NDN-based Ethereum Blockchain

Quang Tung Thai, Namseok Ko, and Sun-Me Kim

Electronics and Telecommunications Research Institute (ETRI)

{*tqtung, nsko, kimsunme*}*@etri.re.kr*

December 1, 2020

Motivations

- Blockchain may have many potential use cases
- Data propagation traffic in blockchain networks is hightly redundant. We believe it can take benefits from ICN
- We want to develop the first functioning blockchain on NDN
  - to support decentralized applications in NDN
  - to support NDN-based blockchain research
  - to find what changes are needed for better supporting of blockchain-liked distributed systems

Why Ethereum blockchain?

- Supporting smart-contracts which may have many potential use cases
- Already has many smart-contract based decentralized applications
- Its network has been operating securely for years
- Source code is stable, well-supported by an active developer community
- A popular platform for supporting academic research

Data propagation in Ethereum blockchain network

- Blocks and transactions are broadcasted
- Gossip protocol over P2P overlay
- UDP-based Kademlia DHT for constructing a structured P2P overlay
- Some peers are selected from Kademila routing table for state synchronization
- TCP connections are established with these peers for data propagation

Data propagation in Ethereum blockchain network

- Small objects are pushed directly to all managed peers
- Large objects are pushed to a few managed peers
- .. and announced to remaining peers for downloading
- Redundant traffic : many copies of an object are sent and received
- Downloading object from far distant node

Design concerns

- Blocks/transactions broadcasting resembles the data synchronization problem
- Can we utilize existing protocols such as ChronoSync, PSync and VectorSync?

- Do we need P2P overlay?

Design concerns

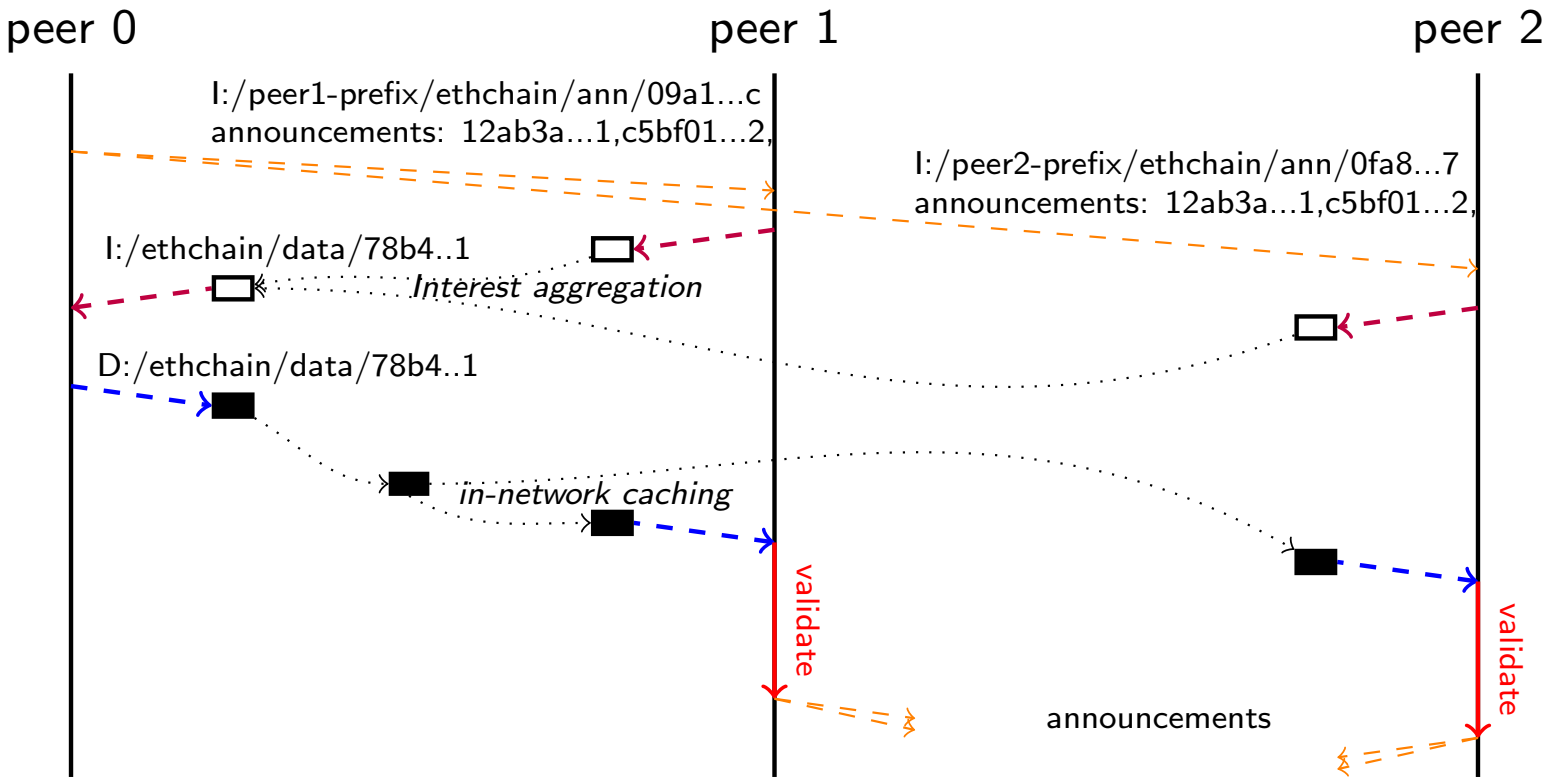
- Blocks/transactions broadcasting resembles the data synchronization problem
- Can we utilize existing protocols such as ChronoSync, PSync and VectorSync? No!
  - Churning — no → managed membership
  - Scalability — do not use → Interest multicast
  - Malicious nodes — should not → cause nodes halt for state reconsilation by sending invalid states
- Do we need P2P overlay? Yes!
  - Validating data before propagating

Ideas

- Assumption:
  - Blockchain nodes have routable prefixes (node names)
- All data objects (blocks/transactions) have globally unique names `/ethchain/data/0x152ac3eb...`
  - enabling in-network caching
  - enabling interest aggregation
- Separating object name and forwarding information by specifying node name as ForwardingHints
- Announce-Pull data broadcasting
  - identity of new object is announced through p2p overlay
  - The object is then retrieved using Interest/Data exchange with its identity being used for deriving Interest's name
  - Nodes validate the data before forwarding, invalid data is not broadcasted

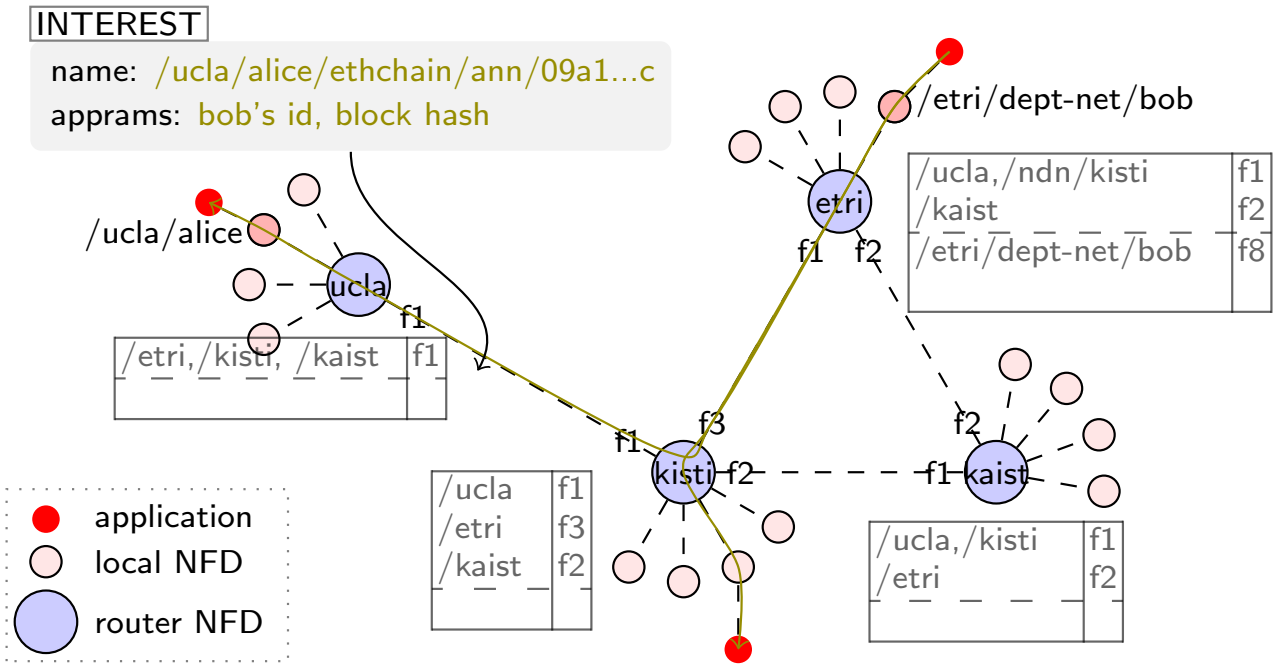
Announce-pull data broadcasting

- data announcing on p2p; data retrieving with Interest/data exchange
- unique names for Interest aggregation and in-network caching



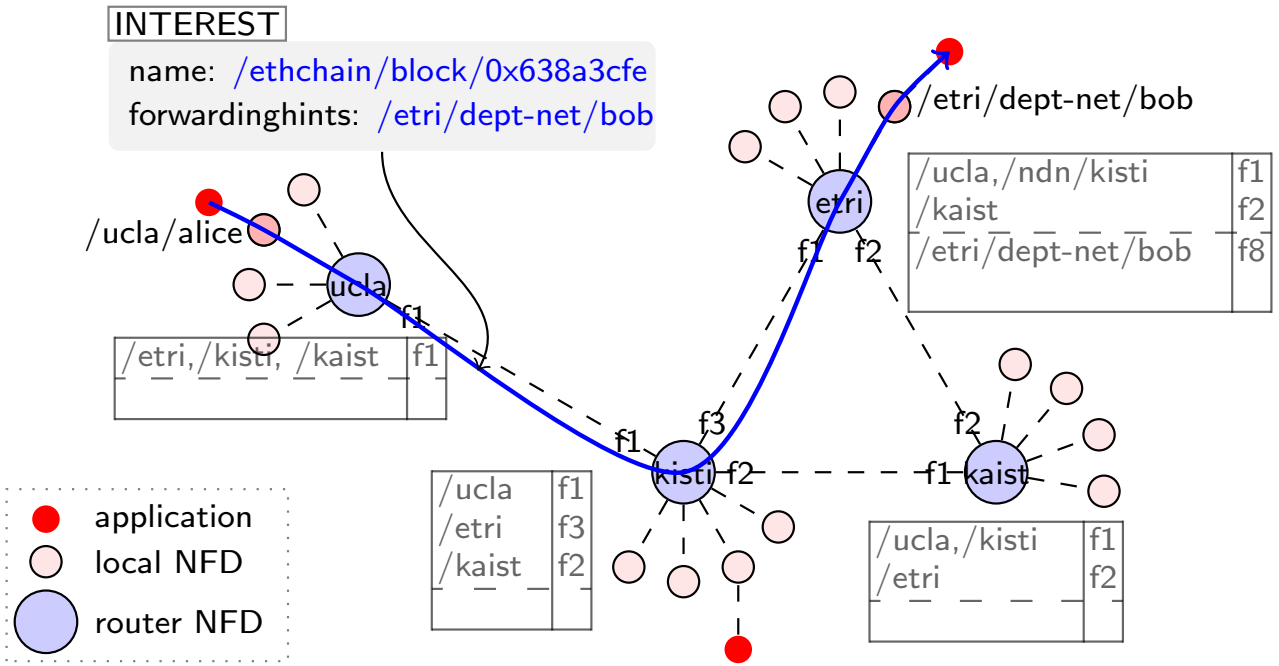
Locating location-independent named data objects

- gossiping announcement let nodes know the data name and who is having it
- p2p overlay plays the NDNS role: `/ethchain/data/78b4..1` → `/etri/dept-net/bob`



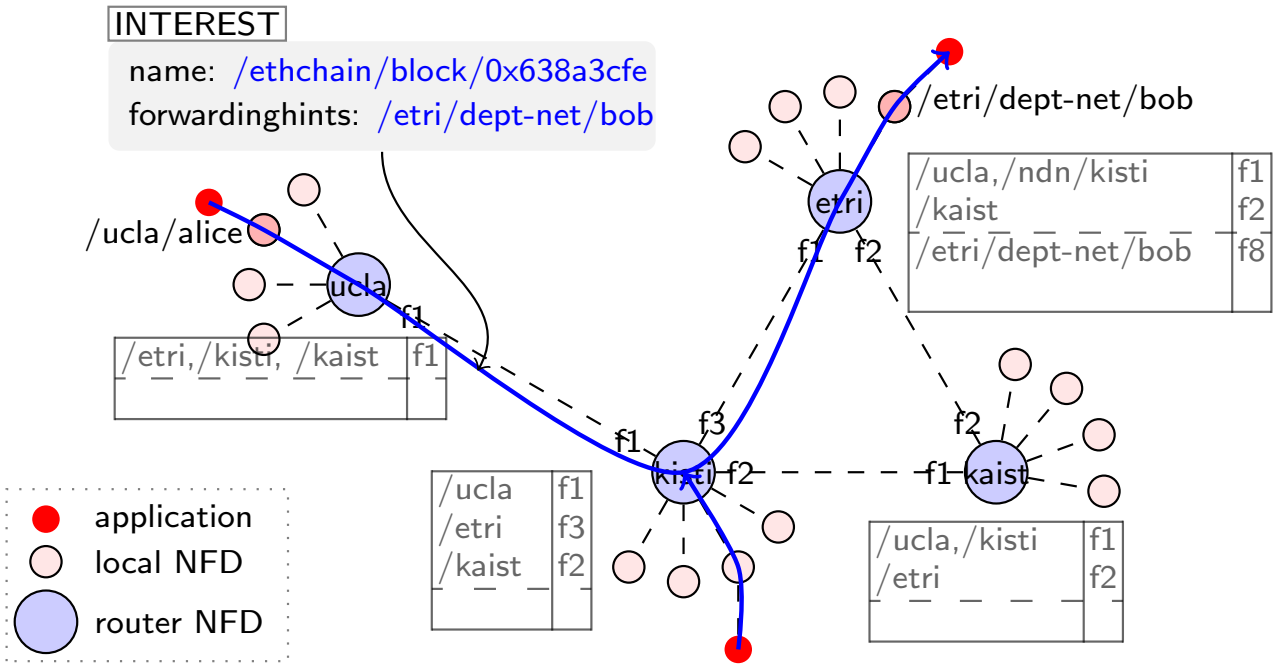
Locating location-independent named data objects

- gossiping announcement let nodes know the data name and who is having it
- p2p overlay plays the NDNS role: `/ethchain/data/78b4..1` → `/etri/dept-net/bob`



Locating location-independent named data objects

- gossiping announcement let nodes know the data name and who is having it
- p2p overlay plays the NDNS role: `/ethchain/data/78b4..1` → `/etri/dept-net/bob`

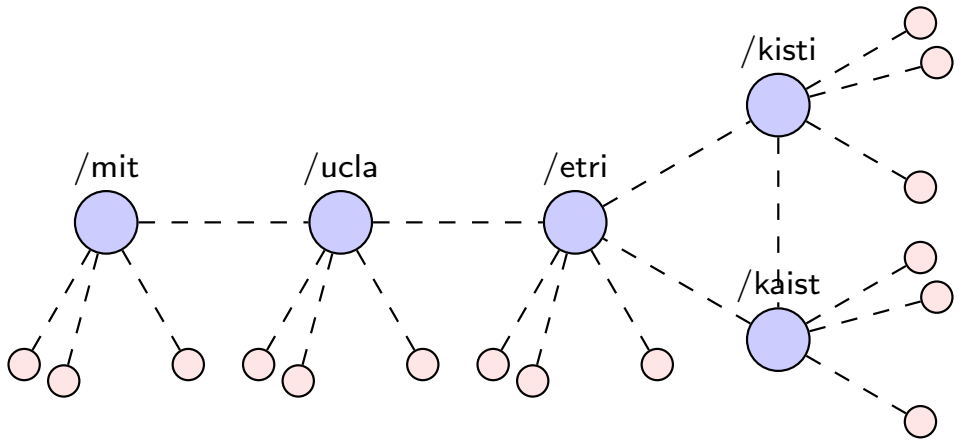


Implementation

- A fully functional NDN-based ethereum blockchain client
- Developed from the go Ethereum official client (`geth`)
- We need to develope a minimal go ndn client
- TCP/IP based p2p module is replaced by NDN-based p2p module
- Block/transaction broadcasting and chain synchronization protocols are redesigned and implemented for NDN
- Stable enough for conducting experiments

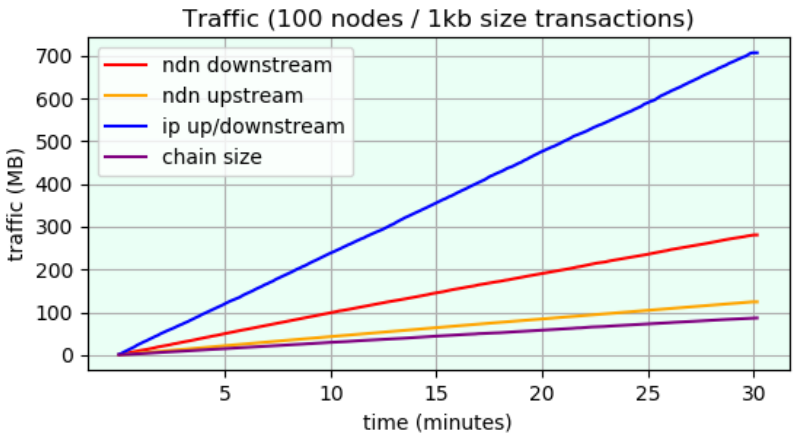
Experiments

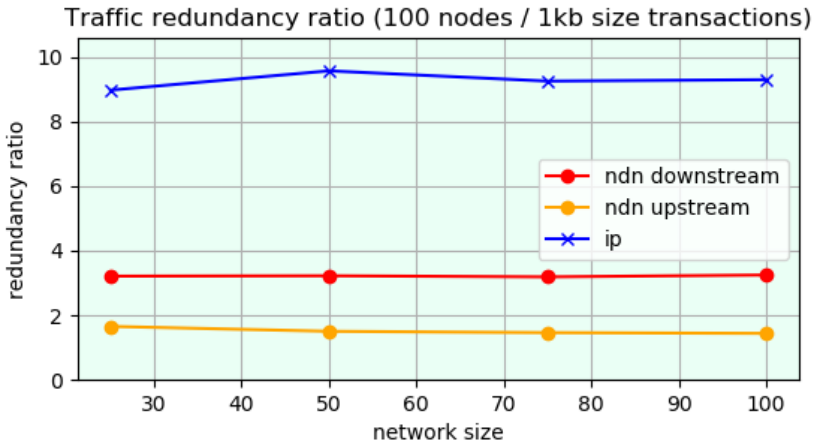
- Comparing to IP-based Ethereum blockchain in terms of traffic use
- Settings:
  - A networks of 5 NDN domains, each having 5,10,15 to 20 blockchain nodes
  - Sending 1kb transactions at constant rate
  - Upstream and downstream traffics are measured at every nodes
  - Traffic redundancy ratio $= \frac{traffic}{chainstoragesize}$
  - Caching ratio $= \frac{downstream - upstream}{downstream}$



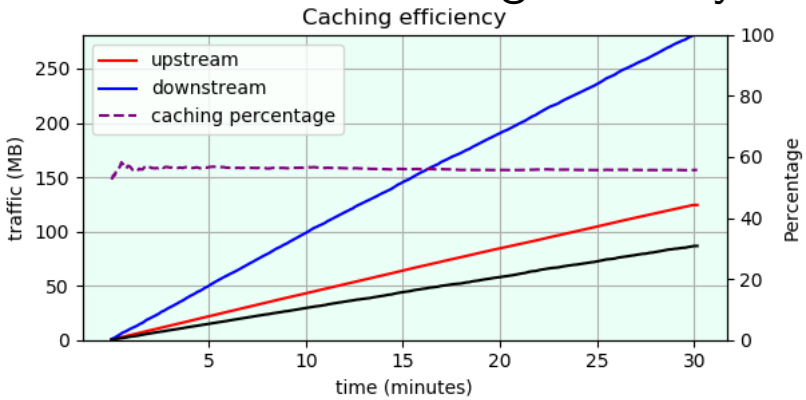
Results

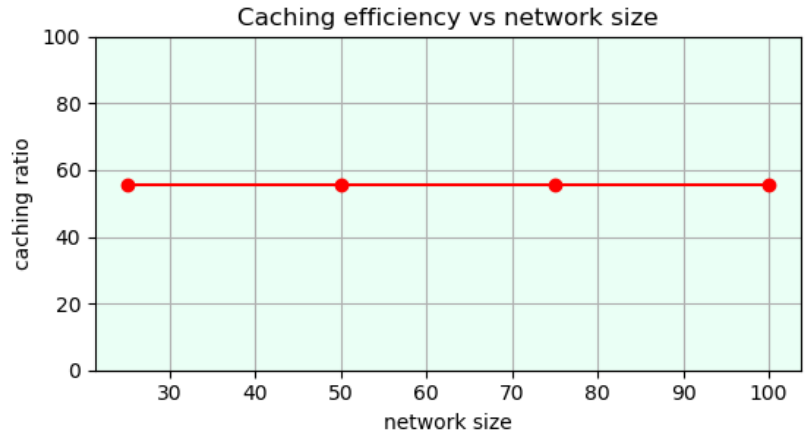
traffic use: IP vs NDN





caching efficiency in NDN-based blockchain

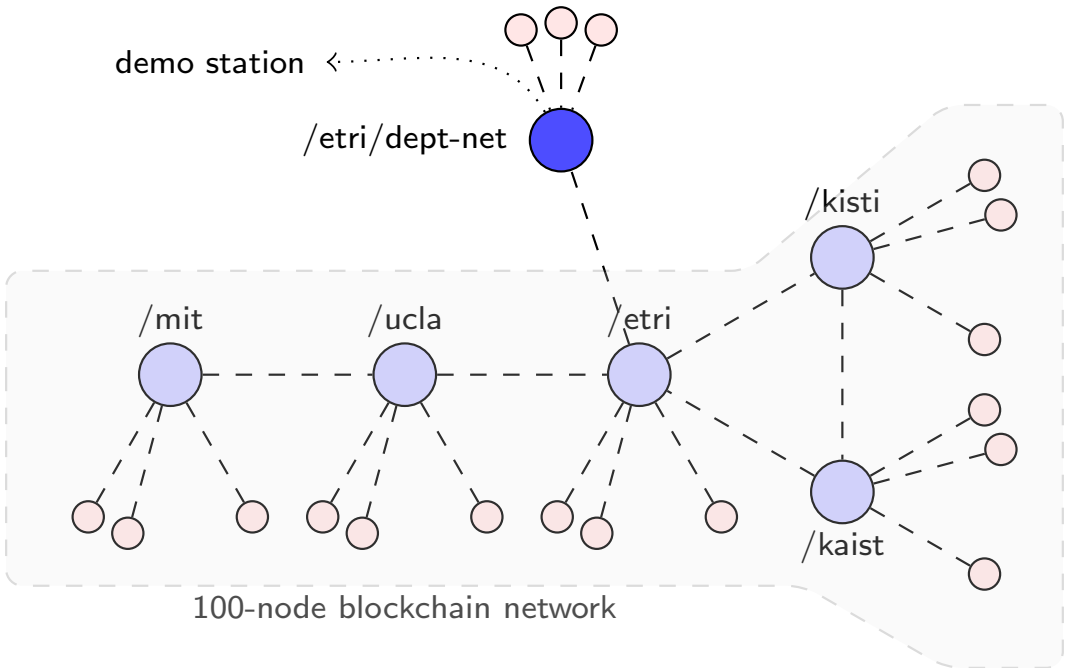




Conclusions & Discussions

- Traffic redundancy is reduced in NDN-based blockchain
- More tests on testbed networks for latency measurement
- Plan to publish as open source package
- Announcements are pushed in Interest packets (not ICN way?)

Demo


demo station
/etri/dept-net
/kisti
/mit
/ucla
/etri
/kaist
100-node blockchain network