NDN Research in Tsinghua University

Jun Bi

Network Research Center, Tsinghua Univ.

AsiaFI 2012 NDN workshop

2012.03.20

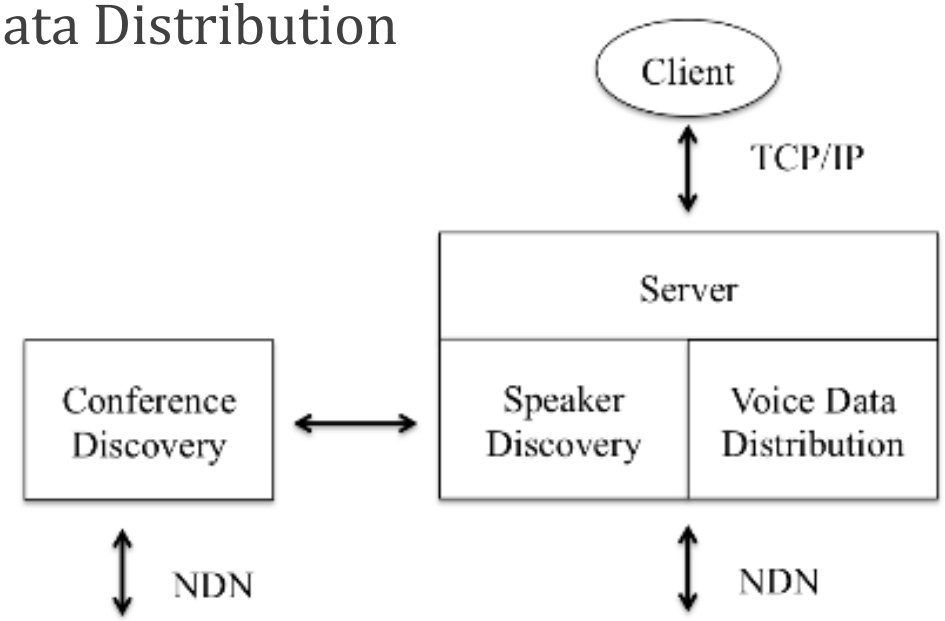
Research on NDN Applications/Testbed

Audio Conference Tool Over NDN

- Joint research with UCLA/PARC
- Motivation
  - How to name the application data and its relationship with routing scalability
  - A useful tool for the NDN team collaborations
  - Generate “real-world” traffic on an NDN testbed

An Overview of ACT Design

- Three main components:
- Conference Discovery
- Speaker Discovery
- Voice Data Distribution



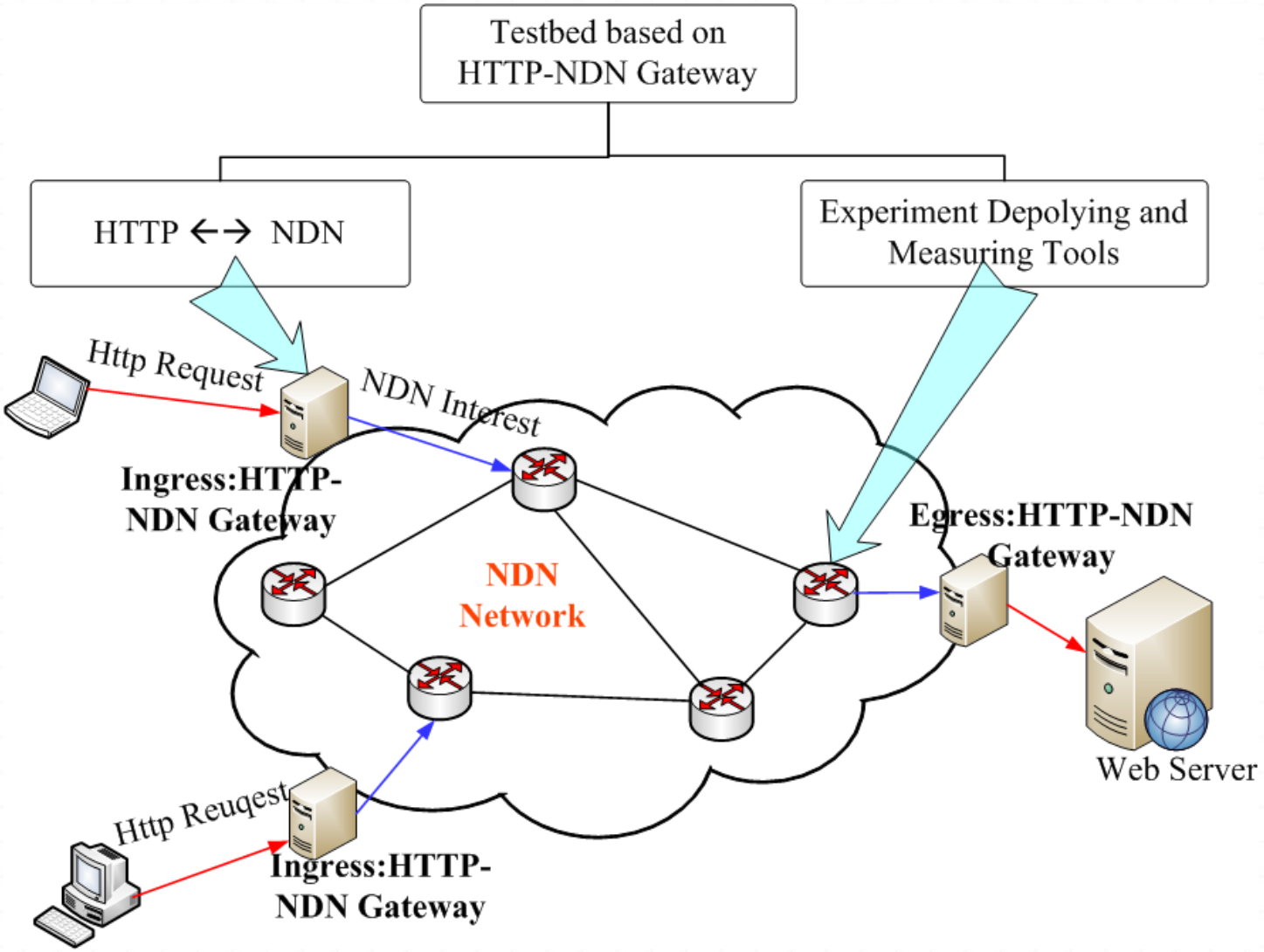
A NDN testbed with real traffic via Http-NDN gateway

- Joint project with Huawei
- Motivations
  - For research experiment: to build a large scale testbed with NDN traffic that is easily generated by a Http-NDN gateway (as http proxy for students) from translating students' daily http traffic (short-term)
  - For NDN dissemination: To make full use of NDN's benefits (such as caching in NDN router) to improve student's web browsing performance, and/or to save campus operators bandwidth (mid-term)

Some Experience Gained

- User/server transparency is good for experiments
  - No change for HTTP browser/server
  - Easily deployment
- The match between HTTP “semantics” and CCN is pretty decent
  - To simply do “syntax” translation can achieve most “semantics”
  - Translation is easy for simple browsing task
- Gateway still need to fully understand HTTP “semantics”
  - Not just delivering packets by “syntax” translation, for advanced browsing task

Overview of the Testbed

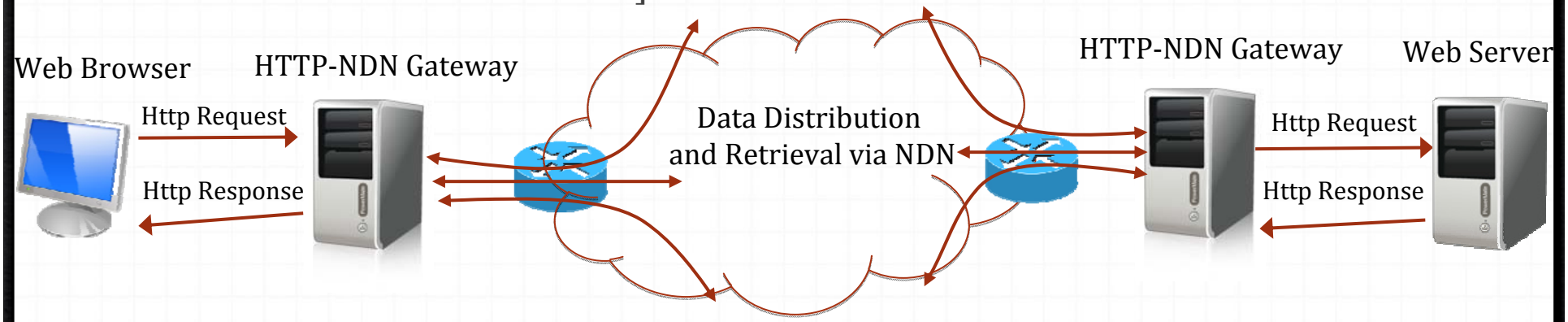


Main Components of Gateway

- HTTP proxy
  - Capture HTTP requests[Ingress]
  - Send HTTP requests[Egress]
  - Retrieve HTTP responses[Egress]
  - Send HTTP responses[Ingress]
- Protocol translation
  - Bi-translation between HTTP packets and NDN packets
  - HTTP request [1]←→[n]NDN interests[Ingress]
  - HTTP response[1]←→[n]NDN contents[Egress]
  - Content lifetime calculating according HTTP headers
- NDN data distribution
  - Send the first interest to initialize a HTTP request [Ingress]
  - Split received HTTP response into NDN Content chunks and send via NDN[Egress]
  - Handle incoming content and express next interest for remaining slices[Ingress]

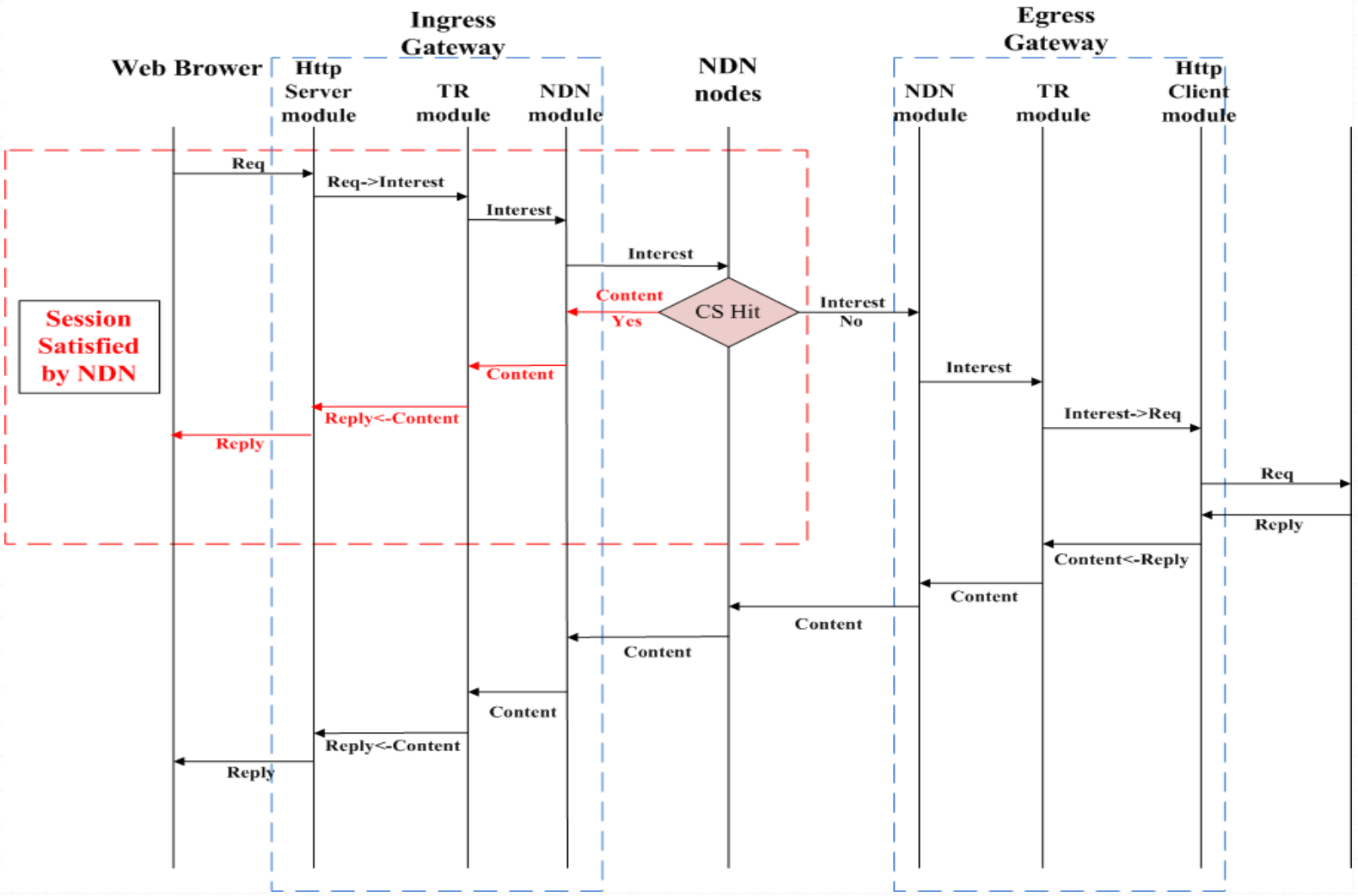
Session Maintenance

- Three kinds of session
  - Session involves an ingress gateway and one or more egress gateway[due to multipath feature]
  - Session involves just an ingress gateway and some intermediate routers[due to cache hit]
  - Session involves an ingress gateway, some intermediate routers and one or more egress gateway[due to partial caching]
  - Session identify
  - HTTP side: recognized by <IP:Port> pair
  - NDN side: recognized by unique name[url and some other information combined]



Example of a whole session: two http connections and two ndn connections

A Trip of a HTTP Request



Protocol Translation

- HTTP request to NDN interest (basically, "syntax" translation)
  - Request url to identify an interest
  - Additional information include request method and some HTTP headers stuffed into interest name
  - HTTP headers considered:
    - cache-control related headers, need to calculate lifetime of a serial of contents
    - Other headers are simply discarded
  - A common prefix is used to limited the routing scope to a small subset of the routers
  - TODO
    - Cookies related headers: local cookie caching in gateway
    - Accept related headers: in case of client unsupported content type or encoding

HTTP request: http://www.baidu.com/s?wd=yi&rsv_bp=0&rsv_spt=3&inputT=1488
Cache-Control: max-age=0
Cookie: MCITY=-289%3A; BAIDUID=6A6E9A3863C42DEBBF861A5B4C2FAA1D:FG=1
User-Agent:
Mozilla/5.0 (Windows NT 5.1) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.963.79 Safari/535.11

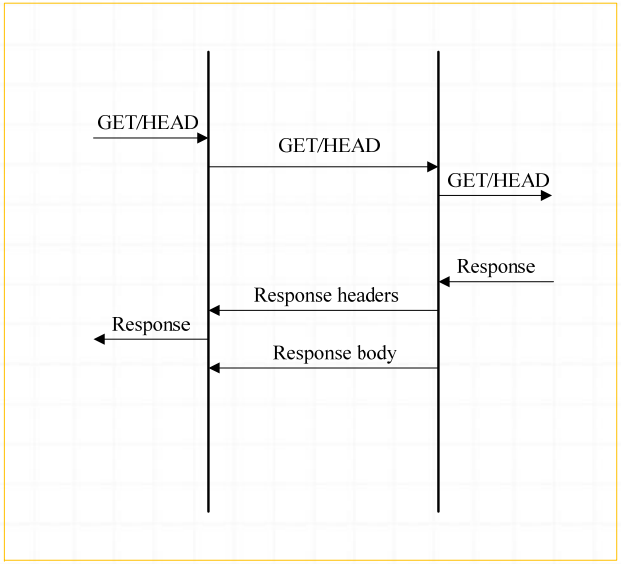
NDN Interest:
ccnx:/.gateway/http/242/GET/www.baidu.com/s?wd=yi&rsv_bp=0&rsv_spt=3&inputT=1488/Cache-Control: max-age=0/

Protocol Translation(cont.)

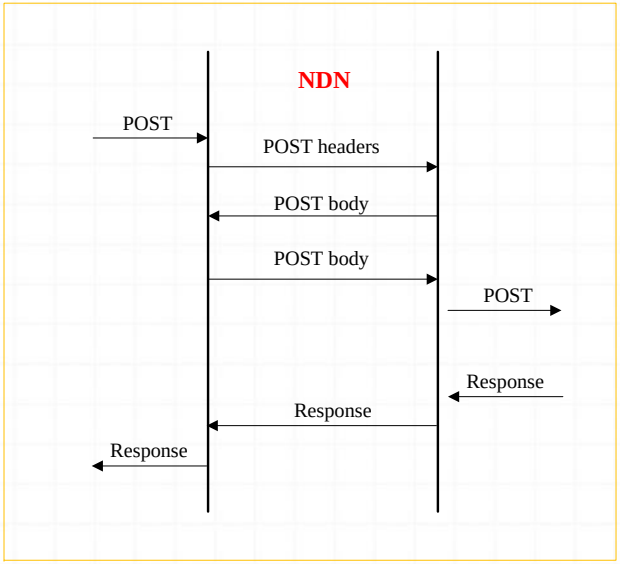
- NDN interest to HTTP request
- Request method, url and headers are decoded from interest name
- Method and url are used for reconstruct a HTTP request
- Headers are used to calculate lifetime of ndn contents
- HTTP response to NDN content
  - Content name is the same of its interest
  - Response are sent via NDN in two steps
    - Status, response headers and the length of response body is sent first[encoded in one ndn packet], marked as segment 0
    - Response body is split into 4k Bytes slices , starting from segment 1
    - The final slice encodes as a CCN_ FinalBlock component to indicate the request or of a termination
- NDN content to HTTP response (with some semantics translation)
  - Decode response status, header, final url[redirection] and the body length from the first incoming content
  - Send the response header to browser with a Content-Length field and tell the browser don't close the connection by insert a Connection: keep-alive header
  - Decode response body from following content slice by slice and send it to browser

Protocol Cases

- Different HTTP methods are handled respectively
  - Sent GET/HEAD request in one interest
  - Sent POST request headers first, peer is responsible for asking post body



GET/HEAD

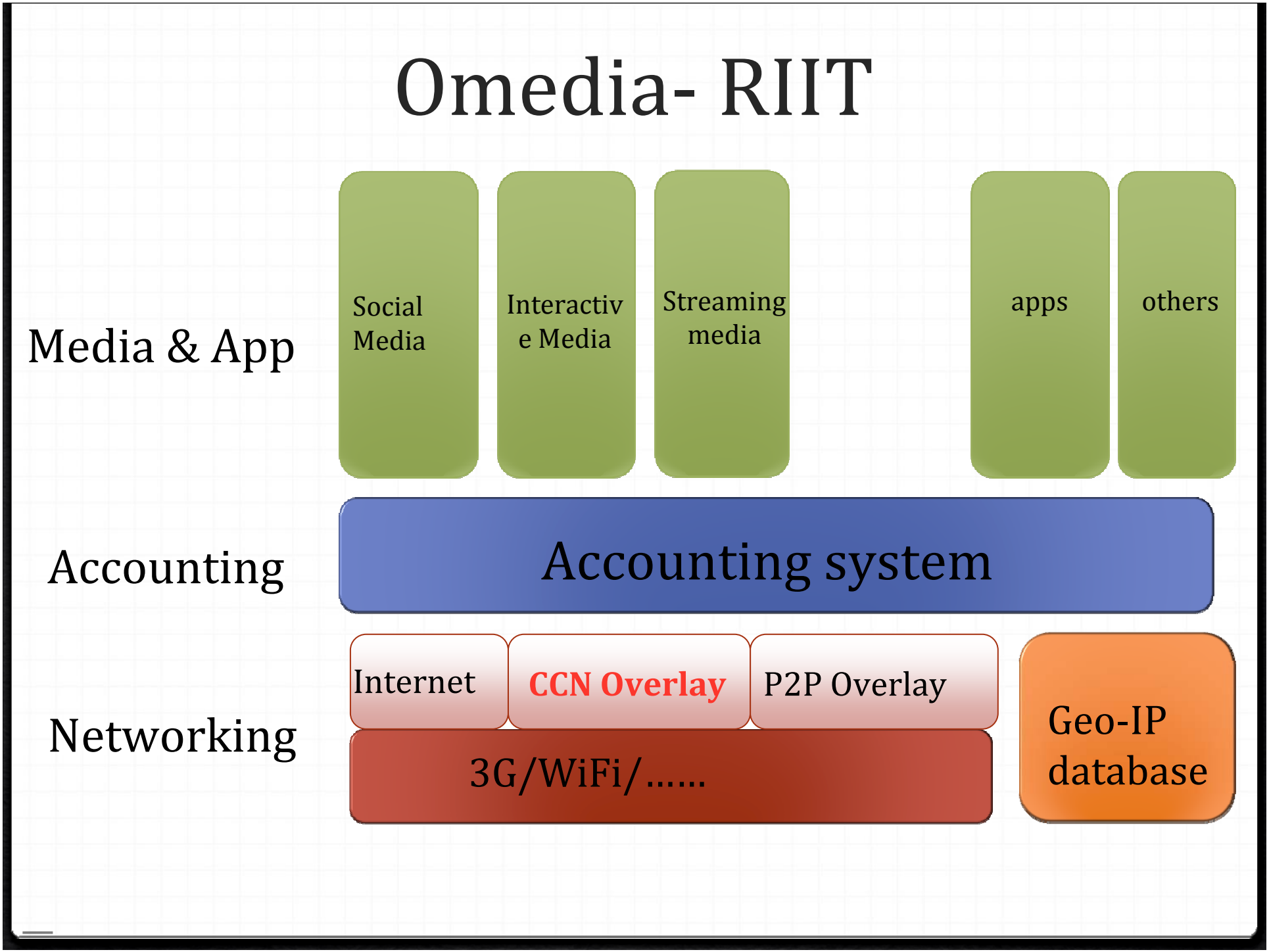


POST

Ongoing Work

- Congestion control for big video file
  - To translate HTTP “semantics”
    - One HTTP get for one big file post -> One Interest pkt for multiple fragmented NDN DATA pkts (size 4KB)
    - One HTTP get, one big file post -> One Interest pkt for every fragmented DATA pkt
  - NDN packet size
    - How to cut a big file into NDN packets
- Implementation of a monitoring/management tool for large scale testbed
- Adding new NDN routing/forwarding/caching mechanisms into CCN router to observe results under generated NDN traffic

Omedia- RIIT


Media & App
Social Media
Interactive Media
Streaming media
apps
others
Accounting
Accounting system
Networking
Internet
CCN Overlay
P2P Overlay
3G/WiFi/……
Geo-IP database

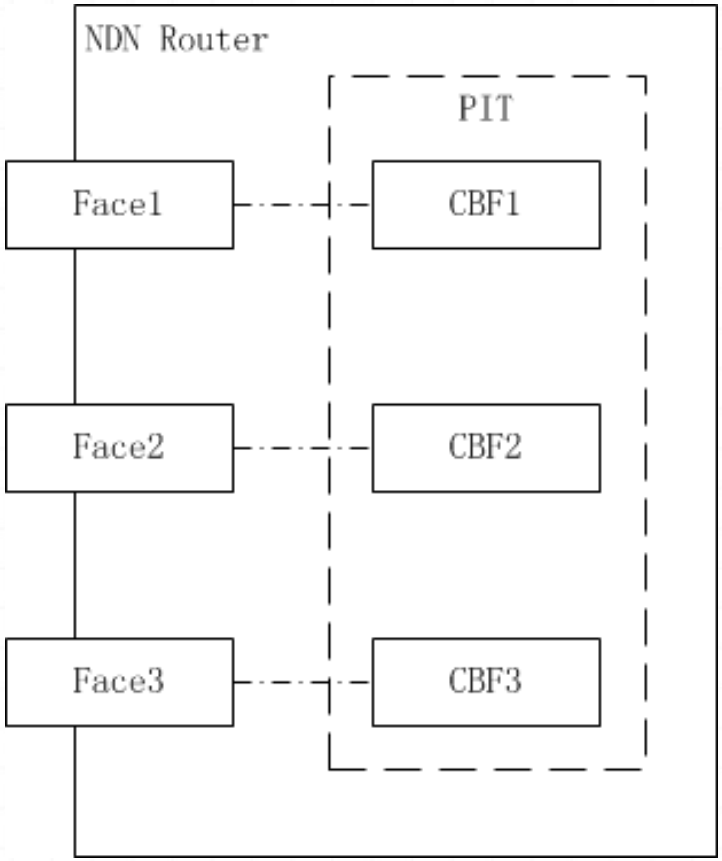
Research on NDN Scalability

PIT Compression

- PIT may face scalability issue
  - Soft state in PIT: may keep millions of entries
- Our study
  - Current PIT: Key=name, Value=a set of faces
  - Problem:
    - Key space is sparse (a waste of storage)
    - Bloom Filter causes false positive
  - Our research: to study the effect of false positive if uses Bloom Filter but lacking accurate PIT
    - Key=face, Value=Bloom Filter (name set)
    - Use CBF (Counting Bloom Filter) to reduce False Positive of Bloom Filter

PIT Architecture & Algorithm

j faces (j=3)



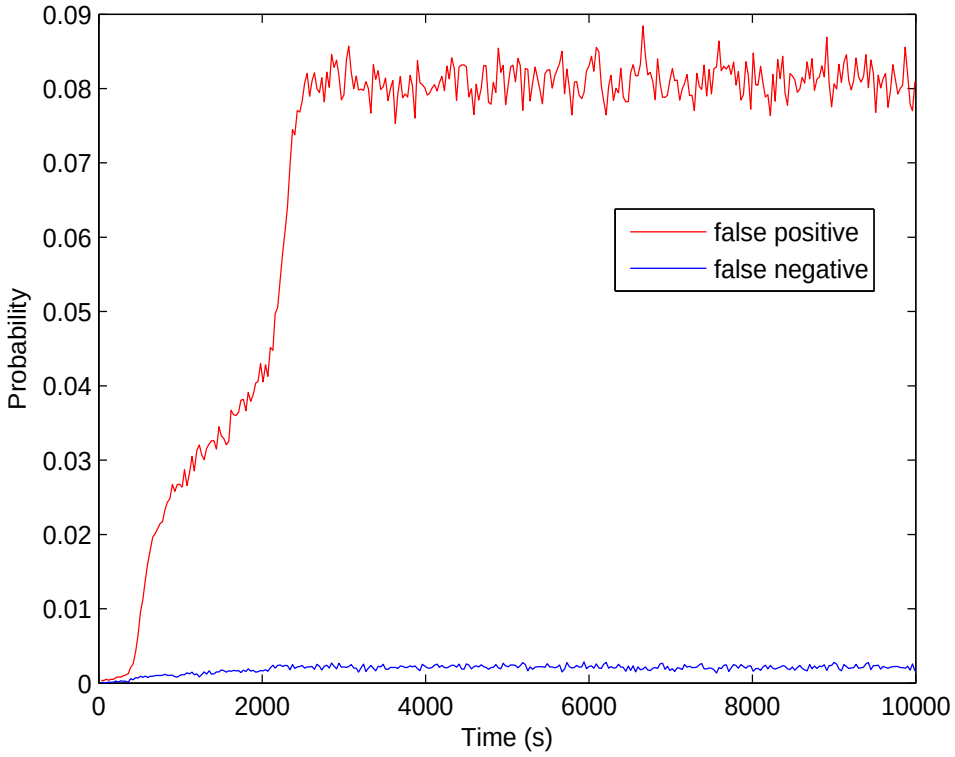
Receive Interest(*Name*) from Face[*i*]

```
if Name match in CBF[i]
  Drop Interest
else
  Add Name into CBF[i]
  if Name doesn't match in other CBFs
    Forward it according to FIB
```

Receive Data(*Name*) from Face[*i*]

```
For k = 1 to j (not i)
  if Name match in CBF[k]
    Delete Name from CBF[k]
    Forward it to k
```

Evaluation



i	1	2	3	4	5	6	7	8	9	>10
p_i‰	993.5	1.830	0.638	0.686	0.497	0.390	0.317	0.288	0.245	1.459

- Compression >50%, if false positive is <0.14%
  - FP means Interest pkt retransmission
- FP of CBF deletion may cause high FP for less than 1% users
  - Solution: might set a “slow path” bit at interest packet if the retransmission exceed the limit (e.g. 2), by users
  - If that bit is set, then the interest pkt is forwarded to a high end router with accurate PIT

Research on NDN Caching

Motivation

- In-network caching is considered as one of the most significant properties of NDN
- To study how to achieve the optimized caching result for a given objective
  - By modeling and calculation theoretically, then compared with simulation results
- To compare performance between various cache policies
- Try to propose new cache policies

Problem Formulation

- Optimizing objective by MILP modeling (Mixed-integer linear programming)
  - find a feasible assignment of caching copies of each content to routers in order to minimize the overall resource consumption of the network.

- Objective Function: minimizing the overall average hops
  - $\min \sum_{v \in V} \sum_{c \in C} q_{v,c} * m(c) * \min_{0 \le i < u\left(p_{v,l(c)}\right),\ \delta_{c,g\left(p_{v,l(c)},i\right)} != 0} \left(\delta_{c,g\left(p_{v,l(c)},i\right)} * i,\ u\left(p_{v,l(c)}\right)\right)$ (1)
- Capacity Constraints
  - $\sum_{c \in C} \delta_{c,v} * m(c) \le b(v), \forall v \in V$ (2)
- Introduce extra variables to linearize the objective fuction:
  - Objective Function : $\min \sum_{v \in V} \sum_{c \in C} q_{v,c} * m(c) * \mu_{v,c}$ (7)
  - $\sum_{c \in C} \delta_{c,v} * m(c) \le b(v), \forall v \in V$ (2)
  - $\sum_{i=0}^{i \le u(p_{v,l(c)})} \sigma_{v,c,i} = 1, \forall v \in V, \forall c \in C$ (3)
  - $\delta_{c,g\left(p_{v,l(c)},i\right)} \ge \sigma_{v,c,i}, \forall v \in V, \forall c \in C, 0 \le i < u\left(p_{v,l(c)}\right)$ (4)
  - $\mu_{v,c} \ge \delta_{c,g\left(p_{v,l(c)},i\right)} * i - \left(\max_{v \in V, c \in C} u\left(p_{v,l(c)}\right)\right) * \left(1 - \sigma_{v,c,i}\right), \forall v \in V, \forall c \in C, 0 \le i < u\left(p_{v,l(c)}\right)$ (5)
  - $\mu_{v,c} \ge u\left(p_{v,l(c)}\right) - \left(\max_{v \in V, c \in C} u\left(p_{v,l(c)}\right)\right) * \left(1 - \sigma_{v,c,u\left(p_{v,l(c)}\right)}\right),\ \forall v \in V, \forall c \in C$ (6)

Cache Policy

- Preliminary results with theoretical analysis and NS-3 simulation
  - Perfect-LFU can achieve a relatively better performance than other policies (e.g. LRU Random)
  - The performance gain might be limit with complex cache policies relying on collaboration among NDN routers

NDN Caching - DCS

- Proposed a age-based cooperative caching scheme
- Using NDN's caching to improve DTN Networking
  - In-network caching
  - Content distribution
  - Multicast

Thank you!