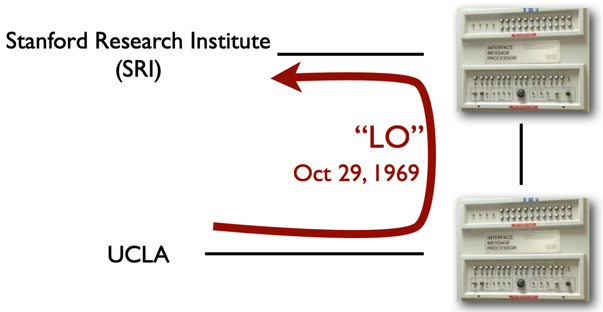
- Message broken into short packets, each handled separately
- One operation: send packet
- Packets stored/queued in each router, forwarded to appropriate neighbor

History of Internet: ARPANET

1967 Lawrence Roberts publishes plan for the ARPANET computer network

1968 Bolt Beranek and Newman (BBN) wins bid to build packet switches, the Interface Message Processor (IMP)

1969 BBN delivers first IMP to Kleinrock's lab at UCLA



Oct 29, 1969: ARPANET went live!

The intended message was "login", but the system crashed after "o"

History of Internet: ARPANET grows

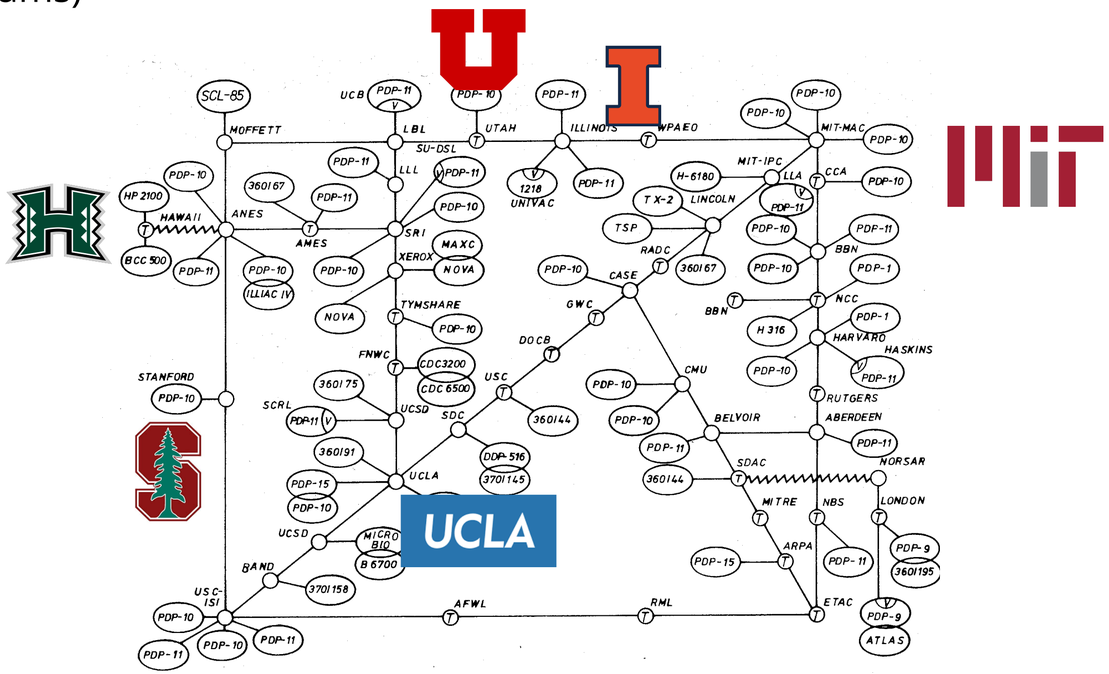
1971 Telnet, FTP

1972 Email (Ray Tomlinson, BBN)

1973 USENET (precursor of Internet forums)

Originally for military computer networking, later expanded for universities

GOOD TO KNOW
ALOHAnet@Hawaii: first public wireless data network (inspiration for Ethernet and WiFi networks)



History of Internet: network of networks

In the meantime of ARPANET

- Other networks, such as PRnet, SATNET developed
- May 1973: Vinton G. Cerf and Robert E. Kahn present first paper on interconnecting networks

Concept of **connecting diverse networks**, unreliable datagrams, global addressing, etc. → became TCP/IP

IEEE TRANSACTIONS ON COMMUNICATIONS, VOL. COM-22, NO. 5, MAY 1974

A Protocol for Packet Network Intercommunication

VINTON G. CERF AND ROBERT E. KAHN, MEMBER, IEEE

***Abstract*—A protocol that supports the sharing of resources that exist in different packet switching networks is presented. The protocol provides for variation in individual network packet sizes, transmission failures, sequencing, flow control, end-to-end error checking, and the creation and destruction of logical process-to-process connections. Some implementation issues are considered, and problems such as internetwork routing, accounting, and timeouts are exposed.**

INTRODUCTION

IN THE LAST few years considerable effort has been expended on the design and implementation of packet switching networks [1]–[7],[14],[17]. A principle reason for developing such networks has been to facilitate the sharing of computer resources. A packet communication network includes a transportation mechanism for delivering data between computers or between computers and terminals. To make the data meaningful, computers and terminals share a common protocol (i.e., a set of agreed upon conventions). Several protocols have already been developed for this purpose [8]–[12],[16]. However, these protocols have addressed only the problem of computer resources called HOSTS, a set of one or more *packet switches*, and a collection of communication media that interconnect the packet switches. Within each HOST, we assume that there exist *processes* which must communicate with processes in their own or other HOSTS. Any current definition of a process will be adequate for our purposes [13]. These processes are generally the ultimate source and destination of data in the network. Typically, within an individual network, there exists a protocol for communication between any source and destination process. Only the source and destination processes require knowledge of this convention for communication to take place. Processes in two distinct networks would ordinarily use different protocols for this purpose. The ensemble of packet switches and communication media is called the *packet switching subnet*. Fig. 1 illustrates these ideas.

In a typical packet switching subnet, data of a fixed maximum size are accepted from a source HOST, together with a formatted destination address which is used to

http://pbg.cs.illinois.edu/courses/cs598fa09/readings/ck74.pdf


I didn't invent
the Internet
I did not
invent the Web

History of Internet: standards

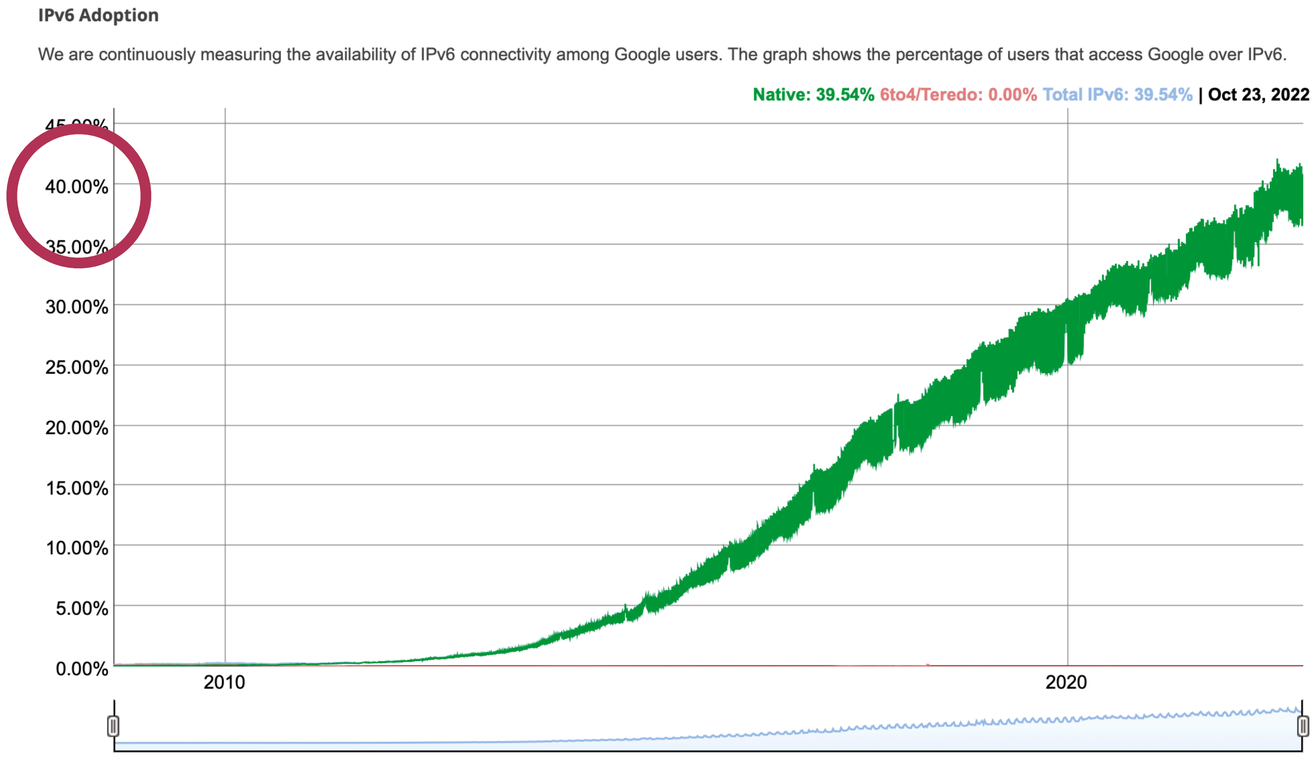
TCP/IP: interconnecting networks

- TCP/IP implemented on mainframes by groups at Stanford, BBN, and UCL
- David Clark implements it on Xerox Alto and IBM PC
- 1982: International Organization for Standards (ISO) releases Open Systems Interconnection (OSI) reference model
- Jan 1, 1983: **“flag day”** NCP (Network Control Protocol) to TCP/IP transition on ARPANET

Ethernet: local area networking

- 1976: R. Metcalfe and D. Boggs
- 1985: Radia Perlman, Spanning Tree Protocol (STP)

Another flag day is almost impossible nowadays



The global IPv4 → IPv6 transition is extremely low...

https://www.google.com/intl/en/ipv6/statistics.html

History of Internet: fast development

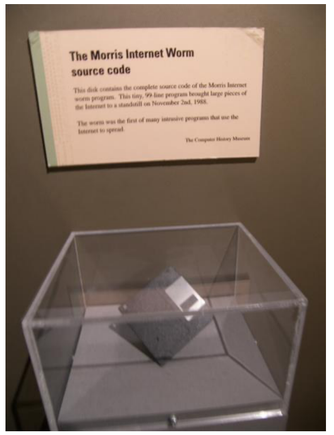
1983 DNS developed by Jon Postel, Paul, Mockapetris (USC/ISI), Craig Partridge (BBN)

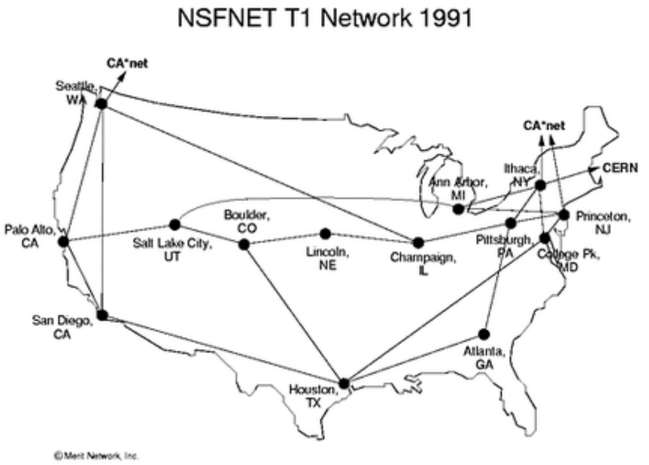
1984 Hierarchical routing: EGP, IGP (later to become eBGP and iBGP)
NSFNET for US higher education
- Served many users, not just one field
- Encouraged development of private infrastructure (e.g., backbone required to be used for research and education)
- Stimulated investment in commercial long-haul networks

1988 Morris worm – first computer worm

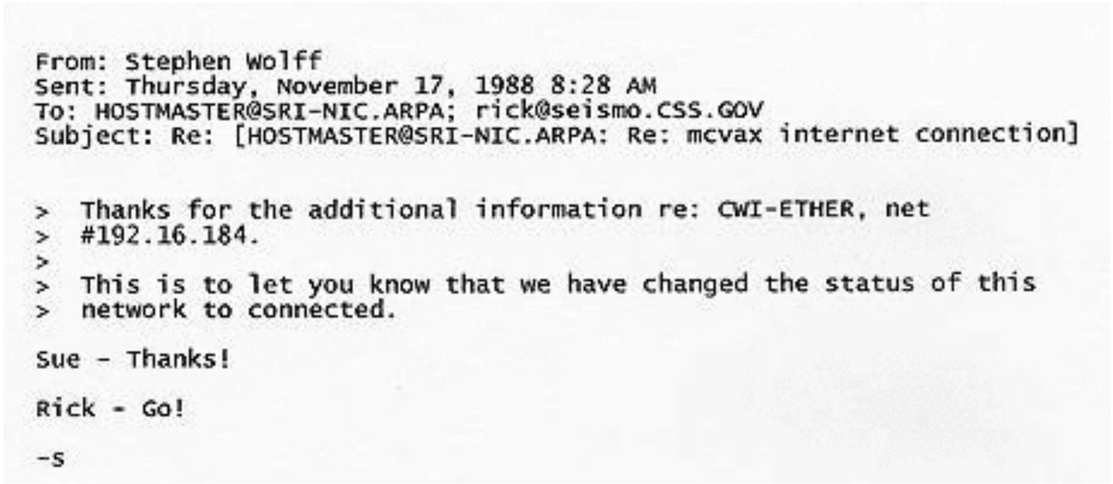
1990 ARPANET ends

1995 NSFNET decommissioned





Internet in The Netherlands



From: Stephen Wolff
Sent: Thursday, November 17, 1988 8:28 AM
To: HOSTMASTER@SRI-NIC.ARPA; rick@seismo.CSS.GOV
Subject: Re: [HOSTMASTER@SRI-NIC.ARPA: Re: mcvax internet connection]

> Thanks for the additional information re: CWI-ETHER, net
> #192.16.184.
>
> This is to let you know that we have changed the status of this
> network to connected.

Sue - Thanks!

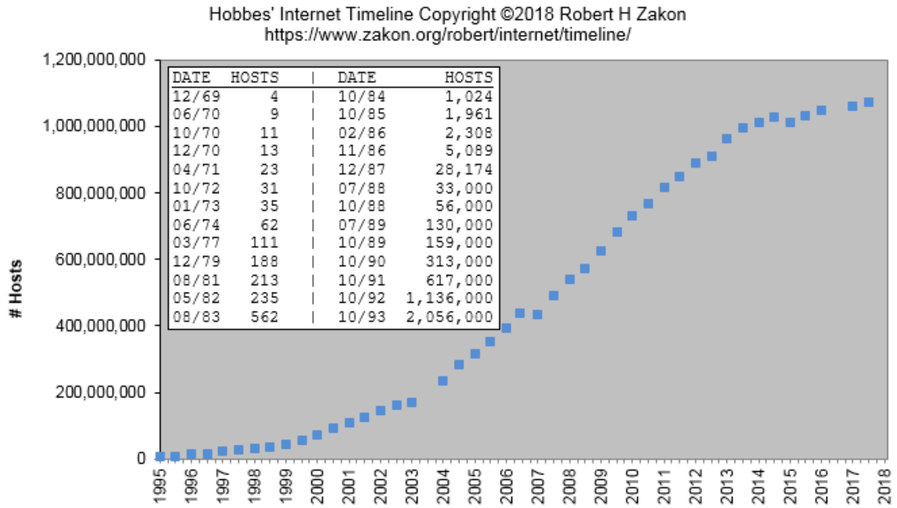
Rick - Go!

-s

First email via the first transatlantic connection (Nov 17, 1988)

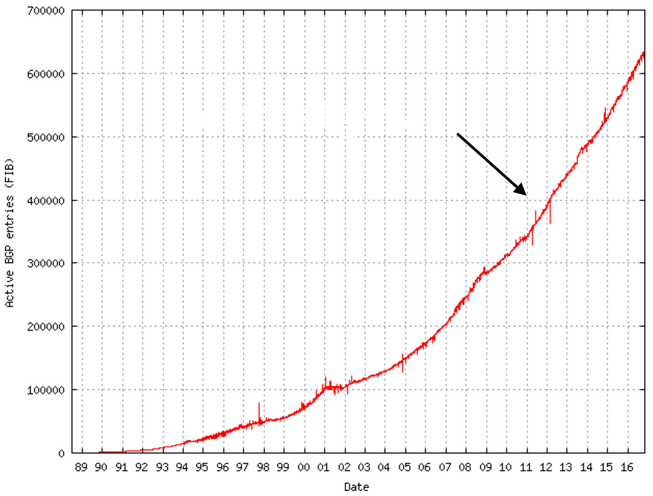
Piet Beertema (CWI): https://godfatherof.nl

Internet growth



DATE	HOSTS	DATE	HOSTS
12/69	4	10/84	1,024
06/70	9	10/85	1,961
10/70	11	02/86	2,308
12/70	13	11/86	5,089
04/71	23	12/87	28,174
10/72	31	07/88	33,000
01/73	35	10/88	56,000
06/74	62	07/89	130,000
03/77	111	10/89	159,000
12/79	188	10/90	313,000
08/81	213	10/91	617,000
05/82	235	10/92	1,136,000
08/83	562	10/93	2,056,000

Internet forwarding table size



Year

Questions?

What do we want from the network?

Performance: latency? bandwidth?

Reliability, availability, security?

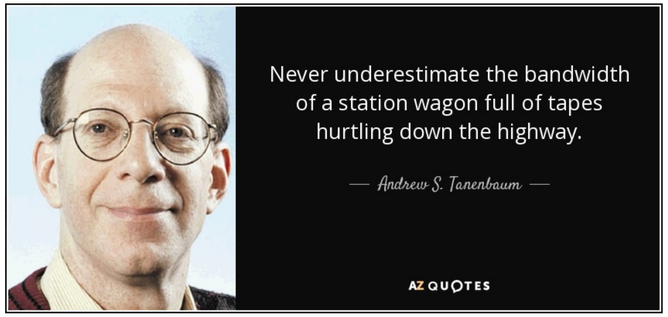
Flexibility, manageability?

Others?

Network performance

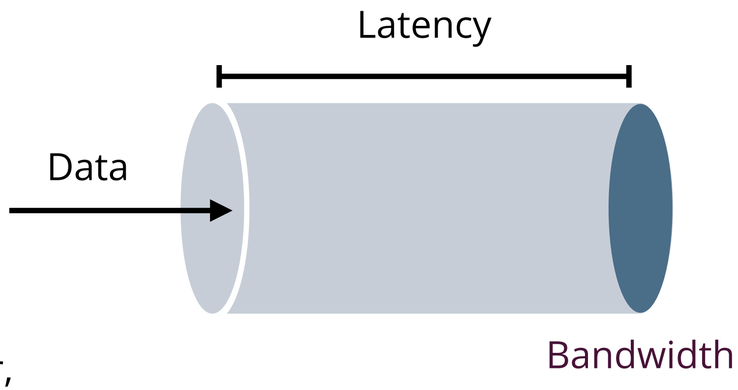
Bandwidth

- Definition: maximum amount of data transfer across a given network path in a given amount of time
- "*Never underestimate the bandwidth of a station wagon full of tapes hurtling down the highway.*" [1]



Latency

- Definition: the amount of time it takes to deliver some data from the source to the destination across the network
- “Latency lags bandwidth” — David A. Patterson



[1] Andrew S. Tanenbaum paraphrasing Dr. Warren Jackson, Director, University of Toronto Computing Services (UTCS) circa 1985

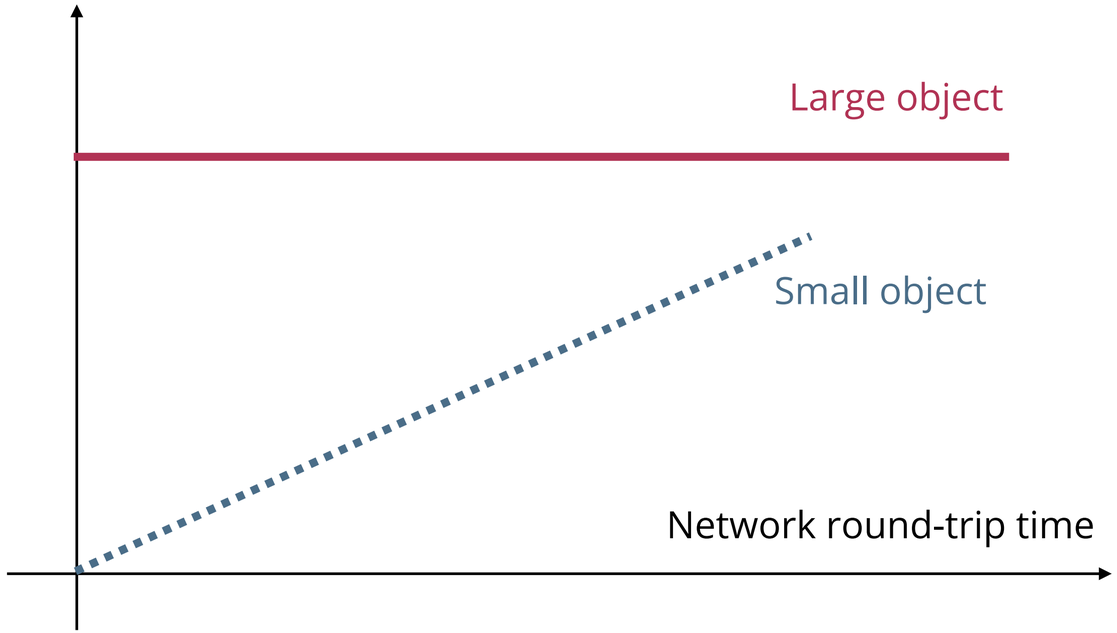
Network latency

	Barcelona	Paris	Tokyo	Toronto	Washington
Amsterdam	31.861ms	10.867ms	244.908ms	93.658ms	78.877ms
Auckland	273.837ms	269.753ms	180.19ms	222.641ms	240.231ms
Copenhagen	39.22ms	22.013ms	242.061ms	95.813ms	90.678ms
Dallas	127.285ms	113.482ms	138.571ms	37.592ms	43.562ms
Frankfurt	23.778ms	10.283ms	216.044ms	102.751ms	150.243ms
London	28.053ms	8.133ms	226.941ms	89.71ms	76.85ms
Los Angeles	152.733ms	142.583ms	100.183ms	75.196ms	71.105ms
Moscow	75.802ms	49.123ms	271.002ms	134.922ms	186.522ms
New York	100.964ms	72.693ms	176.968ms	23.118ms	8.073ms
Paris	22.847ms	—	235.576ms	93.516ms	81.167ms
Stockholm	52.198ms	31.29ms	244.968ms	100.352ms	104.932ms
Tokyo	213.107ms	235.594ms	—	166.963ms	172.307ms

https://wondernetwork.com/pings

What factors are involved in these latency numbers?

Impact of bandwidth vs. latency



Assume a network link with 10Gbps bandwidth and 1ms latency? How long does it take to transfer 1B, 100KB, or 10GB of data?

With **TCP**: 100KB = 1460B * (0 + 2 + 4 + 8 + 16 + 32 + 6) → 7ms in the best case, 0.0143Gbps

Do you know how we come to this?

Performance metric

Flow completion time

- How long does it take to complete a traffic flow?
- How long does it take to complete a set of correlated flows (co-flows)?

What else?

- How long does https://www.google.com/?q=cool+network take?
- What is the best video quality I can watch with, without the annoying “buffering”?
- How to guarantee the per-frame latency (e.g., 20ms) in AR?

Not just about performance

- But also **consistent**, **predictable** performance

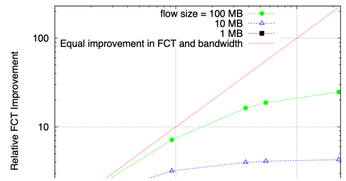
Why Flow-Completion Time is the Right Metric for Congestion Control

Nandita Dukkipati
Computer Systems Laboratory
Stanford University
Stanford, CA 94305-9025
nanditad@stanford.edu

Nick McKeown
Computer Systems Laboratory
Stanford University
Stanford, CA 94305-9025
nickm@stanford.edu

ABSTRACT

Users typically want their flows to complete as quickly as possible. This makes Flow Completion Time (FCT) an important - arguably the most important - performance metric for the user. Yet research on congestion control focuses almost entirely on maximizing link throughput, utilization and fairness, which matter more to the operator than the user. In this paper we show that with typical Internet flow sizes, existing (TCP Reno) and newly proposed (XCP) congestion control algorithms make flows last much longer than necessary - often by one or two orders of magnitude. In contrast, we show how a new and practical algorithm - RCP (Rate Control Protocol) - enables flows to complete close to



ACM SIGCOMM CCR 2006

What about fairness?

Suppose a network is flow-fair. How useful is that?

Flow Rate Fairness: Dismantling a Religion

Bob Briscoe
BT Research & UCL
bob.briscoe@bt.com

ACM SIGCOMM CCR 2007

"Both the thing being allocated (rate) and what it is allocated among (flows) are **completely daft** — both unrealistic and impractical."

Food for thought:

- How to translate microbenchmarks to app-level metrics?
- How to make service providers accountable?
- How to improve the performance and approach fairness?

Network reliability

The end-to-end argument

The fate-sharing principle

Packet vs. circuit switching

The end-to-end argument

TCP provides reliable transport

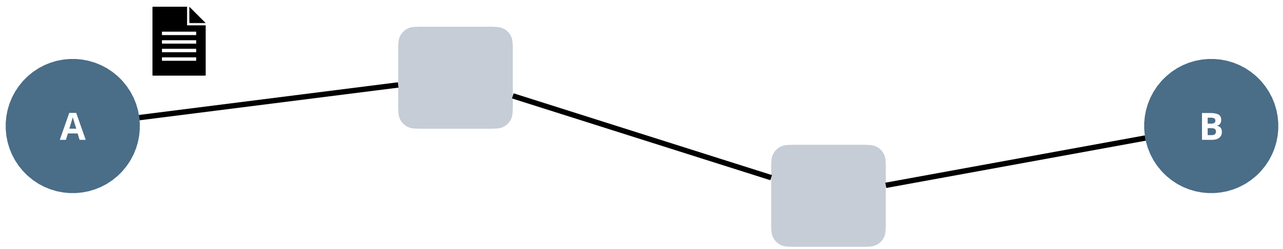
What if no reliable transport is provided?

- Every application that needs reliability has to engineer it from scratch: programming burden, bugs…

What if the network layer tried to provide reliable delivery?

The end-to-end argument

Problem #2: can the network even achieve reliable global packet delivery?



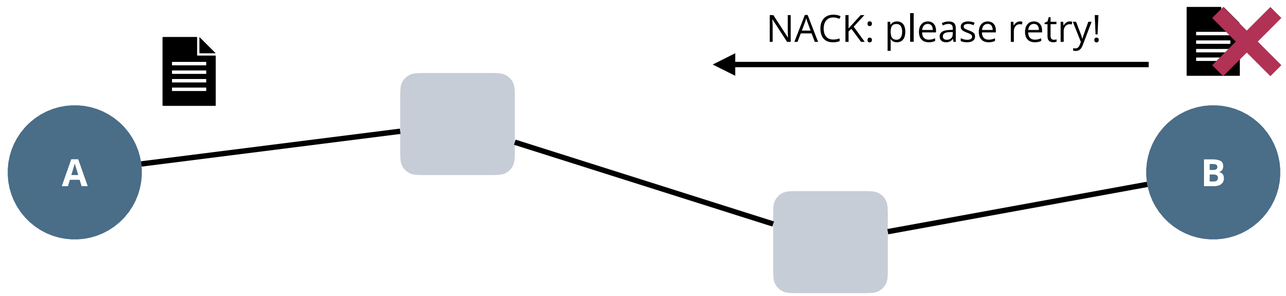
Check reliability at every step (on the network layer)

Problems: bugs, failures are a truth of life

The end-to-end argument

“If a function can only be correctly implemented end-to-end, it must be implemented in the end systems. Implementing it in the network can, at best, only be an optimization.”

Examples?

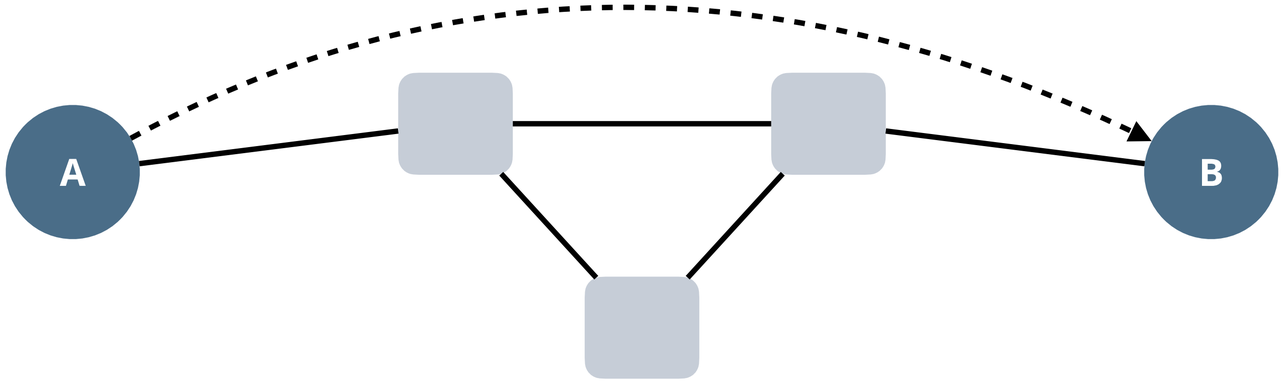


Allow unreliable steps (network layer is best-effort).
B checks correctness. On failure, B tells A to retry.

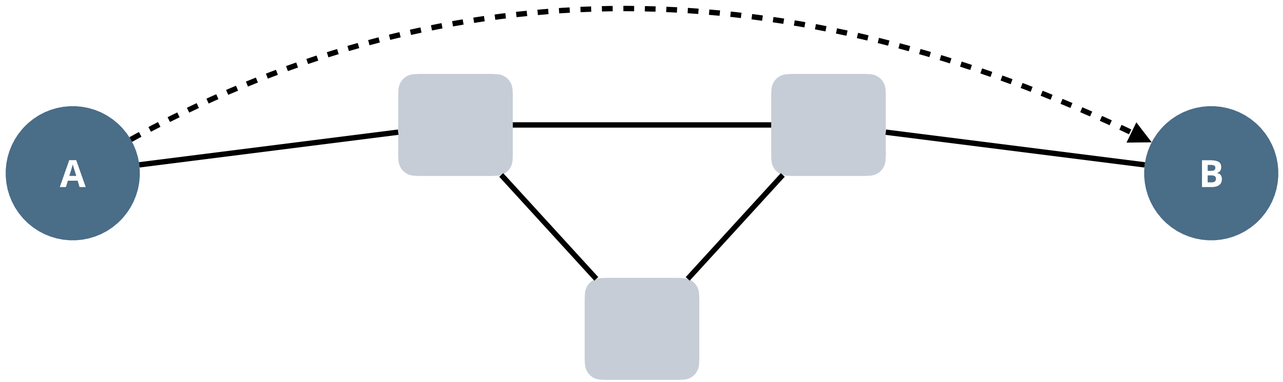
Can still fail, but only if A/B themselves fail →
depends on what end-points themselves control

The fate-sharing principle

Where to maintain A → B connection state?



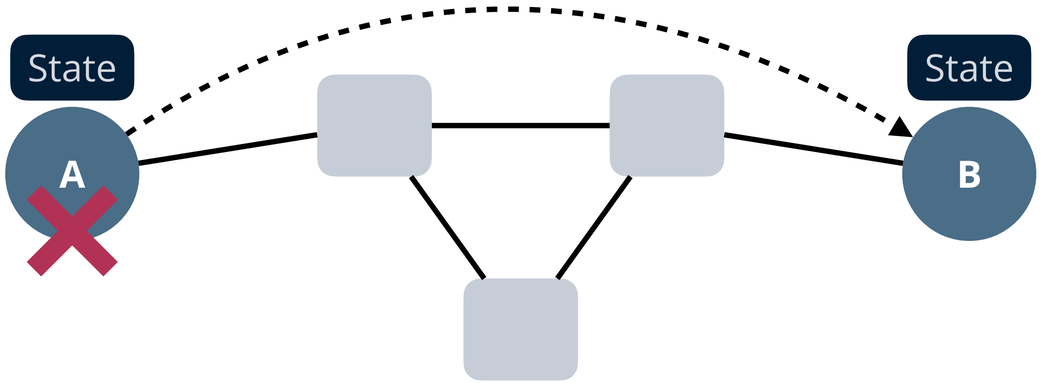
The fate-sharing principle



To deal with potential failures, store critical system state at the nodes which rely on that state. Only way to lose that state is if the node that relies on it fails, in which case it does not matter.

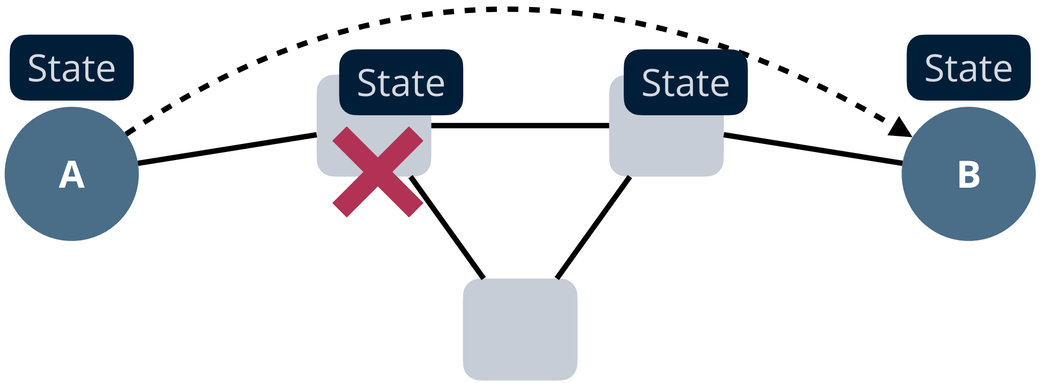
The fate-sharing principle

Keep state on end hosts



In the end-host fails, the connection state is irrelevant anyway

Keep state on network devices



Failures of network devices require the state to be cleaned up or recreated → complex consistency issue!

Packet vs. circuit switching

Predictable performance

Wasteful, if packet is bursty and short

Large latency for small messages, as they wait for circuits creation

Require new circuit setup upon failure

Efficient use of resources

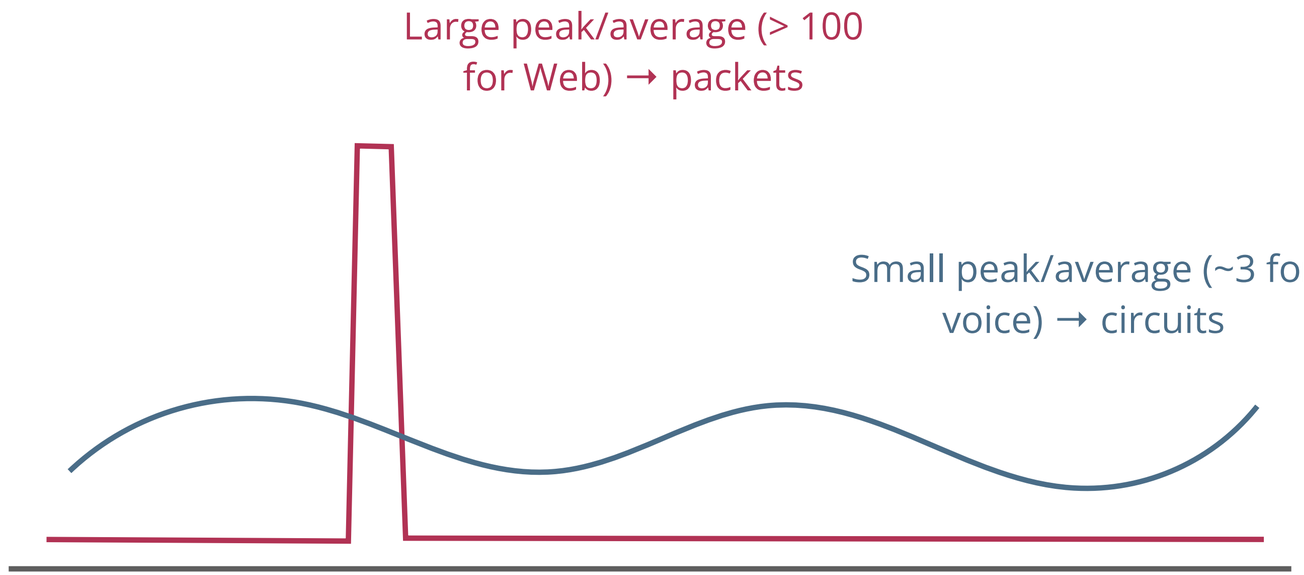
Automatic, in-network rerouting on failures

Unpredictable performance

Requires buffer management and congestion control

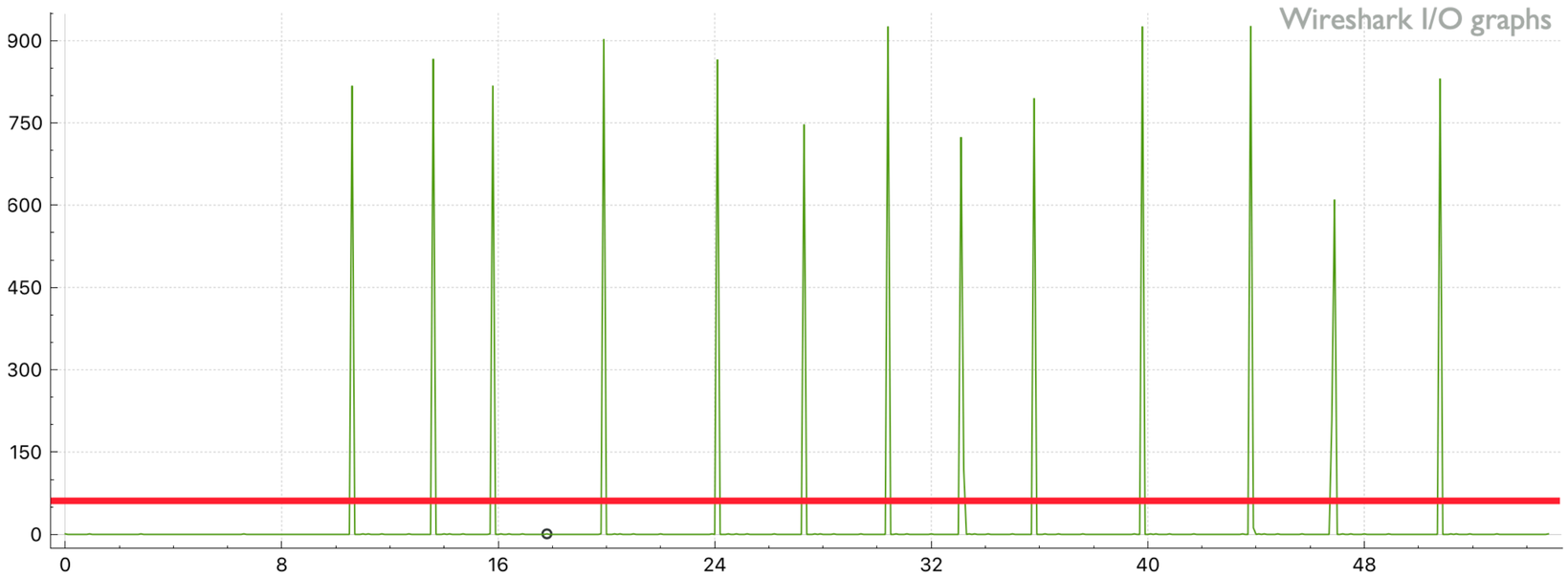
Packet switching beats circuit switching with respect to **resilience** and **efficiency**

Packet vs. circuit switching examples



Which pattern do you think modern video streaming (e.g., YouTube) will follow?

Video streaming today



Do you know why?

Questions?

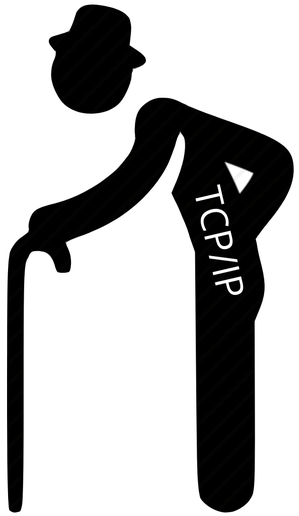
Internet: current status

Internet is there for more than 50 years

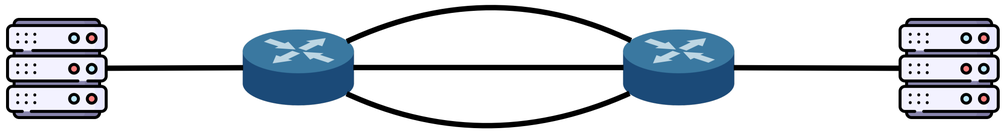
- Networks keep growing and more applications are developed
- TCP/IP is the norm: to program a network application, you simply use the socket APIs

So, the network will keep growing with the same set of technologies?

- Partially, yes, we are still using these old technologies (TCP/IP)
- But, there are also new developments
- This course is to reveal the **state-of-the-art** of computer networking



Still TCP? Yes, but there are more!



Multipath TCP, QUIC

facebook Engineering

Open Source Platforms Infrastructure Systems Physical Infrastructure Video Engineering & AR/VR

POSTED ON OCT 21, 2020 TO ANDROID, DATA INFRASTRUCTURE, IOS, NETWORKING & TRAFFIC, WEB

How Facebook is bringing QUIC to billions

We Need a Replacement for TCP in the Datacenter

John Ousterhout
Stanford University
(paper currently under submission)

Abstract

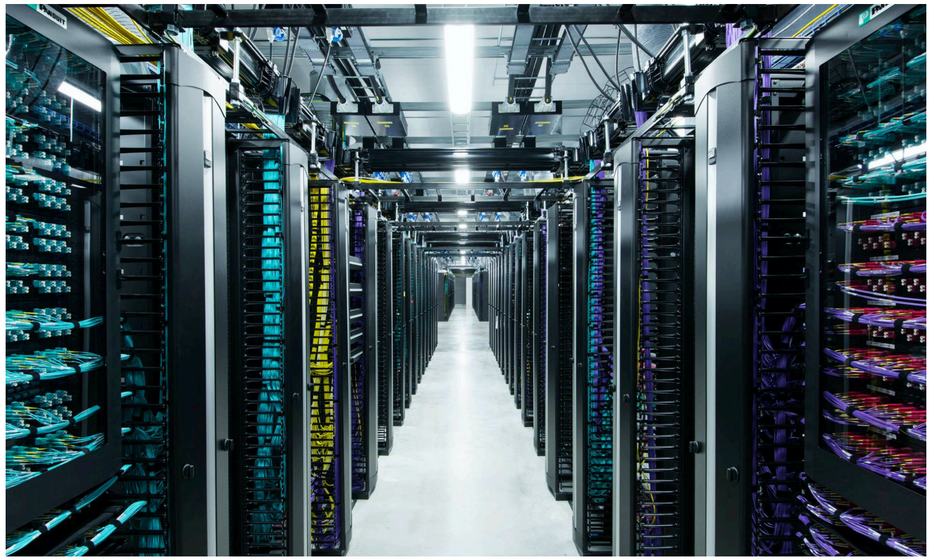
In spite of its long and successful history, TCP is a poor transport protocol for modern datacenters. Every significant element of TCP, from its stream orientation to its requirement of in-order packet delivery, is wrong for the datacenter. It is time to recognize that TCP's problems are too fundamental and interrelated to be fixed; the only way to harness the full performance potential of modern networks is to introduce a new transport protocol into the datacenter. Homa demonstrates that it is possible to create a transport protocol that avoids all of TCP's problems. Although Homa is not API-compatible with TCP, it should be possible to bring it into widespread usage by integrating it with RPC frameworks.

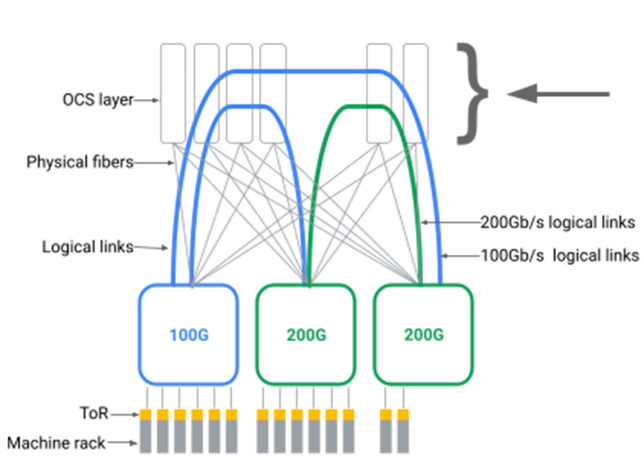
1 Introduction

The TCP transport protocol [6] has proven to be phenomenally successful and adaptable. At the time of TCP's design in the late 1970's, there were only about 100 hosts attached One example is load balancing, which is essential in datacenters in order to process high loads currently. Load balancing did not exist at the time TCP was designed, and TCP interferes with load balancing both in the network and in software.

Section 4 argues that TCP cannot be fixed in an evolutionary fashion; there are too many problems and too many interlocking design decisions. Instead, we must find a way to introduce a radically different transport protocol into the datacenter. Section 5 discusses what a good transport protocol for datacenters should look like, using Homa [16, 18] as an example. Homa was designed in a clean-slate fashion to meet the needs of datacenter computing, and virtually every one of its major design decisions was made differently than for TCP. As a result, some problems, such as core congestion, are eliminated entirely. Other problems, such as congestion control and load balancing, become much easier to address. Homa demonstrates that it is possible to solve all of TCP's problems.

How does Google construct their data center networks?





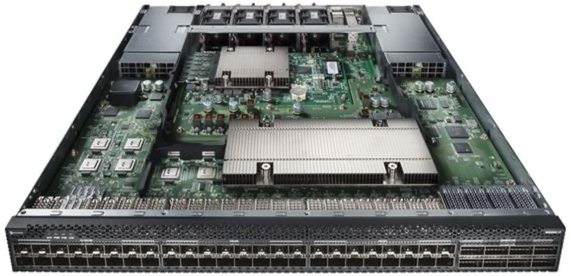
https://cloud.google.com/blog/topics/systems/the-evolution-of-googles-jupiter-data-center-network

New networks

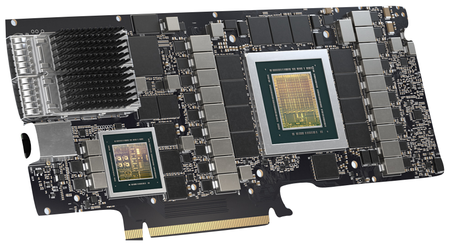
https://www.starlink.com

New network devices

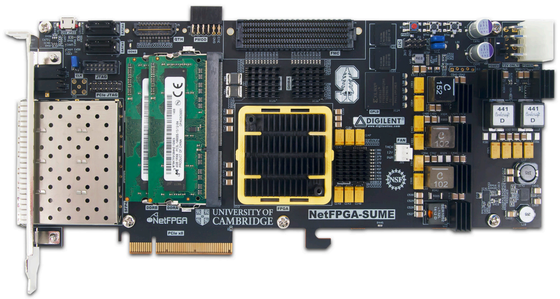
Available at our lab, contact us for fun projects!



Intel Tofino switching ASIC



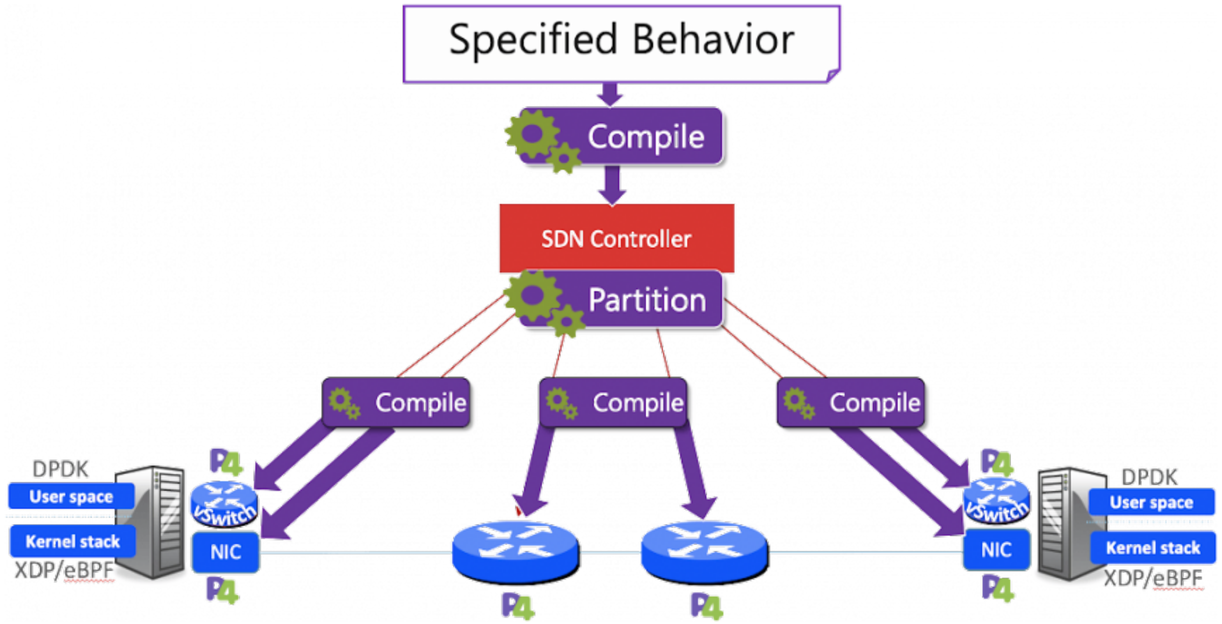
Nvidia SmartNICs and DPUs



NetFPGA

Available in our research group.

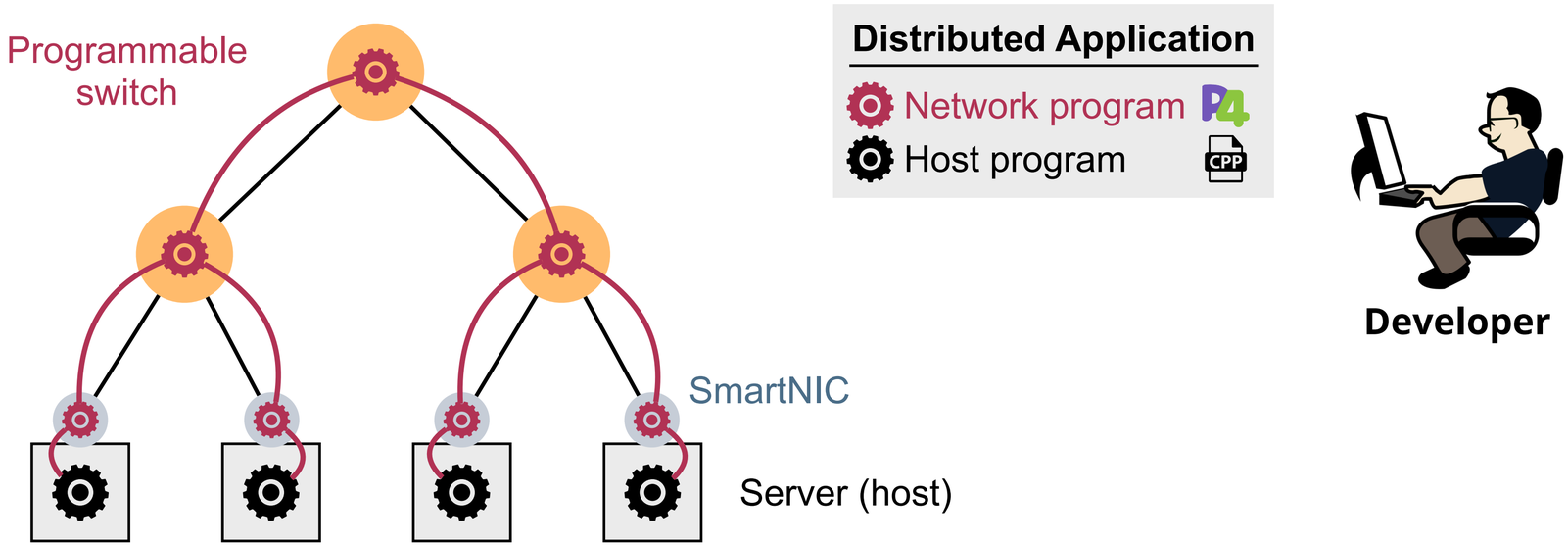
Programming the network as simple as programming the computer



https://prontoproject.org

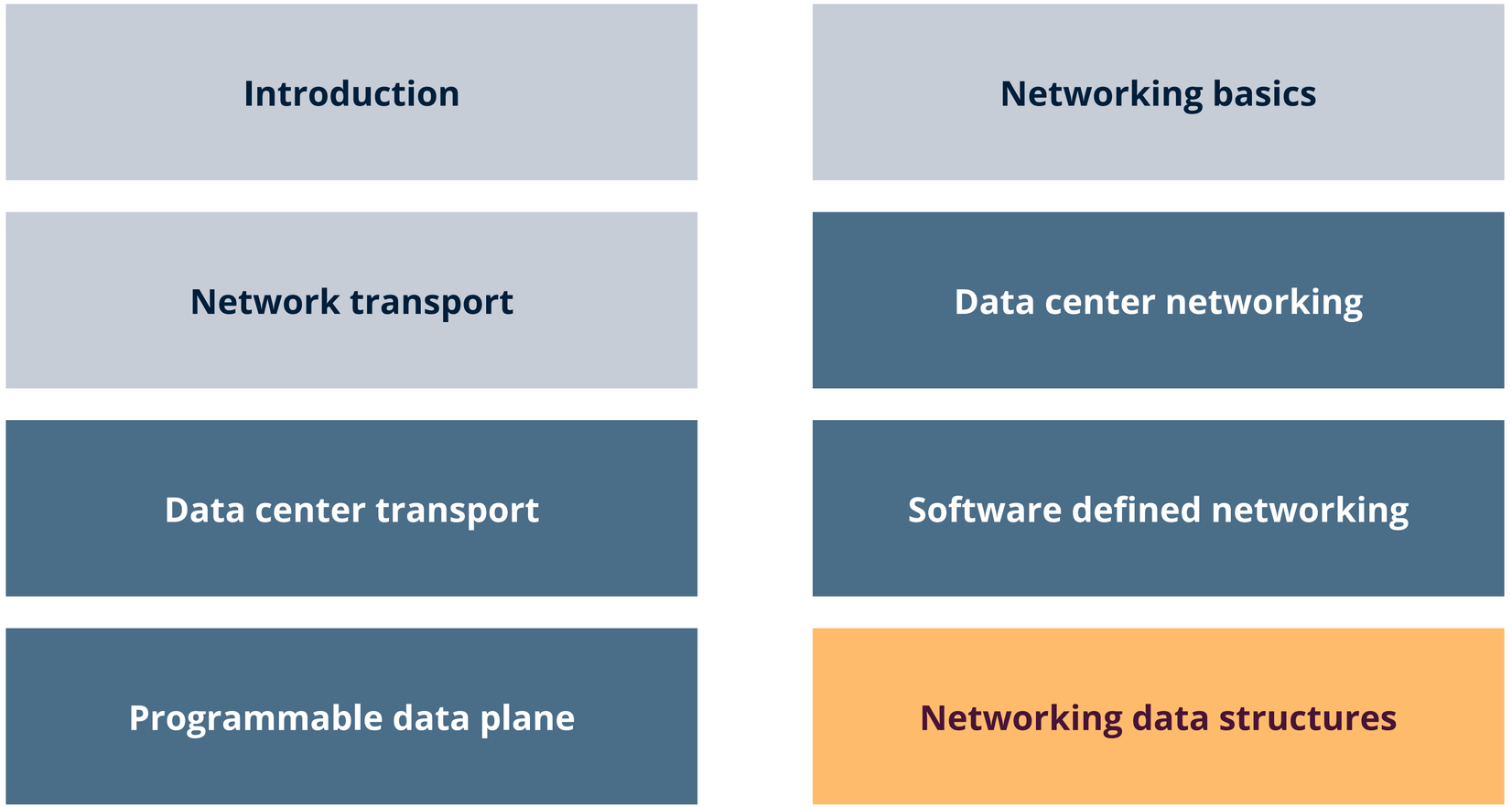
In-network computing

When the network becomes the computer!



Example services: aggregation, caching, agreement, DB query acceleration, machine learning…

Course structure



Course structure

Network monitoring

Programmable switch architecture

In-network computing - applications

In-network computing - hardware

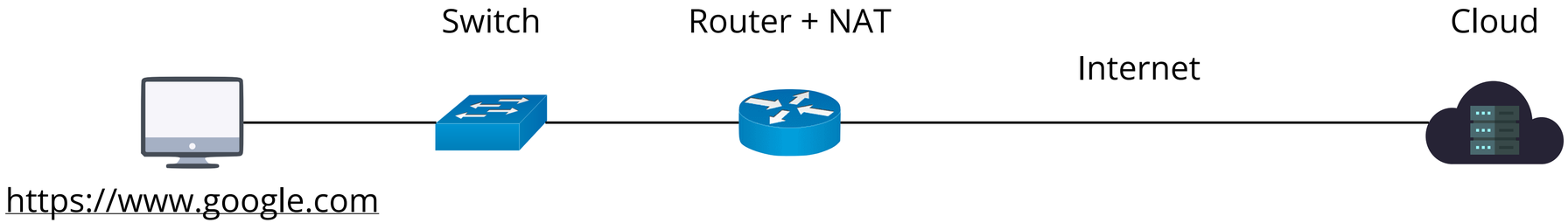
Machine learning for networking

Guest lecture by Fernando Ramos (University of Lisbon)

Exam

Next lecture: networking basics

What happens when you visit Google in your browser?



DNS, TCP, NAT, IP routing, ARP, Ethernet