ACM ICN 2022 OSAKA

NAMED DATA NETWORKING

Tutorial:

NDN Evaluation Tools: ndnSIM and Mini-NDN

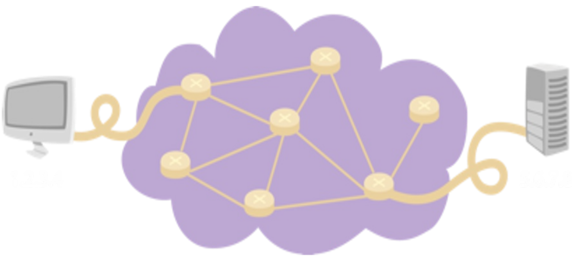
Alexander Afanasyev[1], Saurab Dulal[2], Varun Patil[3], Adam Thieme[2], and Lixia Zhang[3]

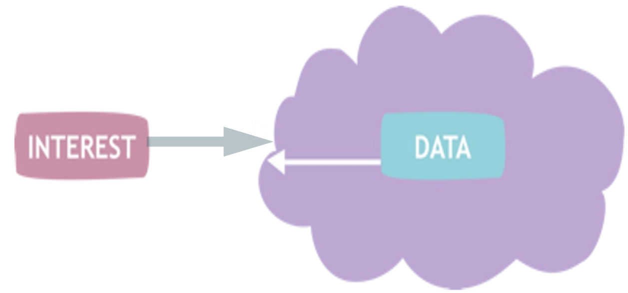
[1]Florida International University
[2]University of Memphis
[3]UCLA

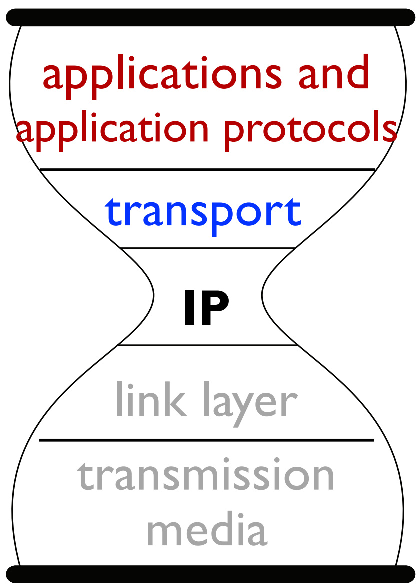
To evaluate a network architecture

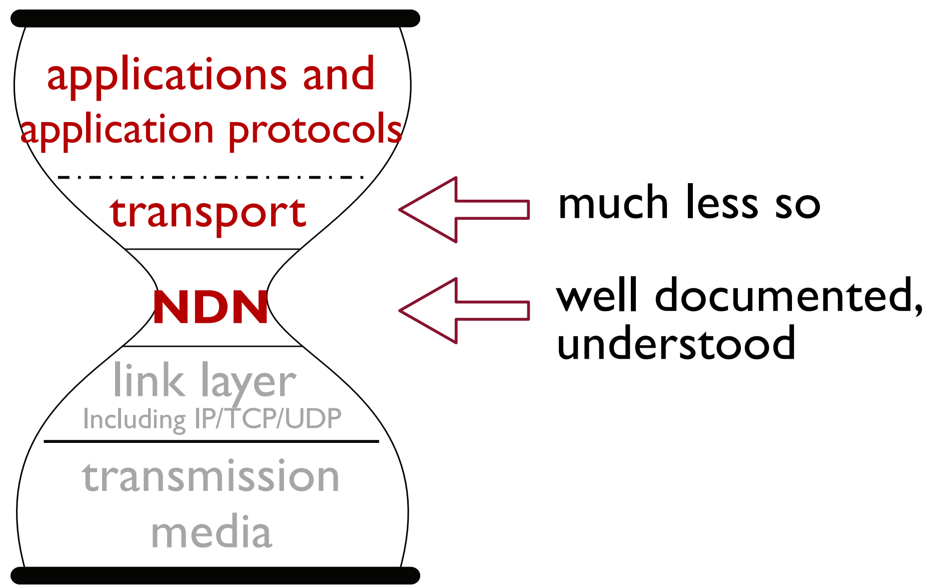
1. A clear understanding of how the architecture works
2. Figuring out what performance metrics to measure, and how to measure

From IP to NDN: a *conceptually* simple change





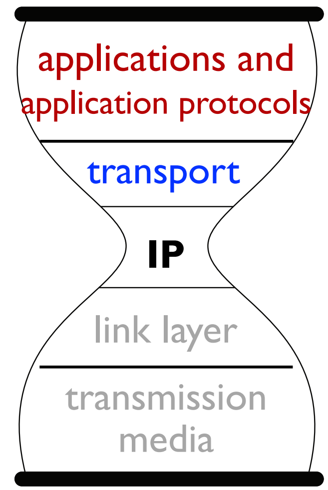




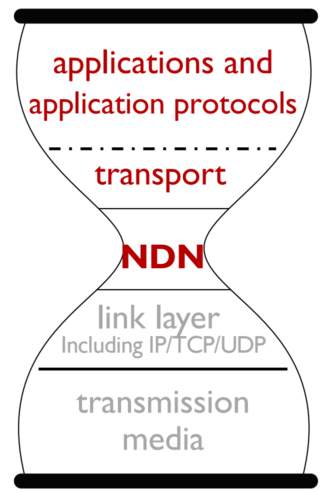
What transport service does

- In today's TCP/IP Internet
  - Demultiplexing (using port #)
  - Congestion control (leverage transport's 2-way packet exchange)
  - Reliable data delivery

  All the above are based on *point-to-point* IP connectivity

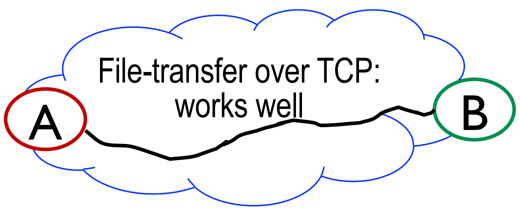


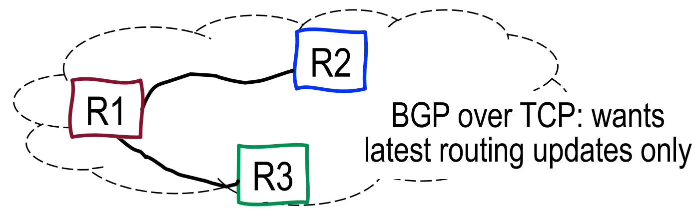
- In an NDN network
  - Demultiplexing: done by names
  - Congestion control: responsibility of network layer
  - Reliable data delivery: remains as the transport function
    - Apps just want the service
    - Network does not want the job



Supporting reliable data delivery: a challenging task

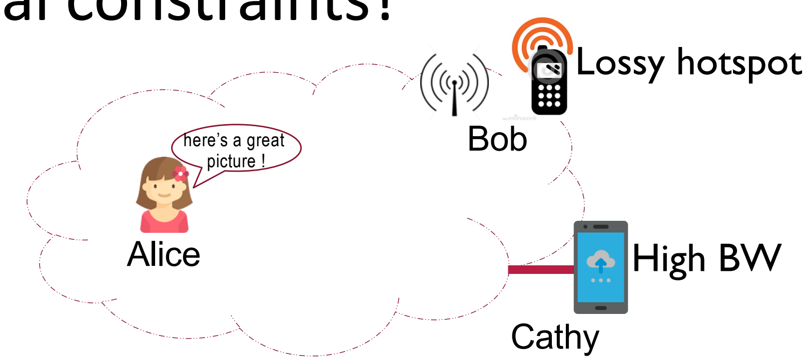
- Can one transport service support apps of different reliable delivery requirements?
  - Lessons learned: TCP's one size (reliable byte-stream) does not fit all





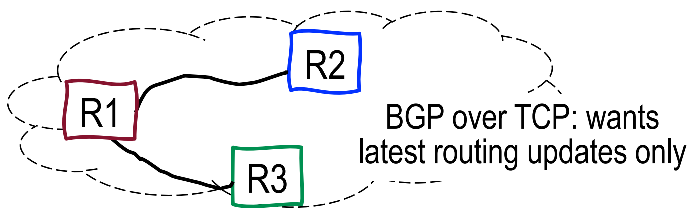
R3 got all updates from R1; R2 missed update-2 when R1 gets update-3...

- Can one transport service tailor data delivery for multiparty apps with each party having different local constraints?

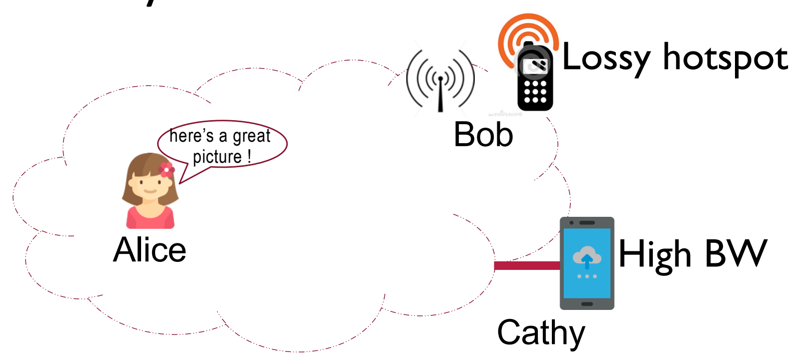


Addressing the challenge: one mechanism fits all

- Let data consumers fetch desired data
  - Higher layers decide whether to fetch, and which piece most urgent
  - Lower layers decide when/what's the best way to fetch



R2 can just fetch the latest updates, even if an earlier one got lost

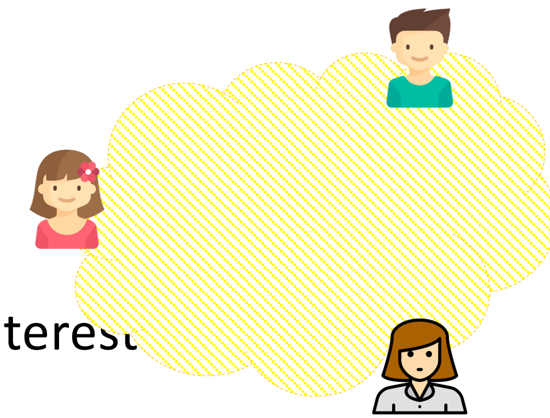


Cathy fetches Alice' picture right away; Bob will fetch only if he gets better connectivity quick, otherwise forget it.

- There is no silver bullet ...
  - how can Cathy and Bob know Alice produces new data?
    - Once learned the names of all data, one can fetch desired data by names at the right time
      - If it is something urgent, Bob could also request a low resolution version...

NDN transport: Synchronizing dataset names

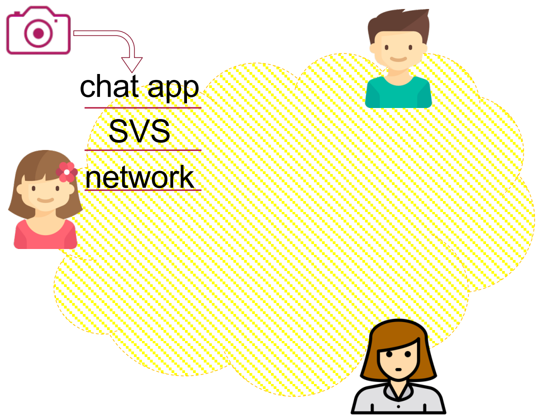
- State Vector Sync (SVS)
  - The latest sync protocol after 10+ earlier designs; used in this tutorial
  - See paper "SoK: The evolution of distributed dataset synchronization solutions in NDN" for more details
- Basic idea:
  - Alice, Bob & Cathy running a distributed app
    - All join a Sync group
  - Whoever produces a piece of data: send Sync interest to inform the others
    - Whoever wants the data: fetch by the data name



- Protocol spec:
  https://named-data.net/publications/techreports/ndn-0073-r1-svs/
- Code: https://github.com/named-data/ndn-svs

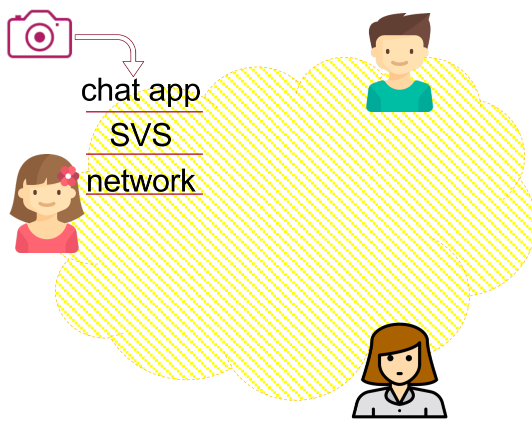
An example app: chatroom

- Everyone in the same chatroom joins a Sync group
- Alice produces a picture, inputs to chat app
  - The app passes the image data to SVS
- SVS informs the group of the new data
  - State-vector: Alice:1; Bob: 0; Cathy:0
  - Sync Interest name: /chat/friends/state-vector/
  - Sync Interest is multicast to the group; no data reply
- Cathy sends an interest to fetch the picture



Not shown in the previous picture: NDN has security built-in

- When starting an NDN-based app: each participant goes through a bootstrapping step
  - Obtain its name in the app, associated certificate, the app's trust anchor, and security policies



- App: source code authentication via software distribution channel
- Alice starts a chatroom, sets trust anchor and policies
- Alice invites Bob and Cathy to join
  - Issuing each a certificate signed by the chatroom's trust anchor, passing the app's trust anchor and security policies

See "Enabling Plug-n-Play in Named Data Networking" for more details
http://web.cs.ucla.edu/~lixia/papers/2121MilcomPnP-paper.pdf

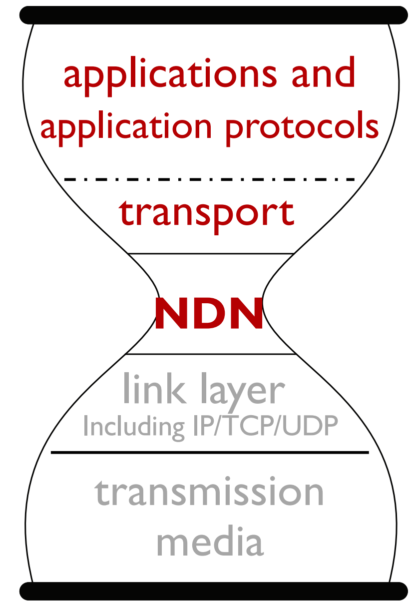
Disclaimer: neither ndnSIM nor Mini-NDN can be used to evaluate NDN security solutions 😣

Encourage everyone to develop innovative security solutions by playing with/developing NDN apps!

https://named-data.net/codebase/platform/

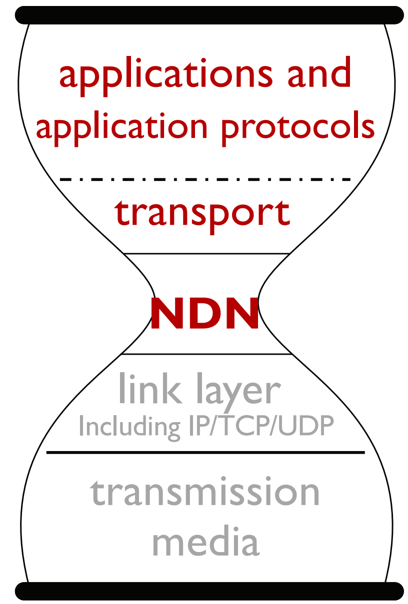
What the current tools can help: performance evaluation

- Measure transport performance: *dataset namespace update delay*
  - Sync Interests can be lost
  - time between data production and reception by each node in the group
- Measure application performance:
  - Fetching delay: the time period from sending an interest packet till the requested data packet received
  - Throughput



Measuring network performance

- The number of Interest and Data packets sent
  - By each node
  - Over each link
- # of hops Interest (and data) packets traverses
  - Reflection of interest aggregation and data caching
- Pening Interest Table
  - Size
  - PIT entry life time distribution
  - Satisfied / unsatisfied Interest



- Packet queueing delay: effectiveness of congestion control
  - “Effective NDN congestion control based on queue size feedback”@ICN’22
    - The reference list shows many existing works on this topic

Measuring performance: what about caching?

- Cache hit-ratio[1]
  - Per node
  - Per namespace
- Cache space occupancy distribution?
  - If the cache management design pre-emptively remove cached content

[1] Isn't this largely depending on the topology and traffic patterns?

Let the real tutorial begin: next: ndnSIM