Learning the Art and Science in Internet Protocol Designs

LIXIA ZHANG

UCLA

AUGUST 12, 2020

Acknowledgment

- Dr. Fleur Yano (Professor Emeritus, Cal State LA)

- David Clark, Van Jacobson

- NDN teammates: Alex Afanasyev, Jeff Burke, Kim Claffy, Patrick Crowley, Deborah Estrin, Alex Halderman, Dmitri Krioukov, Dan Massey, Christos Papadopoulos, Tarek Abdelzaher, Diana Smetters, Jim Thornton, Gene Tsudik, Lan Wang, Edmund Yeh, Beichuan Zhang
  I know I missed some names here, my sincere apology!

- All my students

- My colleagues and collaborators who helped me along the way over the past 40 years

Computer networking: where came the idea? why? how?

- Asking all the questions

- Looking for all the answers

- Discovered Paul Baran's 12-volume Rand report on packet switching

- Grasped TCP/IP basics
  - *Datagrams: simplest delivery model*
  - *Stateless*
  - *End-to-end principle*
  - *Fully distributed*
    - no dependency on any single node – a great capability

http://www.cs.ucla.edu/~lixia/papers/Baran64.pdf
http://www.cs.ucla.edu/~lixia/papers/Baran_ifip77.pdf

"Why TCP Timers Don't Work Well"

- Why I wrote this paper

- TCP timer setting at the time:

  $SRTT= \alpha \times SRTT + (1- \alpha) \times SRTT$

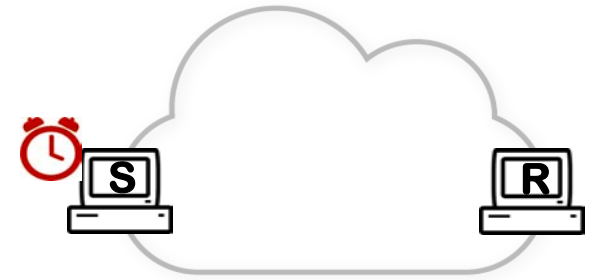 $RTO = min \{ Ubound, max (Lbound, (\beta \times SRTT) \}$

- A number of papers tried to adjust α and β in various ways

  - Jostling a dilemma

SIGCOMM 1986:
"*Why* TCP Timers Don't Work Well"

- Symptom: rxt timer goes off at wrong time

- Root cause ?
  - Multiple random variables affecting the RTT
  - *The sender had little clue what's going on*

- I learned a lot from writing the paper

1. All hosts are blind
2. Need a timer to trigger action, If not hear from the other side
3. More info exchange to sync up end state

SIGCOMM 1987: Clark, Lambert, Zhang

"NETBLT: A High Throughput Transport Protocol"

- Network scenario: high delay, high bandwidth, lossy channels

- Observation: achievable end-to-end throughput an order of magnitude lower than provided bandwidth.

- Root cause?

To see the issues clearly: considering extreme points

- What if no loss: delay doesn't matter
  - When losses exist: each end needs state to keep track things

- What if no delay: loss doesn't matter
  - Delay makes the 2 ends out of sync for at least an RTT

I sent pakcets 1-5

I resv1, 3, 5
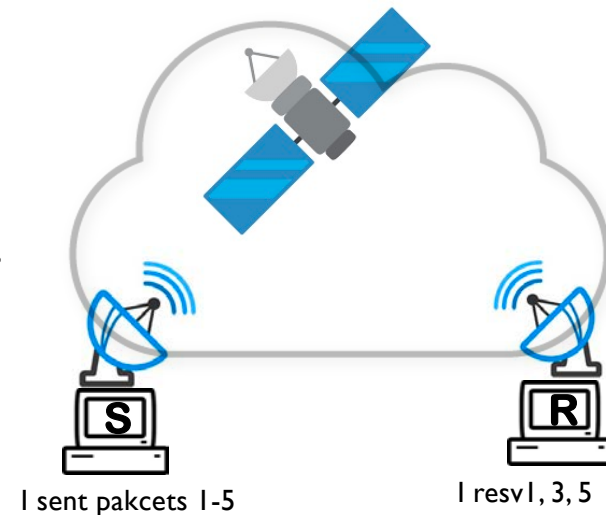
SIGCOMM 1987:
# "NETBLT: A *High Throughput* Transport Protocol"

- Must keep data transmission going while the two ends sorting out losses/retransmissions

- Let *receiver* be in control:
  - S sending at constant rate: R learns packet arrival patterns
    - delay jitter distributions, out-of-order deliveries
  - R requests missing packets, S inserts them into the outgoing queue

- I learned
  - Consider extreme points → expose insight into the problem
  - Let the receiver be in control
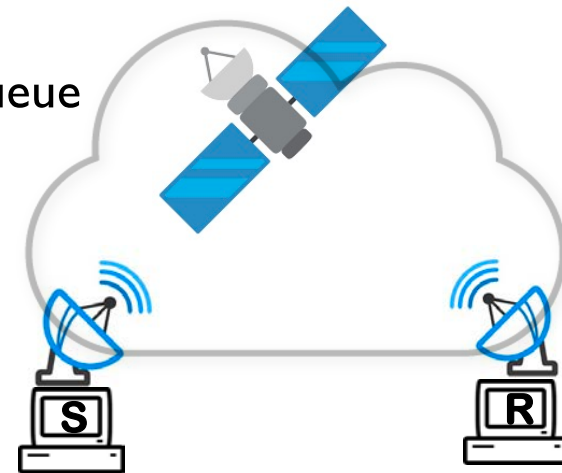
RSVP: A New Resource ReserVation Protocol

- Network needs state *(not the end state)*

- Receiver-driven state setup

- 2-way signaling message exchange
  - close the feedback loop

- A generic messenger carrying "a bag of bits" to pass to routers along the way

- Support both unicast & multicast
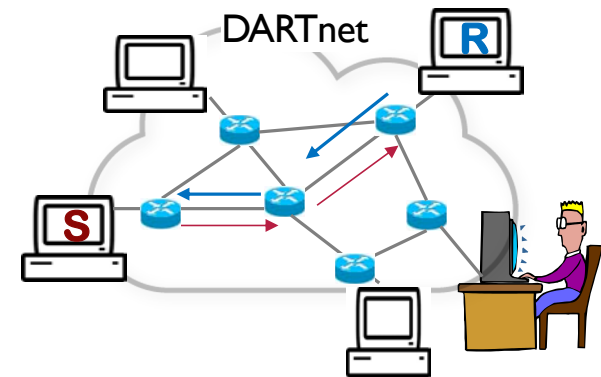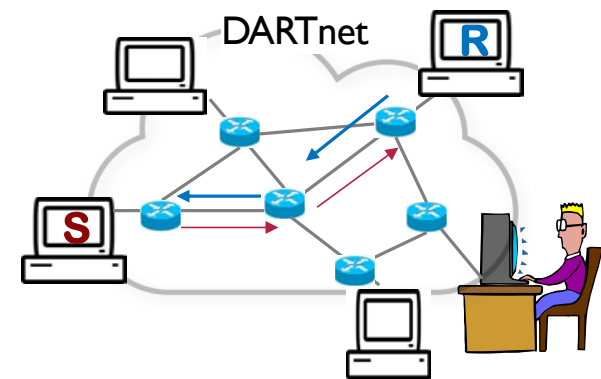
RSVP: A New Resource ReserVation Protocol

- ## What I learned from the RSVP development
  - build *feedback loop* into the protocol
  - "less is more"

The protocol moved on with a life of its own . . .

Landmark **10** *Articles*

RSVP: A NEW RESOURCE RESERVATION PROTOCOL

Lixia Zhang, Stephen Deering, Deborah Estrin, Scott Shenker, and Daniel Zappala
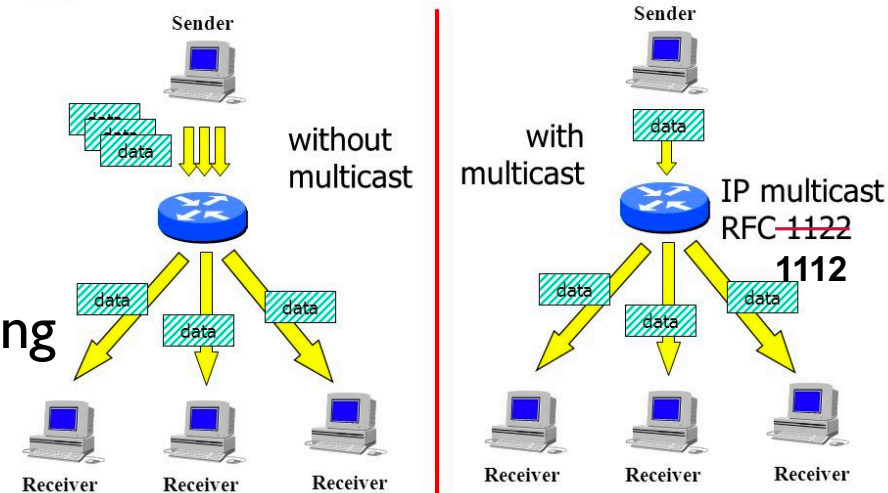
DARTnet

R

S

IP Multicast ...

SIGCOMM 1988: Steve Deering
"Multicast Routing in Internetworks and Extended LANs"

- Great idea
  - Efficient data dissemination
  - Information discovery
  - *I don't get anything that I did not ask for*
- RFC1112: Host Extensions for IP Multicasting
  - Let hosts choose to join multicast group –
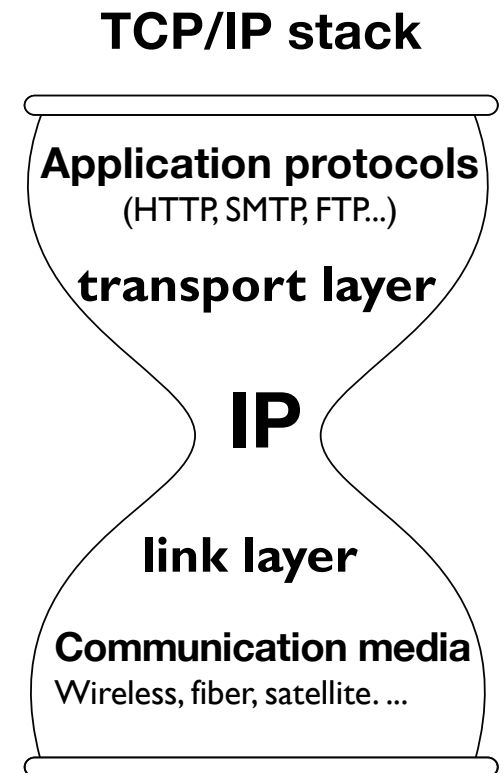   *receiver pull* (remember NETBLT)

How to use IP multicast to enable new applications?

SIGCOMM 1990: Clark and Tennenhouse
"Architectural Considerations for a New Generation of Protocols"

How to make host process packets faster ?

- The existing protocol stack: every protocol layer
  - puts on it own header
  - runs its own processing
  - → Moving packets in and out memory multiple times
- *Application Layer Framing (ALF):*
  - package data into application data units (ADU)
  - enable integrated layer processing (ILP)

TCP/IP stack

Application protocols
(HTTP, SMTP, FTP...)

transport layer

IP

link layer

Communication media
Wireless, fiber, satellite. ...

SIGCOMM 1995: Floyd, Jacobson, McCanne, Liu, Zhang A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing

(this & next slides directly borrowed from Van's 1995 SIGCOMM presentation)

- Since 1991, we have been trying to elaborate the ALF model.

- One piece we've developed is a scalable, reliable multicast framework, **SRM**. It is fully decentralized (no ring or central controller) and handles arbitrarily large groups.

- A complete protocol using the framework has been implemented in the LBL whiteboard tool, *wb,* and tested on the MBone.
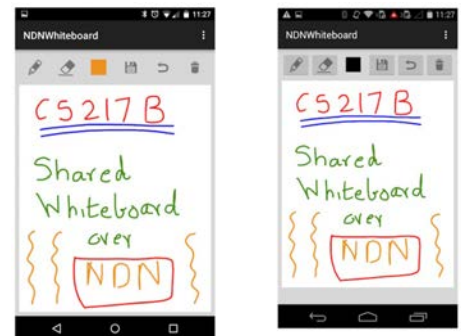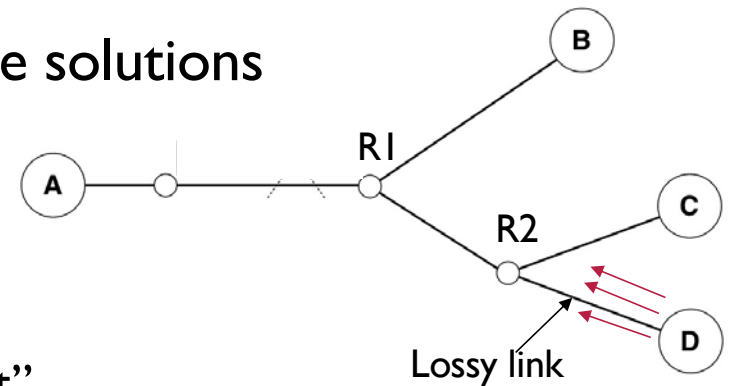
SRM Reliability Machinery

- *All* traffic is multicast.

- Anyone can send if have data

- All members send low-rate 'reports' that contain their current state (of the data reception)
  - exchange info to sync up end state

- Receivers learn they're missing data either from hole in sequence space or from someone's report.
  - don't rely on timeout

- Receivers multicast a 'repair request' to ask for missing data (after random delay)
  - receiver controls loss recovery; eliminating ACK implosion

- *Anyone that has data can reply*, not just original source of data
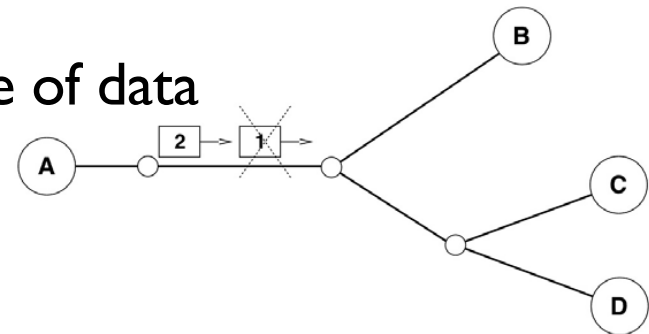
Solved one problem, more remaining

- What about the "crying baby" problem ?
- Tried hard with local recovery, did not find simple solutions
- Symptom:
  - hard to control where D's "repair request" goes

- Root cause ?
  - multicast address ≠ "looking for a specific data packet"
  - R2 saw the packets earlier too, why can't R2 help?
    - Seeing the address; not knowing the content

Lossy link

"Local Error Recovery in SRM: Comparison of Two Approaches", C. Liu, D. Estrin, S. Shenker, L. Zhang, IEEE/ACM Transactions on Networking, December 1998

SIGCOMM 1995 Middleware Workshop
"How to Kill the Internet" — Van Jacobson

http://conferences.sigcomm.org/sigcomm/1995/workshop/agenda.html

http://www.root.org/ip-development/talks/vj-webflame.pdf.gz

- The Internet was designed to survive a nuclear war.

- It lives up to its design.

- How might you kill it?

Easy — Invent the Web.

- Web traffic is destroying the Internet.

- With 25 years of Internet experience, we've learned exactly one way to deal with exponential growth:

Caching

- Data has to find 'local' sources near consumers *rather than always coming from the place it was originally produced.*

  recall seeing the same statement 3 slides back

- For any source of popular data, want a cache distribution tree rooted at that source with leaves near everyone who wants the data.

- *Cache hierarchy should be self-configuring and adaptive.* i.e., use multicast.

- Using multicast implies that *we stop thinking of communication as 'conversations':*

  Instead of asking X to send you Y, simply ask for Y.

Adaptive Web Caching

Principle Investigators:

Lixia Zhang

Sally Floyd

Van Jacobson

DARPA Networking PI Meeting

March 13, 1998

Designing a scalable & robust caching system

❑ Follow the example of a scalable and robust delivery system: IP.
❑ The Internet delivery model
  – routing packets to their destinations
  – global data delivery through *transitive* IP connectivity
❑ Adaptive web caching
  – routing URL requests towards data loci
  – global data dissemination through *transitive* caches

What happened next

- ## We gave talks
  - A few by Sally Floyd still hanging at https://ee.lbl.gov/nrg-talks.html

- ## We wrote papers
  - "Adaptive web caching: towards a new global caching architecture". Michel, Nguyen, Rosenstein, Zhang, Floyd, Jacobson, Computer Networks & ISDN Systems, November 1998.
  - "URL Forwarding and Compression in Adaptive Web Caching", Michel, Nikoloudakis, Reiher,, Zhang, INFOCOM, March 2000.

- ## We built demoware

- ## The students even pulled off a startup and got big venture funding

- ## The design ran into a stone wall

Hard Problems

- Wanted a generic cache overlay running on IP to support all web traffic
  - not web proxies that need config and serve local users only
  - not CDNs that only serving paying customers
- Our difficulties:
  - Hard to make caches interconnect to form a global overlay
    - easy if they were routers
  - Hard to suck browser queries to a nearby cache
    - easy if caches were just local routers
  - Hard for caches to figure out where individual contents were
    - easy if caches just run a routing protocol (of URL prefixes)
- Root cause? We don't network by content names **networking by content names ?**
  - *If we do*, that'd address all the SRM's problems as well
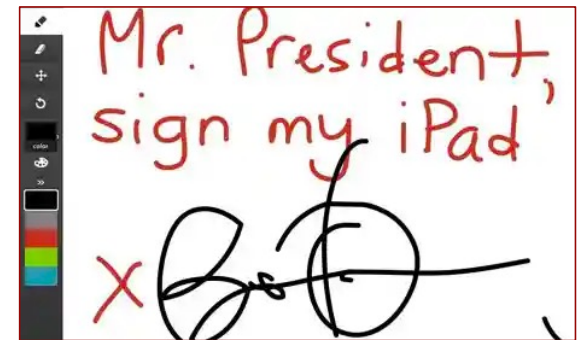
https://www.youtube.com/watch?v=oCZMoY3q2uM
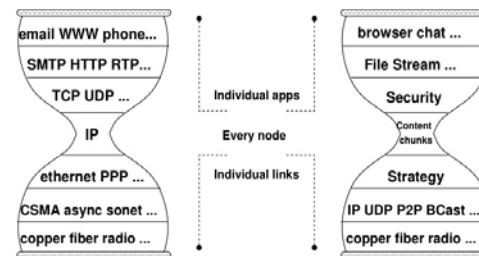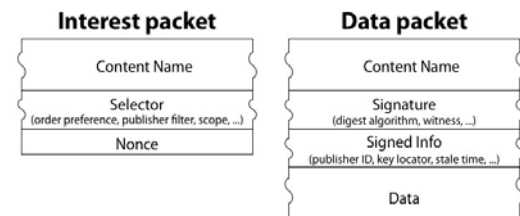
The new way: Named Data Networking (NDN)

- IP delivers packets to numeric IP addresses

2.3.4.5   22.3.7.9

- NDN fetches data by application data object names

INTEREST → DATA
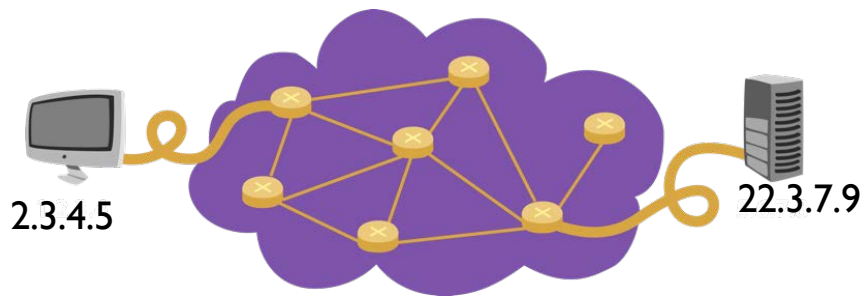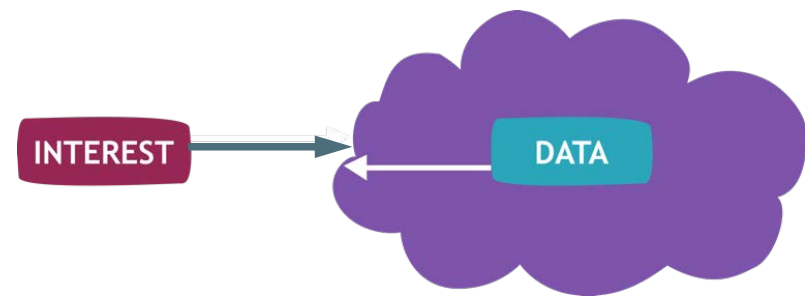
⬦ Example data name:
  ○ www.youtube.com/watch?v=oCZMoY3q2uM
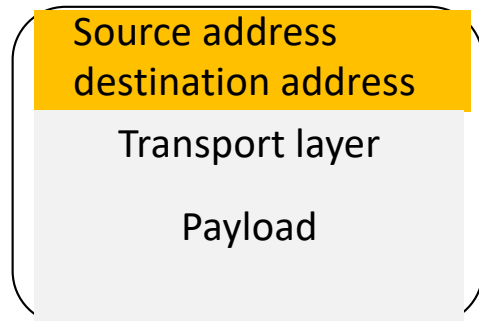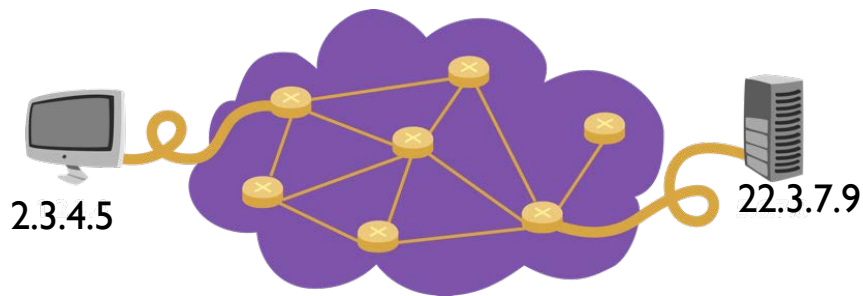 ○ https://en.wikipedia.org/wiki/Van_Jacobson

⬦ Large objects segmented to multiple packets
  ○ each segment uniquely named

The new way: Named Data Networking (NDN)

- IP delivers packets to hosts based on numeric IP addresses

2.3.4.5   22.3.7.9

| Source address destination address |
| Transport layer |
| Payload |

IP packet

- NDN fetches data by application data object names
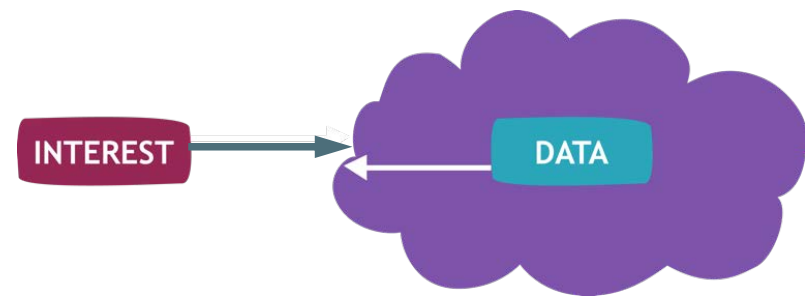
INTEREST → DATA

| Application data name |
| (a few optional parameters) |

Consumer requests data by sending **Interest packet**

| Application data name |
| (a few optional metadata) |
| data |
| **crypto signature** |

Producer binds name to content to create **Data packet**

How an NDN network delivers data

An IP Router

2.3.4.5   22.3.7.9

MPLS,
VPN,
Multicast,
and other
patch-ons

Routing protocol

Forward Information Base (FIB)

synchronous, point-to-point communication

An NDN node

INTEREST   DATA

Forwarding Strategy

Multiple means to fill in FIB

content Store

Pending Interest Table

FIB

(a)synchronous data dissemination, with security, storage, feedback built into the narrow waist
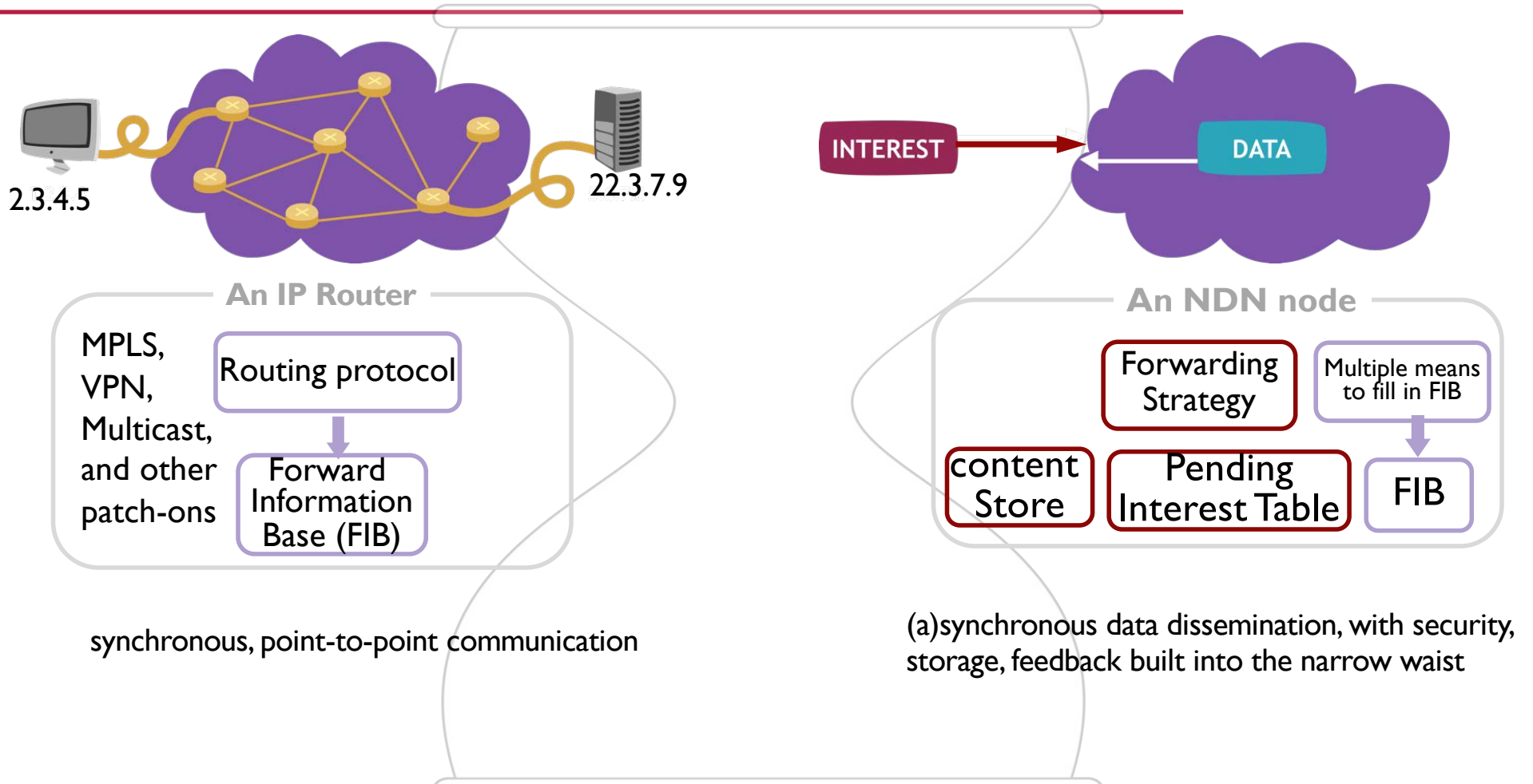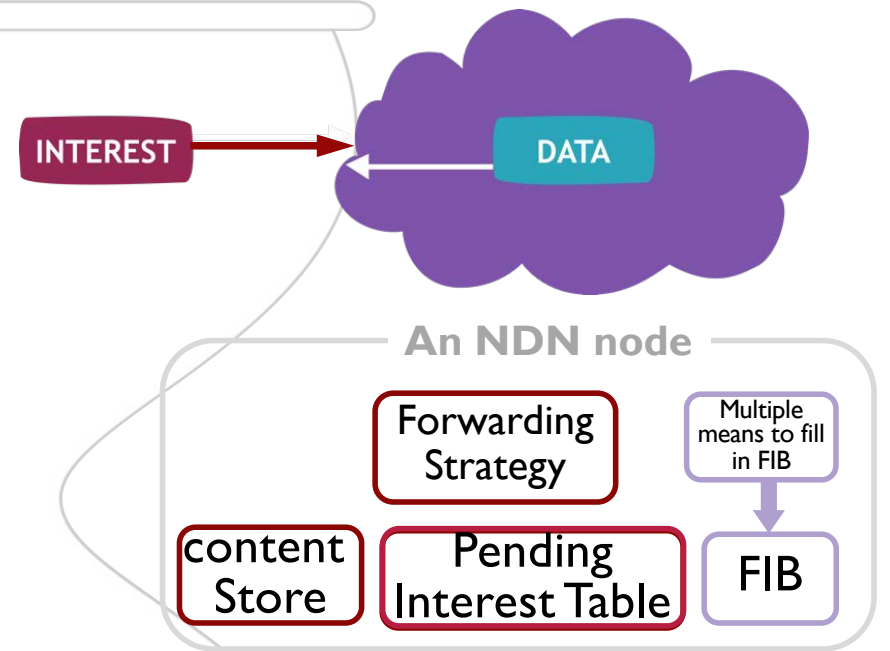
How an NDN network delivers data

- "*Anyone that has data can reply*"
  - Wire and storage treated the same

- Data is secured

- Stateful data plane
  - Hop-by-hop soft datagram state
  - Enabling
    - Multicast delivery
    - Multipath forwarding
    - Flow balance   } Enabled by
    - Congestion control   interest/data
    - Fast adaptation to failures exchange feedback

Yes the PIT and FIB sizes can get large
"A Note on Routing Scalability in Named Data Networking"
http://web.cs.ucla.edu/~lixia/papers/2019ICC-NDNRouting.pdf

INTEREST → DATA ←

An NDN node

Forwarding Strategy

Multiple means to fill in FIB

content Store | Pending Interest Table | FIB

(a)synchronous data dissemination, with security, storage, feedback built into the narrow waist

Securing data directly → end-to-end security

Data producers sign

Data consumers verify
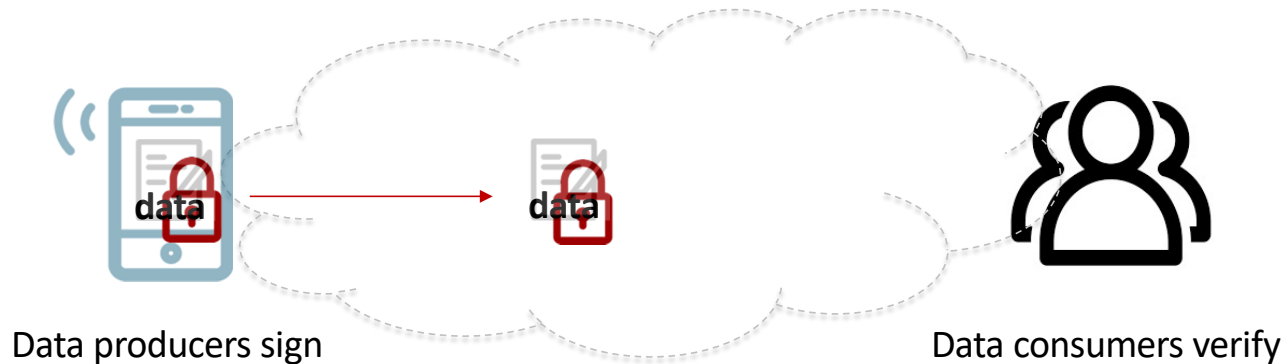
End-to-end data authenticity, also confidentiality if encrypted in face of
  ▷ scalability challenges (unbound volume of requests)
  ▷ connectivity challenges (intermittent, ad hoc, DTN)
  independent from the security (or lack) of intermediate hops

INFORMATION-CENTRIC NETWORKING SECURITY

An Overview of Security Support in
Named Data Networking

THE ESSENCE OF NETWORKING

- Delivering bags of bits for applications
  - secure delivery between trusted entities

- The fundamental question: what namespace to use for that delivery

NDN IS MADE OF 3 SIMPLE IDEAS

- Fetching *application-named* data

- Securing data directly

- Stateful forwarding plane

ADHERE TO/EXTEND IP PRINCIPLES

- *Datagrams: simplest delivery model*

- *End-to-end principle*

- *Fully distributed*
  a great capability

Can a new network architecture ever get rolled out?

- "Look at IP6 rollout: how difficult it has been"
  - Even though it simply a new version of IP

- Two fundamentally different ways of rolling out new protocols
  - Replacement: IPv4 → IPv6
  - Starting anew, ROTT: telephony → IP

Learn from history: Telephony → IP

- TCP/IP in mid 70-80's: promising new technology, largely unknown outside its small community

- Started from the edge, *driven by its own apps*

- Uses any/all available connectivity to deliver IP packets
  - Phone dialup, ALOHA, Ethernet: all the same

- Grew big on top of telephony, but independent from it
  - Telephony went off the stage on its own

An interview with https://www.usenix.org/system/files/login/articles/login_spring16_08_zhang.pdf , spring 2016

The road to a new architecture

- Application-driven architecture development
  - Running code, useful applications
  - tackling emerging environments/applications where IP-based solutions faced challenges
- Open source development, global community engagement
  - NDN codebase: https://github.com/named-data
  - NDN tutorial page: https://named-data.net/publications/tutorials/
- Grassroots growth *from the edge*, the same way as IP did

A number of NDN apps
(demoware) have been developed

Decentralized and Secure Multimedia Sharing Application over Named Data Networking

NpChat, a Multimedia Sharing Application over NDN

https://medium.com/@ritikk/npchat-604663a7047d

In the world dominated by big internet players like Google, Facebook, etc., most of our day to day internet traffic is routed to their servers. Virtually everything from E-commerce, Social Media, Web Streaming are increasingly controlled by some giant corporation. ...

Such a connected world requires a *decentralised end-to-end encrypted social multimedia app*. And when looking on it from Information-centric network perspective, NpChat seems quite promising.
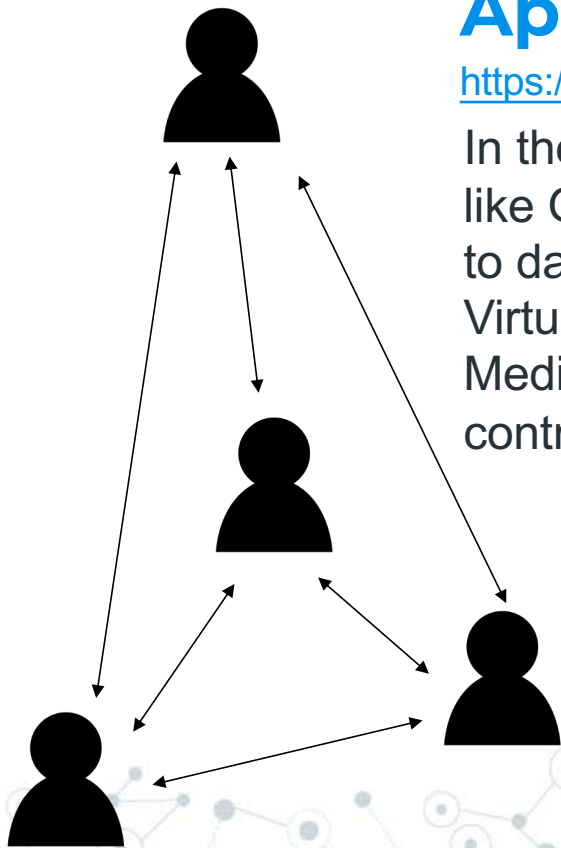
No central entity

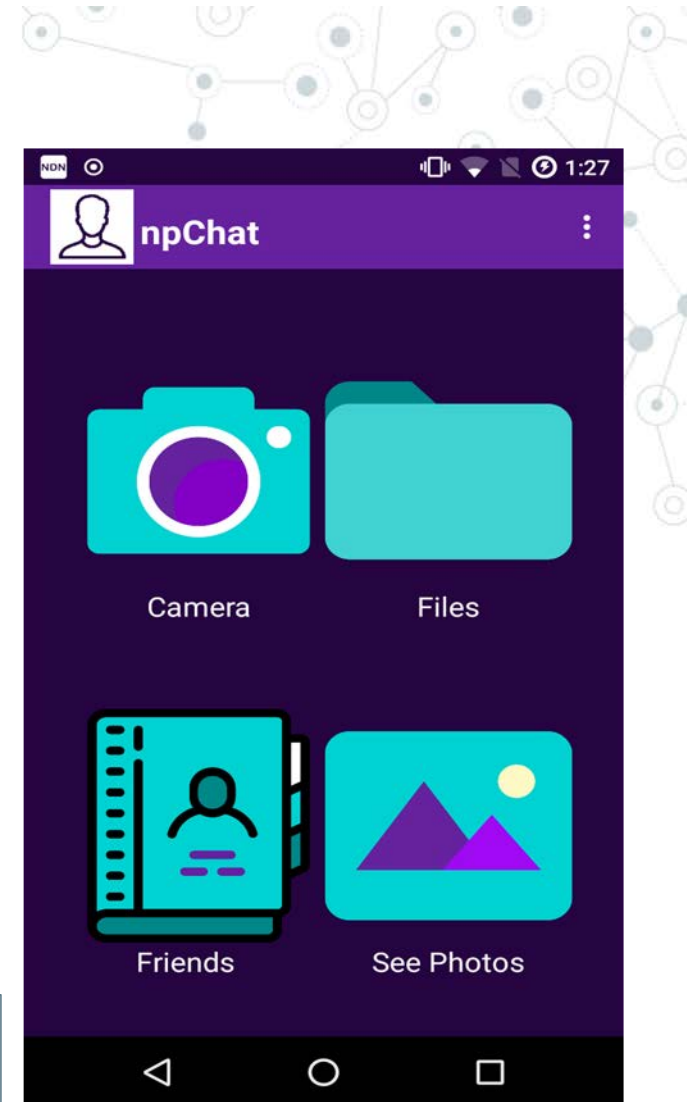No single user directory

No special infrastructure

No single trust anchor

User control of data

- Decentralized social media platform
- Blueprint for other developers

npChat

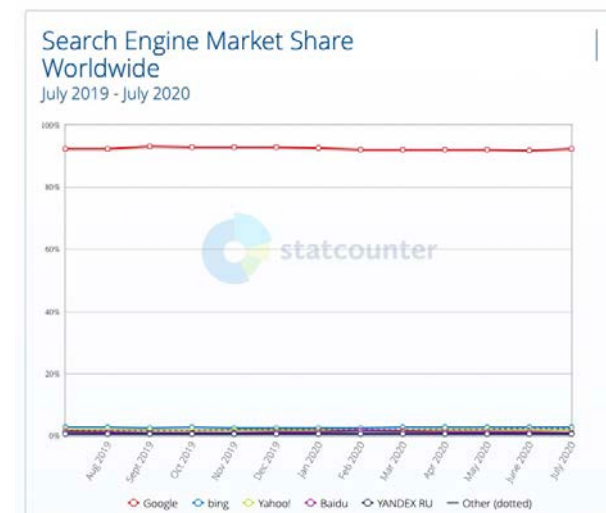Camera

Files

Friends

See Photos

Common features of NDN apps

- Peer-to-peer communication, with strong security designed in
- New use cases approach us, turn to great NDN-based solutions
- Empowering end user communities
- Facilitating distributed app development
  - can utilize, without reliance, on cloud services

As a side note:

"On Cyber Governance" by Geoff Huston

- .....

- It's truly amazing that the sum of human knowledge is at my fingertips, instantly accessible from anywhere at any time. That's incredibly empowering.

- It's truly frightening that all this information is only accessible through a single entity, who funds this service through an insidious economy based on surveillance capitalism.

Search Engine Market Share Worldwide
July 2019 - July 2020

https://gs.statcounter.com/search-engine-market-share

https://www.potaroo.net/ispcol/2020-08/cgov.html

Summing up:
Learning the Art and Science in Internet Protocol Designs

"Protocol Design Arts"

Collecting a bag of tricks/ideas

- Use timers; use it right
- Soft-state (timers in disguise)
- Feedback loop
- Very often less is more
- slow-start; randomization (and a few others that I didn't touch on during this talk)
- Find the root cause

Is networking research a scientific undertaking?

- Physics:
  - Nature/God made the world
 - Physicists interpret how it works

- Networking:
  - No one built the Internet for us; we did
  - There is not a bible to tell us how to do it either
  - design $\Rightarrow$ implement $\Rightarrow$ experiment $\Rightarrow$ draw lessons from practice
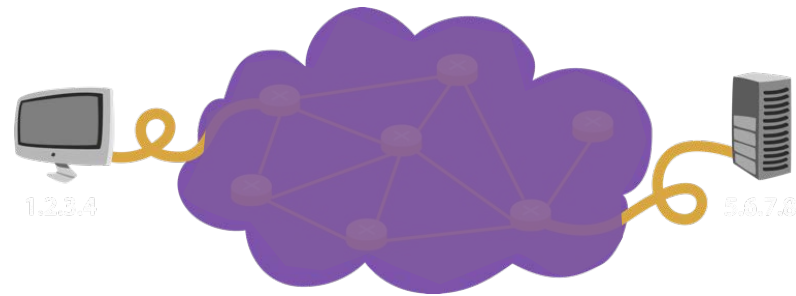
    - Going through this cycle frequently for incremental improvements
    - Going through this cycle *infrequently* for architectural changes
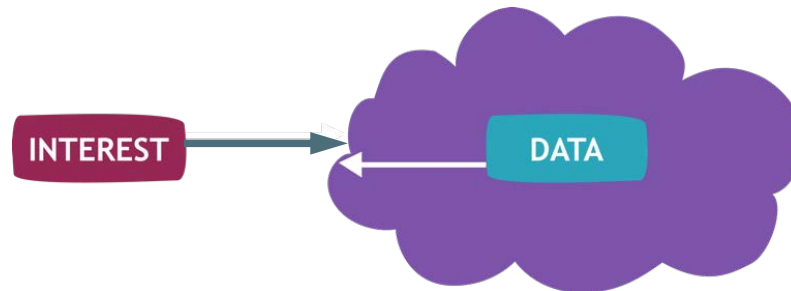
punctuated
equilibrium

The future of networking lies in recognizing the right *communication abstraction*
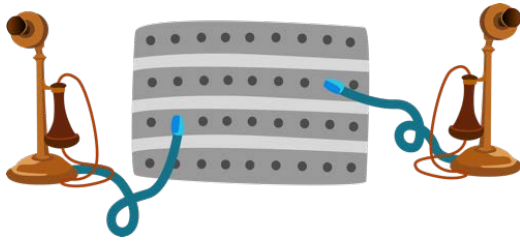
Telephone Network: Focused on building *wires between 2 ends*

Internet Protocol (RFC791): Focused on delivering packets between 2 *nodes*

INTEREST → DATA

NDN: Focuses on fetching *named, secured data*