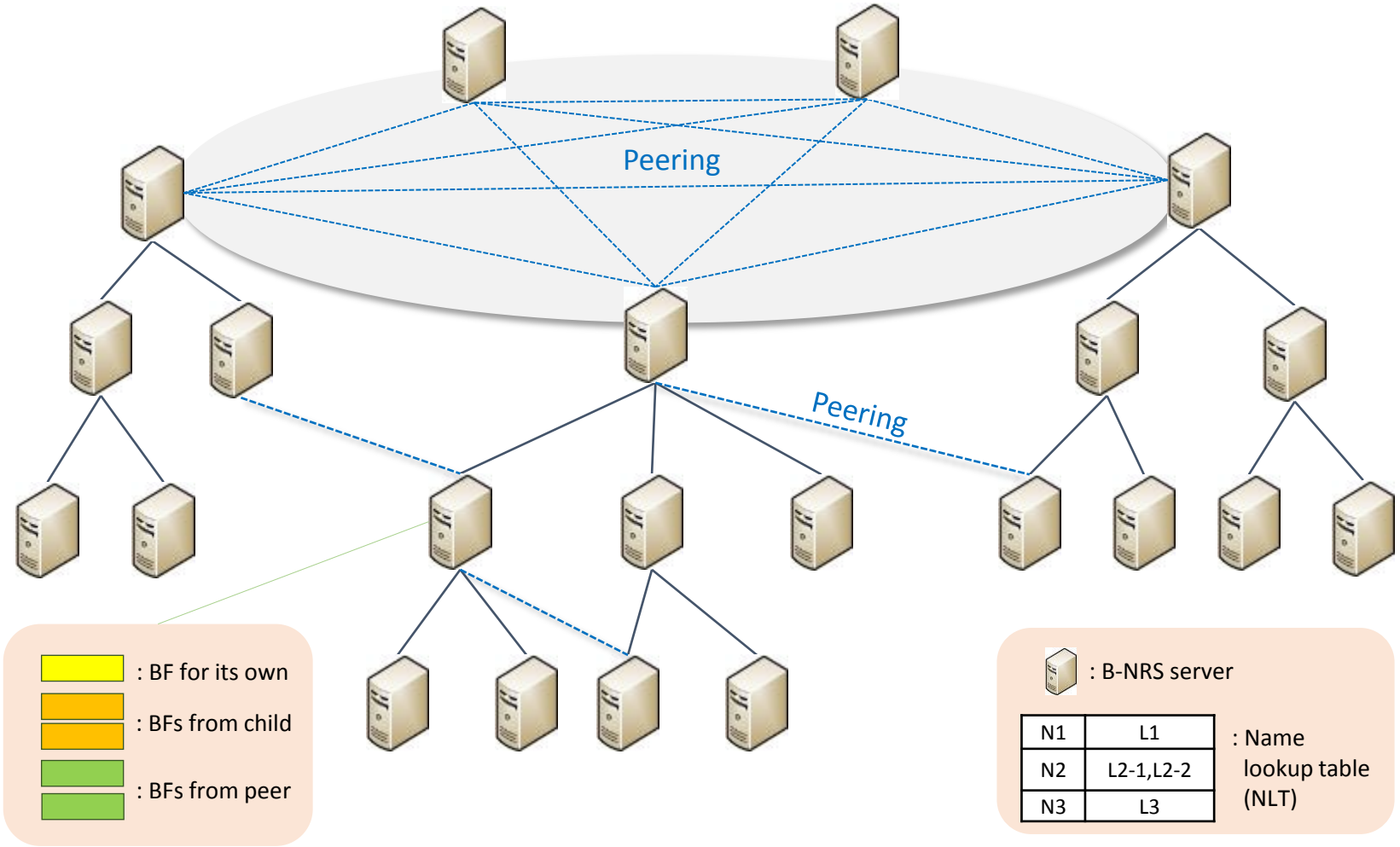
Some Results of Implementing Bloom Filter based NRS for ICN

ICNRG Interim Meeting, Prague

Jungha Hong, Woojik Chun, and Heeyoung Jung

July 19, 2015

B-NRS structure



Protocol messages (1)

- From client to B-NRS server
  - REQ_NAME_REGISTER
  - REQ_LOC_UPDATE
  - REQ_LOOKUP
  - REQ_NAME_DEREGISTER

Protocol messages (2)

- Between B-NRS severs in a relation of parent-child
  - CMD_BF_UPDATE
  - CMD_LOOKUP
  - CMD_LOOKUP_NACK
  - CMD_LOOKUP_TOP_DOWN
  - CMD_LOC_UPDATE
  - CMD_LOC_UPDATE_NACK
  - CMD_LOC_UPDATE_TOP_DOWN

Protocol messages (3)

- Between B-NRS severs in a relation of peering
  - PEER_BF_UPDATE
  - PEER_LOOKUP
  - PEER_LOOKUP_NACK
  - PEER_LOOKUP_TOP
  - PEER_LOC_UPDATE
  - PEER_LOC_UPDATE_NACK
  - PEER_LOC_UPDATE_TOP

- From B-NRS server to client
  - REP_LOOKUP

REQ_NAME_REGISTER

- prot : Protocol type
- Name : 24 Byte
- Store the name in the name lookup table
- Send CMD_BF_UPDATE to parent and peers

REQ_LOC_UPDATE

0	1	2	26	27	27+vlen
prot	mode	name	vlen	LOC	

- LOC : variable length string
- mode : type of LOC update
  - mode == 1 : Add
    - Append the LOC in the table
  - mode == 2 : Delete
    - Delete the LOC from the table
  - mode == 3 : Replace
    - Delete all LOCs stored in the table and store the given LOC
- When LOC is changed,
  - Old LOC is deleted first and then new LOC is stored
- vlen : length of LOC

REQ_LOOKUP

- Checks it own BF first
  - If the name is found, then REP_LOOKUP is directly sent to client
  - Otherwise, send CMD_LOOKUP to the corresponding servers

REQ_NAME_DEREGISTER

- Delete the all information of the name from the table

CMD_BF_UPDATE

- Updates its own BF
- Sends CMD_BF_UPDATE to parent and PEER_BF_UPDATE to peers

CMD_LOOKUP (1)

0	1	25	29	30	34
prot	name	ip_org	up/down	depth	

- ip-org : IP address of client who sends the lookup message
- up : when forwarded from child
- down : when forwarded from parent
- depth : increase by one whenever goes down to child
  - Up → depth = 0
- Checks it own BF first
  - If the name is found, then REP_LOOKUP is directly sent to client

CMD_LOOKUP (2)

0	1	25	29	30	34
prot	name	ip_org	up/down	depth	

- Case that any BF of child and peer answers positively
  - Up
    - Store the name and the list of all the positive child and peers including false positive cases in a DB
    - Send CMD_LOOKUP/PEER_LOOKUP to all the positive child and peers
    - If it gets NACK from all of them in the list, then send CMD_LOOKUP to parent
  - Down
    - Send CMD_LOOKUP/PEER_LOOKUP to all the positive child and peers without keeping the information in a DB
- Case of no positive BF
  - if depth = 0, send CMD_LOOKUP to parent
  - if depth > 0, send CMD_LOOKUP_NACK to parent

CMD_LOOKUP_NACK

0	1	25	29	30	34
prot	name	ip_org	up/down	depth	

- Sent to parent when no positive BFs
- If depth > 0, forward CMD_LOOKUP_NACK to parent by decrease the depth by one
- If depth = 0,
  - Mark the corresponding child in the list as negative
  - When all the child and peers in the list turn into negative, send CMD_LOOKUP to parent

CMD_LOOKUP_TOP_DOWN

0	1 … 25	29
prot	name	ip_org

- It is sent from top server to its child for lookup
- Since it is coming down from top server,
  - The given name has to be eventually found
  - It is forwarded only to down
    - No need of up/down
- If any positive child
  - Just forward the message to the corresponding child
- If no positive child
  - Just discard the message

CMD_LOC_UPDATE

0	1	2	26	27	27+vlen	28+vlen	32+vlen
prot	mode	name	vlen	LOC	up/down	depth	

- It is a message of name lookup for LOC update
- Works as the same as the CMD_LOOKUP until the name is found
- Once the name is found, just updates the LOC in the table according to the mode

CMD_LOC_UPDATE_NACK

0	1	2	26	27	27+vlen	28+vlen	32+vlen
prot	mode	name	vlen	LOC	up/down	depth	

- It is a NACK of name lookup for LOC update
- Works as the same as the CMD_LOOKUP_NACK

CMD_LOC_UPDATE_TOP_DOWN

0	1	2	26	27	27+vlen
prot	mode	name	vlen	LOC	

- It is sent from top server to its child
- Works as same as the CMD_LOOKUP_TOP_DOWN
- If the name is found, then just updates the LOC in the table according to the mode
- Otherwise, discard the message

PEER_BF_UPDATE

- Updates the corresponding BF
- Do not sent to parent
- It works the same on the top peering

PEER_LOOKUP

- It is a lookup message from peer to peer
- If any positive BFs,
  - Store the name and peer information in DB
  - Send CMD_LOOKUP or PEER_LOOKUP to the corresponding servers
  - If gets NACKs from all of them, send PEER_LOOKUP_NACK according to the information in DB
- If no positive BF, send PEER_LOOKUP_NACK

PEER_LOOKUP_NACK

- It is a NACK message to peer
- Send it to the corresponding peer according to the information in DB

PEER_LOOKUP_TOP

0	1	25	29
prot	name	ip_org	

- It is a lookup message between top servers
- No need to store the name and peer information in DB
- If any positive BFs,
  - Send CMD_LOOKUP_TOP_DOWN
- Otherwise, discard it

PEER_LOC_UPDATE

0	1	2	26	27	27+vlen
prot	mode	name	vlen	LOC	

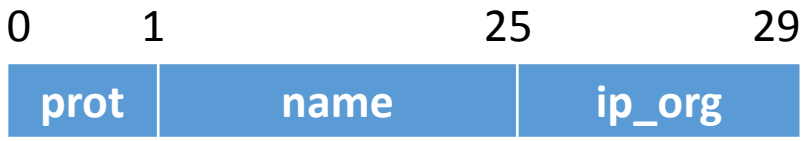
- It is a message of name lookup for LOC update from peer
- Works as the same as the PEER_LOOKUP until the name is found
- Once the name is found, just updates the LOC in the table according to the mode

PEER_LOC_UPDATE_NACK

0	1	2	26	27	27+vlen
prot	mode	name	vlen	LOC	

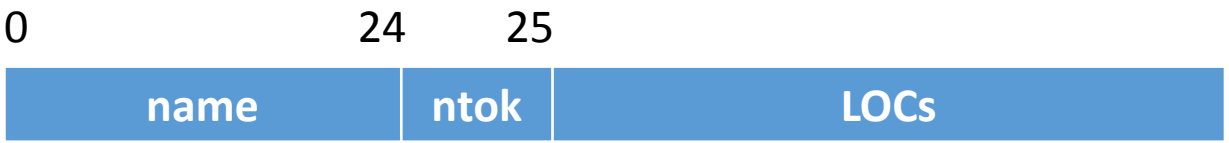
- It is a NACK of name lookup for LOC update to peer
- Works as the same as the PEER_LOOKUP_NACK

PEER_LOC_UPDATE_TOP



- It is a lookup message between top servers for LOC update
- No need to store the name and peer information in DB
- If any positive BFs,
  - Send CMD_LOC_UPDATE_TOP_DOWN
- Otherwise, discard it

REP_LOOKUP

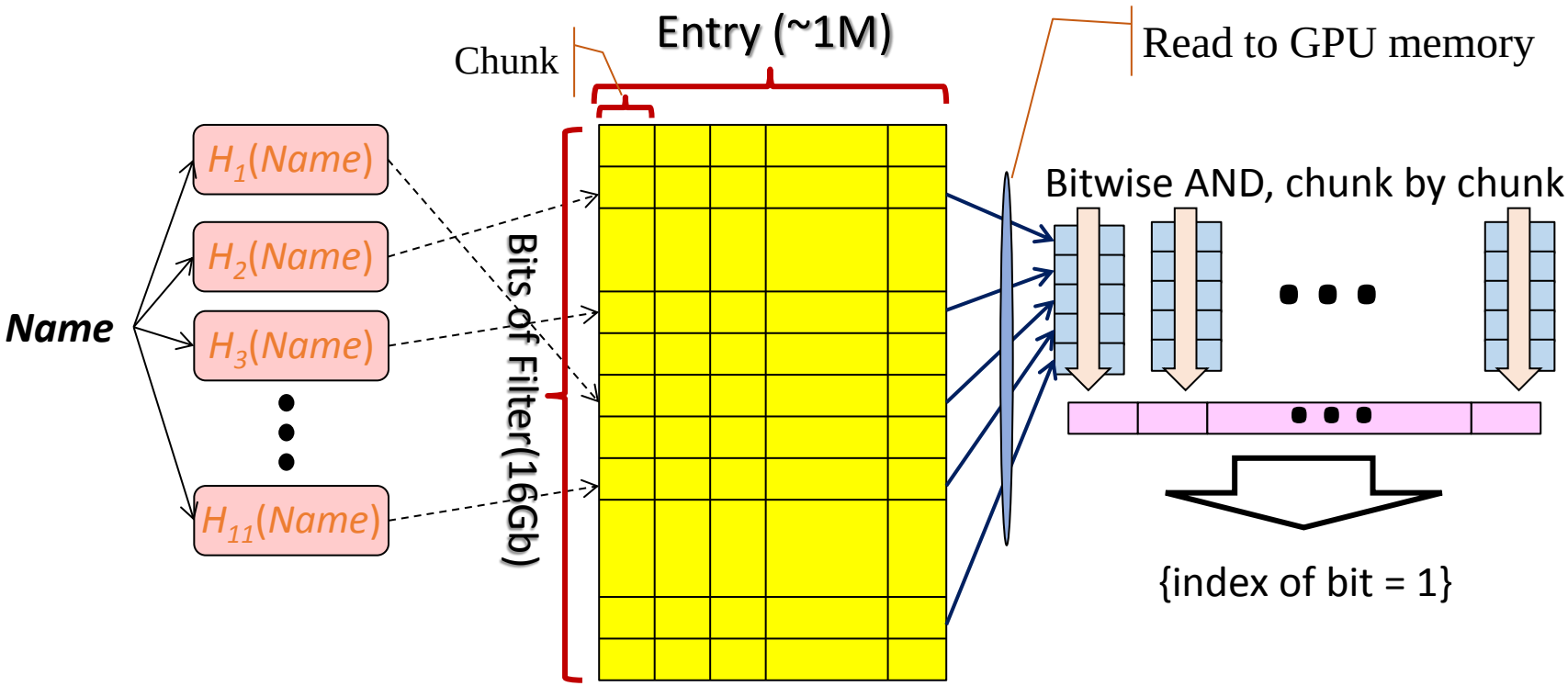


- It is a reply to the client
- ntok : number of LOCs
- LOCs includes all the locators of the name

Implementation environments

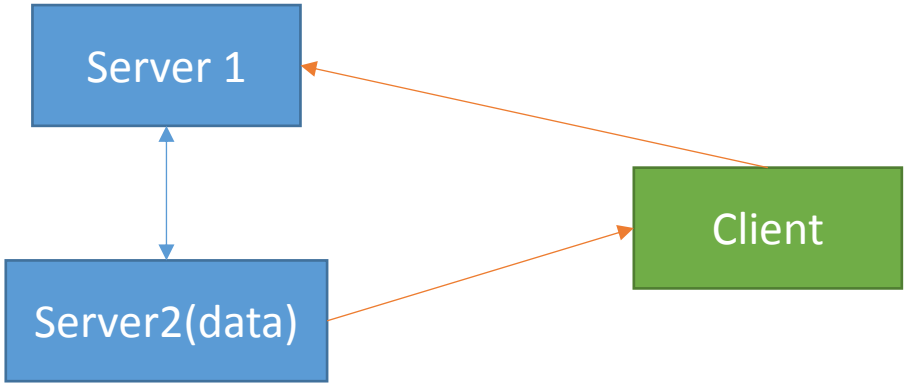
- Name : 24 byte
- LOC : variable length string
- BF size : 2MB
- 11 hash functions
- Maximum number of names in a BF : 10^6
- False positive probability < $4.586 * 10^{-4}$
- UDP is used
- CPU : AMD Opteron(tm) Processor 4180 x 12 cores
  - Memory : 32GB
- GPU : NVIDIA TITAN X 6GB

GPU assisted implementation

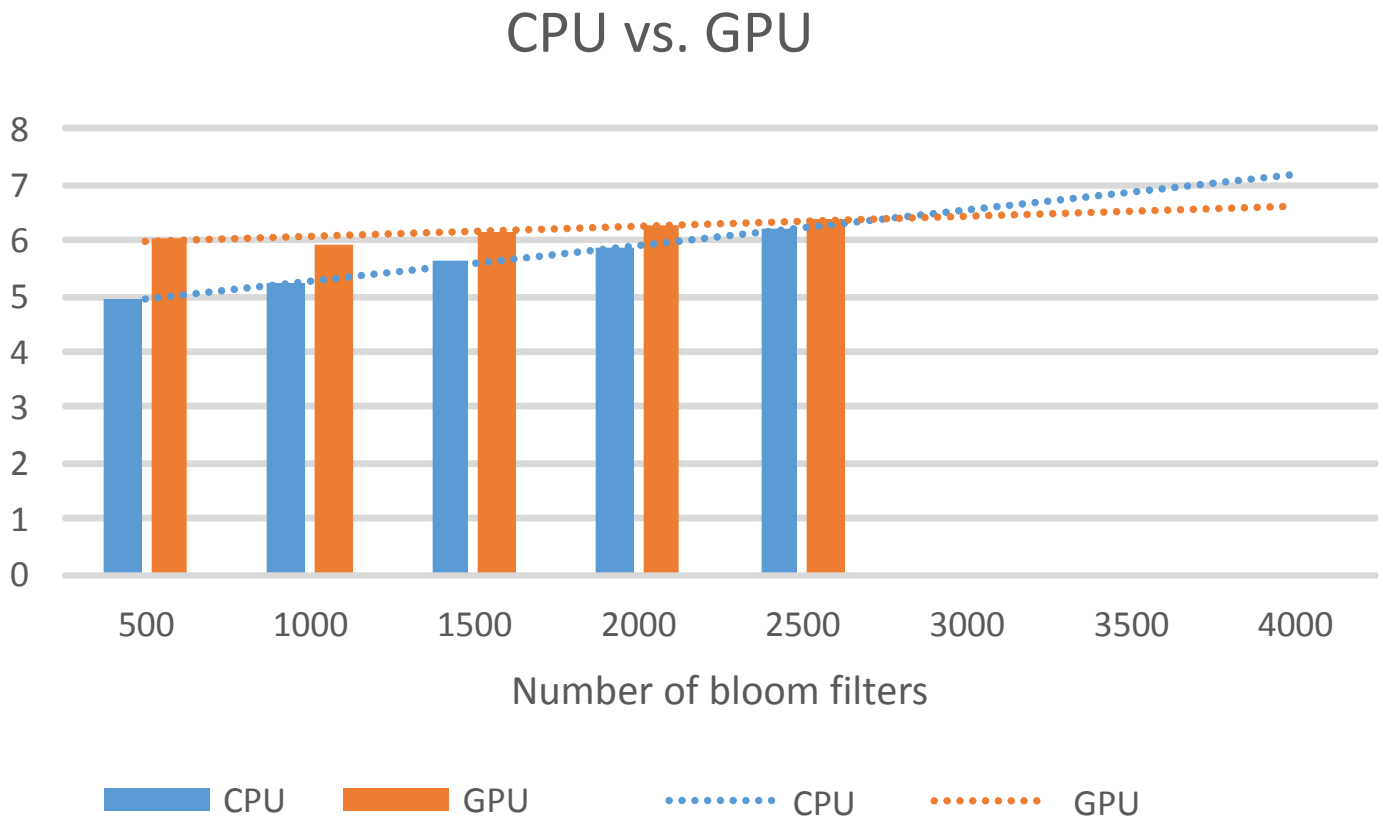


BF searching time

- The server has a number of BFs
- Insert 1000 random names into each BF
- Client sends 10^4 lookup messages to the server



Execution time (s)



Questions or Comments

www.idnet.re.kr