DCAR: Dual-mode Content Aware Router

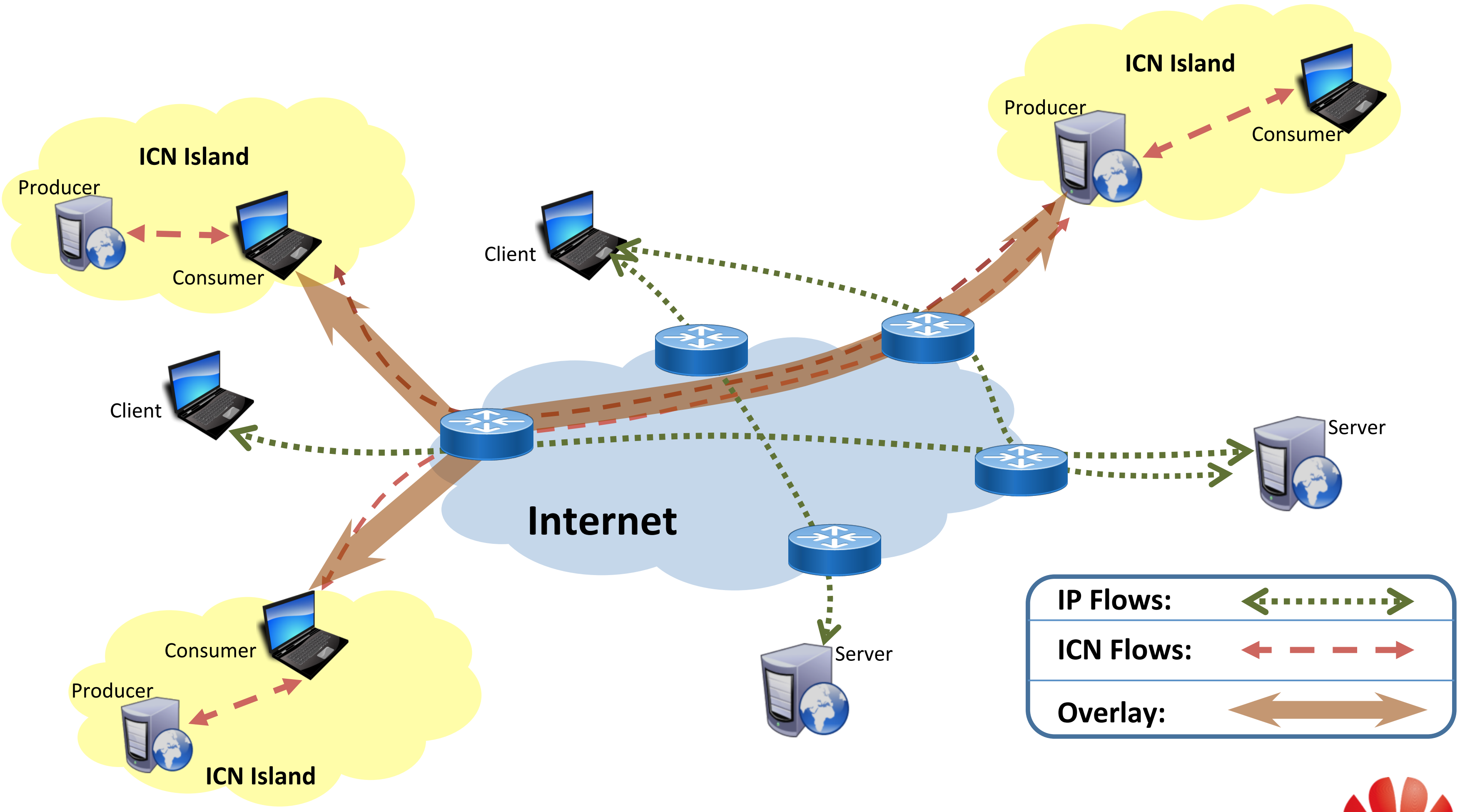
Syed Obaid Amin, Ravi Ravindran and GQ Wang
Huawei Research Center, USA (Futurewei)
{obaid.amin, ravi.ravindran, gq.wang}@huawei.com

CCNxCon 2015, PARC

Contents

- Motivation
- Proposal
- Advantages
- Implementation Details
- Evaluation
- Final Comments

Current Practices in ICN Deployment



Problems

- Diminishes the main advantage of ICN i.e. fetching contents from nearby sources or caches
- Interest packets are still routed to the end-point of the tunnel
- Contents are only cached at the end-points
- Dumb intermediate IP routers cannot help end-hosts in fetching the content

Goal

- **Making intermediate IP routers useful for the ICN flows**

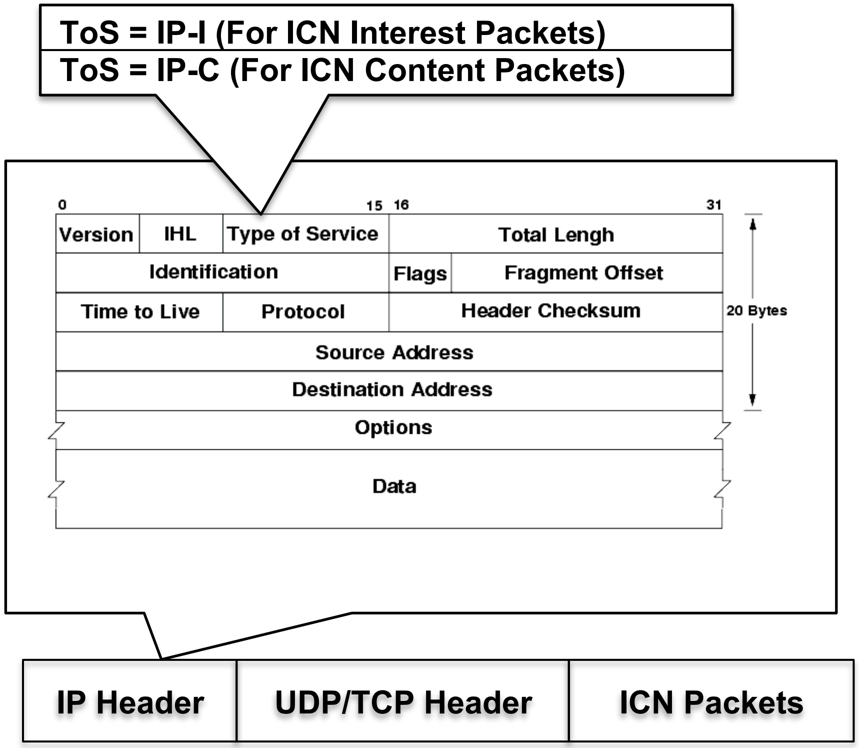
- **Constraints:**
  - The regular IP flows shouldn’t be affected
  - The changes should be minimal so that they can be easily adapted

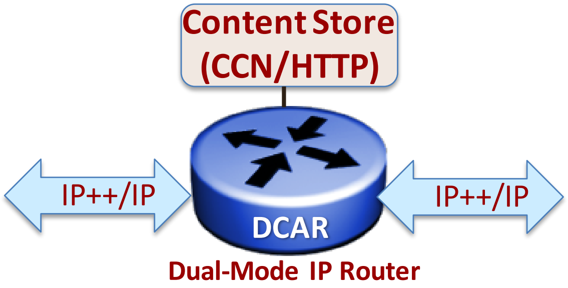
Proposal

- **DCAR (Dual-mode Content Aware Router)**
  - Introduce a router that:
    - can understand both ICN and IP flows, hence Dual-mode
    - can understand ICN content abstraction, hence Content Aware
    - can also cache the contents, so that future requests for the same content can be fulfilled

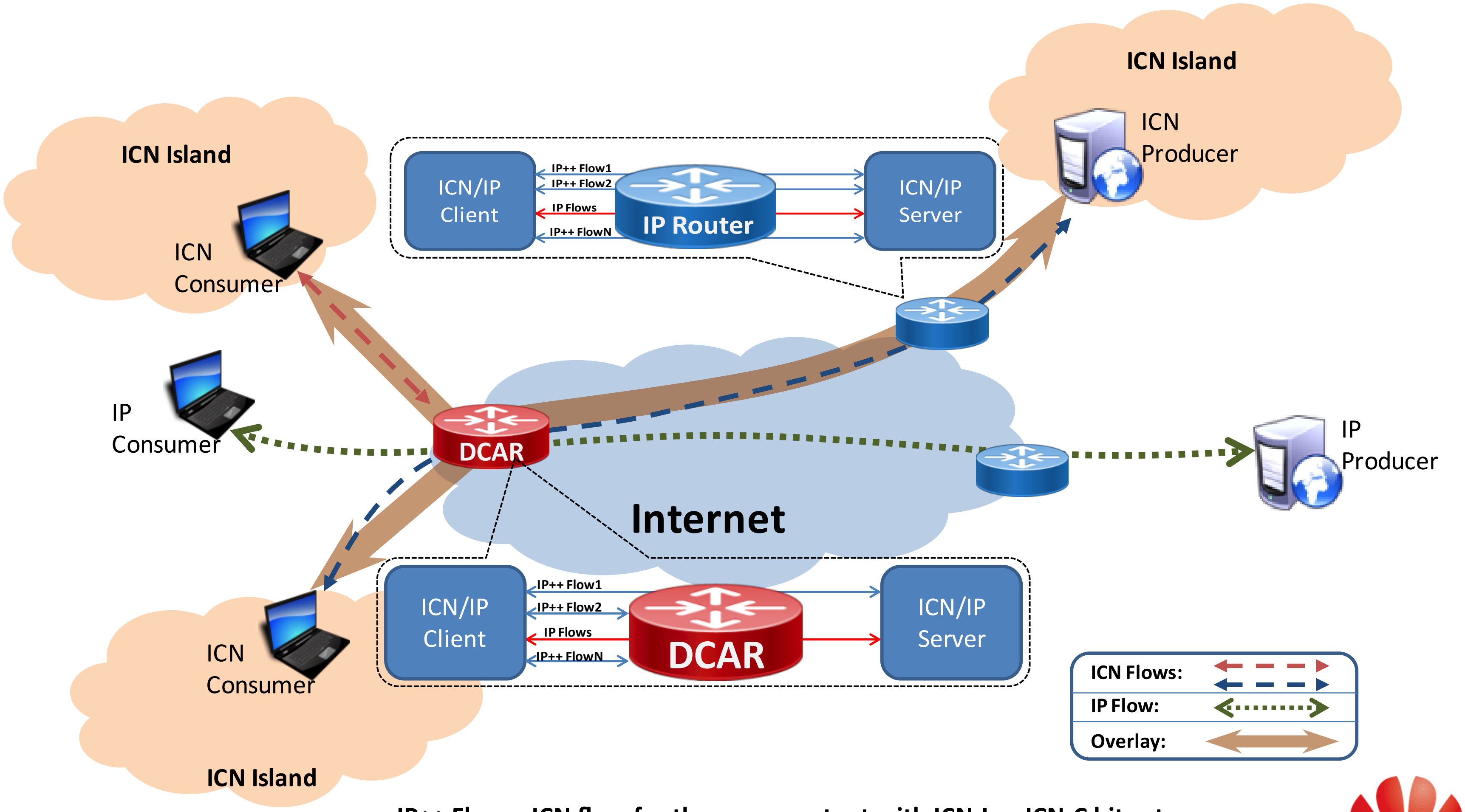
How to differentiate ICN and IP flows

- In overlay, ICN packets are encapsulated by the IP packets
- **How to make IP routers aware of the ICN flows?**
- Introduced two new IP primitives to differentiate between ICN and IP flows
  - **IP-I (IP Interest)**
    - Indicates that the IP packet encapsulates an ICN Interest
  - **IP-C (IP Content)**
    - Indicates that the IP packet encapsulates an ICN Content/Data
- IP flows with above primitives would be mentioned as ***IP++*** flows





Network Architecture with DCAR



IP++ Flow = ICN flow for the same content with ICN-I or ICN-C bit set

Advantages

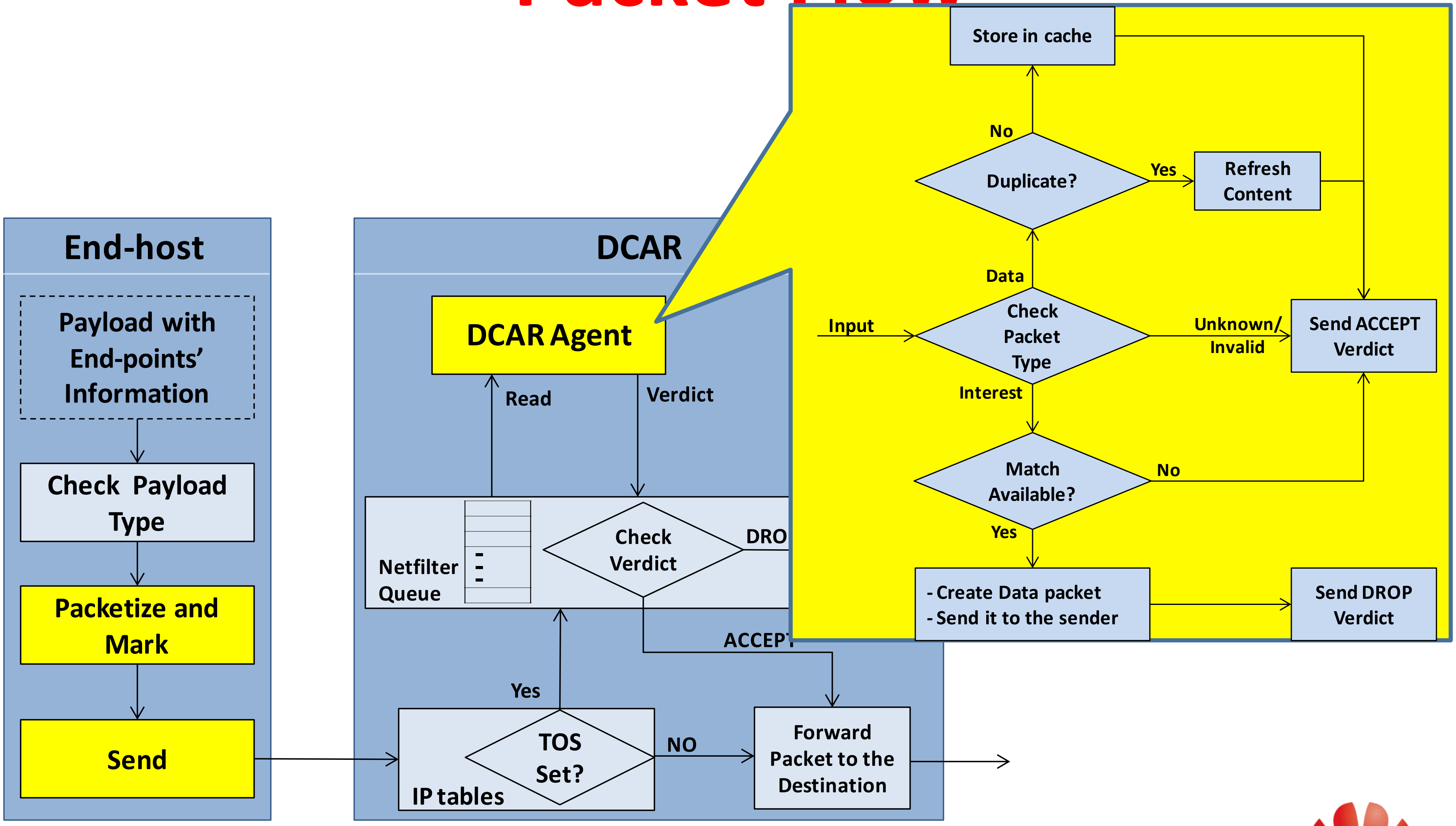
- Make intermediate routers useful in content sharing, with minimal changes and lightweight protocol
- Doesn't affect line rate forwarding of non-ICN packets
- Saves the bandwidth, as well as load on the producer
- Can be deployed easily in existing infrastructure
- In short, DCAR provides one of the main services of ICN i.e. Caching with minimal changes in the infrastructure

Other Use Cases for DCAR

- Using CCNx/NDN content abstraction in IP networks instead of HTTP
- Advantages:
  - content transfer is secure yet matches the performance of HTTP[1]
  - Better than HTTPS[1]
- Without using other components, like FIB and PIT provides ICN's main feature i.e. caching

[1] Van Jacobson et al, "Networking Named Content", Conext 09.

Packet Flow

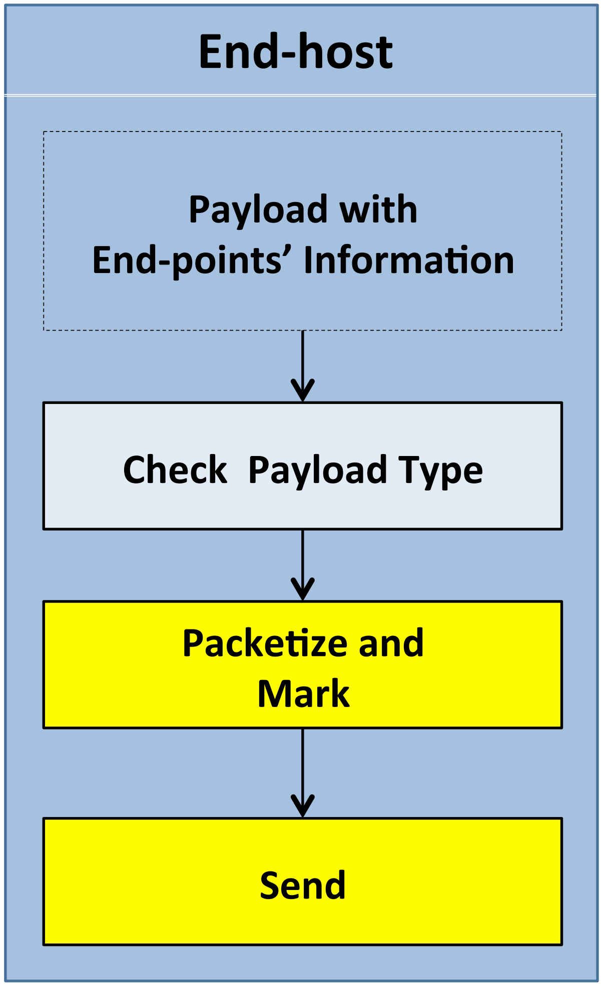


*** ACCEPT** means, handle the packet according to default rules.

DROP means that DCAR Agent can handle the packet and there is no need to forward it further.

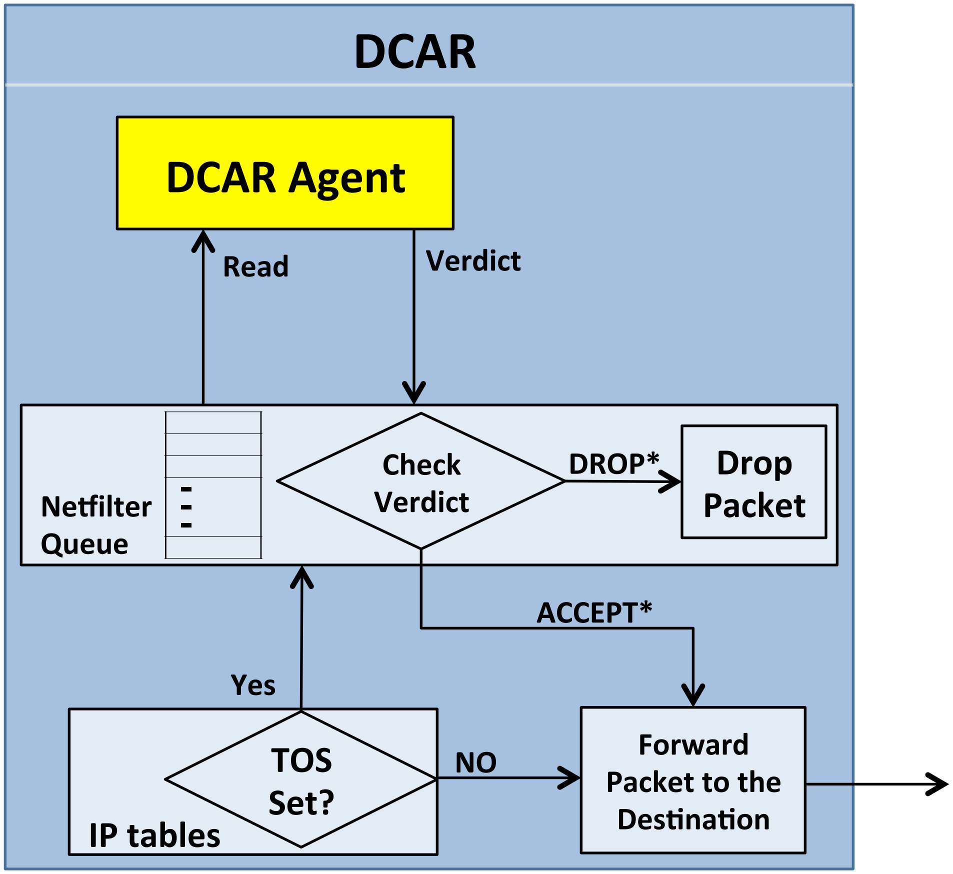
Implementation Details

- **On End-hosts:**
  - Modified ccnx 0.8.2
  - Marking can be enabled or disabled using environment variables
  - Disabled piggybacking, so that packets can be marked individually
  - Ensured that the content chunks are lower than the MTU size (limitations of the tools used)



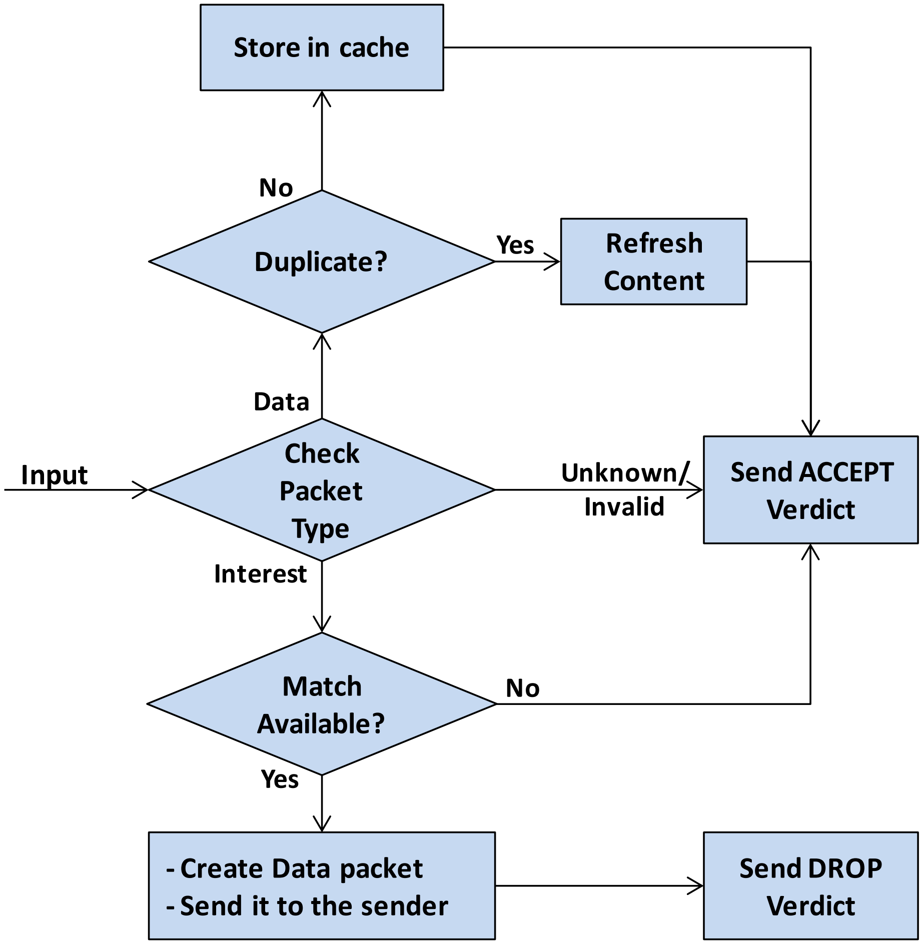
Implementation Details

- **On DCAR**
  - Needed an agent that can intercept and take decision on packets
  - Can be done on:
    - kernel, efficient, but not easy to implement
    - userspace, not efficient, but good for prototype
  - Used Netfilter that provides good framework for intercepting and processing packets



Implementation Details

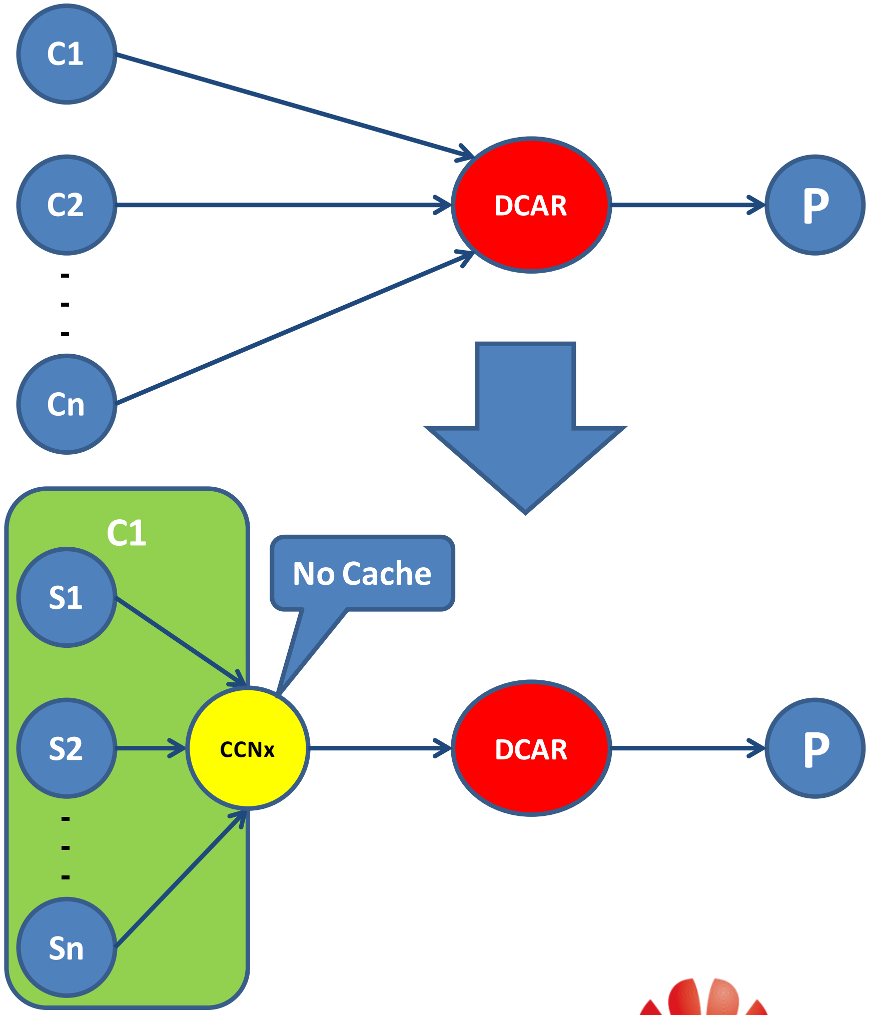
- **DCAR Agent**
  - A trimmed down version of “ccnd.c”
  - **No face management, No PIT, No FIB**
  - Only 1615 lines of code as compared to 6244 lines of ccnd.c,
  - Around 175 lines in the header, as compared to 531 of ccnd_private.h



Evaluation Results

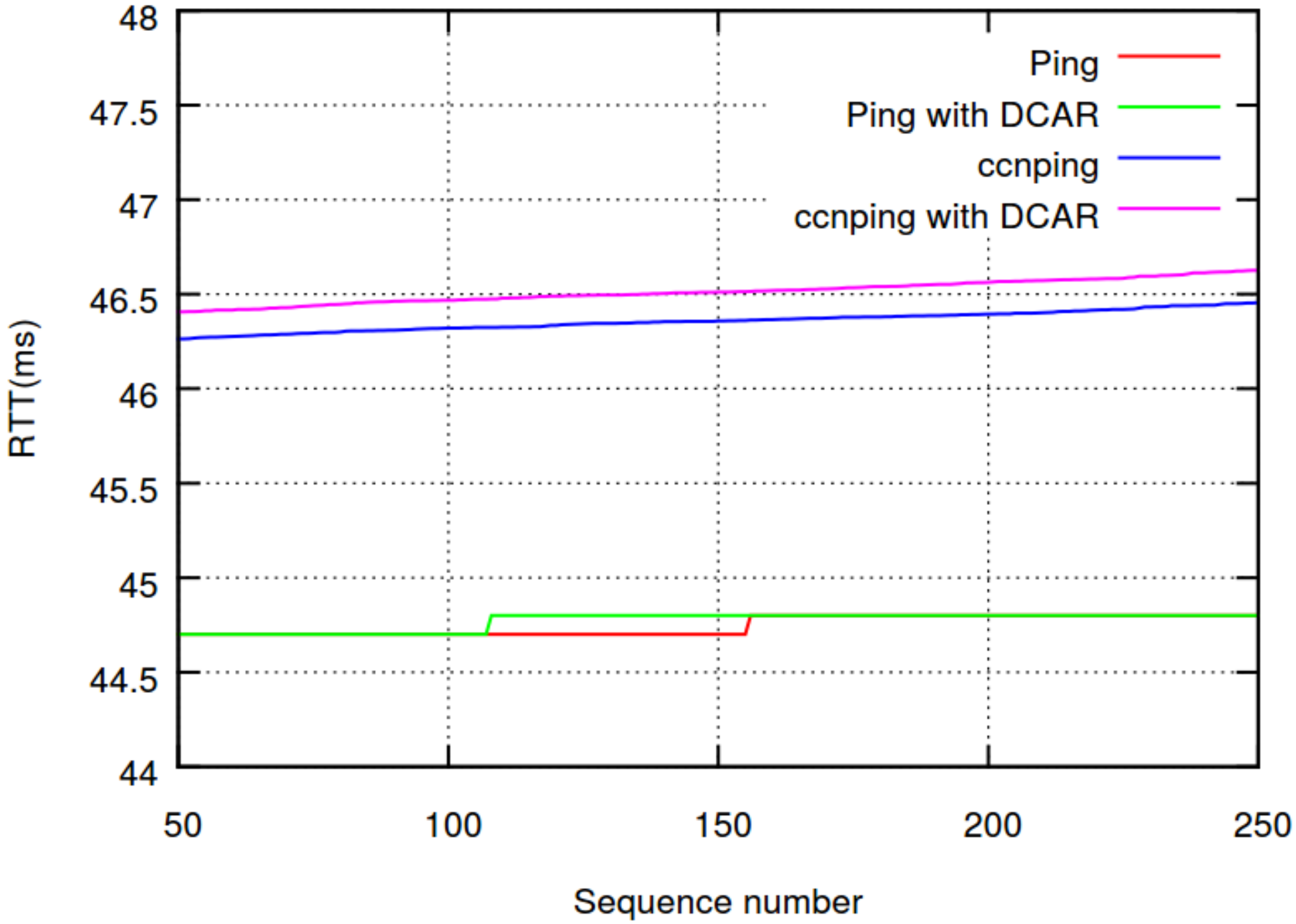
Experiment Setup

- Three nodes connected in a line topology
- RTT=44ms,
  - Consumer to DCAR=10ms[1]
  - DCAR to Producer=34ms [2]
- ***Disabled client side cache***
- **For file transfer tests:**
  - used ccnseqwriter and ccncat with default pipeline value i.e. 4
  - Chunk size = 8KB, MTU = 9000B
  - Client sends multiple requests for the same content (denoted as S)
  - The requests are sent exponentially at mean = 1 sec



[1] S. Sundaresan et al., "Broadband Internet Performance: A View From the Gateway." , ACM SIGCOMM'11
[2] http://ipnetwork.bgtmo.ip.att.net/pws/network_delay.html

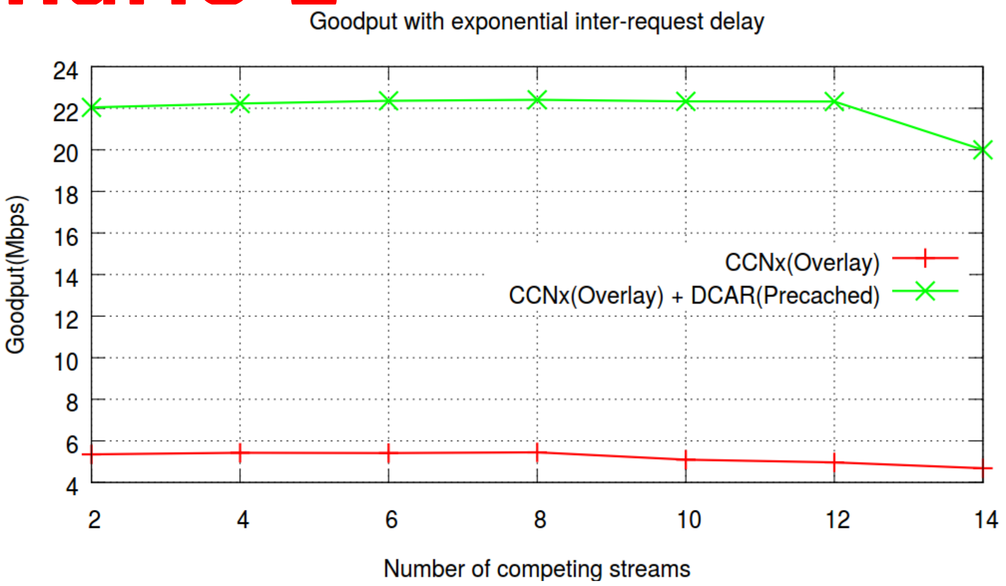
Processing Delay

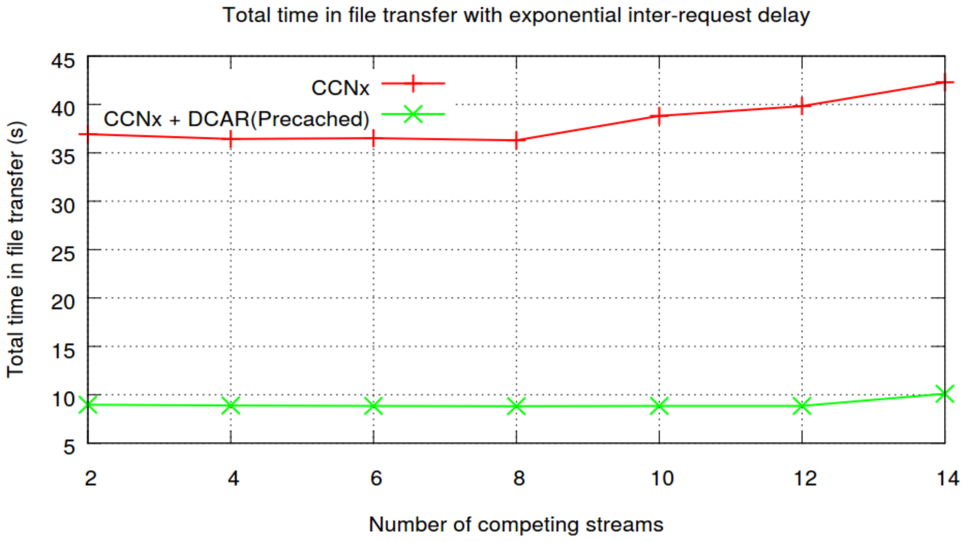


- Shows the RTT observed by packets in 4 different scenarios

Scenario 1

- Contents are pre-cached
  - E.g. Comcast VoD, software updates
- Benefits are quite obvious
- As the contents are coming from nearby sources, throughput is greatly improved.

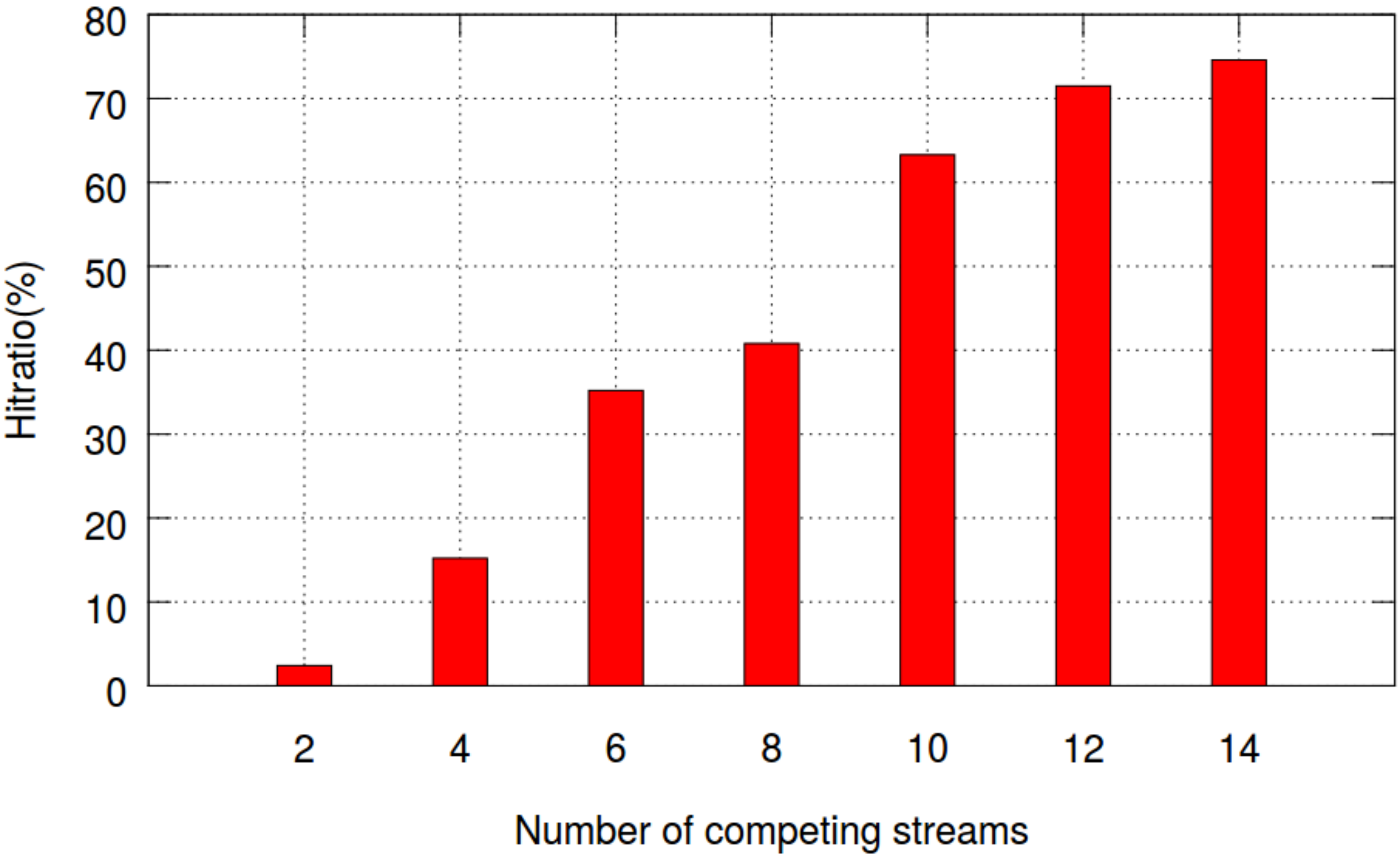

Goodput with exponential inter-request delay


Total time in file transfer with exponential inter-request delay

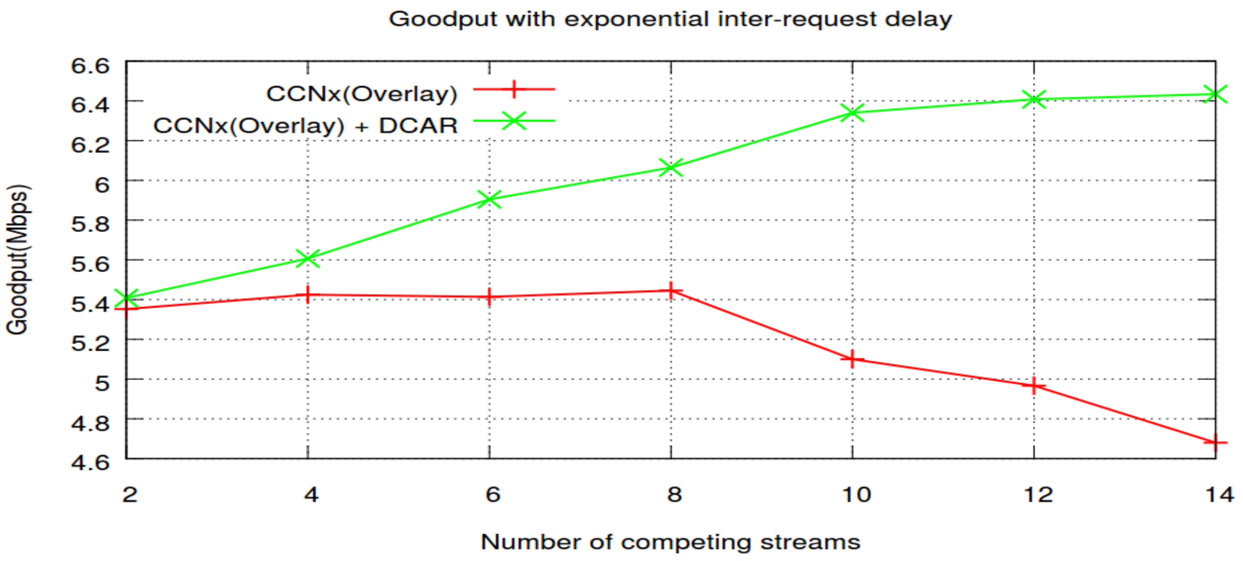
Scenario 2

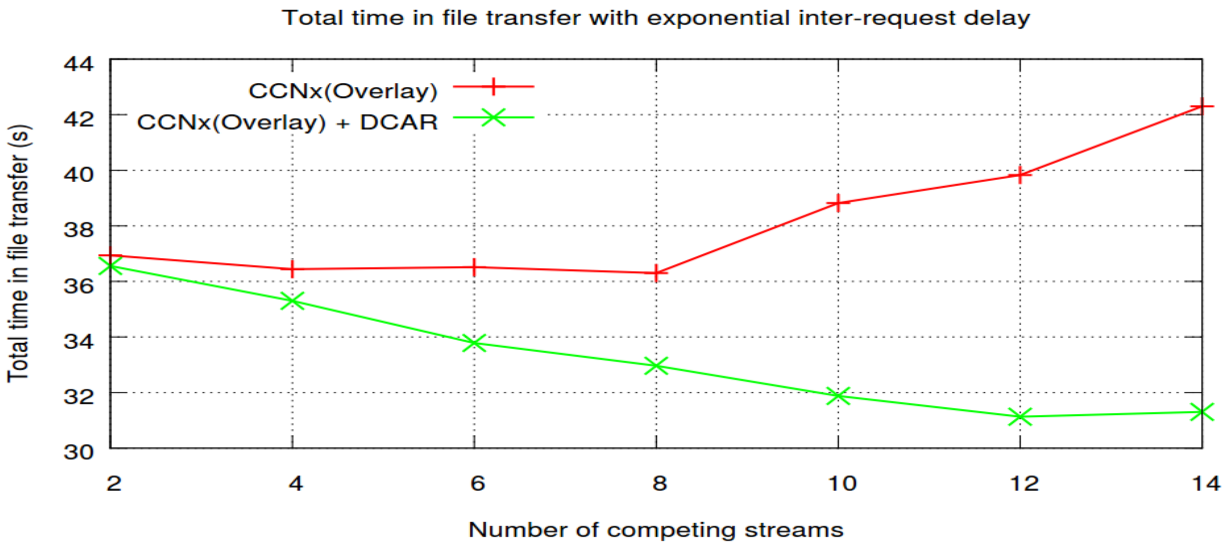
- **No pre-caching**
  - E.g. Live streaming of superbowl, realtime video conferencing
- Results shows that even live contents can be benefited by caching
- As the number of competing streams increases, the hit ratio of Interest for a similar content increases, improves both throughput and latency

Hit Ratio


Hitratio(%)
80
70
60
50
40
30
20
10
0
2
4
6
8
10
12
14
Number of competing streams

Goodput and Latency





Final comments

- Currently, DCAR supports CCNx abstraction
- We are working on
  - NDN
  - HTTP
- CCN/NDN provides strong content abstraction, why it is not promoted alone as an alternative of HTTP?
- Do we need FIB, PIT, name-based routing, etc in initial deployments of ICN?
- My two cents: promote CCN/NDN abstraction first, the other parts of the system will come eventually