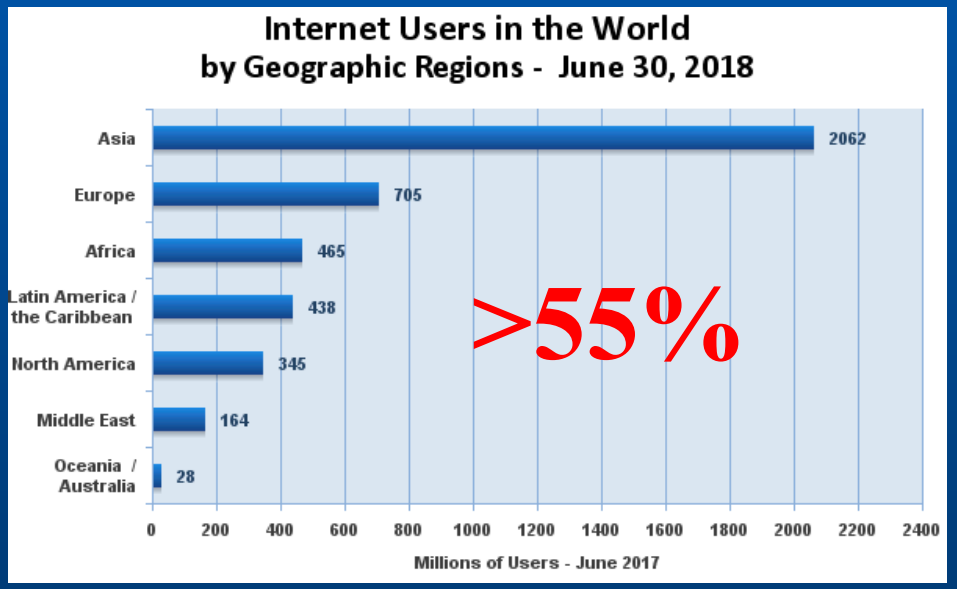
Communication Architecture for the future Internet

Walid Dabbous
INRIA Centre de Sophia-Antipolis Méditerranée
https://team.inria.fr/diana/

Networking was invented in this world

It was about sharing resources, not data.

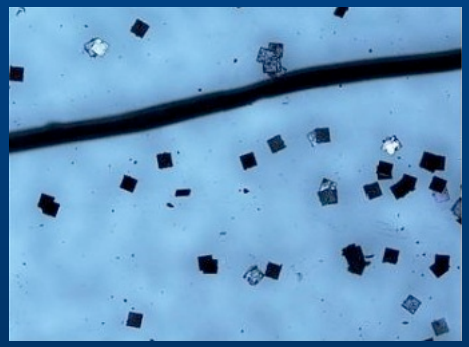
Today we have the Internet

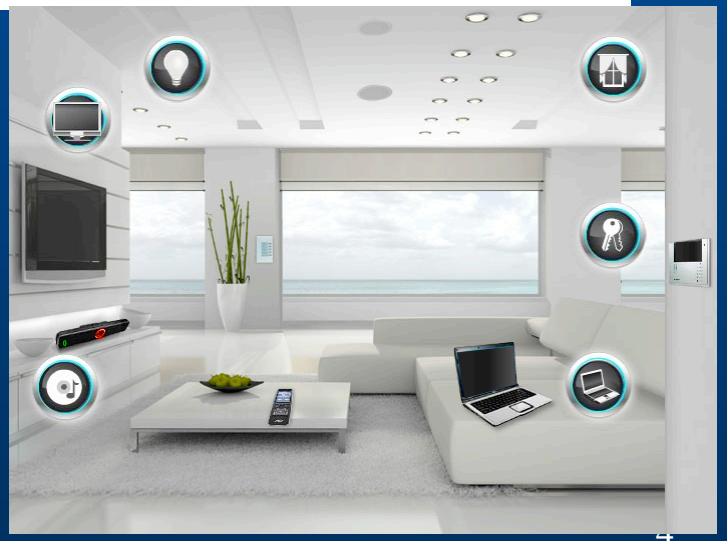


Estimated to 4,208,571,287 users
7,634,758,428 persons

June 2018

Internet Evolution



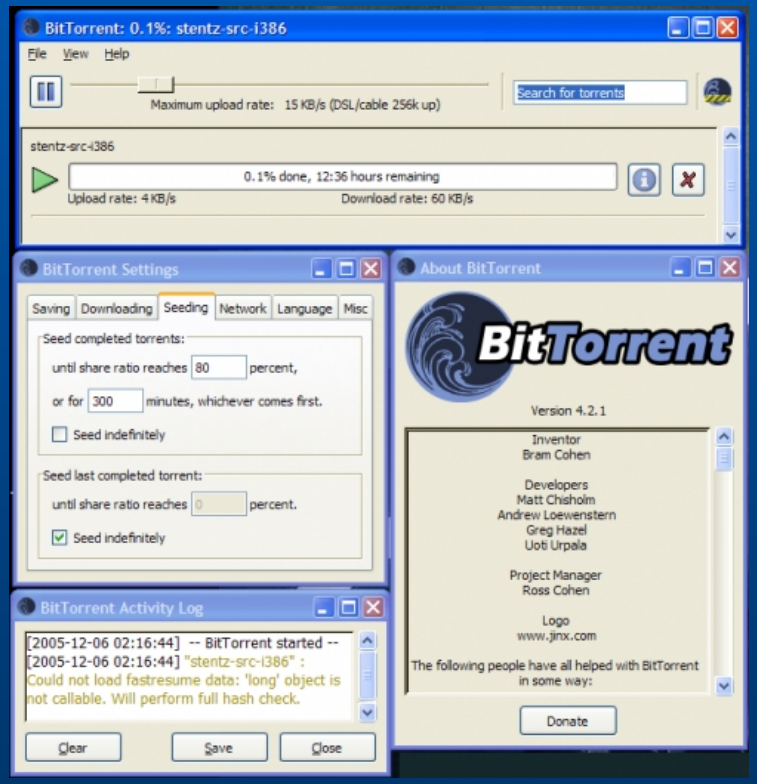


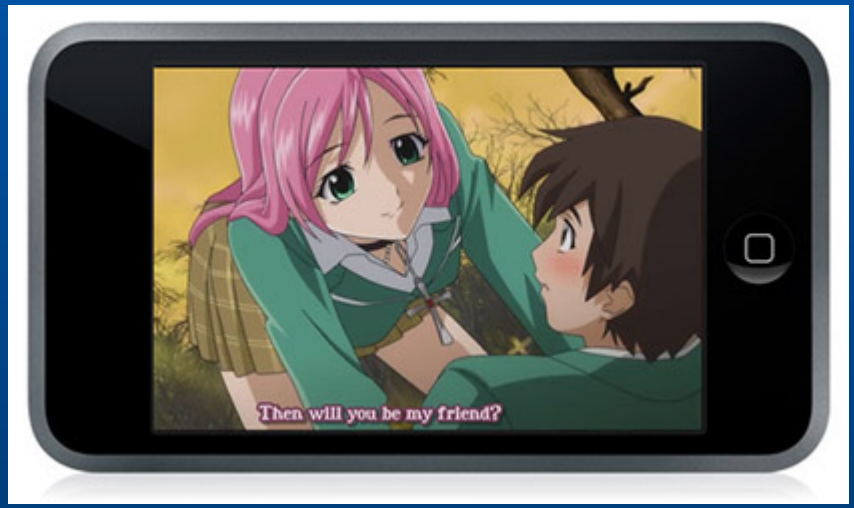
Internet Evolution

- **Mobility and episodic connectivity**

Internet Evolution

- **Dissemination of real time data**





Content is King

Wonderful … But

- Ubiquitous wireless
- Devices connectivity
- Wealth of information

- Doesn’t work everywhere
- Multiple (out of sync) devices
- Information related to hosts on which it resides

Point “patches” for ubiquitous problems

Networking History

- The Phone System
  - Focus on the wires
- The ATM network
  - Focus on virtual circuits
- The Internet today
  - Focus on the endpoints
- The future Internet
  - Focus on the data

The Phone System

- Connecting wires to other wires
  - Utility depends on running wires to *every* home & office
  - Wires are the dominant cost
  - Revenue comes from path construction
- Not about making "calls"
  - For an operator
    - a call is a "circuit" not a "conversation"
    - a phone number is a program to build the path not the callee *address*
  - Business model based on making revenues from calls that are a "side effect"
  - Revolutionized communications!

Concepts

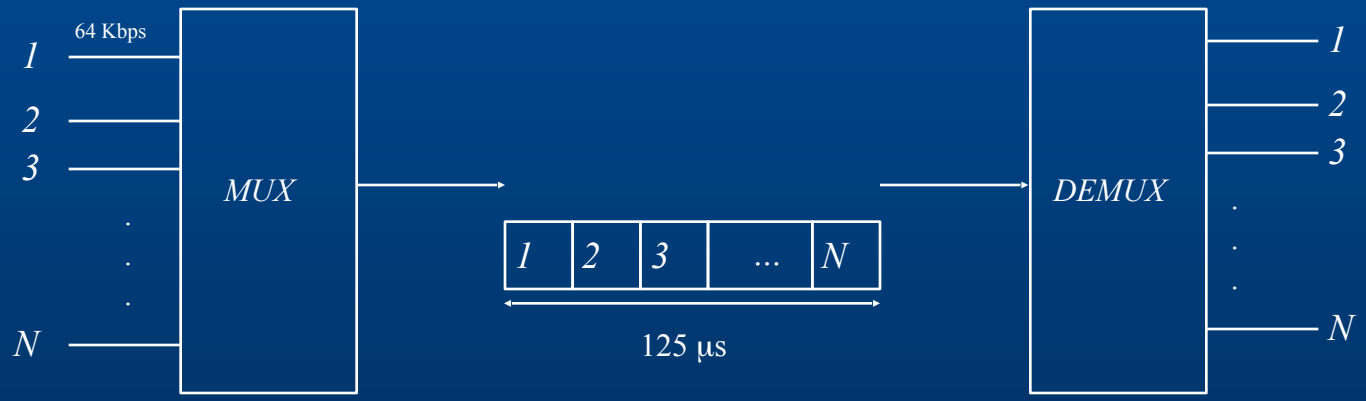
- Single basic service: two-way voice
  - low end-to-end delay
  - guarantee that an accepted call will run to completion
- Endpoints connected by a *circuit*
  - like an electrical circuit
  - signals flow both ways (*full duplex*)
  - associated with bandwidth and buffer *resources*

Links

- Link characteristics
  - information carrying capacity (bandwidth)
    - information sent as *symbols*
    - 1 symbol ≥ 1 bit
  - propagation delay
    - time for electromagnetic signal to reach other end
    - light travels at 0.7c in fiber ~ 5 ms/km
    - Nice to Paris => 5 ms; London to NY => 27 ms ; ~250 ms for earth-sat-earth on GEO satellites
  - attenuation
    - degradation in signal quality with distance
    - long lines need regenerators
    - but recent links need regeneration each 5000 Km and optical amplifiers exist

Synchronous transmission

- Most trunks time division multiplex voice samples
- At a central office, trunk is demultiplexed and distributed to active circuits
- Synchronous multiplexor
  - N input lines (associated with a buffer to store at least one sample)
  - Output runs N times as fast as input



Synchronous transmission

- Demultiplexor
  - one input line and N outputs that run N times slower
  - samples are placed in output buffer in round robin order
- Neither multiplexor nor demultiplexor needs addressing information (why?)
  - requires however accurate timing information
- Can cascade multiplexors
  - need a standard
  - example: DS hierarchy in the US and Japan

Digital Signaling hierarchy

Digital Signal Number	Number of previous level circuits	Number of Voice circuits	Bandwidth
DS0		1	64 Kbps
DS1 - T1	24	24	1.544Mbps
DS2	4	96	6.312 Mbps
DS3 - T3	7	672 = 28 T1	44.736 Mbps

Problems with STM

- Problems with STM
  - idle users consume bandwidth (STM is inefficient)
  - Arbitrary schedules result in complicated operation
    - links are shared with a fixed cyclical schedule => quantization of link capacity (corresponds to 64 Kbps circuits in telephone)
    - can't 'dial' bandwidth e.g. 91 Kbps.
  - STM service is inflexible

Better than STM for data?

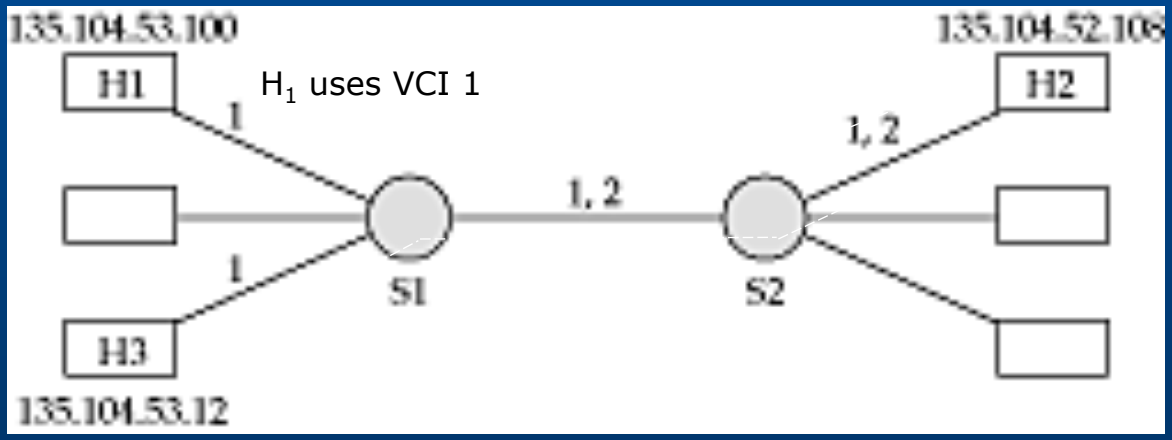
- STM is easy to overcome
  - use *packets* instead
  - meta-data (header) indicates src/dest
    - allows to store packets at switches and forward them when convenient
    - no wasted bandwidth (identify cell by source address not only order in frame) - more *efficient*
    - arbitrary schedule (cells of same source can occur more than once in frame) - more *flexible*
- Two ways to use packets
  - carry only an identifier (The ATM network)
  - carry entire destination address in header (IP)

The ATM network

1. Virtual circuits
2. Fixed-size packets (*cells*)
3. Small packet size
4. Statistical multiplexing
5. Integrated services

Virtual circuits

- Identifiers save on header space
- But need to be pre-established
- We also need to switch Ids at intermediate points
  - VCIs are allocated locally
- Need *translation table* (for VCI swapping) and *connection setup*



Features of virtual circuits

- All packets must follow the same path
  - if any switch along the route fails -> the VC fails
- Switches store per-VC state (entry in translation table)
  - can also store QoS information (priority, reserved bandwidth)
- Call set-up (or signaling) => separation of *data* and *control*
  - control in software over slow time scale, data transfer in hardware
- Virtual circuits do not automatically guarantee reliability
  - possible packet loss
- Small Identifiers can be looked up quickly in hardware
  - harder to do this with IP addresses

More features

- Setup must precede data transfer
  - delays short messages
- Switched vs. Permanent virtual circuits
- Ways to reduce setup latency
  - preallocate a range of VCIs along a path
    - *Virtual Path*
    - *reduces also the size of the translation table*
  - dedicate a VCI to carry datagrams, reassembled at each hop

2. Fixed-size packets

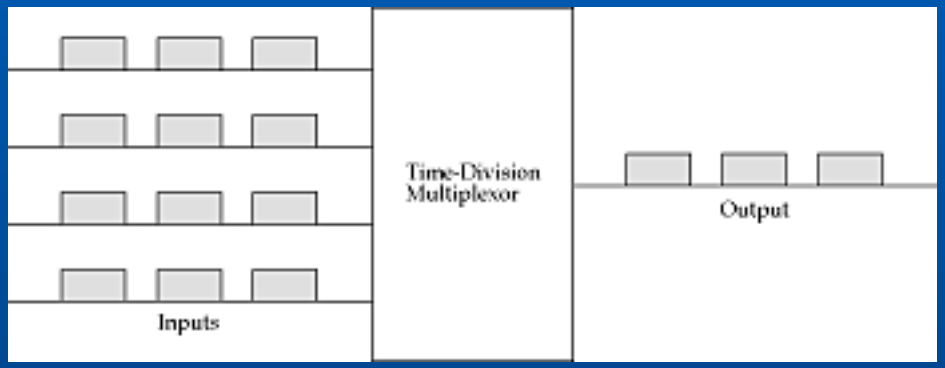
- Pros
  - Simpler buffer hardware
    - packet arrival and departure requires us to manage fixed buffer sizes (easier, no memory fragmentation)
  - Simpler line scheduling
    - each cell takes a constant chunk of bandwidth to transmit -> harder to achieve simple ratios with variable size packets
  - Easier to build large *parallel* packet switches
    - input buffers, parallel switch fabrics, output buffers -> *maximum parallelism if same packet size*
- Cons
  - If the chosen size < ADU => overhead
  - segmentation and reassembly cost
  - last unfilled cell after segmentation wastes bandwidth

3. Small packet size

- At 8KHz, each byte is 125 microseconds
- The smaller the cell, the less an endpoint has to wait to fill it
  - *packetization delay*
- The smaller the packet, the larger the header overhead
- EU and Japan: reduce cell size (32 bytes cell, 4 ms packetization delay)
- US telcos: reduce header cost (existing echo cancellation equipment) (64 bytes cell, 8ms packetization delay)
- Standards body balanced the two to prescribe 48 bytes + 5 byte header = 53 bytes
  - => ATM maximal efficiency of 90.57%

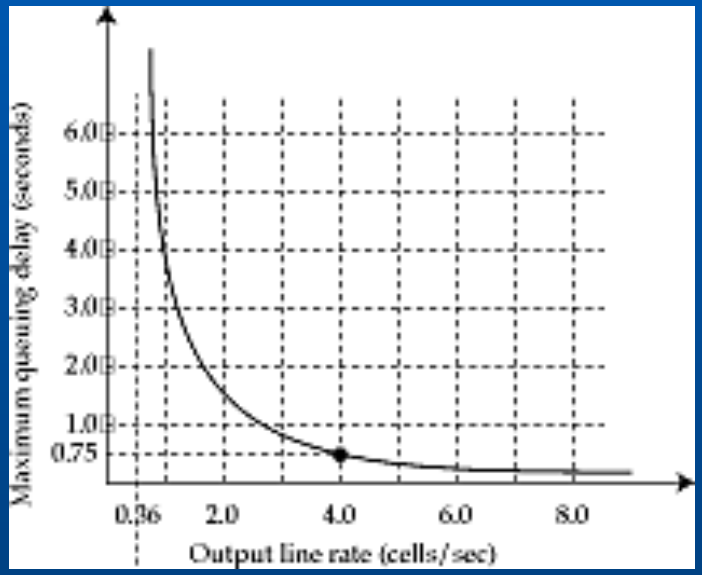
IETF TShirsts

4. Statistical multiplexing



- output rate: 4cells/s. queuing delay <= 3/4s.
- Suppose cells arrive in bursts
  - each burst has 10 cells evenly spaced 1 second apart
  - mean gap between bursts = 100 seconds (average rate = 0.0909 cell/s)
- What should be service rate of output line?
  - No single answer (4c/s? 0.36c/s? 1c/s?)

Statistical multiplexing



- We can trade off *worst-case delay* against *speed of output trunk*
- Statistical Multiplexing Gain = sum of peak input/output rate
  - A cell switch exploits SMG in the same way as a TD multiplexor.
- Whenever long term average rate *differs* from peak, we can trade off service rate for delay (requires buffers for zero loss)
  - key to building packet-switched networks with QoS

Generalized SMG

- *n* bursty source that have *p* peak rate and *a* average rate
- Worst case: simultaneous arrivals -> conservatively serve at *n.p*
- To reduce cost, can serve at *r* with $n.a < r < n.p$
  - Requires buffering -> higher delays
- SMG = $n.p/r$
- general principle:
  - if long-term average rate < peak rate; trade-off service rate for mean delay
- ATM cells can be stored & long distance BW expensive
  - -> SMG applicable
- Not if average rate close to peak rate

5. Integrated services?

- Traditionally, voice, video, and data traffic on separate networks
- Integration
  - easier to manage
  - innovative new services (e.g. Vconferencing)
- How do ATM networks allow for integrated service?
  - lots of (switching) capacity: hardware-oriented switching
  - support for different traffic types
    - signaling for call set-up
    - admission control, Traffic descriptor, policing
    - resource reservation
    - requires intelligent link scheduling for voice/data integration (more flexible than telephone because of headers)

Problems with Connection Oriented approaches for data

- Path construction is non-local and encourages centralization and monopoly (know/control resources)
  - Scheduling is NP hard
- *System* reliability goes down exponentially with scale
  - Requires high reliability *elements*
- Requires a path set-up phase
  - Not efficient for data (especially for large BDP: 100ms at Gbps is 12 MB!)

Datagrams

- A different style of communication
  - Change of Point of view : Focus on endpoints
  - The wires are already there!
- Data is sent in independent chunks of reasonable size with the destination address
  - Fairly share the path
- Simple relaying/routing of datagrams
  - “Connecting” adjacent hops
  - Based on addresses

Datagrams (contd)

- Internet was built on top of the phone system
  - Using the wires differently
- Speed agnostic
  - No set up phase
- But for operators it was:
  - Just an inefficient way to use their network!
  - A pure overlay
- Delivery technology agnostic
  - Could be overlaid over "everything"
  - Phone, Ethernet, satellite, radio, etc.

"TCP/IP"

- Reliability increases exponentially with the system size
- No call setup
  - Higher efficiency
- Distributed inter-domain routing
  - Works on any topology (simple hop count)
  - Tends to spread load
  - Network repairs from failures and
  - "hooks itself up" initially (due to the use of *explicit* address) – a big democratization
- Great for getting ubiquitous communication infrastructure

My how you've grown!

- In 1996, 10 million computers had joined the Internet
- By July 1997, 10 million more have joined
- By Jan 2001, 100 million hosts
- By March 2002, 400 million users
- By 2004, 800 million users
- By June 2008, 1.46 billion users
- By September 2009, 1.73 billion users
- By June 2018, 4.2 billion users
- Now, everyone who has a phone is likely to also have one or several email accounts

What does it look like?

- Loose collection of networks organized into a multilevel hierarchy
  - 10-100 machines connected to a *hub* or a *router*
    - service providers also provide direct dialup access
    - or over a wireless link
  - 10s of routers on a *department backbone*
  - 10s of department backbones connected to *campus backbone*
  - 10s of campus backbones connected to *regional service providers*
  - 100s of regional service providers connected by *national backbone*
  - 10s of national backbones connected by *international trunks*

Example of message routing

```
# traceroute parmesan.cs.wisc.edu (three probes at each TTL value)
traceroute to parmesan.cs.wisc.edu (128.105.167.16), 30 hops max, 38 byte packets
 1  t4-gw.inria.fr (138.96.32.250)  0.314 ms  0.271 ms  0.332 ms
 2  nice.cssi.renater.fr (195.220.98.117)  7.953 ms  10.770 ms  2.018 ms
 3  nio-n1.cssi.renater.fr (195.220.98.101)  17.489 ms  22.218 ms  14.136 ms
 4  nio-i.cssi.renater.fr (193.51.206.14)  14.080 ms  23.882 ms  18.131 ms
 5  opentransit-nio-i.cssi.renater.fr (193.51.206.42)  22.554 ms  15.353 ms  15.653 ms
 6  P3-0.PASCR2.Pastourelle.opentransit.net (193.251.241.158)  25.020 ms  16.662 ms  20.514 ms
 7  P11-0.PASCR1.Pastourelle.opentransit.net (193.251.241.97)  18.202 ms  15.704 ms  16.216 ms
 8  P12-0.NYKCR2.New-york.opentransit.net (193.251.241.134)  90.137 ms  90.190 ms  89.799 ms
 9  P6-0.NYKBB3.New-york.opentransit.net (193.251.241.238)  96.411 ms  97.740 ms  96.006 ms
10  BBN.GW.opentransit.net (193.251.250.138)  112.554 ms  116.028 ms  110.994 ms
11  p3-0.nycmny1-nbr2.bbnplanet.net (4.24.10.69)  119.815 ms  113.583 ms  108.599 ms
12  * p15-0.nycmny1-nbr1.bbnplanet.net (4.24.10.209)  115.725 ms  115.237 ms
13  so-6-0-0.chcgil2-br2.bbnplanet.net (4.24.4.17)  115.999 ms  124.484 ms  119.278 ms
14  so-7-0-0.chcgil2-br1.bbnplanet.net (4.24.5.217)  116.533 ms  120.644 ms  115.783 ms
15  p1-0.chcgil2-cr7.bbnplanet.net (4.24.8.106)  119.212 ms  117.684 ms  117.374 ms
16  a0.uwisc.bbnplanet.net (4.24.223.22)  123.337 ms  119.627 ms  126.541 ms
17  r-peer-WNMadison-gw.net.wisc.edu (216.56.1.18)  123.403 ms  127.295 ms  129.175 ms
18  144.92.128.226 (144.92.128.226)  124.777 ms  123.212 ms  131.111 ms
19  144.92.128.196 (144.92.128.196)  121.280 ms  126.488 ms  123.018 ms
20  e1-2.foundry2.cs.wisc.edu (128.105.1.6)  132.539 ms  127.177 ms  122.419 ms
21  parmesan.cs.wisc.edu (128.105.167.16)  123.928 ms *  124.471 ms
```

What holds the Internet together?

- Addressing
  - how to refer to a machine on the Internet
- Routing
  - how to get there
- Internet Protocol (IP)
  - what to speak to be understood at the "inter-network" level

Endpoint control - the end2end argument

- Key design philosophy
  - do as much as possible at the endpoint
  - dumb network
  - exactly the opposite philosophy of telephone network
- Layer above IP compensates for network defects
  - Transmission Control Protocol (TCP)
- Can run over any available link technology
  - but no quality of service
  - modification to TCP requires a change at every endpoint
  - telephone network technology upgrade transparent to users

Is there an architectural problem in the Internet?

- Hosts are tied to IP addresses
  - Mobility and multi-homing pose problems
- Services are tied to hosts
  - A service is more than just one host: replication, migration, composition

Internet Naming is *Host-Centric*

- Two global namespaces: DNS and IP addresses
- These namespaces are host-centric
  - IP addresses: network location of host
  - DNS names: domain of host
  - Both closely tied to an underlying structure
  - Motivated by host-centric applications

The Trouble with Host-Centric Names

- Host-centric names are *fragile*
  - If a name is based on mutable properties of its referent, it is fragile
  - Example: If your Web page http://www.unice.fr/~ubinetter moves to www.wallstreetstiffs.com/~yuppie, Web links to his page break
- Fragile names constrain movement
  - IP addresses are not stable host names
  - DNS URLs are not stable data names

Networking produced today's world of content but was never designed for it

- The central abstraction is a host identifier
- The fundamental communication model is a point-to-point conversation between two hosts.

Unfortunate consequences

- Networking hates wireless, mobility and intermittent connectivity.
- Cognitive mismatch - user/app model is *'what'*, network wants *'who'*. Mapping between models requires a lot of convention and configuration (middleware & wetware).
- No useful security - content is opaque to the net and it can't secure something it knows nothing about.

So the problem has changed

- When TCP was invented there were a lot of users per machine
- Now there is a lot of machines per user with data to be synchronized and shared
- Conversations are the central architectural elements in today's networks
- But 90% of the use of today's networks is to get a named chunk of data (web, mail)
- It's not conversation it is a *dissemination*

Similar shift than data over telephone

- Phone system was to build paths
- Used for making calls
- TCP invented to make conversations
- The Internet is used mainly for web access and mail
  - Not a conversation
  - "Does Any body know where we are?"
    - Point2multipoint or mp2mp
  - Dissemination
  - Superset of the conversational model

Data matters not the supplier

- It’s possible to disseminate over conversations and get the data as a side effect, but..
- Security is difficult: Channels are secured not the data
  - An SSL connection does not stop the spam
- It’s inefficient
  - same content, different destinations
- Users have to manually set up the plumbing to make things work using TCP
  - E.g. VPNs set up to get mail offsite
- The network has no knowledge of the content

‘Dissemination’ Networking

- Focus on data not on where it lives
- Data is requested by *name* using all means available
- Anything that hears the request and has a valid copy can respond
- Returned data is signed and secured so that the integrity and association with the name can be validated
  - Lemonde.fr today news
- An appliance can realize dissemination for the user
  - Collecting data and credentials

Naming data

- Data has a name not a location
  - It does not matter where the data is
  - Different from current `caching' which is getting a closer copy of remote data
- Integrity and trust are derived *from the data*
  - Not remote agents
- Any thing that move bits in time and/or space can be used for communication
  - Cut through, store (buffer) and forward, store carry and forward

Enhanced Communication

- Communicates intent to network and it will do things on behalf
  - Translate top level intent to the right semantics at lower levels
  - E.g. establish VPN to get my mail if out
- User can fine grain control the QoS (on access) due to req/resp model
- Popular content won't generate congestion

Communication

- Leverage broadcast
  - Since nodes don't need names, wireless & sensor nets can use simple local protocols (proximity, diffusion)
- No distinction between bits on a wire, in a memory or disk
- Data can be remembered
  - intermittent operation does not preclude communication
- Can use opportunistic transport
  - Planes, cars, etc.

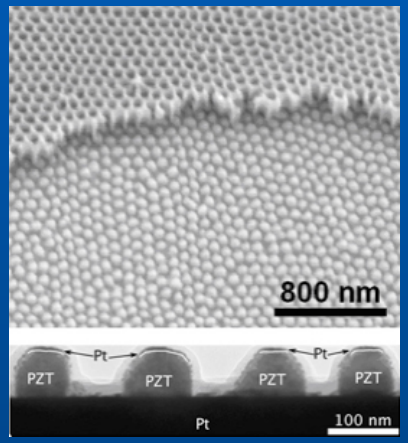
Data Communications today is about moving content

- There is a lot of content: As of Dec 2008 the Internet was moving 8 Exabytes/month.
- IDC reports that 180 Exabytes of new content was created and copied in 2006.
- 4.4 zettabytes in 2013
- More than 44 zettabytes per year in 2020!

Networking & storage cost evolution

- Disk cost/byte has fallen 3% per week for the last 25 years!
- US OC-3 $ per Mbps per Mile remained almost constant

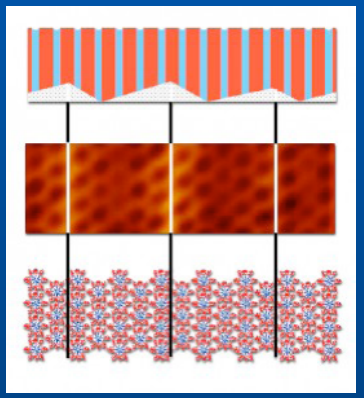
and storage is going to get a lot cheaper…



200 Gb/in² PZT nano-capacitor non-volatile memory

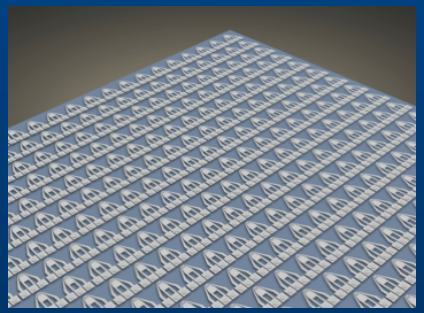
Max Planck Institute, June 2008

10 Tb/in² co-polymer magnetic memory

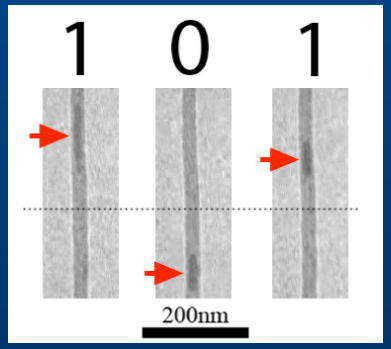


LBL, Feb. 2009

4 Tb/in² MEMS memory array



Univ. Twente, July 2009



Tb/in² carbon nanotube magnetic memory

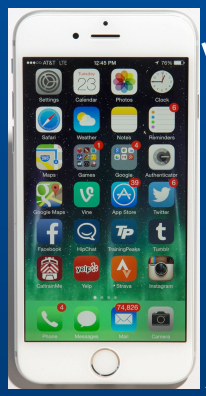
LBL, May 2009

Cost evolution favors trading storage for bandwidth but ...

- Storage names say what we want,
- Network names say who we want.
  - Mapping between these two models requires a lot of plumbing (middleware & wetware).

- Can we design a network architecture based on named data instead of named hosts?

Making content move itself



Ubineter's calendar

IMNA course at 9:00

IMNA course at 9:00

Ubineter's calendar

Devices express 'interest' in data collections.

- Devices with data in collection respond.

Moving content

- Users specify the objective, not how to accomplish it.
- Data appears wherever it needs to be.
- Model loves wireless and broadcast (802.11, RFID, Bluetooth, NFC, ...).
- There's no distinction between bits in a memory and bits in a wire.
- Data security and integrity are the architectural foundation, not an add-on.

Security

- Trust & data integrity are foundation of the design not an add-on
  - Phishing, Pharming, Spam can easily be made impossible
  - You can tell from the data you got who sent it, what is it
- Trust is associated with user level objects not irrelevant abstractions like an SSL connection
- It's hard for an adversary to disrupt a network that uses any thing, any time, any where to communicate

Content Centric Networking

Content Centric Networking (CCN): Goals

- Create a simple, universal, flexible communication architecture that:
  - Matches today's communication problems
  - Matches today's application design patterns
  - Is at least as scalable & efficient as TCP/IP
  - Is much more secure
  - Requires far less configuration

Universal?

- Any architecture that runs over anything is an overlay (IP is an overlay).
- IP started as a phone system overlay; today much of the phone system is an IP overlay. System theorists would say 'IP is universal'.
- CCN has the same character: it can run over anything, including IP, and anything can run over CCN, including IP.
- And CCN has a simpler, more general relationship with lower layers than IP.

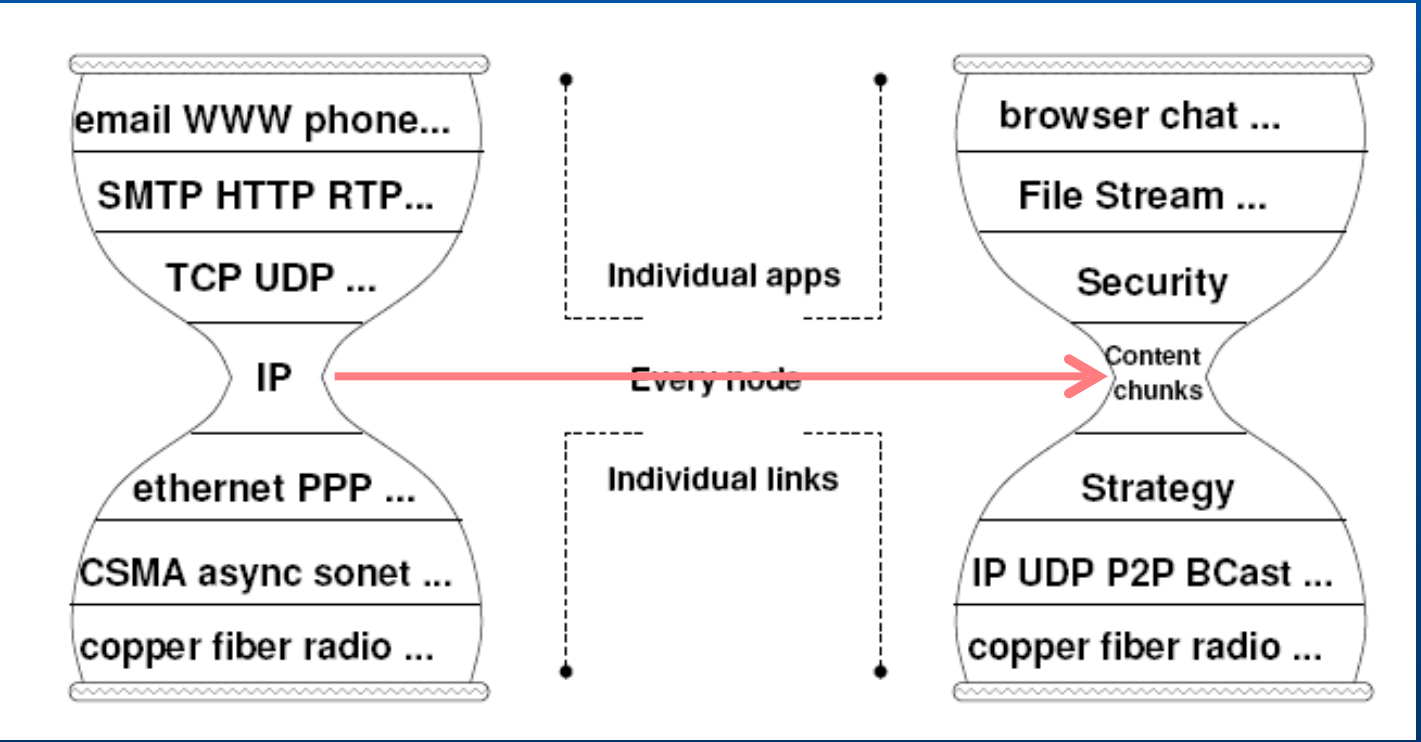
There are two ways to view CCN

- a ‘universal’middleware
- an IP for content

The difference between these is deployment time horizon

IP to "chunks of named content"

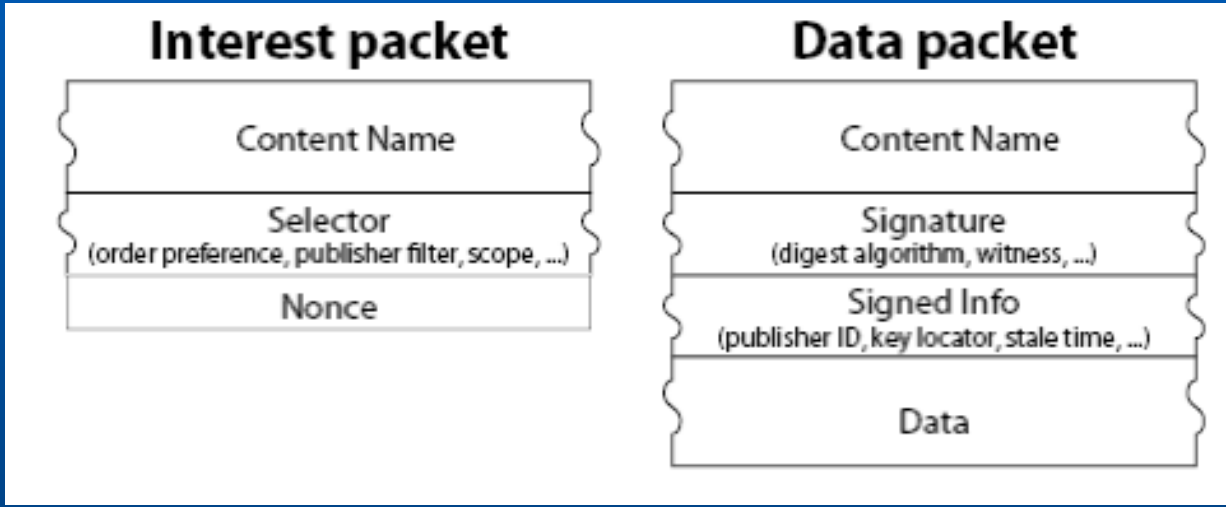
- Strategy: exploit multiple simultaneous connectivities for data delivery (e.g., ethernet, 3G, WiFi) due to its simpler relationship with layer 2



Current Internet

CCN

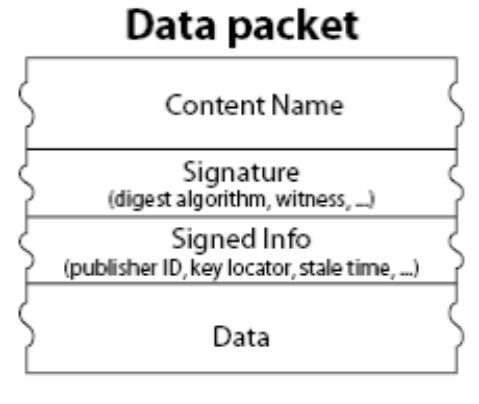
Node Model: CCN packets

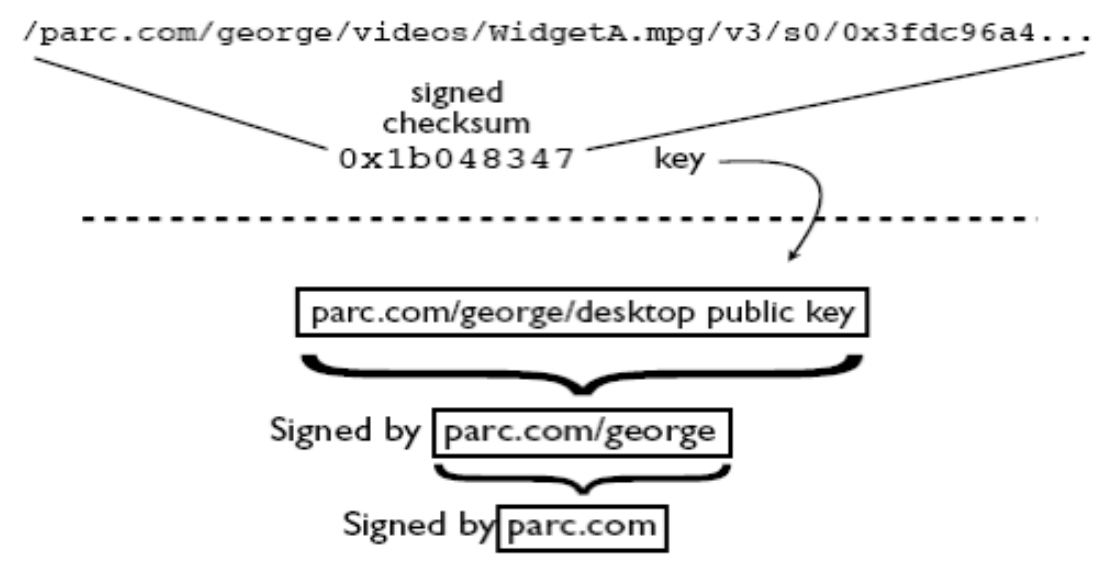


There are two CCN packet types: *interest (similar to http "get") and data* (similar to http response). Both are encoded in an efficient binary XML.

Content-Based Security

- **Name-content mapping verification via per-data packet signature**
  - **Data packet is authenticated with digital signature**





CCN trust establishment by associating content namespaces w/ public keys

Basic CCN forwarding

- Consumer 'broadcasts' an 'interest' over any & all available communications media:
  get '/parc.com/van/presentation.pdf'
- Interest identifies a *collection of data - all data* items whose name has the interest as a prefix.
- Anything that hears the interest and has an element of the collection can respond with that data:
  HereIs '/parc.com/van/presentation.pdf/p1' <data>

Basic CCN transport

- Data that matches an interest ‘consumes’ it.
- Interest must be re-expressed to get new data. (Controlling the re-expression allows for traffic management and environmental adaptation.)
- Multiple (distinct) interests in same collection may be expressed (similar to TCP window).

Internally, CCN names are opaque, structured byte strings

/parc.com/van/cal/417.vcf/v3/s0/0x3fdc96a4...

is represented as a component count
then, for each component, a byte count followed by that many bytes:

The *only assumption CCN makes about names is hierarchical structure.*
E.g., names or components can be encrypted or contain arbitrary binary data.

Using Names

- The hierarchical structure is used to do ‘longest match’ lookups (similar to IP prefix lookups) which helps guarantee log(n) state scaling for globally accessible data.
- Although CCN names are longer than IP identifiers, their *explicit structure allows* lookups as efficient as IP’s.

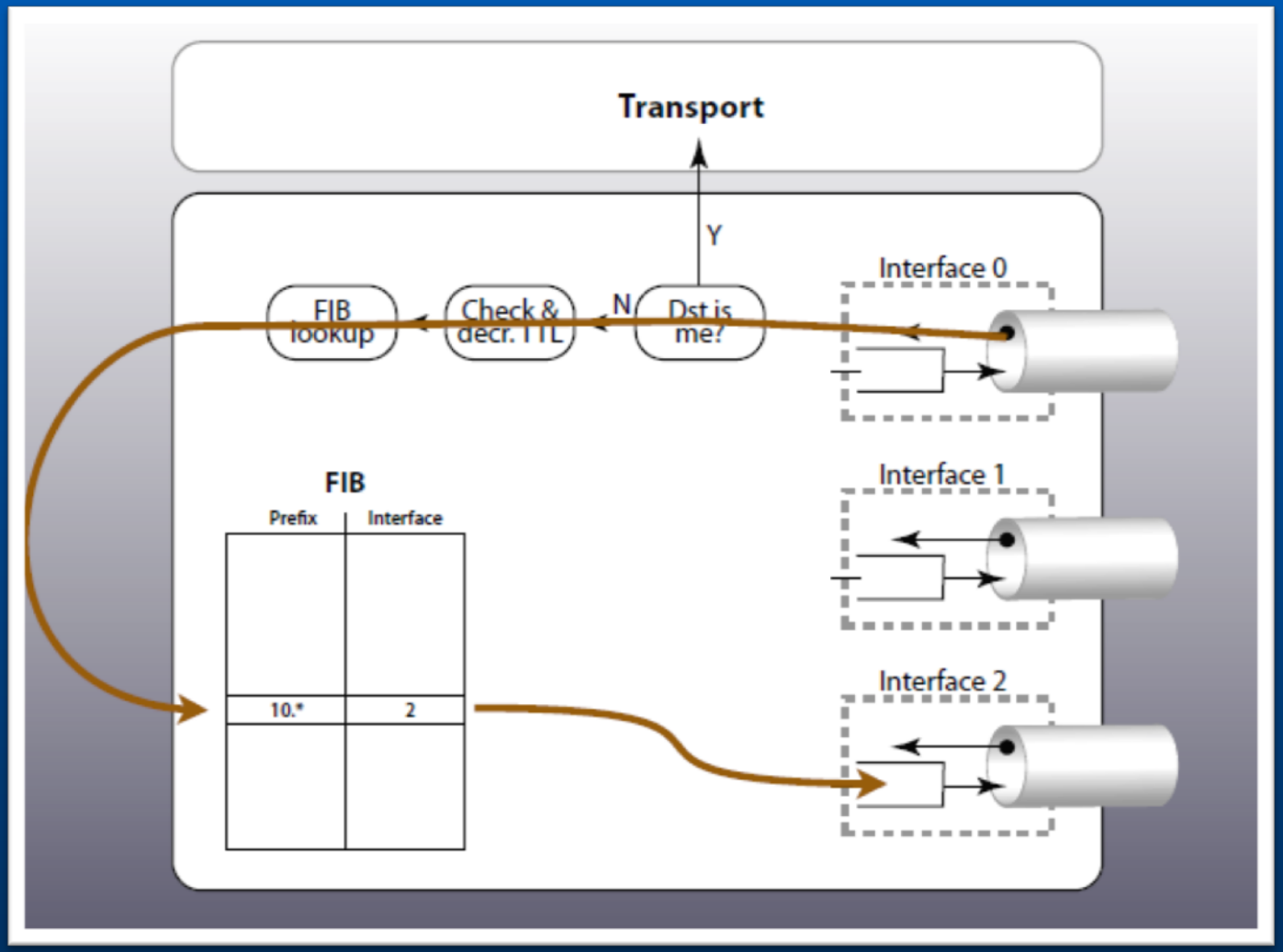
  (see hashing work by Rasmus Pagh and Martin Dietzfelbinger)

Names and meaning

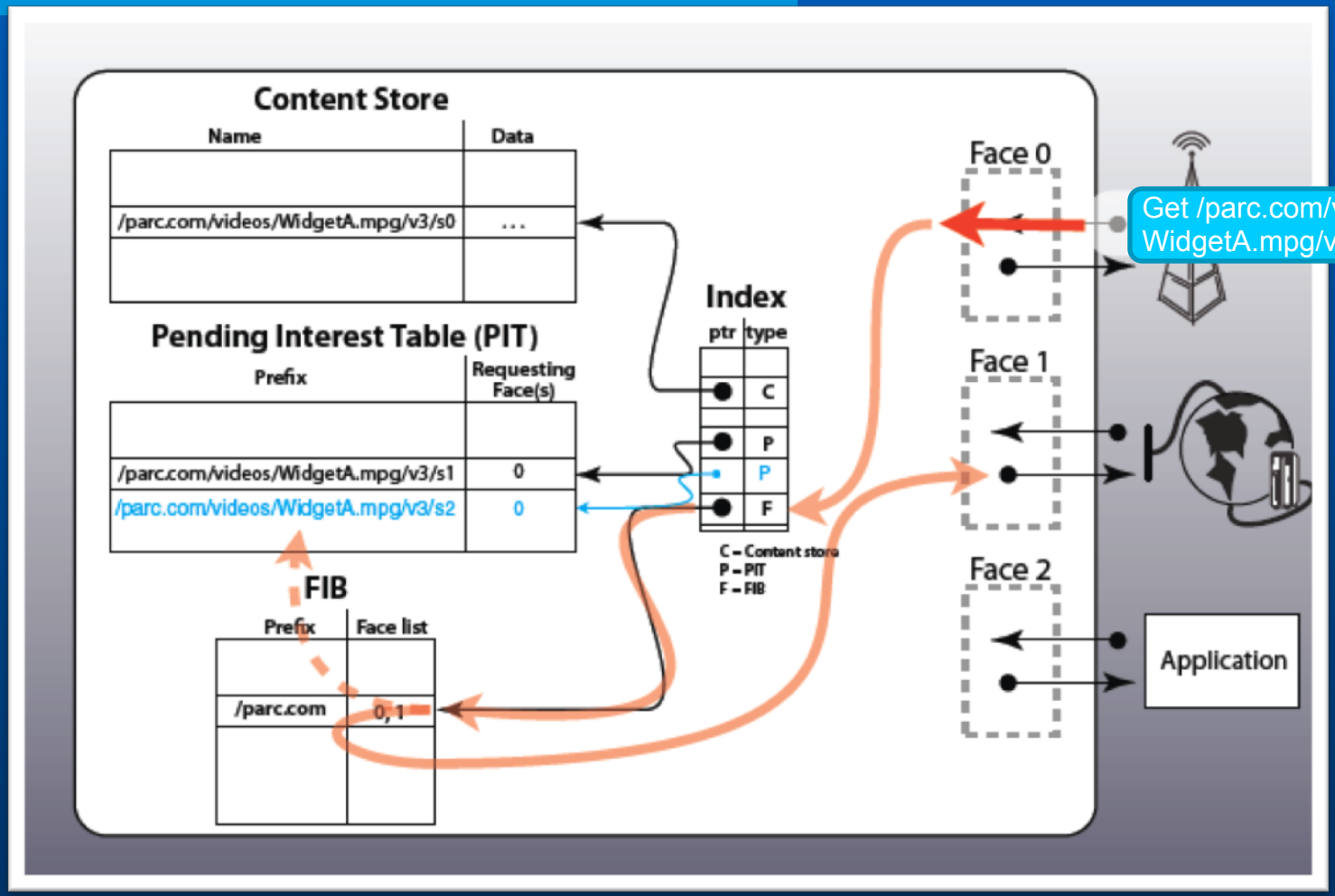
- Like IP, a CCN node imposes no semantics on names.
  Meaning comes from application, institution and global conventions reflected in prefix forwarding rules.

  For example,
  /parc.com/people/van/presentations/FISS09
  might be the name of a presentation's data and
  /thisRoom/projector
  the name of the projector it should display on.
- The former is a globally meaningful name leveraging the DNS global naming structure. The latter is local and context sensitive—it refers to different objects depending on the room you're in.

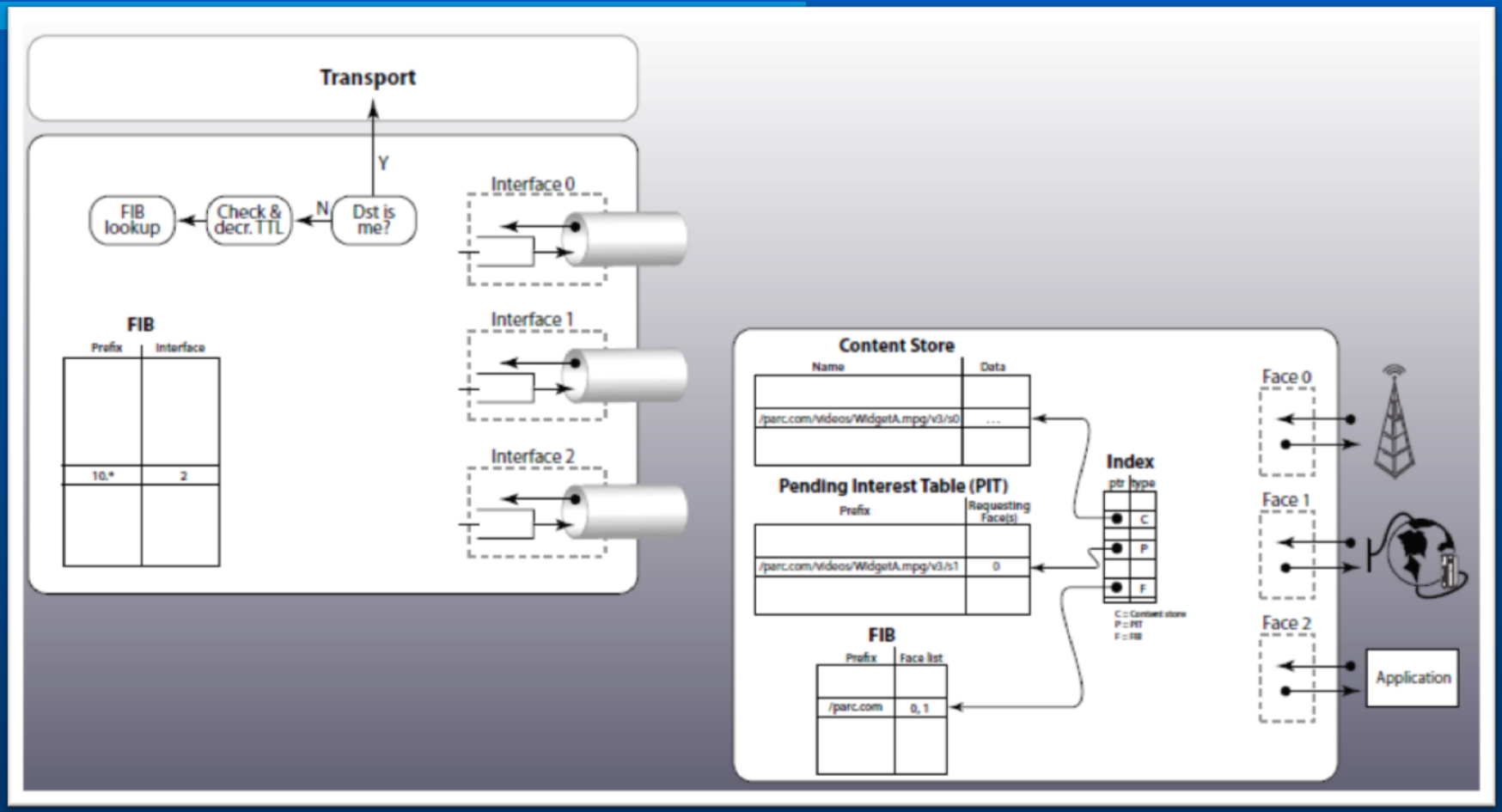
IP node model



CCN node model



Comparison


Transport
Y
N
Dst is me?
Check & decr. TTL
FIB lookup
Interface 0
Interface 1
Interface 2
FIB
Prefix
Interface
10.*
2
Content Store
Name
Data
/parc.com/Videos/WidgetA.mpg/v3/s0
Pending Interest Table (PIT)
Prefix
Requesting Face(s)
/parc.com/Videos/WidgetA.mpg/v3/s1
0
Index
ptr
type
C
P
F
FIB
Prefix
Face list
/parc.com
0, 1
Face 0
Face 1
Face 2
Application

Strategy layer (mobility management)

- When you don't care who you're talking to, you don't care if they change.
- When you're not having a conversation, there's no conversation state to migrate.
- Multi-point gives you multi-interface for free.
- When all communication is locally flow balanced, your stack knows exactly what's working and how well.

Quality of Service (QoS)

- In the current Internet, QoS problems are highly localized.
- Roughly half the problems arise from the serial dependencies created by queues.
- The other half are caused the lack of receiver based control of bottleneck links.
- Unlike IP, CCN is local, doesn't have queues and receivers have complete, fine-grained control.
- But it does aggregate traffic and has face-specific controls on the aggregation.

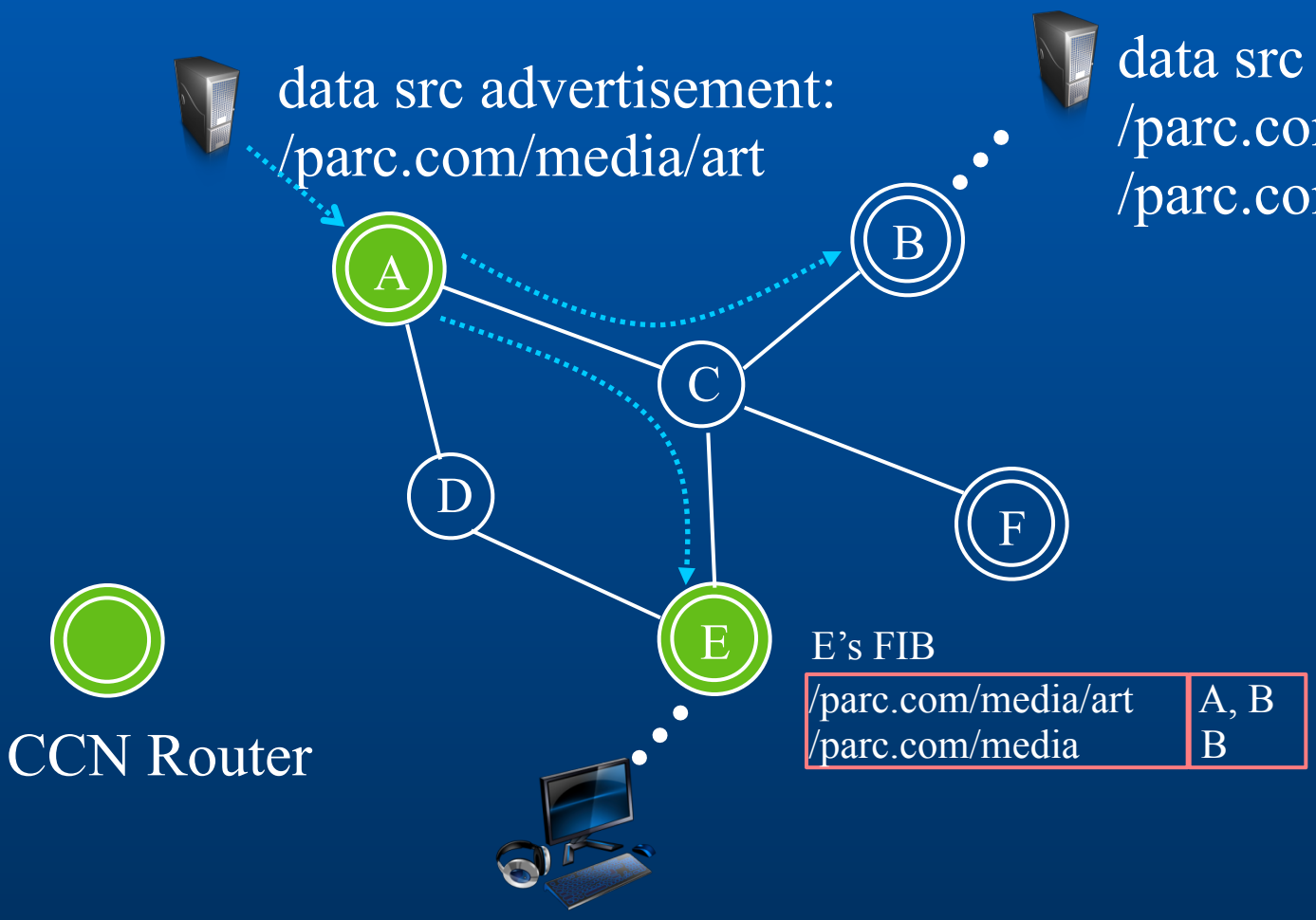
Routing

- There are many (emerging) ways to do routing, e.g., Small Worlds, Geographic Hyperbolic, Pseudo potential Gradient, Epidemic percolation.
- In general they're easier to implement and work better for CCN than for IP:
  - no looping data ⇒ no convergence issues.
  - multi-destination ⇒ state can be approximate (false positives ok).
  - CCN transport model matches routing's and adds security.

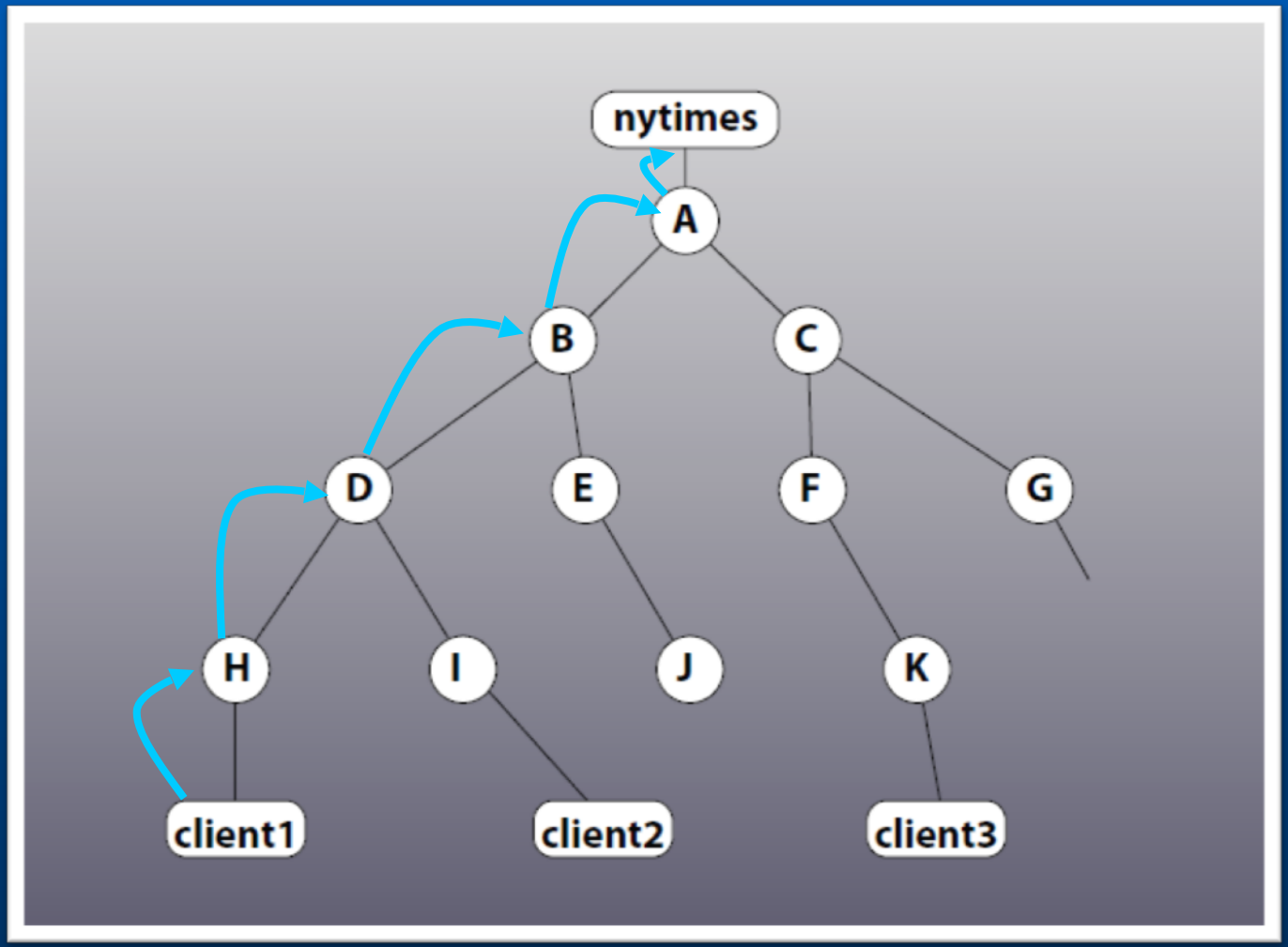
Short term

- I'm only going to talk about embedding CCN in existing Internet routing.
  - This is an easy evolutionary path (it allows for immediate, incremental deployment).
  - It offers some intuition on scaling (same scaling as IP routing).
  - The basics are the same for any routing scheme.

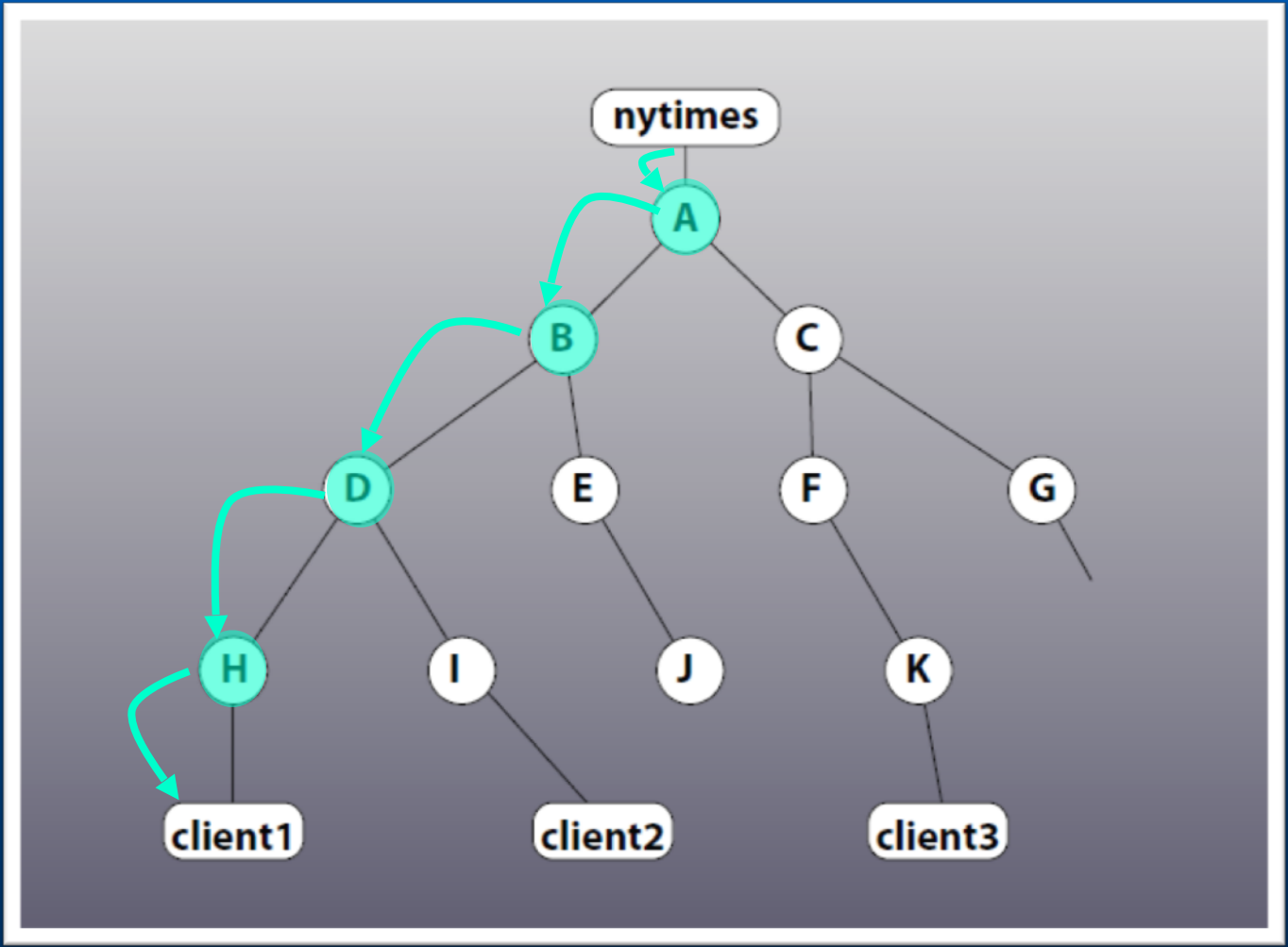
Existing link-state routing protocols can be used, unmodified, to construct a CCN FIB



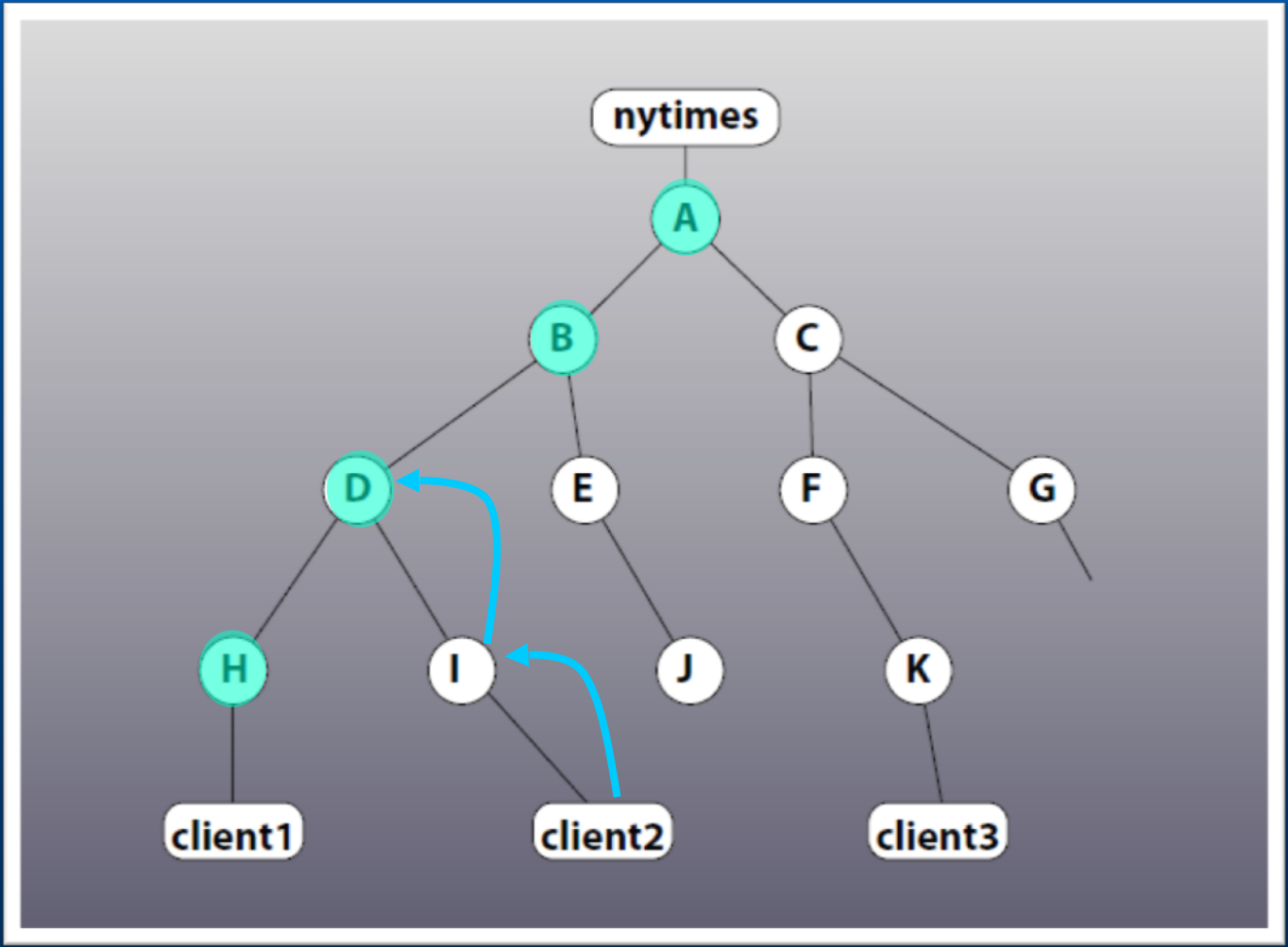
Example: Content Distribution



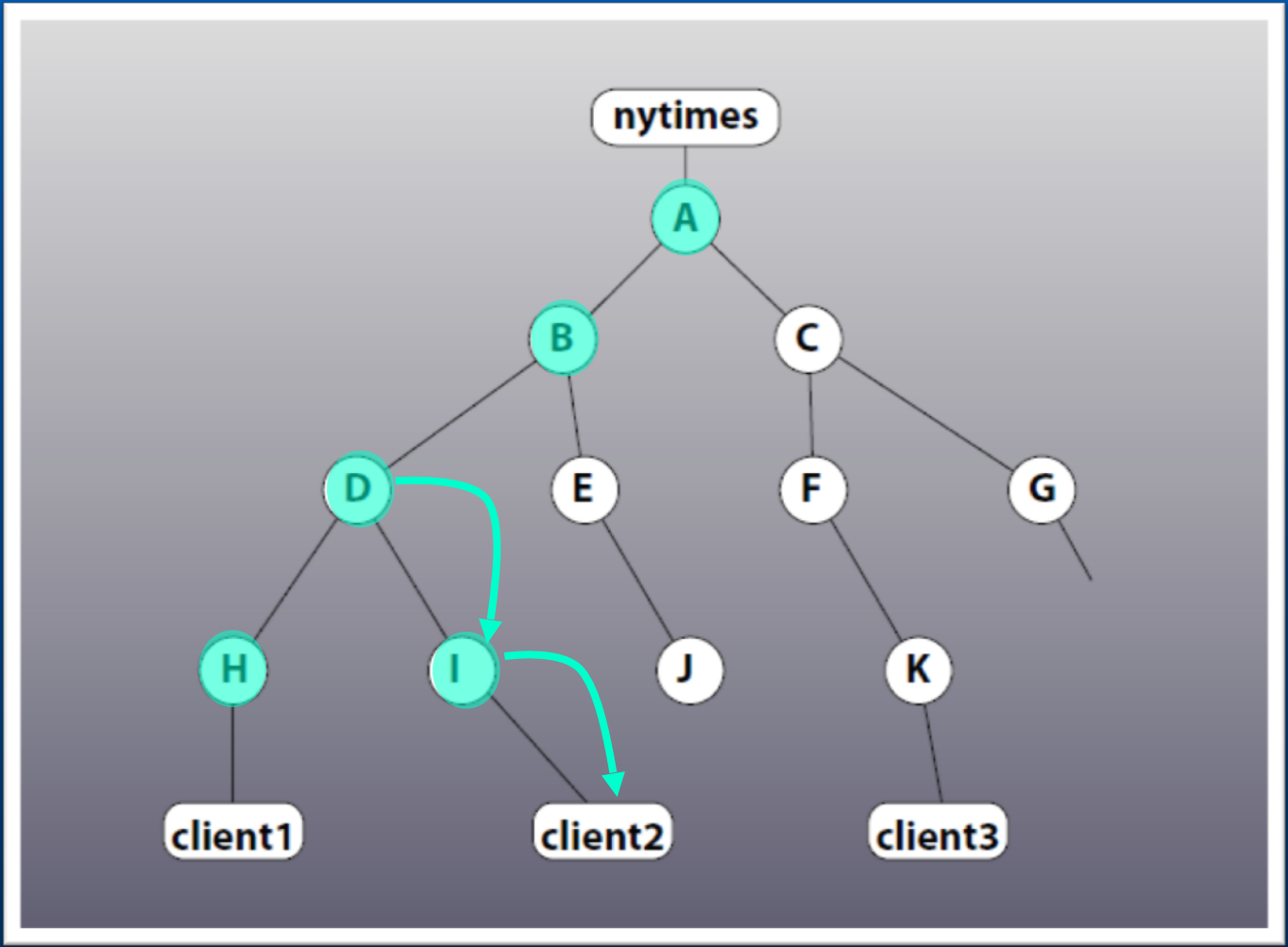
Example: Content Distribution



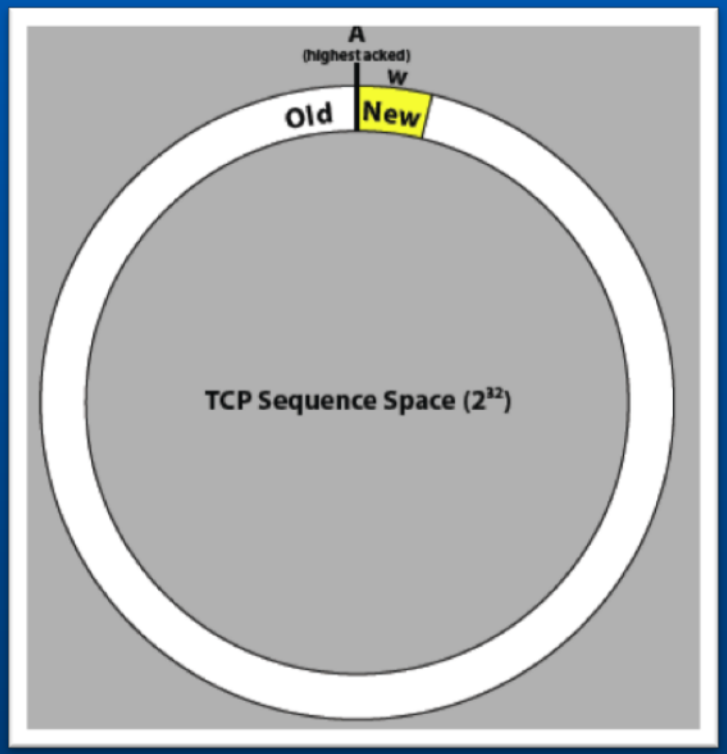
Example: Content Distribution



Example: Content Distribution

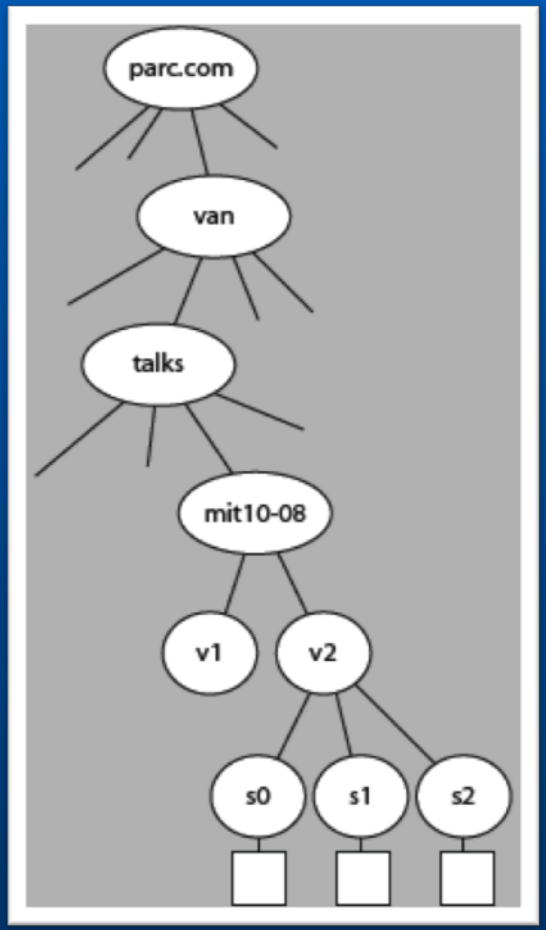


Transport State



- Conversation transport state is very compact:
  - One dynamic state variable (tcp sequence / ack number) conveys what ends know.
  - Additional static variable (tcp window) conveys what they want.

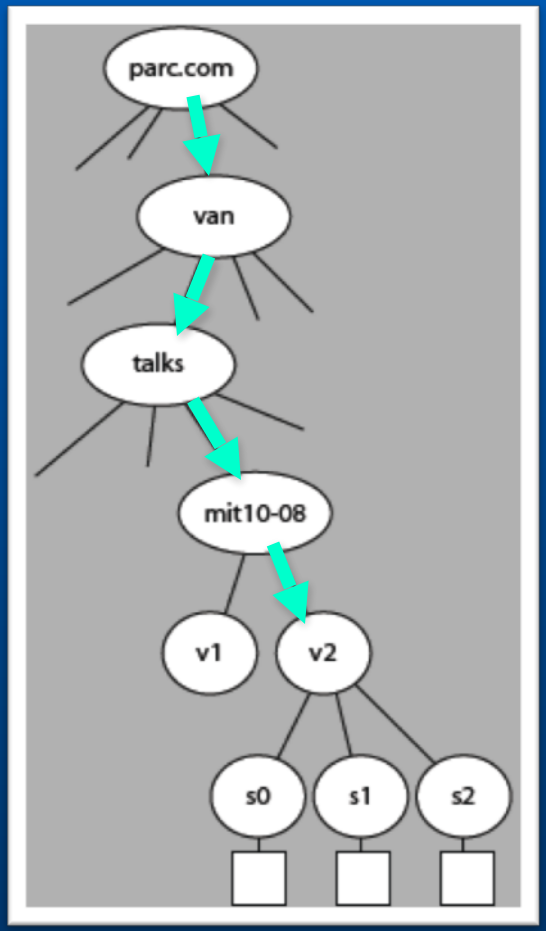
Annotated path in CCN name tree serves as transport state



Conventions:

- *name tree child nodes are lexically ordered*
- *<next> assumed if no relationship specified*

Annotated path in CCN name tree serves as transport state



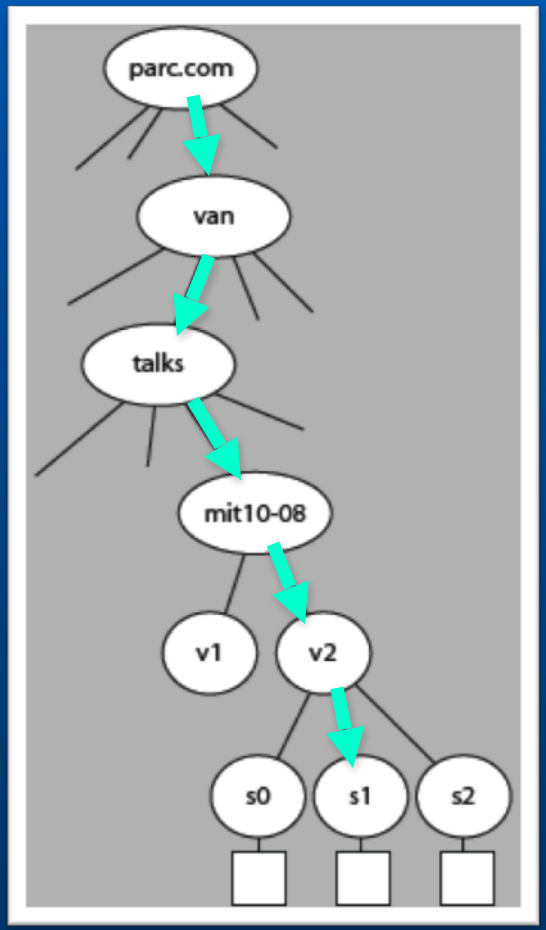
- Most recent version of slides for this talk:

parc.com/van/talks/mit10-08 <rightmost child>

Conventions:

- *name tree child nodes are lexically ordered*
- *<next> assumed if no relationship specified*

Annotated path in CCN name tree serves as transport state



- Most recent version of slides for this talk:

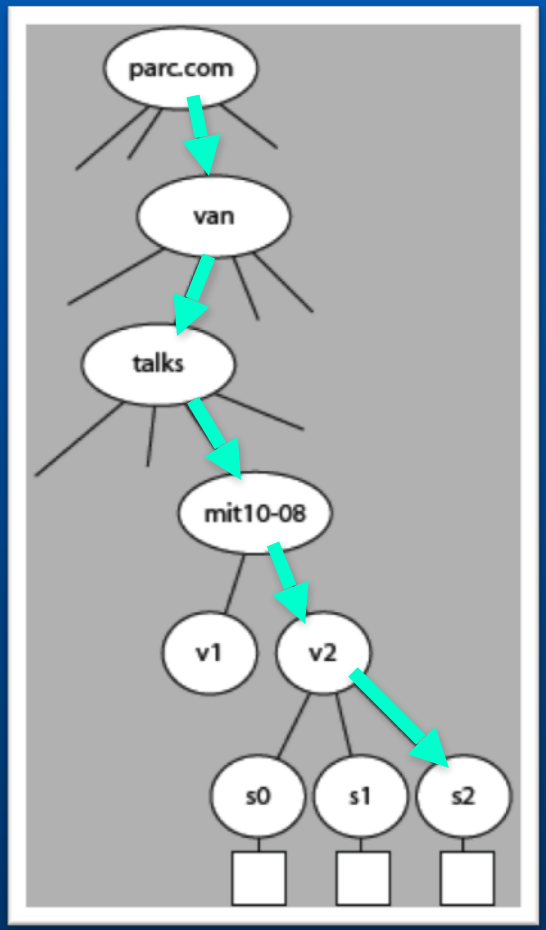
parc.com/van/talks/mit10-08 <rightmost child>

- Newest version after v2:

parc.com/van/talks/mit10-08/v2 <rightmost sibling>
(fails since there is no newer version)

Conventions:
- *name tree child nodes are lexically ordered*
- *<next> assumed if no relationship specified*

Annotated path in CCN name tree serves as transport state



- Most recent version of slides for this talk:

parc.com/van/talks/mit10-08 <rightmost child>

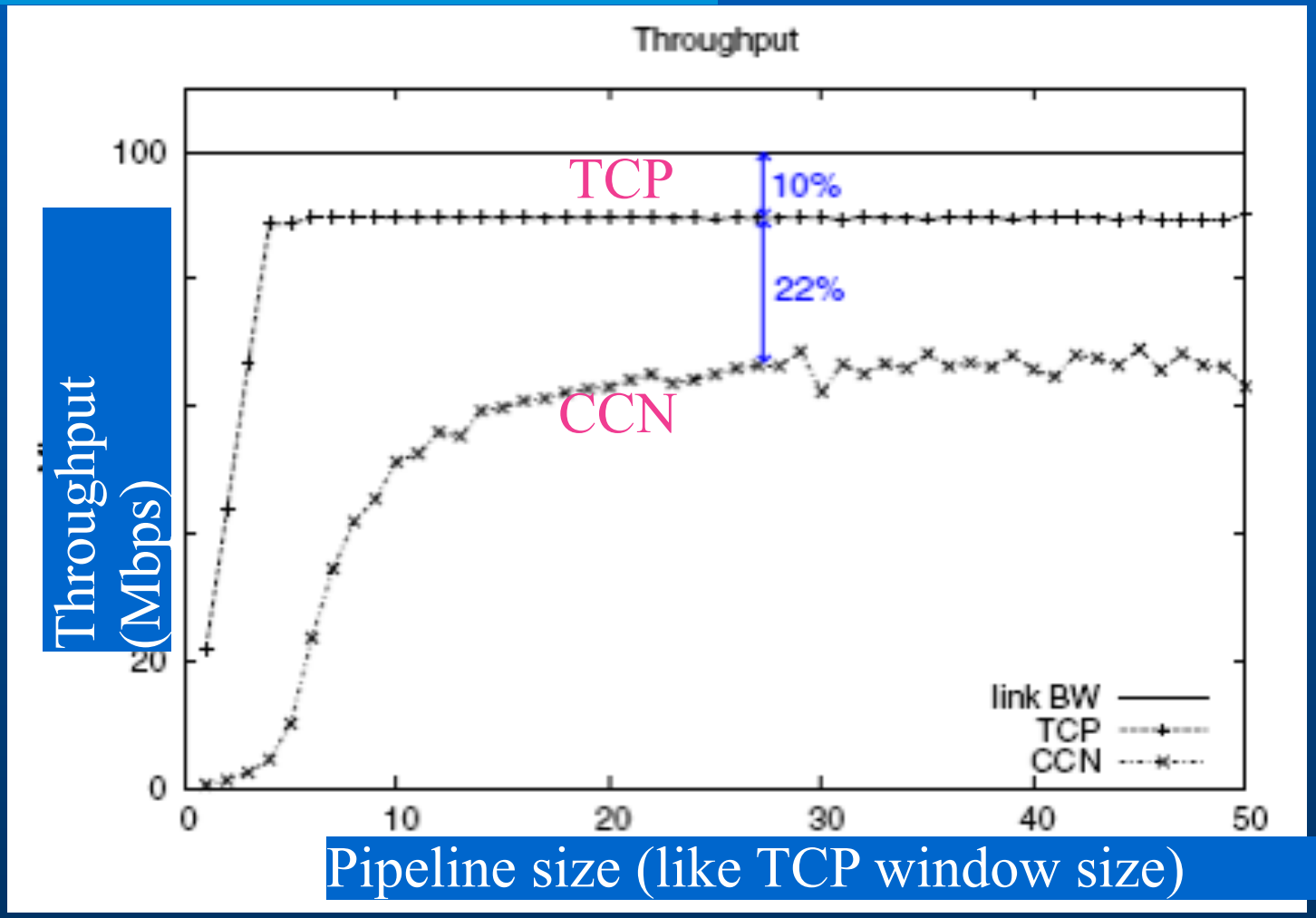
- Newest version after v2:

parc.com/van/talks/mit10-08/v2 <rightmost sibling>
(fails since there is no newer version)

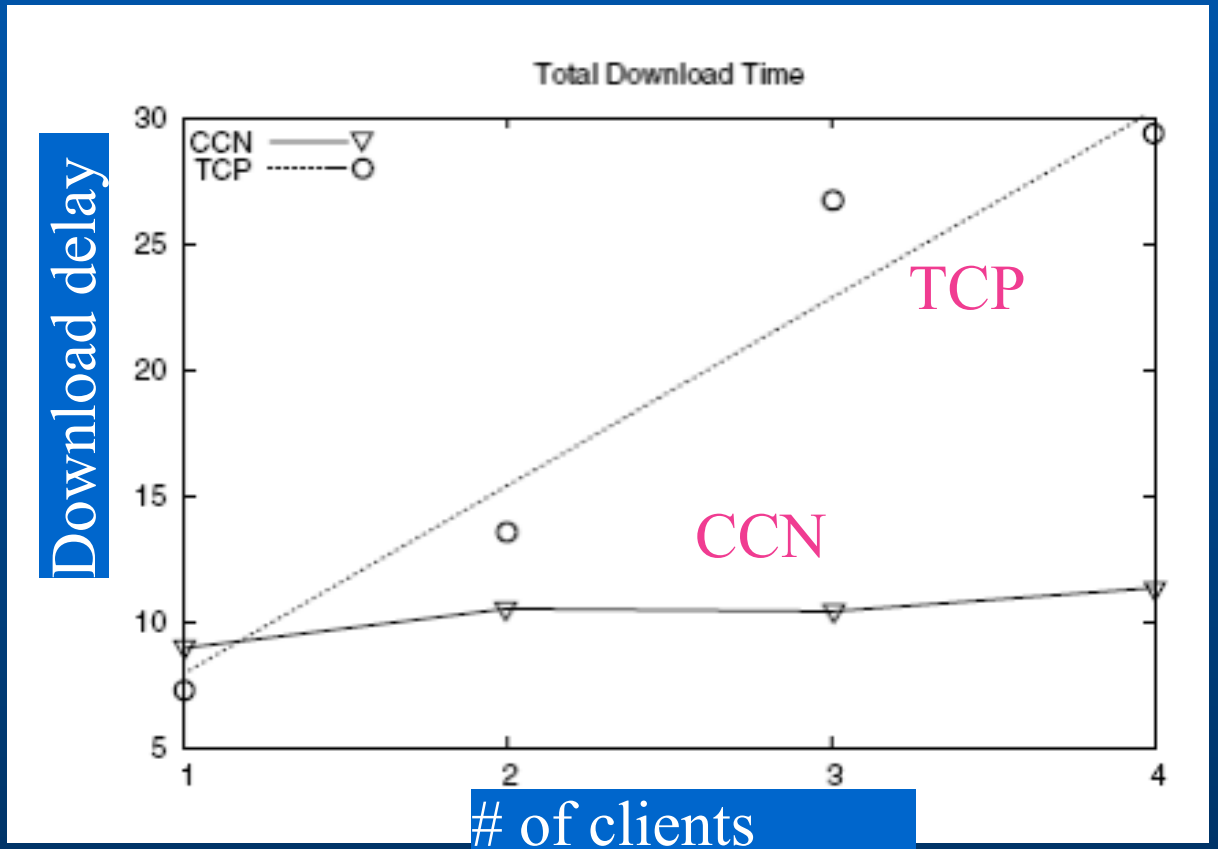
- Next available chunk after s1:
  parc.com/van/talks/mit10-08/v2/s1

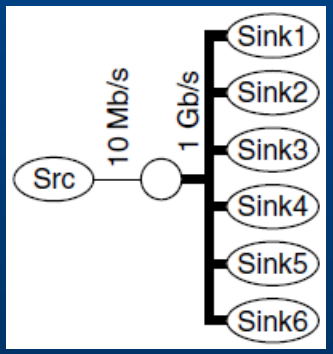
Conventions:
- *name tree child nodes are lexically ordered*
- *<next> assumed if no relationship specified*

Bulk-data transfer performance comparison



Shared-content performance comparison





Wrapup

- CCN node is as simple as an IP node:
  - same memory requirements
  - same computational requirements (with option to increase security)
- CCN offers simple, robust, secure single-point configuration.
- CCN does near optimal content distribution.
- Network, applications and users all share the same model of communication.