Travels with ICN

The road traversed and the road ahead

Dave Oran
Network Systems Research & Design / MIT

NETWORK SYSTEMS RESEARCH & DESIGN

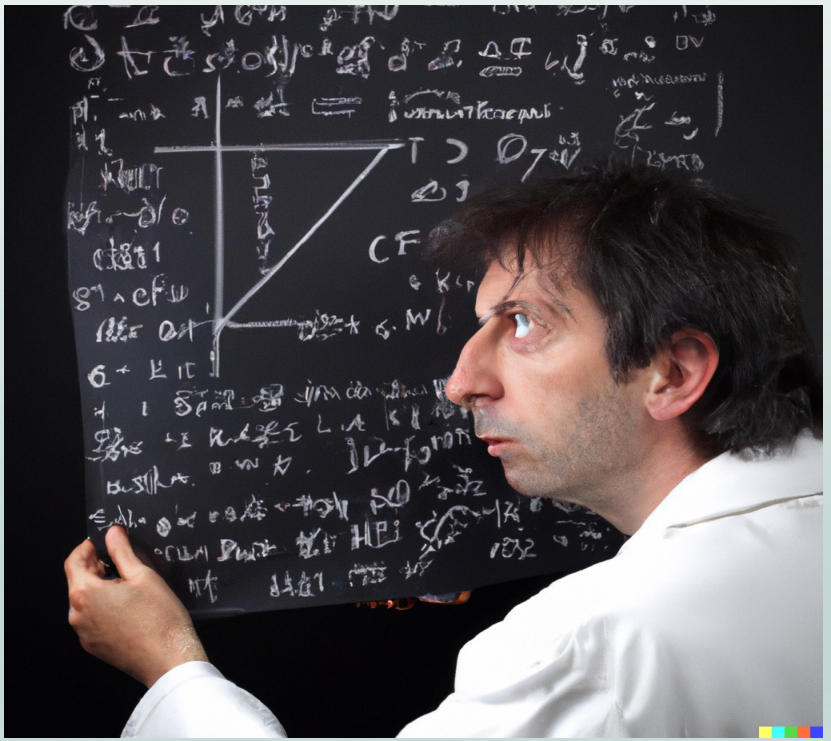
September 20, 2022
ICN 2022, Osaka Japan

Some Historical context – I've been around a while

Late 1960's	Satellite downlink data scrubbing	
Early 1970's	Banking Teller system + National Lottery (México)	
Late 1970's	DECnet + SNA Gateways	
Early 1980s	Internet Routing Protocols	
Late 1980s	RPC + Distributed Naming Services (OSF DCE)	
Early 1990s	Wireless LANs & Distributed File System	
1996-2006	VoIP	
2004-2010	IPTV	
2010-2016	Adaptive Internet Streaming	

Why did I get interested in ICN (2011±)

- Potentially disruptive to the Networking business
- Opportunity to revisit hard problems in networking from different set of assumptions
- "Crazy" to keep per packet state in the network …but… memory is a lot cheaper than it used to be



So, what did I learn (giving away the punchline)

Information-centric Networking

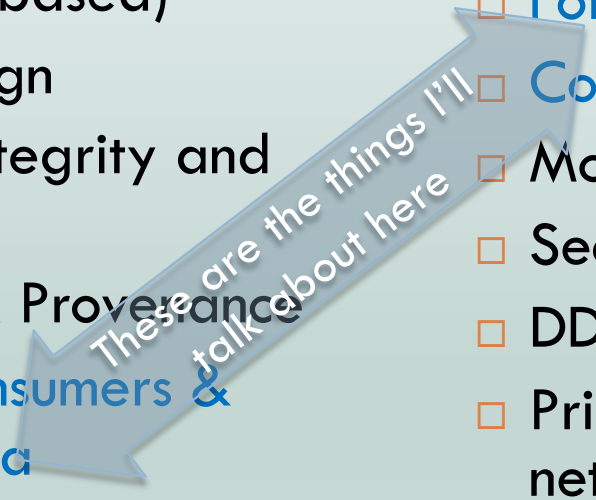
≠

Information-centric **Networking**

Different Emphases

Information-centric

- Naming forms (hierarchy, graph, attribute-based)
- Namespace design
- Named object integrity and confidentiality
- Trust schemata & Provenance
- Interaction of consumers & producers of data



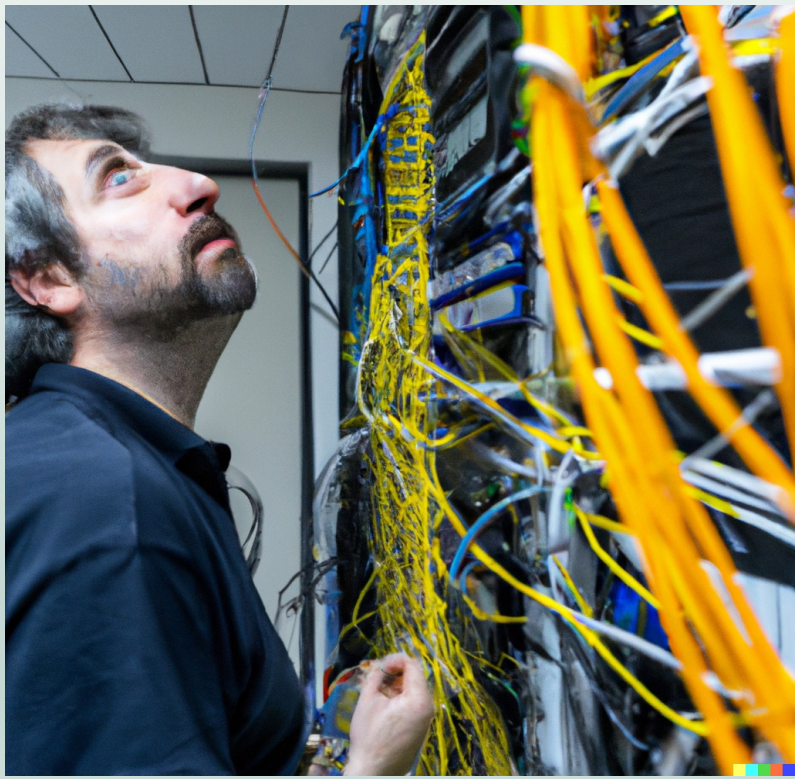
Networking

- Routing
- Forwarding
- Congestion Control
- Mobility
- Security of network devices
- DDoS prevention/mitigation
- Privacy against surveillance of network traffic

Routing

Name-based Routing is Seductive!

- Finally – location-independent routing using content names!
- Routers have lots more bits in the packet format to play with!
- But…
  - hierarchy doesn't help if names don't follow the network topology
  - …and can we route on attributes? Or graph names?
  - Routing protocols scale as {State x Rate}



We've tried lots of things for Name-based routing

- Link-State IGPs (e.g. NLSR)
- Multi-ring DHTs – (e.g. Liu, Foy, Zhang in ICN 2012)
- Bloom Filters – (e.g. Braun, Salamatian, Thomos in CCNC 2018)
- On-demand calculation – (e.g. Ascigil, Psaras, Pavlou) in ICN 2018)
- …bunch of others
- But none of them have achieved:
  - Internet wide scale
  - Multi- AS operation
- So far, we've wound up punting to a Name Translation service
  - Fall back to DNS (sigh…)
  - Try a translation service that claims to scale (e.g, GNS of Mobility First)
  - Use Manifests containing multiple topological names

Is there a lesson here?

Lesson 1: After 40+ years, no magic bullet to escape {State x Rate} scaling for routing

Lesson 2: All is not lost – lots of interesting applications don't need internet scale routing, but

Lesson 3: Few (any?) applications can get away with single AS administrative scope

Forwarding

The early approach to forwarding in CCNx & NDN

- Rich functionality was believed to be needed to demonstrate superiority of an L3 approach to ICN, for example:
  - Prefix matching against cached data allows content discovery to be "built in" to the base protocol
  - Selectors allow consumers to flexibly traverse collections
  - Exclusions needed to bypass data you really wish hadn't wound up in the caches
- But don't worry (yet) about performance, because…
- Doing premature optimizations to get performance risks freezing the architecture too soon

I didn't believe this

- Making things go fast is interesting by itself
- You can't find "big O" problems if you don't care about performance
- Good to know what the bottlenecks are right away if you have very different resource tradeoffs (CPU, memory, bandwidth, latency) from conventional protocols

So we took a real router and made ICN work

- Ran on Service Card of Cisco 9K Router
- We got ~1.3M PPS, and 20 Gbps
  - …not bad for mostly software in 2013
- We learned a whole bunch of interesting things (next slides)
- A number of the key algorithms were adopted in later efforts (e.g. NIST forwarder)

Some things we learned about forwarding, Part 1

- Router hardware was (is?) optimized for moving packets across busses/crossbars fast, not doing lots of computation or mutating state
- Memory bandwidth was the bottleneck, and seems to be the *persistent* bottleneck for this kind of forwarding across generations of hardware
- Algorithm selection the single biggest effect (yes, big O again!)
  - Hashing won over TRIES and other popular FIB approaches
  - Some fast hashes (e.g. CityHash64) were massively insecure, so chose a secure hash (SIPHash), which the had side benefit of being restartable!
- Clever engineering also matters
  - Data structures that allow you to use SIMD instructions
  - Mix of long names and short names required some cleverness to bound worst case while having good average case forwarding lookup performance

Some things we learned about forwarding, Part 2

- Three NDN and CCNx features were performance killers, so we gave up trying to implement them:
  - Prefix matching against CS blew memory access budget by 1-2 orders of magnitude
  - Ditto for Selectors
  - Exclusions (we didn't even try)

Lesson 1: Trying to go fast can inform protocol design even in early stages

Lesson 2: Remember that network routers/switches are not servers even if they have general purpose CPUs and DRAM to play with

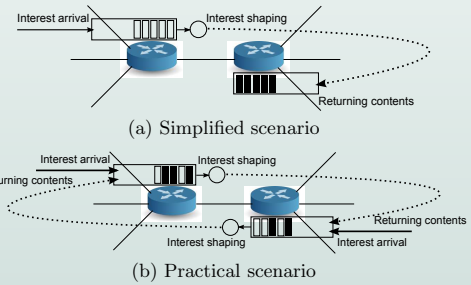
Congestion Control

Stateful forwarding makes Congestion Control *really* different from IP

- Hop-by-hop feedback
- Can get feedback at RTT to bottleneck rather than RTT end-to-end
- Not constrained by limited bits in IP header
- Symmetric routing means congestion state is also symmetric

How do we exploit these properties to get better congestion control?

- We can shape traffic hop-by-hop
  - By shaping Interests we can control rate of returning Data
  - Can get much higher link utilization without loss than IP
- We can make rate-based schemes work!
  - Room in the packets for actual rate of bottleneck, cheap to compute and propagate back in Data
  - If congested, can NACK from bottleneck rather than dropping, giving shorter feedback RTT and retransmission timeliness
- We can police Interests when overloaded since we know rate of reverse link due to symmetric forwarding



An Improved Hop-by-hop Interest Shaper for Congestion Control in Named Data Networking

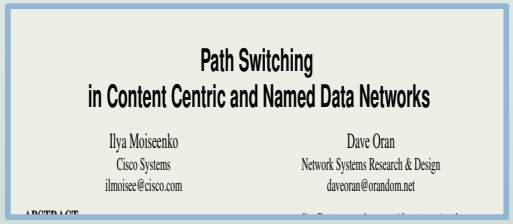
Yaogong Wang, North Carolina State University, USA, ywang15@ncsu.edu
Natalya Rozhnova, Université Pierre et Marie Curie (UPMC), France, natalya.rozhnova@lip6.fr
Ashok Narayanan, Cisco Systems, USA, ashokn@cisco.com
David Oran, Cisco Systems, USA, oran@cisco.com
Injong Rhee, North Carolina State University, USA, rhee@ncsu.edu

MIRCC: Multipath-aware ICN Rate-based Congestion Control

Milad Mahdian, Northeastern University, mmahdian@ece.neu.edu
Somaya Arianfar, Cisco Systems, sarianfa@cisco.com
Jim Gibson, Cisco Systems, gibson@cisco.com
Dave Oran, Cisco Systems, oran@cisco.com

Multipath rate based schemes practical

- We can learn multiple paths and maintain rate for each active path:
- We can steer packets explicitly onto paths
  - This allows proportional traffic splitting over paths


Path Switching
in Content Centric and Named Data Networks

Ilya Moiseenko
Cisco Systems
ilmoisee@cisco.com

Dave Oran
Network Systems Research & Design
daveoran@orandom.net

This rosy picture does have a few limitations

- From Interest arrival, you actually don't know how big a Data packet will come back, only that it will be at most one
  - Some ways to fix this, (e.g. consumer often knows and can say so in the Interest) but they haven't been tried
  - Ratio of Interest size to Data size affects efficiency – how much capacity for Interests on reverse link?
- No Magic for some universal congestion control complications:
  - End-to-end RTT uncertainty can be quite large – how pessimistic should we be? Long interest lifetimes make this even trickier.
  - If link rate changes faster than the link RTT (e.g. wireless) shaping errors will occur

Not everything we tried worked out

- Congestion signaling hop-by-hop should make in-network retransmission attractive, but experiments seem to show it causes more problems than it solves
  - Inflates RTT artificially – consumer can't tell what part of RTT is propagation, accumulated queuing delay on successful path, versus delay via retransmission retries.
  - This also could be fixed, but at possibly significant complexity cost.

Some lessons about ICN Congestion Control

Lesson 1: Congestion Control research is not a project, it's a career

Lesson 2: If you've paid the (high) price of per-packet state in the network, you should leverage it as much as possible

Lesson 3: None of this magically satisfies QoS fantasies, but it does protect the network from overload and can be made fair

Interaction Models

Interaction Models various applications use

- Content fetch
  - Been there, done that…
- Synchronize shared data
  - Variety of Sync protocols developed for NDN, but not a "solved problem" yet
- Publish/Subscribe
- Remote Invocation (aka RPC) & Distributed Computations
  - Some useful early research through NFN and NtaaS
  - More recent work with RICE, CEN, and RF
- Restful Transactions (aka web)

This is what I'll talk about here

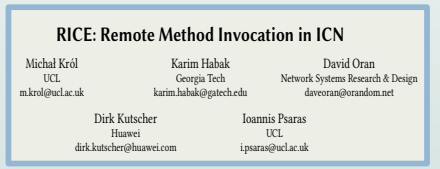
I'll talk a bit about this later

Computations are different from fetching data

- Need to deal with different timescales
  - Computations can run for many network RTTs
  - Long Interest Lifetimes suck – too much state and not robust against network failures
- Input Data matters – do you Push it?
  - Too much unsolicited data screws up congestion control
- Do you trust the caller?
  - Proving identity via request sacrifices consumer anonymity
- Most interactions need more than a two-way handshake to avoid expensive retries to ascertain if something worked.

Evolution of research into ICN for Distributed Computing: RICE, CFN, and RF

- RICE (ICN'18) addressed
  - Avoiding big request messages
  - Separating computation initiation from result delivery
  - Client authorization without losing anonymity to the network
  - Clean support of both idempotent and non-idempotent functions
- Compute First Networking (ICN'19) addressed
  - How to name computations and generate the interaction procedure directly from application code
  - How to orchestrate the placement of functions with Name-based routing
- Reflexive Forwarding (ICNRG Draft'22) cleans up and generalizes the handshake protocol machinery that RICE had hacked together
  - Lighter weight state management using "PIT Tokens"
  - Cleaner naming convention for the Interests going from producer to consumer


RICE: Remote Method Invocation in ICN

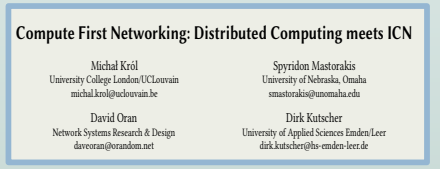
Michał Król
UCL
m.krol@ucl.ac.uk

Karim Habak
Georgia Tech
karim.habak@gatech.edu

David Oran
Network Systems Research & Design
daveoran@orandom.net

Dirk Kutscher
Huawei
dirk.kutscher@huawei.com

Ioannis Psaras
UCL
i.psaras@ucl.ac.uk

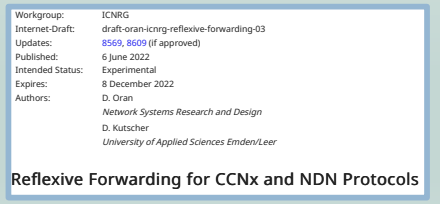

Compute First Networking: Distributed Computing meets ICN

Michał Król
University College London/UCLouvain
michal.krol@uclouvain.be

Spyridon Mastorakis
University of Nebraska, Omaha
smastorakis@unomaha.edu

David Oran
Network Systems Research & Design
daveoran@orandom.net

Dirk Kutscher
University of Applied Sciences Emden/Leer
dirk.kutscher@hs-emden-leer.de


Workgroup: ICNRG
Internet-Draft: draft-oran-icnrg-reflexive-forwarding-03
Updates: 8569, 8609 (if approved)
Published: 6 June 2022
Intended Status: Experimental
Expires: 8 December 2022
Authors: D. Oran
Network Systems Research and Design
D. Kutscher
University of Applied Sciences Emden/Leer

Reflexive Forwarding for CCNx and NDN Protocols

Some Lessons learned from working on distributed computing with ICN

Lesson 1: Content fetch with two-way handshakes is a poor match for doing distributed computations.

Lesson 2: Extensions to the base protocols can give a flexible underpinning for multiple interaction models

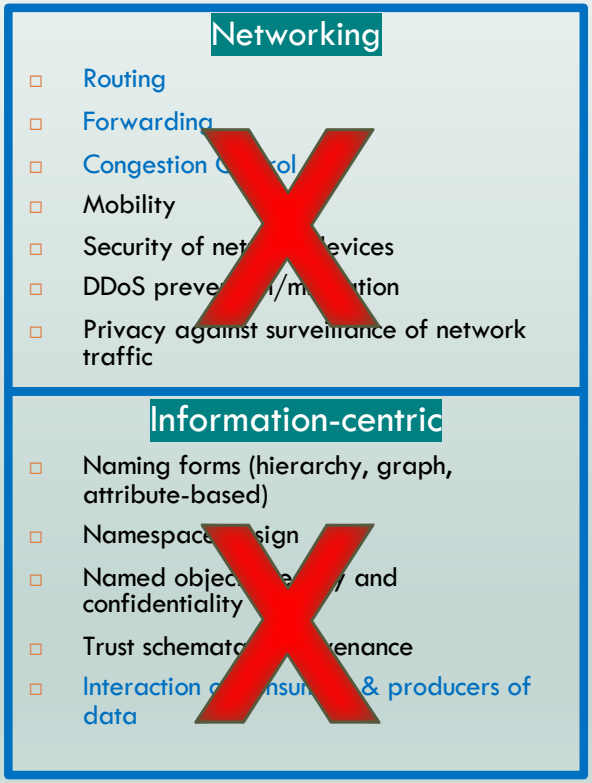
We have two "extreme" data points in the design space to learn from

If you only care about the Network part, look at hICN:

- Use IPv6 packets and steal bits from transport for demux
  - No fancy naming
  - No object security built in
- Get most of the nice routing, forwarding & Congestion control properties of NDN and CCNx

If you only care about the Information-centric part, look at OSCORE:

- Hierarchical naming, Secure objects, Web-like semantics
- But no help from the network
  - Need proxies and L7 caches
  - Interactions tied to transactions rather than objects themselves



Looking Ahead

What can we learn from our prior research to guide future research?

Are the lessons applicable?

WARNING!
Speculation Alert!!!

What belongs in which Layers?

Explore the tradeoffs between doing things at the application layer versus pushing functionality down, perhaps to L2 as well as L3.

- Experimentation with APIs seems central to answering this question
- As are choosing the right metrics: flexibility, throughput, latency, resilience

Application design & Interaction models

Can we have a "narrow waist" that works well for multiple Interaction models?

- We know it works for content fetch
- What about:
  - Sync
  - RPC / Distributed Computing
  - Restful Web transactions
  - Pub/Sub
- Is the multi-way handshake machinery of Reflexive Forwarding adequate (or right) or do we need something else?

Yes, Congestion Control still appeals

Congestion control is a perennial area with many subtle issues to work on

- How much to trust applications to respond to congestion signals (or packet drops) versus enforcement by network protocols
- Multipath congestion control is still ripe for improvements
- Dare I say it? QoS machinery for ICN.

Computing in the Network?

Should we instantiate transport functions and knowledge of interaction models in network devices (ala COIN)?

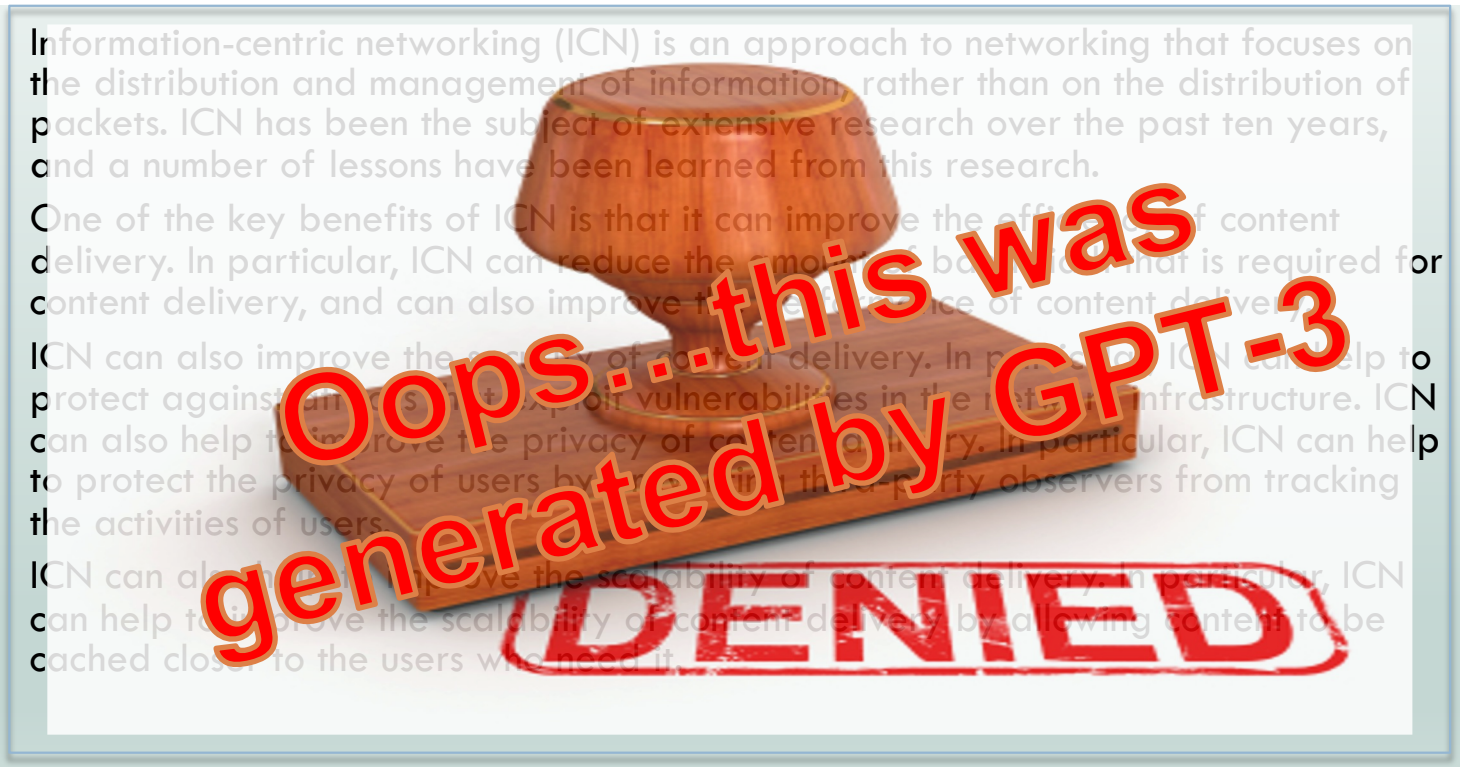
- What ICN functions might belong in switches? What can be done in the face of the constrained programming model?
- Do we even need this? Why can't everything be done in servers?

Is multi-destination delivery a winner in practice?

IP Multicast hasn't been a raging success. Is the same true for multi-destination delivery in ICN?

- ICN multi-destination delivery seems a big win. Can we convincingly demonstrate that?
- On the other hand, Interest Multicast (for things like Sync interactions) seems to suffer from the same pathologies as IP multicast. Can this be overcome?

Summing up the lessons



Let's try again…

- How can the networking insights we've gained from ICN protocols inform the construction of Information Centric systems and applications?
  - Whether and how to utilize name-based routing to achieve robustness and performance scaling for distributed applications?
  - Where does caching help or not help and how to best utilize caches?
  - Does pushing Names down to lower layers help latency? Resilience? Fairness?
- How can the insights we've gained from applying Information Centricity in applications inform what we bother to change the network to do, and what not?
  - Do things like multipath forwarding, in-network retransmission, or reflexive forwarding actually enable applications that are hard or infeasible to do without them?
  - Is there a big win for wireless networks in terms of optimizing a scarce resource or having more robust and responsive mobility characteristics?

Revisiting the Punchline

ICN is

Information-centric ∪ Networking

not

Information-centric ∩ Networking

None of this would be possible without:

Alex Afanasyev
Somaya Arianfar
Mark Baugher
Hila Ben Abraham
Jeff Burke
Ken Calvert
Giovanna Carofiglio
Taejoong Chung
Alberto Compagno
Patrick Crowley
Ralph Droms
Serge Fdida
J.J. Garcia Luna Aceves
Mevlut Garip
Cesar Ghali
Jim Gibson
Cenk Gündoğan
Karim Habak
Michał Król
Dirk Kutscher
Yuanjie Li
Milad Mahdian
Spyros Mastorakis
Maziar Mirzazad
Satyajayant Misra
Ilya Moiseenko
Marie-Jose Montpetit
Marc Mosko
Luca Muscariello
Ashok Narayanan
Börje Ohlman
Christos Papadopoulos
Ioannis Psaras
K.K. Ramakrishnan
Natalya Rozhnova
Thomas Schmidt
Eve Schooler
Wentao Shang
Won So
Mark Stapp
Christian Tschudin
Gene Tsudik
Ersin Uzun
Yaogong Wang
Matthias Waehlisch
Cedric Westphal
Bastiaan Wissingh
Chris Wood
Edmund Yeh
Haowei Yuan
Lixia Zhang

(with Apologies to those I'm sure I missed!)

Thanks for Listening!!*

Many thanks to Dirk Kutscher & Ken Calvert for sage advice on the important things to get across, and many great suggestions

*Most photos are *not* real – they were generated by DALL-E 2 - can you guess which?