Introduction to Information Centric Networking

Andrés Arcia-Moret
N4D Lab, Computer Laboratory
University of Cambridge

Agenda

- Motivation
- Information Centric Networking
- Implementations: NDN, DONA, NetInf, Juno, PURSUIT
- PURSUIT nitty-gritties
- Conclusions

Motivation

- Shift from resource sharing to information sharing
  - host centric: TCP/IP
  - information based: identification, retrieval (communication functions)
- Establishing comm relationship on information interest rather than end-hosts

Information Centric Networking

- Problem mostly addressed from high-level routing or information management perspective
- Lately, it has been also exploited more efficiently (based on BF)
  - in the hardware and processing complexity at FW node
  - resource allocation
  - TE at intra-domain
  - Dissemination of inter-domain

Information Centric Networking

- rather than seeing where is in a name (like IP does) we see what is in a name.
- then we can change physical and topological location transparently.
- exposes the request style abstraction unlike the socket API
- differences with host centric: naming, uniquely naming every (replicated) object. routing, ICN uses bindings between points, and optimal content src. security, ICN secure integrity of objects rather than channels. API, exposed to produce and consume.

Other salient characteristics

- No connection oriented sessions: as communication becomes receiver driven, thus no need for sender cooperation for in-order reliability. Better congestion/flow control due to convenient distribution.

- Content and location scoping: explicit separation between what (objects) and where (location).

- Resilience through replication.

ICN Implementations

DONA: Data Orient
Network Architecture

made me remember the paper saying that with some configuration tricks one can get ICN networks.

- ICN as an alternative to DNS
- Content names are: P:L, P being the cryptographic hash, and L the label that identifies content.
- Resolution Handlers (RH) store <P:L, content location> per domain.
- Resembles BGP tree topology, thus consumer asks local RH. If no reference is found then it propagates in the tree till found. Then shortcut the way back to the consumer (possibly through TCP/IP).
- DONA routes every request embedded within regular data packets.

[Koponen et al., 2007] Koponen, et al (2007). A data-oriented (and beyond) network architecture. In Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM ’07, pages 181–192, New York, NY, USA. ACM.

[Koponen et al., 2007] Koponen, et al (2007). A data-oriented (and beyond) network architecture. In Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM '07, pages 181–192, New York, NY, USA. ACM.

[Tyson et al., 2013] Tyson, G., Sastry, N., Cuevas, R., Rimac, I., and Mauthe, A. (2013). Where is in a name? a survey of mobility in information-centric networks.

DONA

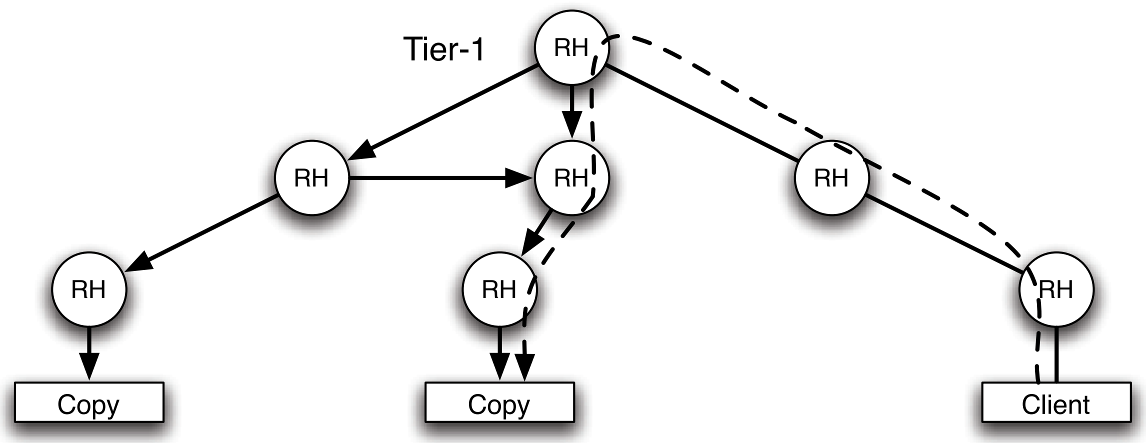


Figure 1: Registration state (solid arrows) in RHs after copies have registered themselves. RHs route client-issued FIND (dashed arrow) to a nearby copy.

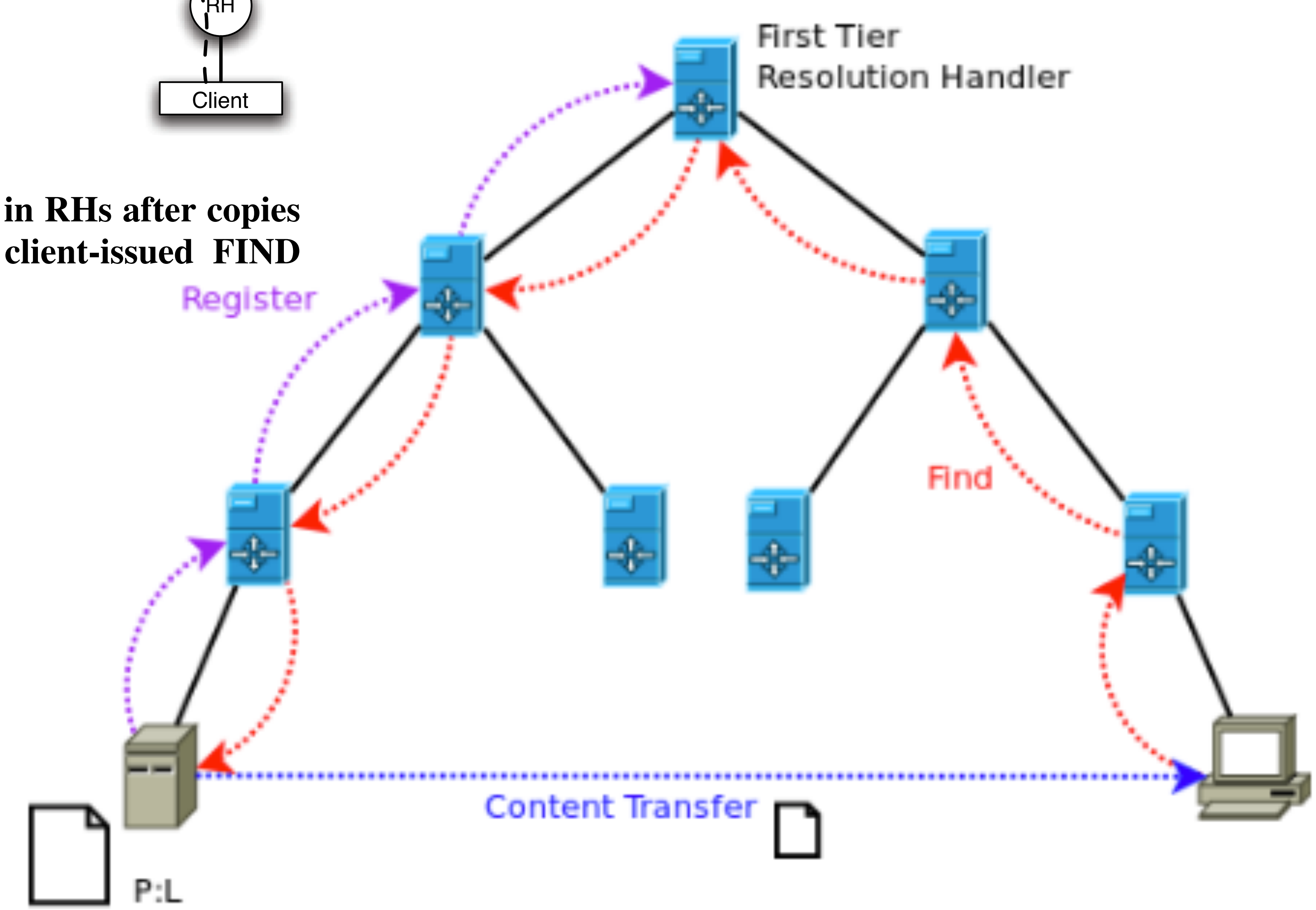


Fig. 2. Overview of DONA

DONA: FIND msg

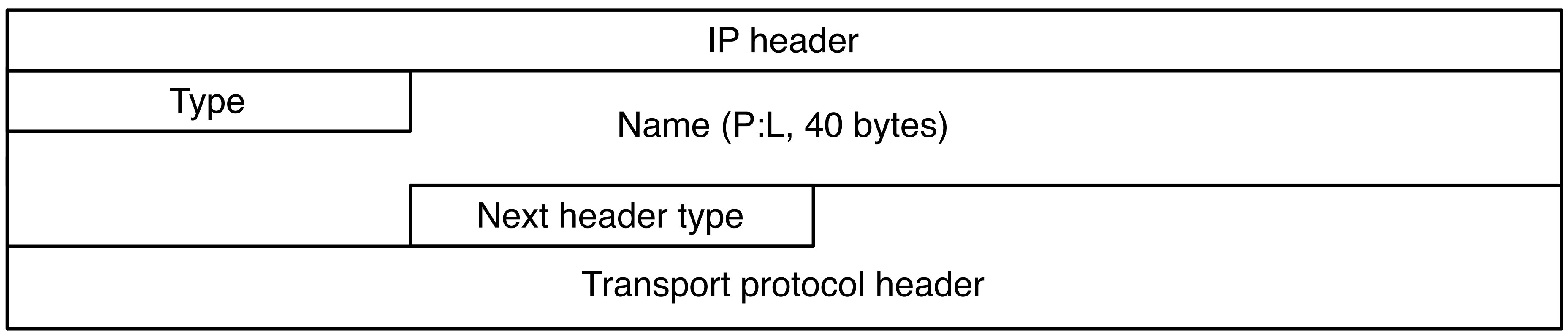


Figure 3: Protocol headers of a FIND packet. Type is to separate FINDs from their responses.

ICN of DONA

- P:L reproduces the scoping model (easily reproducible)
- Data Handlers corresponds to a functional rendezvous (having sub domains)
- IP routing fabric: topology (completely decentralised though in IP), and forwarding (keeps state).

NDN — CCN, CCNX

- Named data networking
- Flexible hierarchical structure allowing various namespaces
- Interest packets sent through to the network to the content.
    - Longest prefix match. Aggregated name hierarchy
- Way back through breadcrumbs in PIT
- Content item’s naming reflect the underlying topology (thus can potentially create state explosion in the core network).

[Tyson et al., 2013] Tyson, G., Sastry, N., Cuevas, R., Rimac, I., and Mauthe, A. (2013). Where is in a name? a survey of mobility in information-centric networks.

NDN- CCN, CCNX

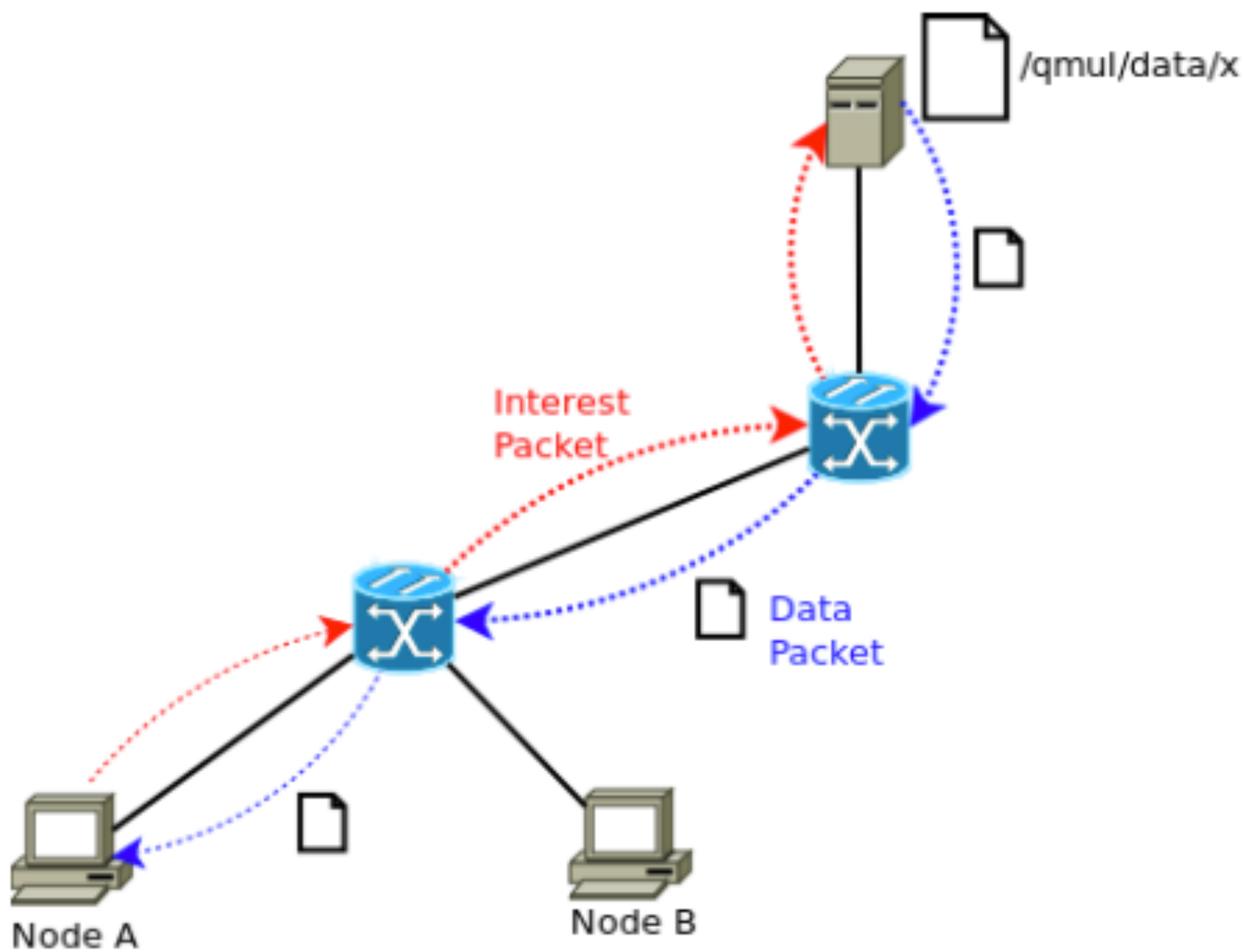
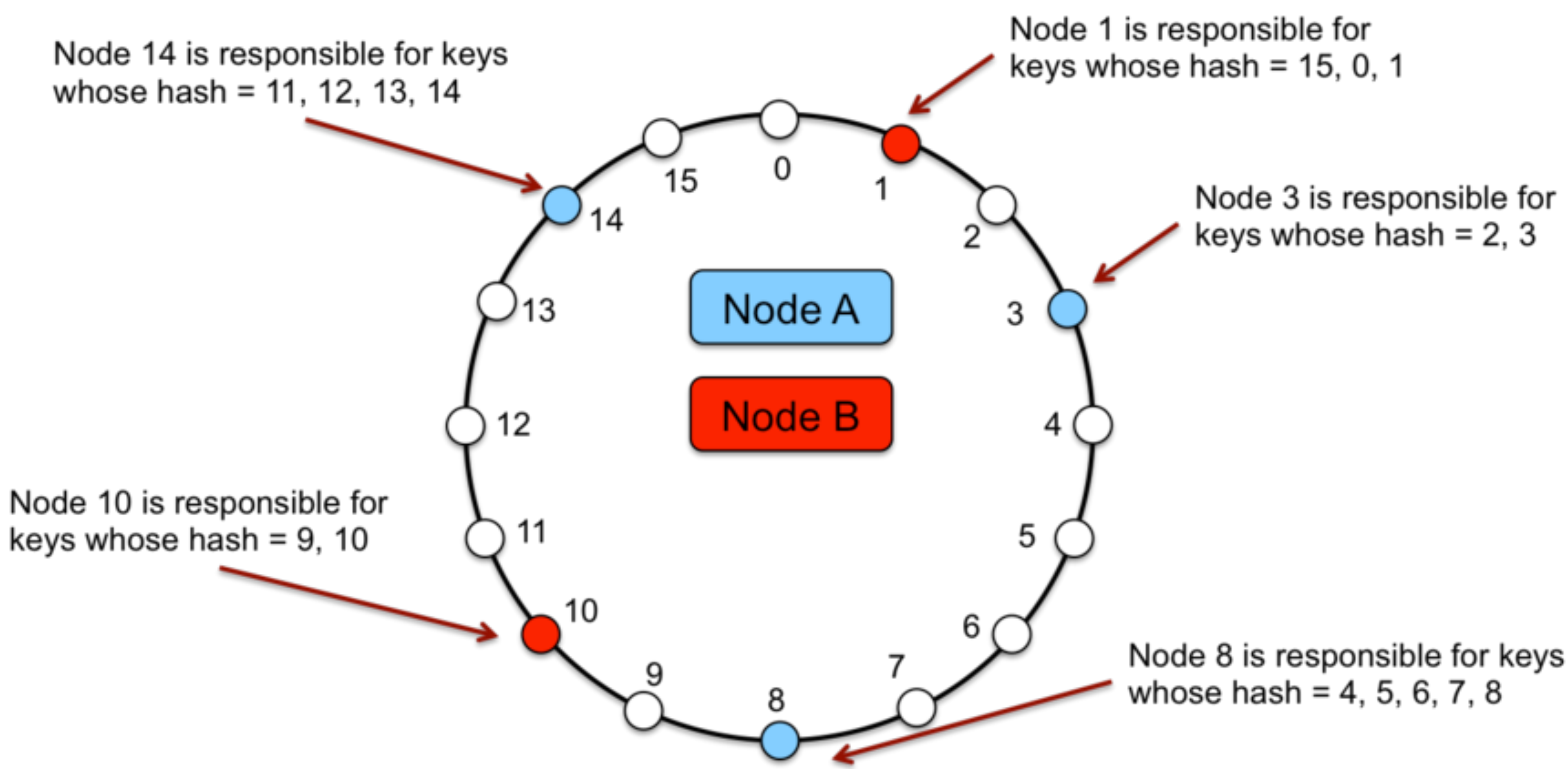


Fig. 1. Overview of NDN

NetInf

- Relies on Name Resolution (NR) service.
- Publishing Named Data Objects and locators (named routing hints) to be discovered later.
- Provide in a multilevel DHTs for finding the location (or the optimal location).
- Self certified NDO mapped to a set of locators.
    - Requester-controlled lookups with eventual list of potencial sources, to choose for optimal (s).
    - MDHT-controlled mode, single consumer matched with single source (by MDHT)
- Content delivery can be done in many ways (e.g., in-router caching)

Distributed Hash Tables



[Dannewitz et al., 2013] Dannewitz, C., Kutscher, D., Ohlman, B., Farrell, S., Ahlgren, B., and Karl, H. (2013). Network of information (netinf) - an information-centric networking architecture. Comput. Com- mun., 36(7):721–735.

[Tyson et al., 2013] Tyson, G., Sastry, N., Cuevas, R., Rimac, I., and Mauthe, A. (2013). Where is in a name? a survey of mobility in information-centric networks.

NetInf

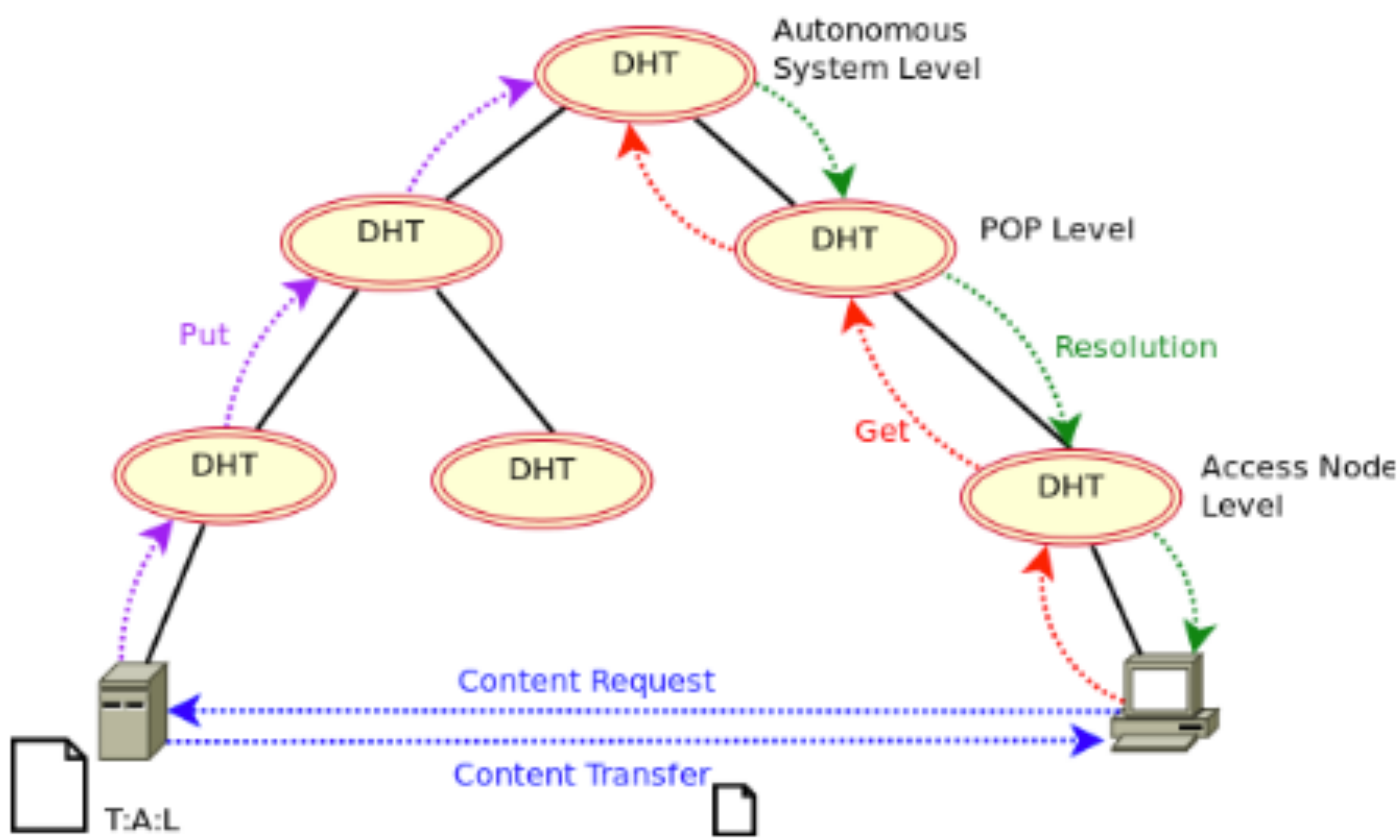


Fig. 3. Overview of NetInf

Juno

- Placement of ICN at the middleware layer
- Flat self-certifying IDs indexed on DHT called Juno Content Discovery Service (JCDS).
- Can probe third party index services such as eMule.
- Delivery framework retrieves the content by using dynamically attachable protocol plug-ins.
- Intelligent reconfiguration for different sources based on: performance, cost, resilience.

[Tyson et al., 2013] Tyson, G., Sastry, N., Cuevas, R., Rimac, I., and Mauthe, A. (2013). Where is in a name? a survey of mobility in information-centric networks.

[Tyson et al., 2012] Tyson, G., Mauthe, A., Kaune, S., Grace, P., Taweel, A., and Plagemann, T. (2012). Juno: A middleware platform for supporting delivery-centric applications. ACM Trans. Internet Technol., 12(2):4:1–4:28.

Juno

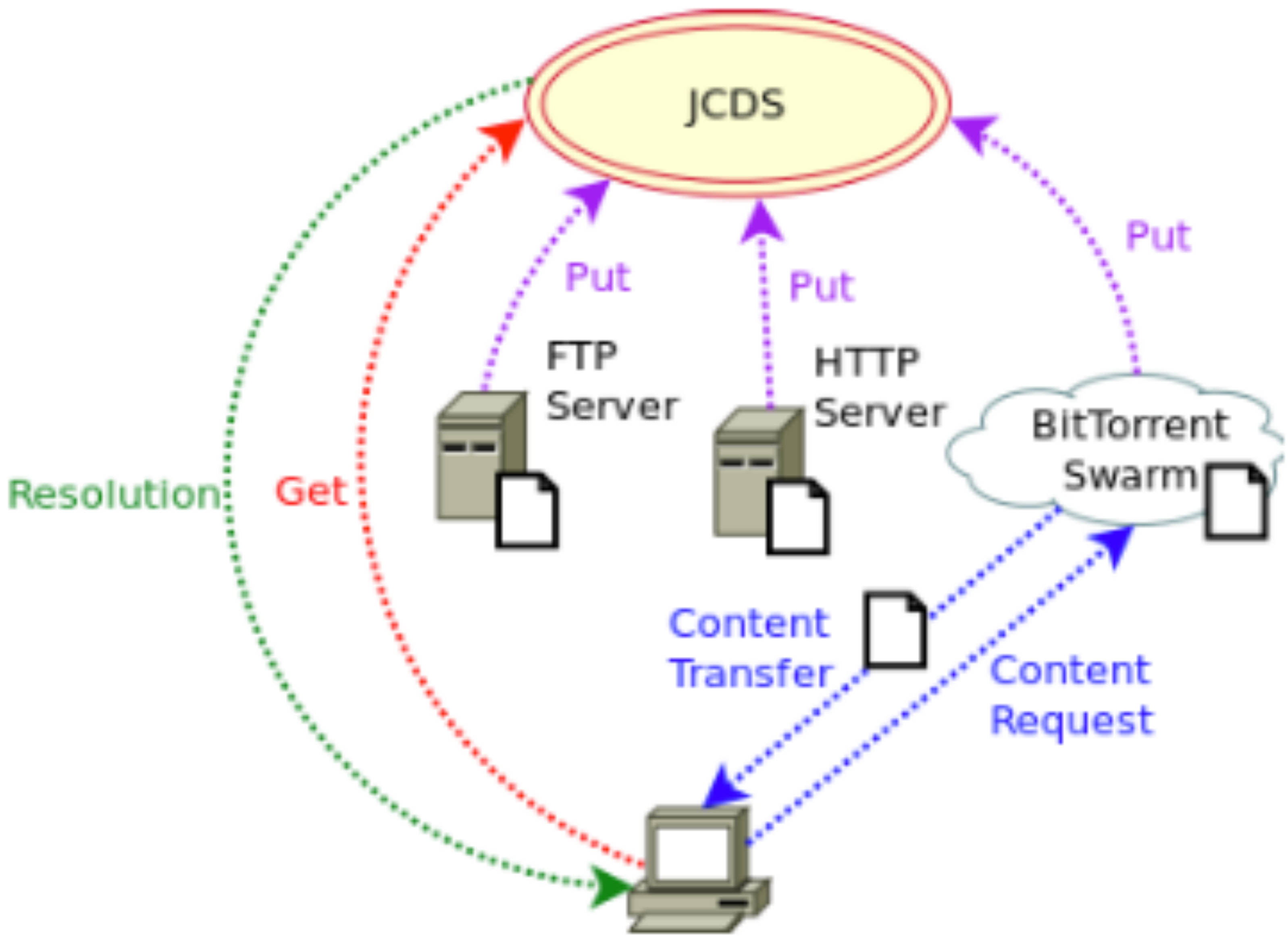
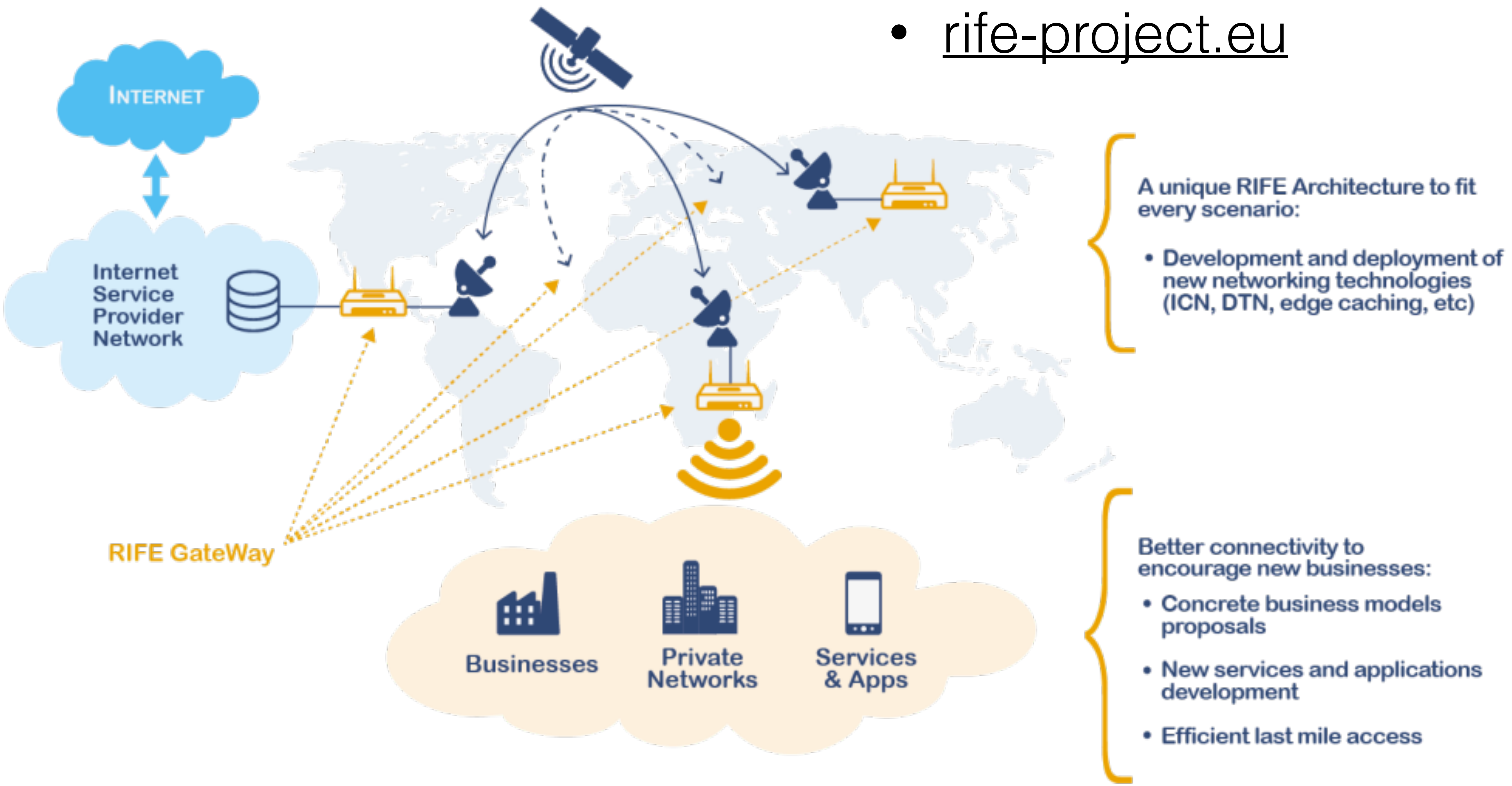


Fig. 5. Overview of Juno

RIFE (a word)

- rife-project.eu



PURSUIT

- A systems approach that operates on graphs of information with a late (as late as possible) binding to a location at which the computation over this graph is going to happen, enables the full potential for optimisation!

- This systems approach requires to marry information & computation (and with it storage) into a single design approach for any resulting distributed system

source: PURSUIT FP7 public dissemination reports.

Starting Point: Solving Problems in Distributed Systems

- One wants to solve a problem, each of which might require solving another problem.

- Example:

    - Send data from A to B(s), eventually solving fragmentation on a restrained link(s)

—> Computation in distributed systems is all about information dissemination (pertaining to a task at hand)

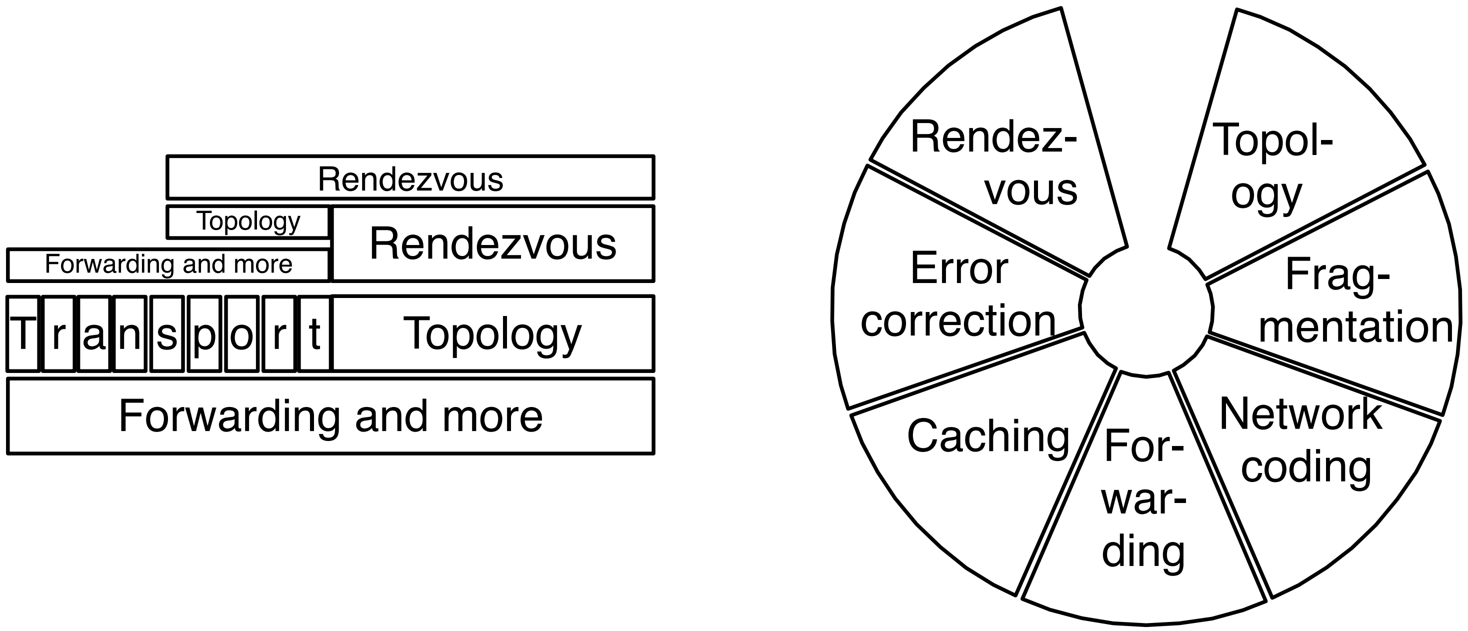
source: PURSUIT FP7 public dissemination reports.

Design Tenets

- Provide means for identifying individual information (items)
  - Can be done via labelling or naming
- Provide means for scoping information
  - Allows for forming DAGs (directed acyclic graphs)
- Expose core functions
  - Rendezvous, topology management, and forwarding
- Common dissemination strategy per sub-structure of information
  - Define particulars of functional implementation and information governance (naming type: flat), adapting to a particular computational problem
- Expose service model
  - Can be pub/sub

Trossen, D. and Parisis, G. (2012). Designing and realizing an information-centric internet. IEEE Communications Magazine, 50(7):60–67.

Layered Model



Recursive models

Figure 1: Rendezvous, Topology, Forwarding

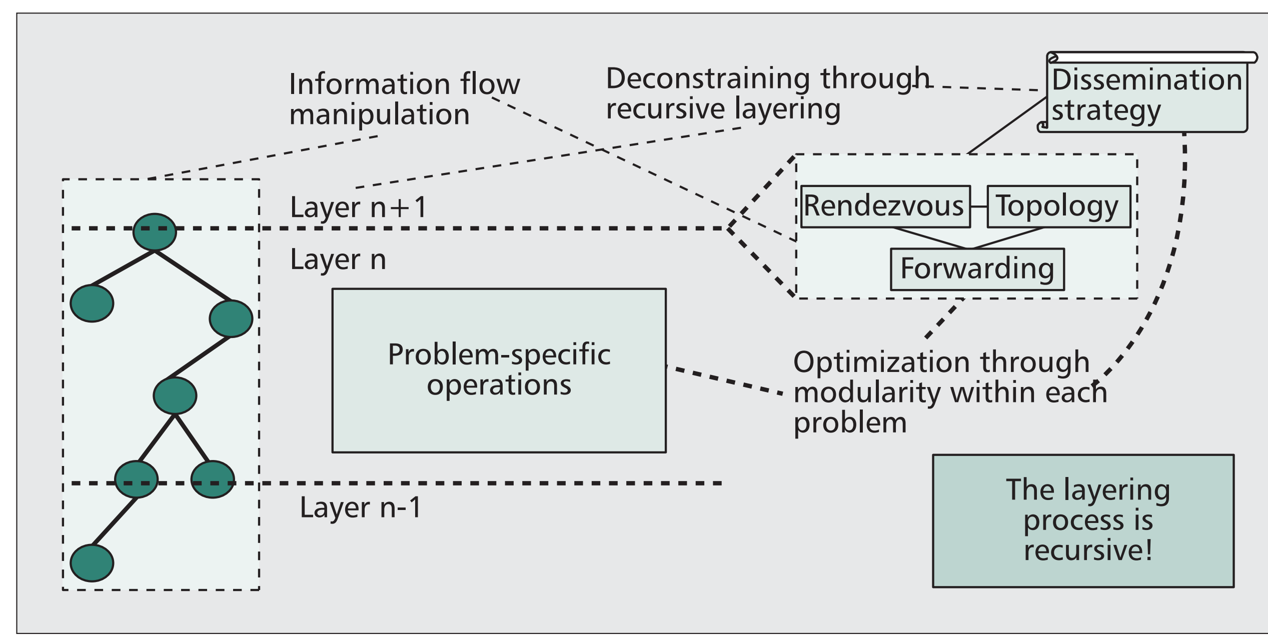
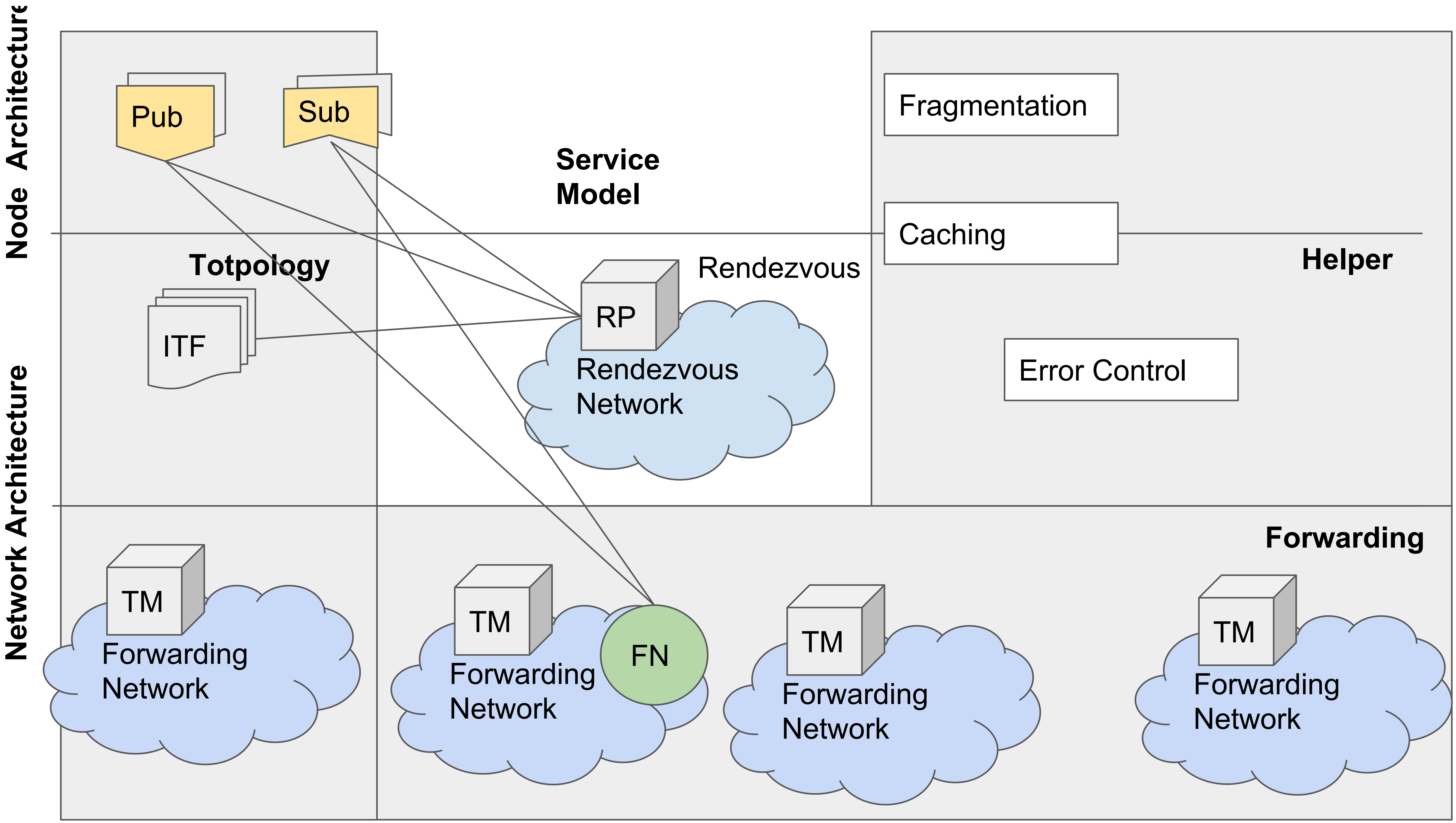


Figure 1. *Functional layered model.*

[Trossen and Parisis] Trossen, D. and Parisis, G. (2012). Designing and realizing an information-centric internet. IEEE Communications Magazine, 50(7):60–67.

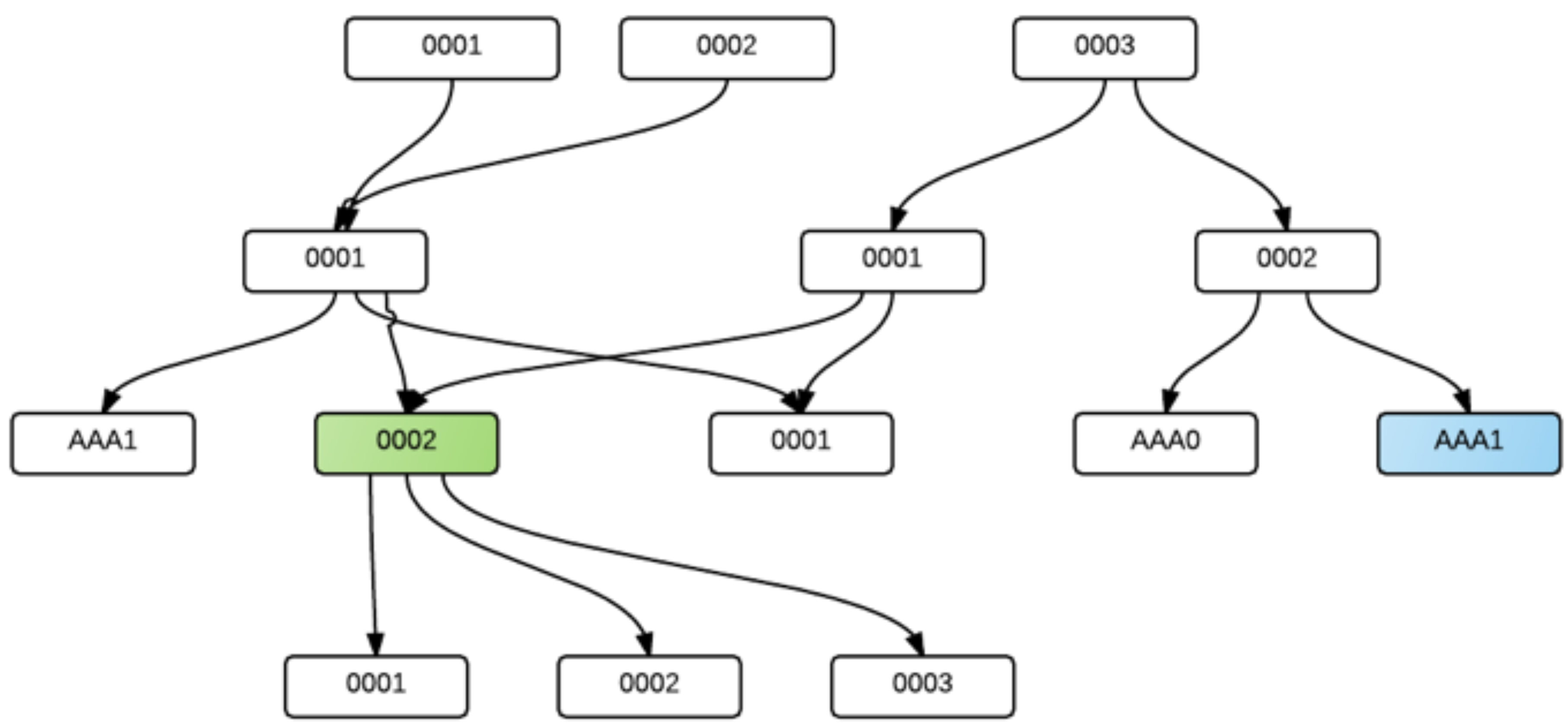
[Jokela et al., 2009] Jokela, P., Zahemszky, A., Esteve Rothenberg, C., Arianfar, S., and Nikander, P. (2009). Lipsin: Line speed publish/subscribe inter-networking. SIGCOMM Comput. Commun. Rev., 39(4):195– 206.

Global Architecture



source: PURSUIT FP7 public dissemination reports.

Information Graph



Information ID: /0001/0002/AAA1

Scope ID: /0001/0001/0002, /0003/0001/0002, /0002/0001/0002

Information Semanti… immutable versus mut…

I should be commenting on the specifics of this work and its relationship to DONA, CCN, etc. (how these other works are seen as dissemination strategies)

- Documents
  - Each RId points to immutable data (e.g., document version)
  - Not well suited for real-time type of traffic
  - Each item is identifiable throughout the network
- Each RId points to channel of data (e.g., a video stream), i.e., the data is mutable (channel in the item)
  - Well-suited for video type of traffic
  - Problems with caching though (since no individual video segment visible)

Built-in multicast capability

- Information is sent along a route of (intra-domain) hops in the Internet
- -> Requires some form of minimal state in each hop
    - If forwarding on names, limiting this state is hard/impossible

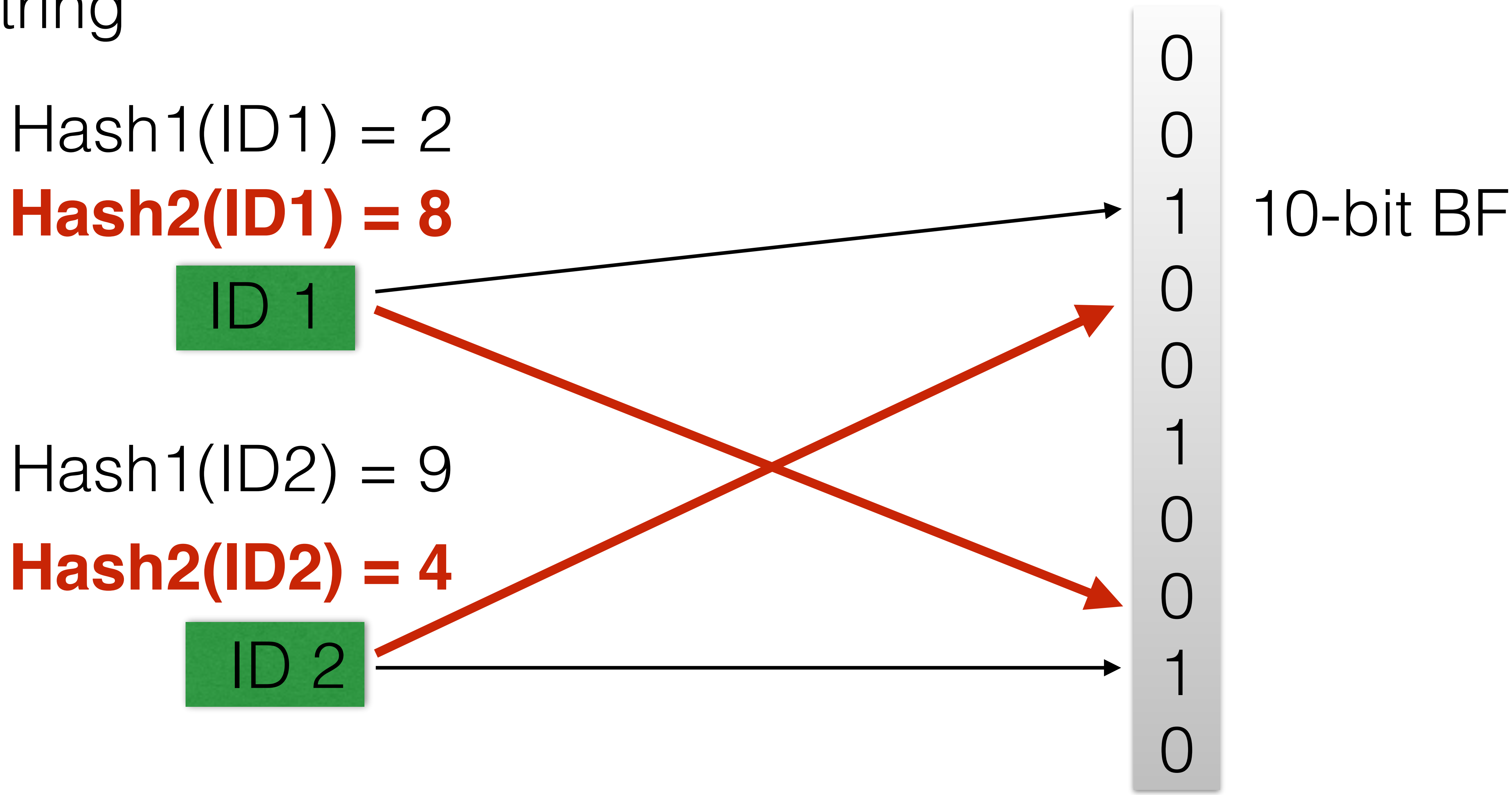
What if we could instead include the state in the packet?

What are Bloom Filters?

- Test if a piece of information has been inserted in the BF:
    - All turned-on after a set of hash functions have been tested? Then, positive response!

Bloom Filters

- Data structure for compressing items into a bit string



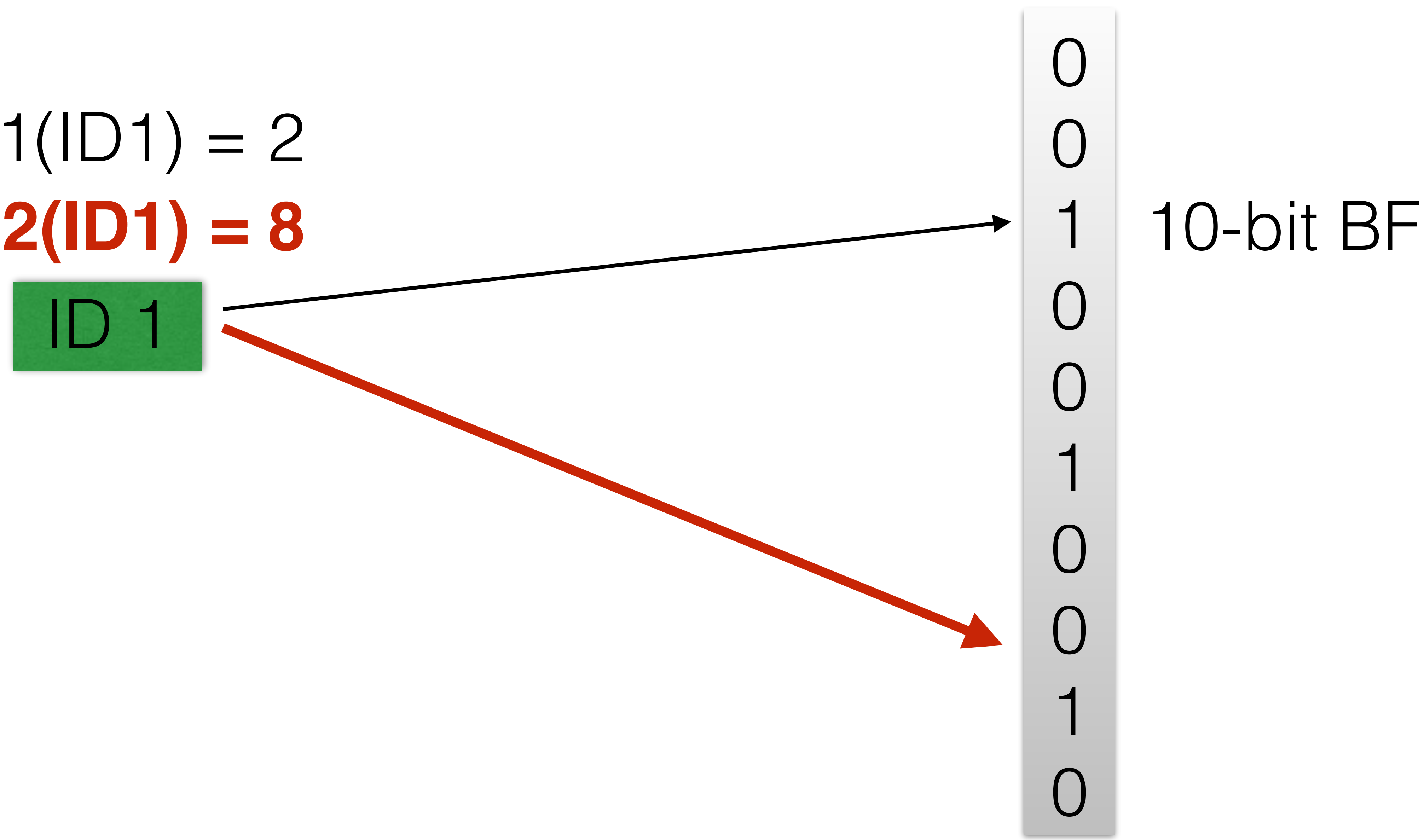
Bloom Filter

Test if “Data 1” has been inserted in the BF

All corresponding bits are set => positive response!

Hash1(ID1) = 2

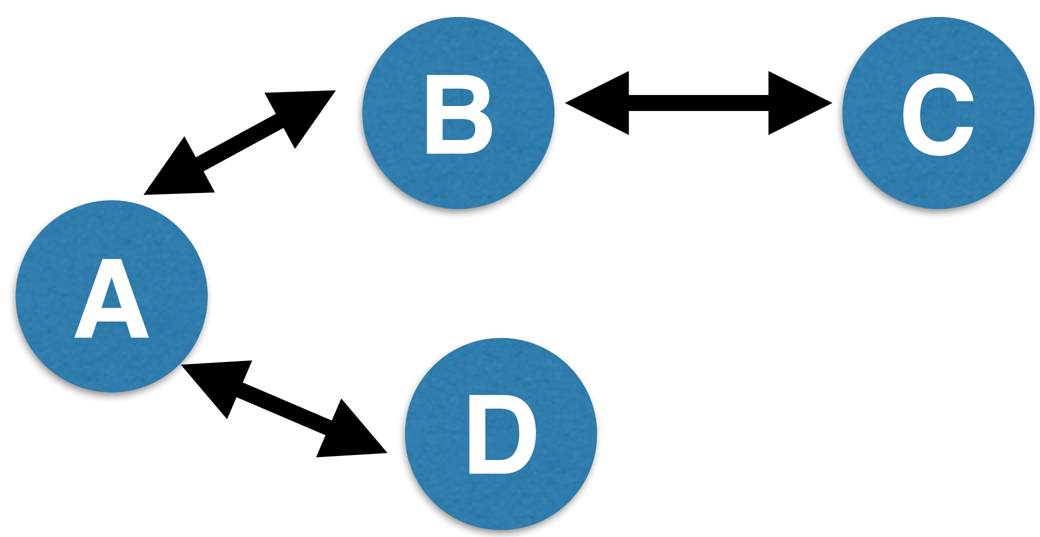
Hash2(ID1) = 8



Line Speed Publish/Subscribe Inter-Network (LIPSIN)

Zorglub

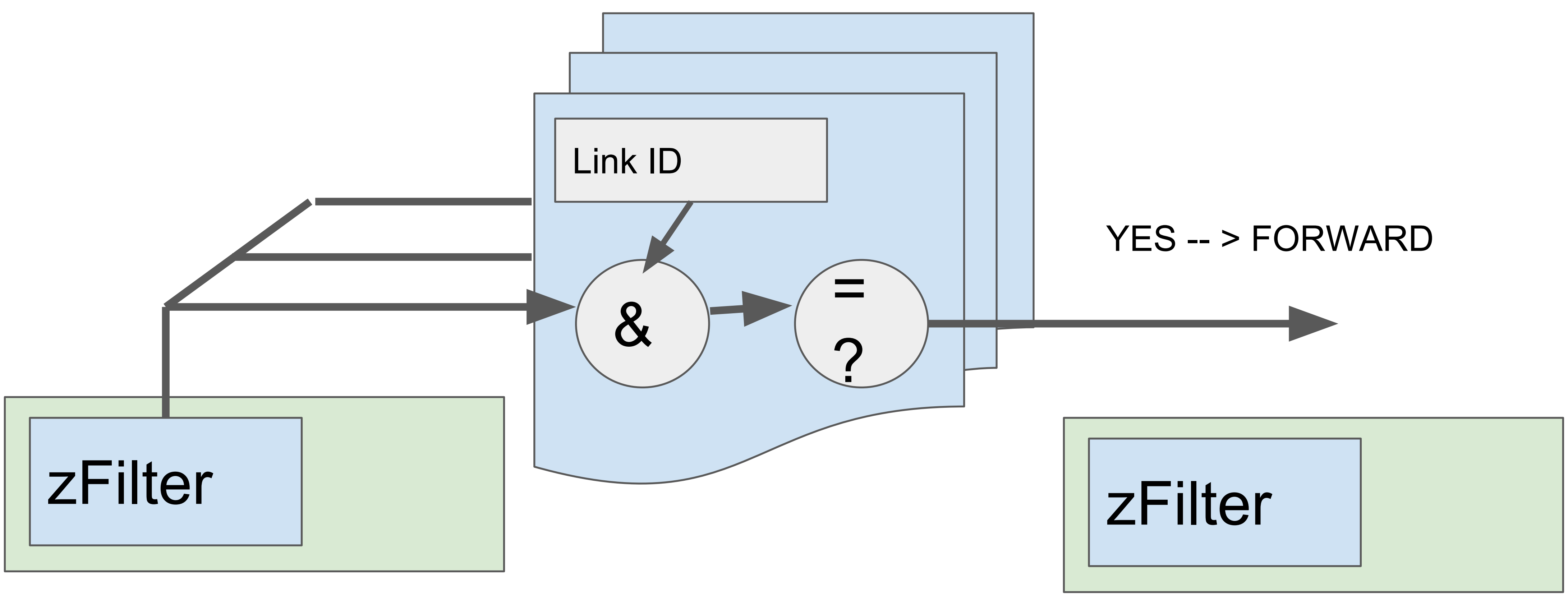
- Line speed forwarding with simplified logic
  - Links are (domain-locally) named instead of nodes (LId), therefore there is no equivalent to IP addresses
- Link identifiers are combined in a bloom filter (called zFilter) that defines the transit path
- Advantages
  - Very fast forwarding
  - No need for routing tables
  - Native multicast support



A->B	0	**1**	0	0	0	**1**	0	0	**1**
B->C	**1**	0	0	0	0	**1**	**1**	0	**1**
zF: A -> B -> C	**1**	**1**	0	0	0	**1**	**1**	0	**1**

Forwarding Decision

- Forwarding decision based on binary AND and CMP
  - zFilter in the packet matched with all outgoing Link IDs
  - Multicasting: zFilter contains more than one outgoing links



False Positive in Forwarding

- False positives occur when test is positive in a given node despite nonhashed
- LId (probability for consecutive false positives is multiplicative!)
    - Increase with number of links in a domain (since more data is hashed into constant length Bloom filter)
- Two immediate solutions:
    - Use Link Identity Tags: tag a single link with N names instead of one, then pick resulting Bloom filter with lowest false positive probability
- Virtual trees: fold “popular” sub-trees into single virtual link, i.e., decrease number of LIds to be used

[Jokela et al., 2009] Jokela, P., Zahemszky, A., Esteve Rothenberg, C., Arianfar, S., and Nikander, P. (2009). Lipsin: Line speed publish/subscribe inter-networking. SIGCOMM Comput. Commun. Rev., 39(4):195– 206.

Forwarding Efficiency

- Simulations with
  - Rocketfuel
  - SNDlib
- Forwarding efficiency with 20 subscribers
  - ~80%

-> suited for MAN-size multicast groups

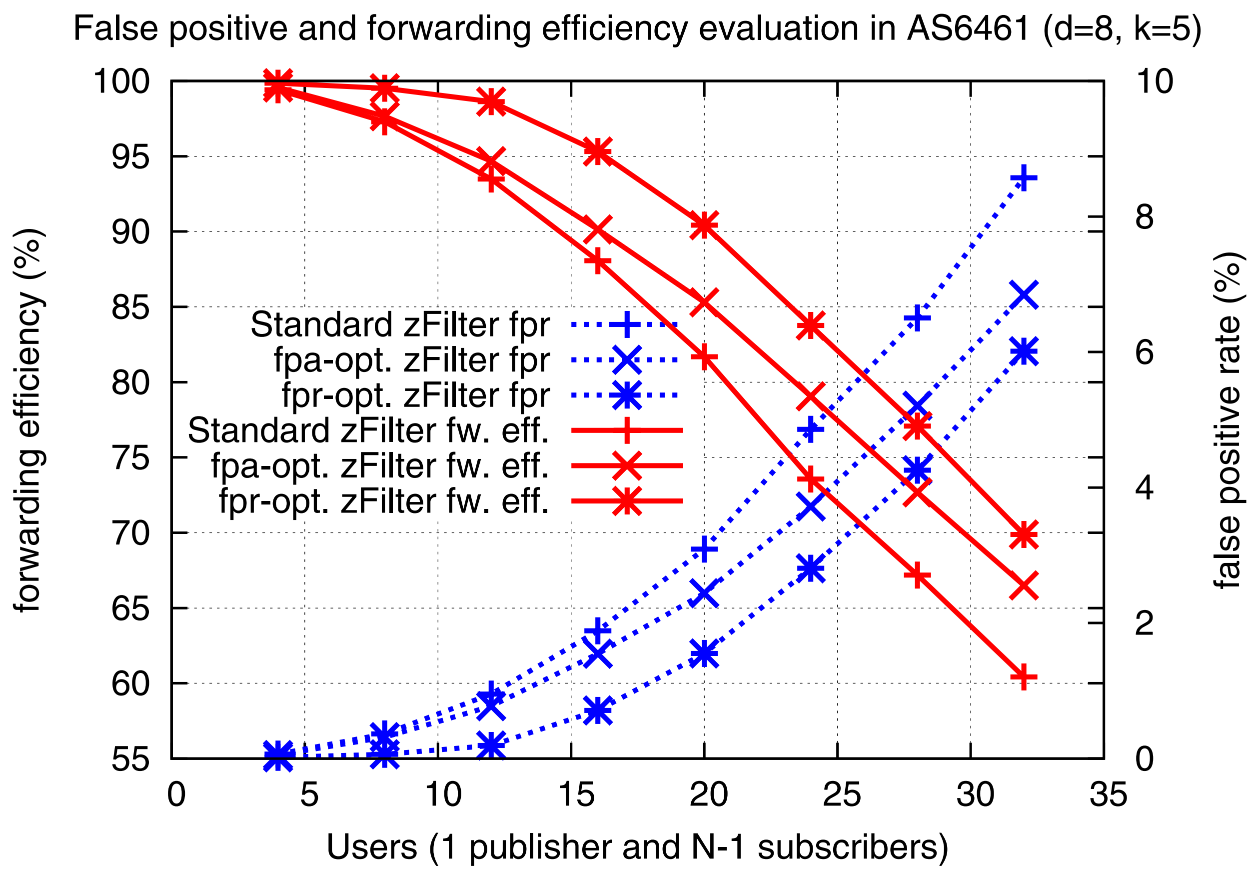
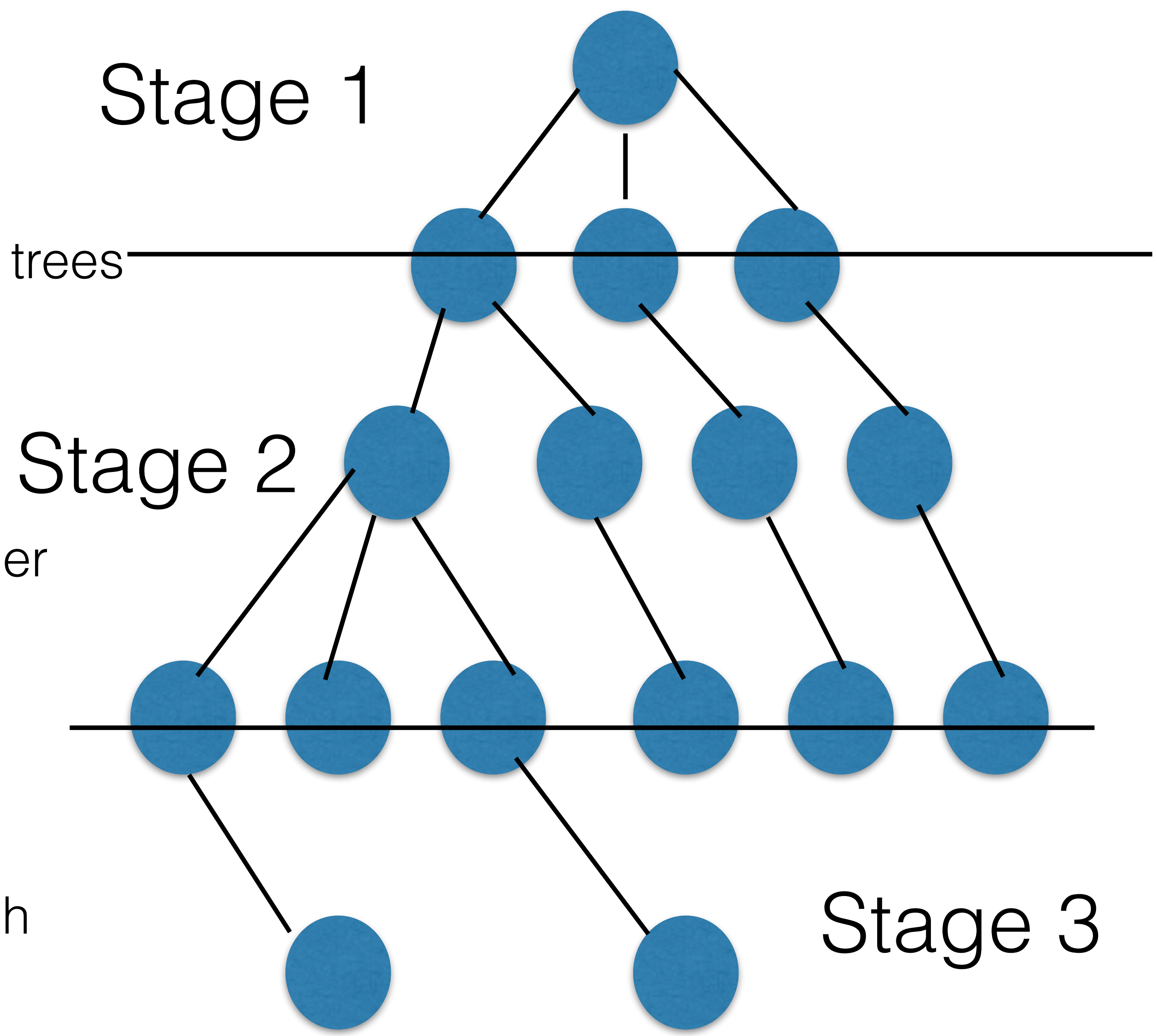


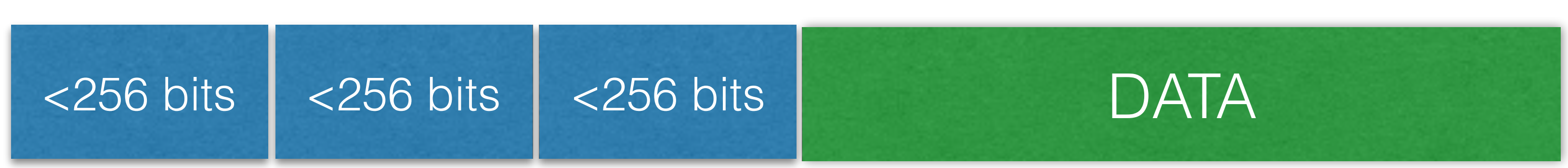
Figure 5: ns-3 simulation results for AS 6461.

AS6461 Abovenet (US) 367 (R -ISP) 1,000 (L -ISP) 2,259 (R - CUST) 1,400 (L - CUST)

Multi Stage BF

- Divide a delivery tree into stages
  - Generally, each stage has individual trees
  - Operation performed at topology manager
- Provide single BF forwarding identifier per stage
- Concatenate all stage into variable size header
  - Perform BF-based forwarding at each stage
  - Remove appropriate BF after each stage

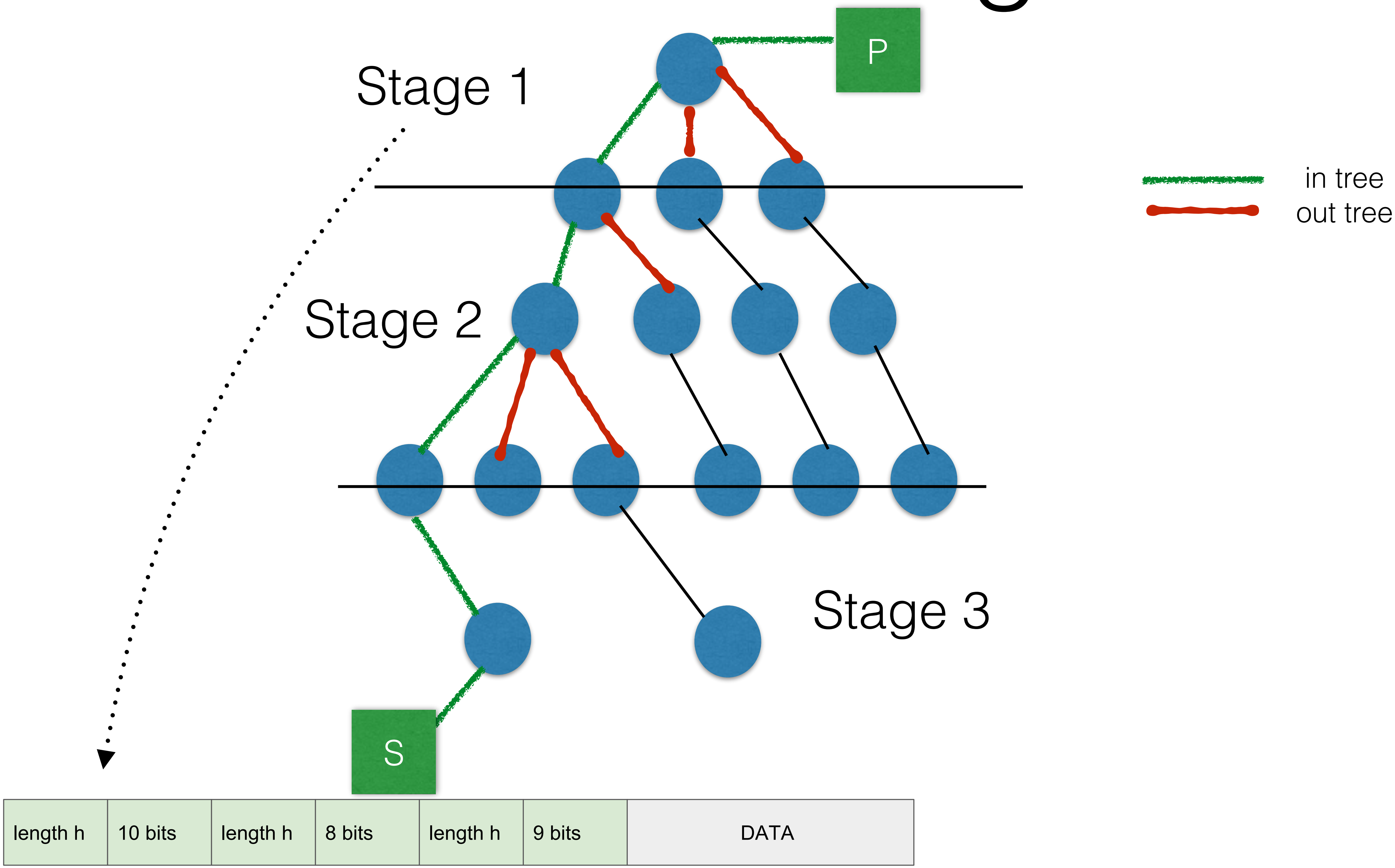




Topology Formation

- Calculate a tree with 0 false positives for given <pub,subs> relationship
  - Within each stage:
  - Define **in_tree** as the set of LIds being in the tree and **out_tree** as the ones that are not
  - Determine minimal length of BF that can hold **in_tree** with *P(false positive)=0* (also taking into account **out_tree**)
- Determine BF through ORing **in_tree** into BF
  - Test if BF would cause false positives, then increase the length if so.
  - Determine overall header by joining all possible stages
  - Write length of stageBF through Elias omega encoding (https://en.wikipedia.org/wiki/Elias_omega_coding)
  - Write the stageBF

Summarising

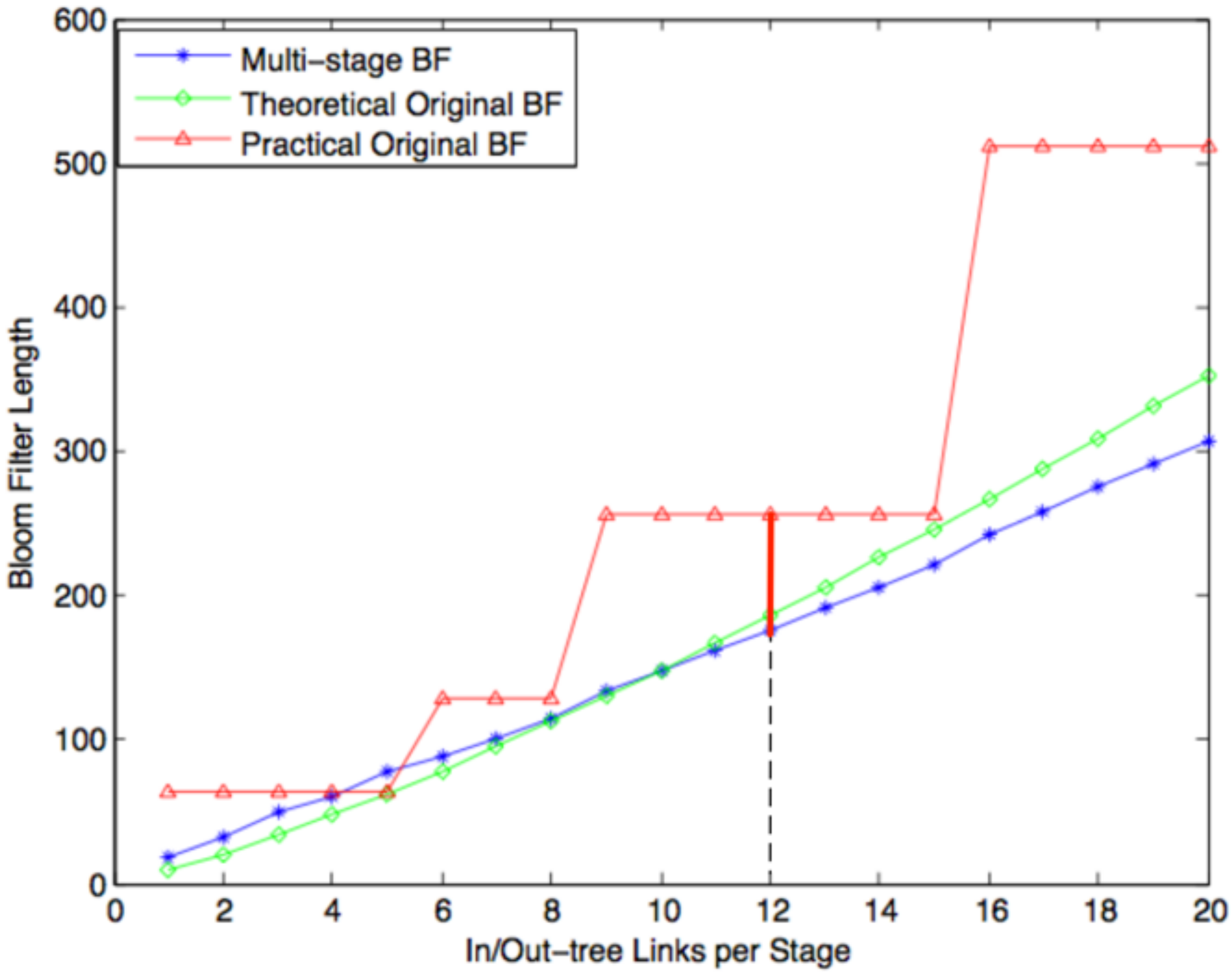


Pros and Cons

- Advantages
  - Arbitrary tree size (limits may exist for maximum size for variable length header)
  - Tradeoff between false positive rate and header size (in this approach false positives is zero)
  - Single hop vs multi-hop stages possible (single hops naturally limit BF anomalies)
  - Lends itself to inter-domain as well as intra-domain forwarding
- Disadvantages
  - Higher complexity in forwarding (decompress/compress)
  - Higher overhead due to variable length, but overhead reduces as you traverse the tree

source: PURSUIT FP7 public dissemination reports.

Header Length



Prototype

Blackadder

- Implements design tenets
- Based on Click router platform
    - Easy user/kernel space support
    - Portable to other OSes
    - Compatible with ns-3
- Available at: https://github.com/fp7-pursuit/blackadder
- Domain-local throughput reaches 1GB/s

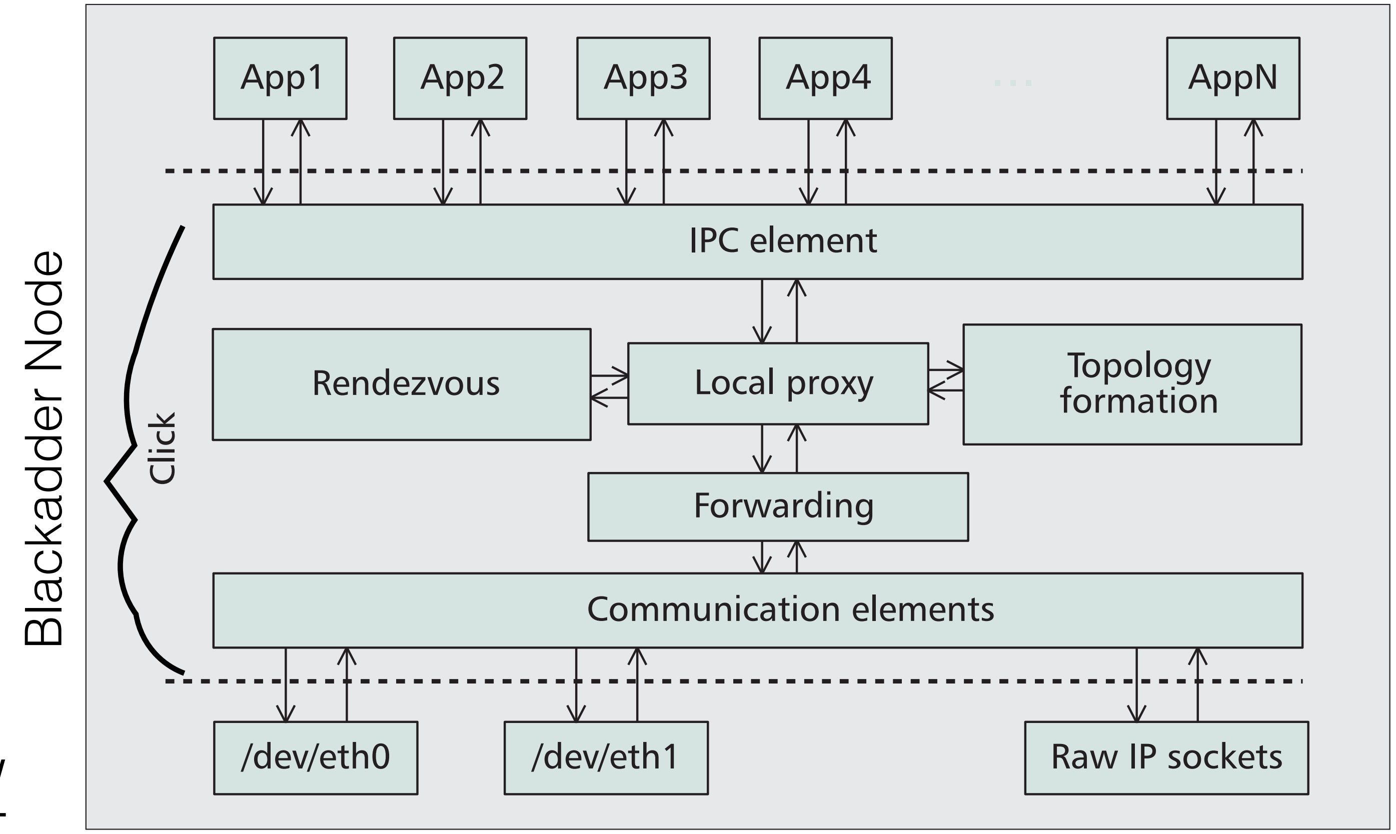
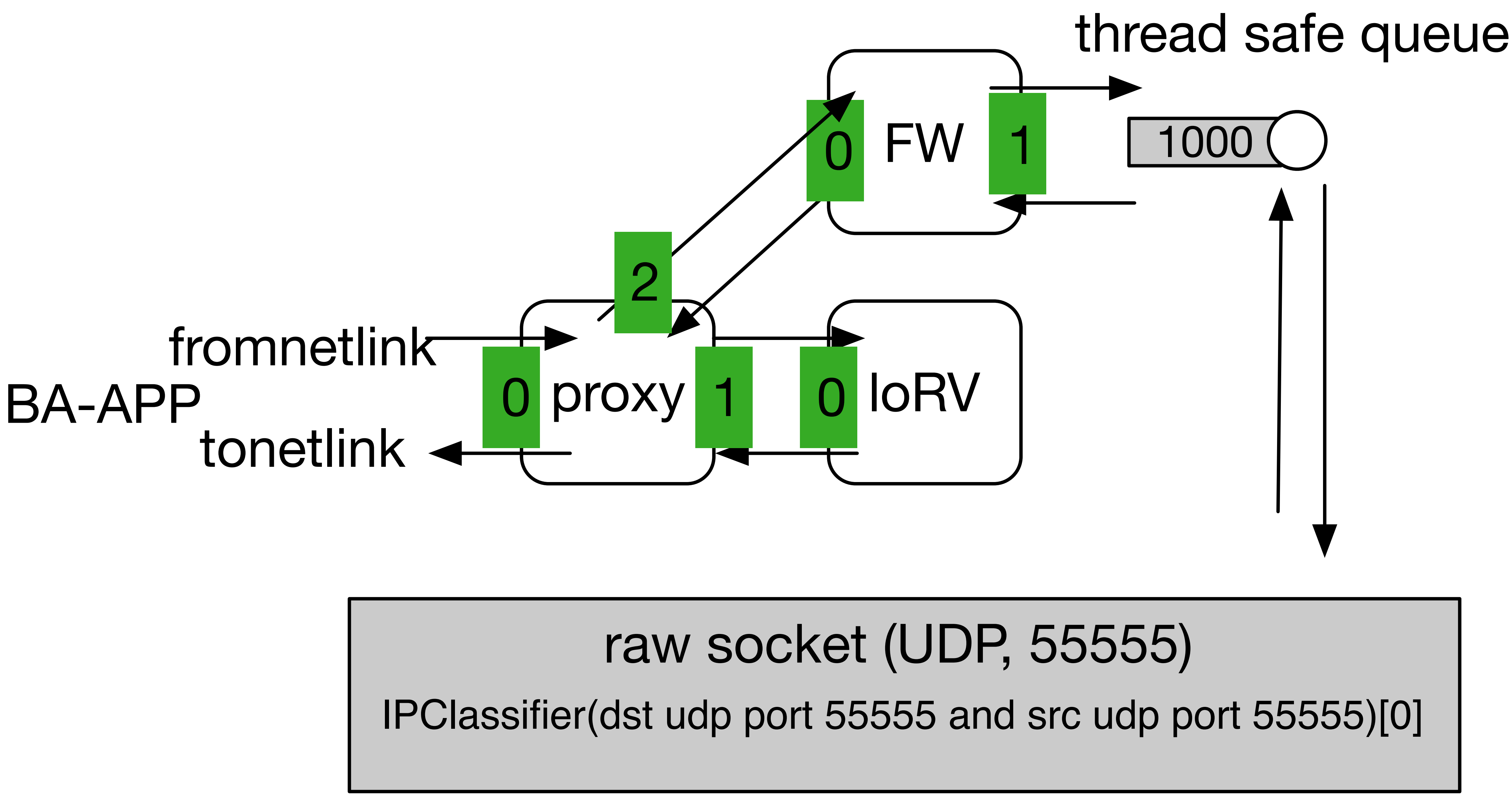


Figure 3. *Node implementation architecture.*

*Trossen, D. and Parisis, G. (2012). Designing and realizing an information-centric internet. IEEE Communications Magazine, 50(7):60–67.

Click scheme



Node Structure

require(blackadder);

globalconf::GlobalConf(MODE ip, NODEID 00000001,
DEFAULTRV
0000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000
0000001000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000,
TMFID
0000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000
0000001000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000,
iLID
0000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000
0000001000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000);

Node Structure

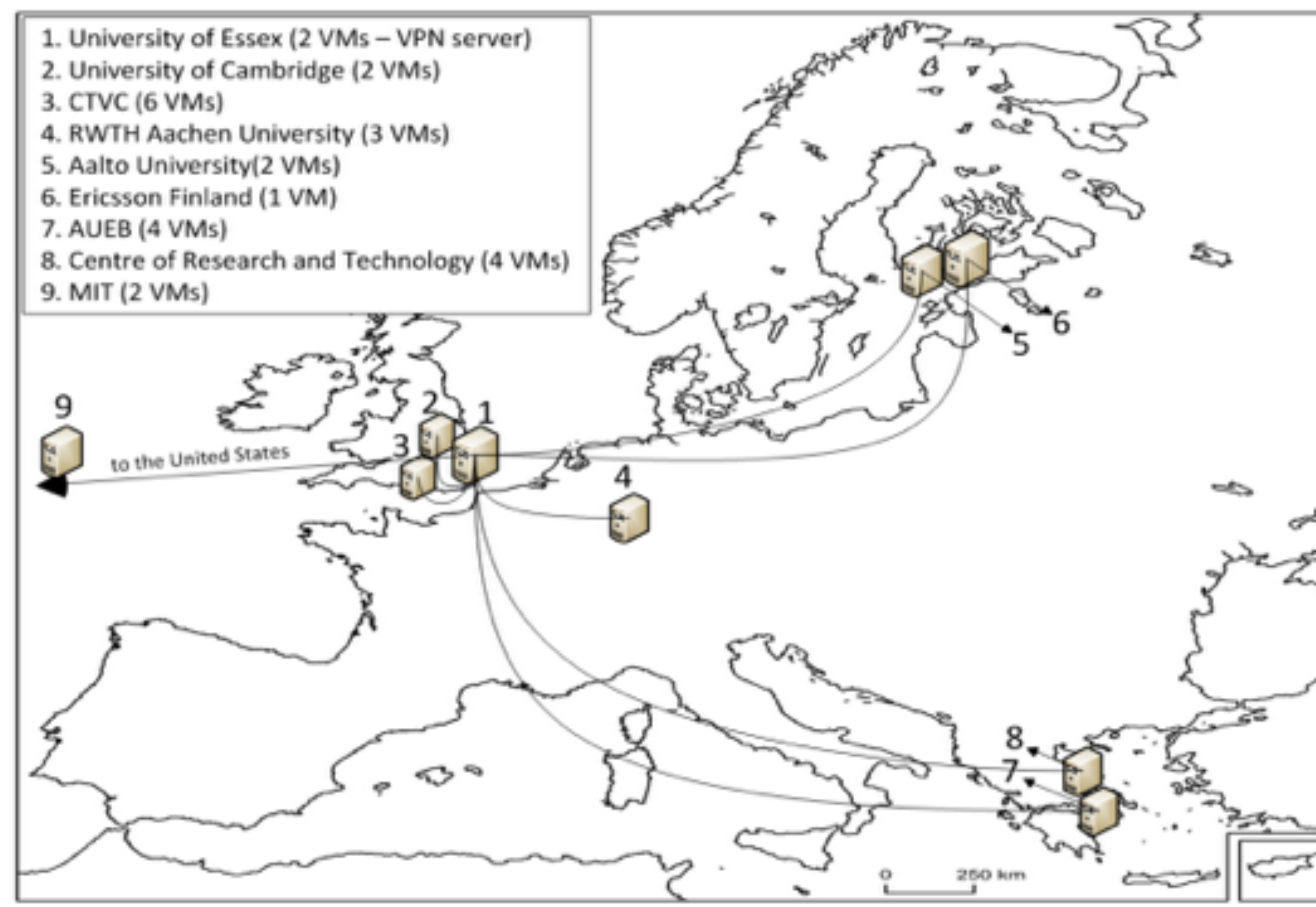
```
localRV::LocalRV(globalconf);
netlink::Netlink();
tonetlink::ToNetlink(netlink);
fromnetlink::FromNetlink(netlink);

proxy::LocalProxy(globalconf);

fw::Forwarder(globalconf,2,
1,192.168.15.4,192.168.15.5,00000000000000000000000000000000000000000000000000
000000000000000100000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000,
1,192.168.15.4,10.0.2.17,000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000100000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0);
```

Testbed

- 9 international sites
- 26 machines with +40 on demand ones
- tunneled via openVPN with configurable topologies



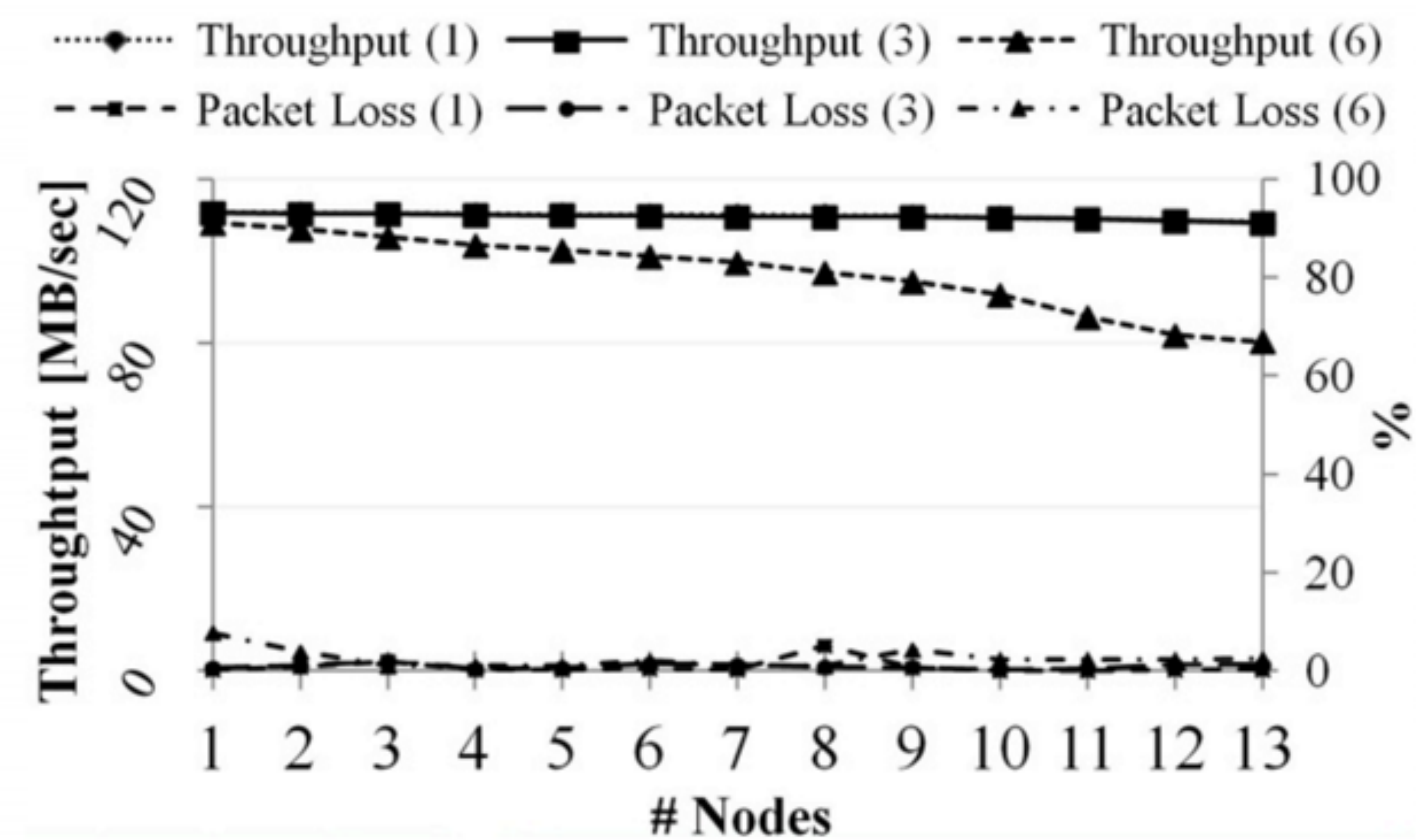
*Trossen, D. and Parisis, G. (2012). Designing and realizing an information-centric internet. IEEE Communications Magazine, 50(7):60–67.

Fast Path Evaluation

Forwarding efficiency

- 15 in a chain
- Multicasting (when nodes is sub)
- ~line speed even when 3 subs per node for 13 nodes

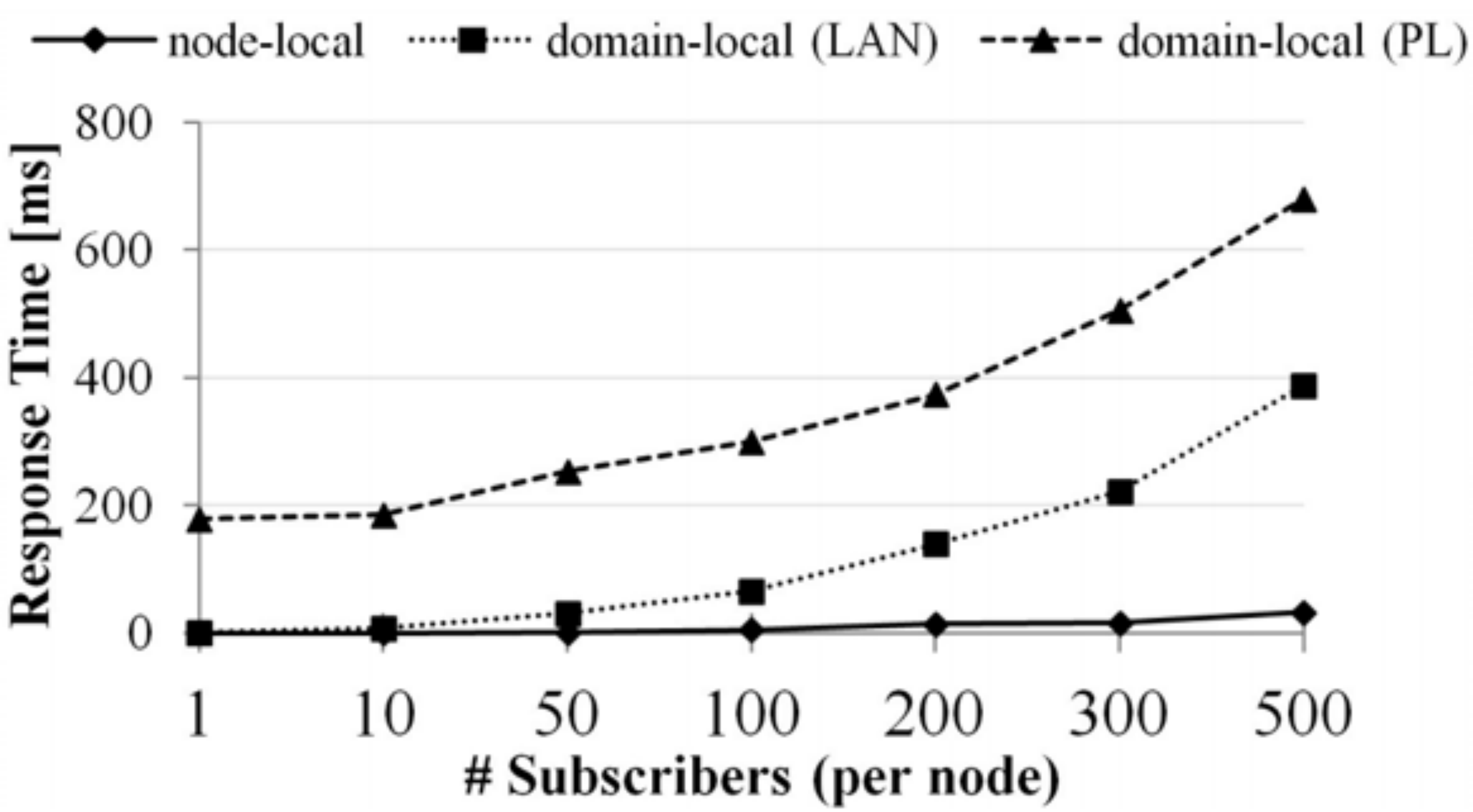
- Degradation when 6 pubs and more due to local copies



*Trossen, D. and Parisis, G. (2012). Designing and realizing an information-centric internet. IEEE Communications Magazine, 50(7):60–67.

source: PURSUIT FP7 public dissemination reports.

Slow Path Evaluation



- 100.000 adverts under single scope
- Subscribers subscribe to random item, wait until receive it and reiterate (500 times)
- -> worst case for slow path (ignores any possible optimisations due to domain-local rendezvous or mutable semantics)

- Node local: No net delays, No TM, 20ms for 500 processes.
- Domain local (Gbit-LAN): Centralised TM, ~400ms for 500 processes per node (7000 subs)
- Domain local (Planet Lab): Large delays, ~250ms for 1 sub per node (73 in total), 680 ms for 500 subs

Conclusions

- Information centric networking as a raising paradigm for dealing with scalable access to information

- Two main different strategies: completely distributed (CCN), and partially distributed (PURSUIT, PSIRP, LIPSIN).

- Potential state explosion in CCN based on naming versus economy of space for LIPSIN/PURSUIT

- Open issues: interfacing existent services over IP, standardisation of interfaces for regular devices: discovery of information and services in local networks, and wider area networks.