Introduction to Sync
Outline

- NDN Sync overview
- Sync protocol design issues
- Sync protocol for mobile ad-hoc networks
What is NDN Sync?

• New transport *service* for data-centric networking [1]
• Synchronizing the namespace of shared dataset for distributed applications among all participants

Do you have some chat messages for /chatroom1 that I haven’t got?
Benefits

• Supports multi-party communication
  • Applications do not need to worry about individual participant addresses or how to reach them

• Desired data can be fetched from any node.
  • Resilient delivery under mobile & intermittent connectivity

Here’s a new chat message I just generated.
How to sync?

- Encode the data set (names) into a compact form
- Participants exchange and compare the dataset state to detect differences
Sync Design Issues

• How to represent one’s dataset state?
  • to make the comparison between datasets with least delay and overhead

• Pair-wise vs. group sync
  • Pair-wise: A – B, B - C, C - D, etc.
  • Group: every node tells all other nodes its data state.

• Sync names vs. sync data
  • Sync protocol may reconcile names only.
  • Individual participants decide whether or when to retrieve missing data based on local conditions
Example: ChronoSync [3]

- Use hash as a compact form of one’s dataset state
- Two-level digest tree
  - Each producer represented by <producer-name, seq#>
- Participants exchange root digest with each other to detect differences in dataset

```
Root Digest = D0
```

```
Prefix = /a
Seq# = 100
Node A
```

```
Prefix = /b
Seq# = 50
Node B
```

```
Prefix = /c
Seq# = 21
Node C
```

```
Prefix = /d
Seq# = 89
Node D
```
ChronoSync (cont’d)

- Multicast sync interests to all participants in an app group

Limitation: cannot directly decode dataset difference from comparing digests
- Applying a few good engineering hints can work well under normal conditions
• Challenge: Intermittent connectivity with mobility
  • Short-lived connections
  • Dynamically changing topology
• Need to rethink Sync design [1]
MANET Sync [2]

• State encoding
  • **State vector** [ A:5, B:7, ... ]
  • Carried in Sync Interest

• State change detection is achieved directly from state vector comparison [5].
  • Whenever one generate new data: send new state vector
  • Periodic transmission of Sync Interests as backup
• New issues in dataset name sync versus data sync
  • Not all nodes are connected at any given time
  • Not all nodes fetch all data
  • One may learn the name of a new piece of data from neighbor which does not have the data

**Solution-1:** only inform neighbors the names if one has the data
**Solution-2:** engineer the networking with data repositories to increase data availability
Summary

- Transport layer service for NDN: dataset name reconciliation for distributed applications
- Represent the dataset using a compact format over data names
- Exchange dataset state to detect missing data names
- Implemented in libraries

Join the NDN session Tuesday afternoon to learn more details: “A Brief Introduction to NDN Dataset Synchronization”


