

MFTP: a Clean-Slate Transport Protocol for the Information Centric MobilityFirst Network

Kai Su (presenting), Francesco Bronzino,
K. K. Ramakrishnan*, and Dipankar Raychaudhuri

WINLAB, Rutgers University

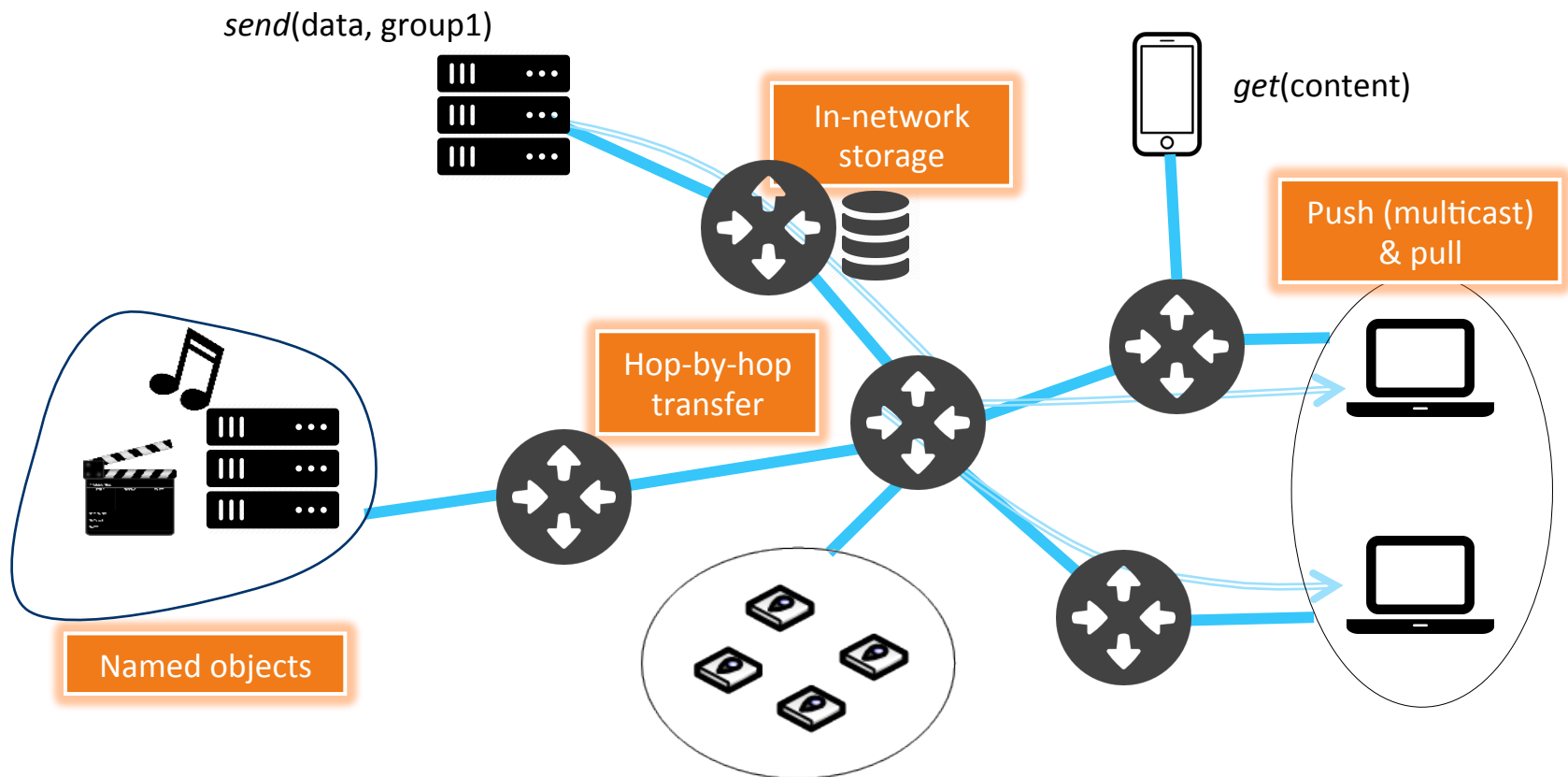
*University of California, Riverside

Motivation

- Name-based, or Information Centric architectures
 - e.g. NDN, CCN, MobilityFirst
 - Are aimed at providing richer set of data transfer semantics:
 - Pull: *get(this_video)*
 - Push (multicast): *sendto(this_video, the_group)*
 - Context-based transfers: *notify(coord, ambulances_in_vicinity)*
 - These patterns are not well-captured by conventional transport, e.g. TCP
 - Was built to provide point-to-point communication, presuming addresses of endpoints are known
 - Single IP address used both as the *identifier* and the *location* of a host

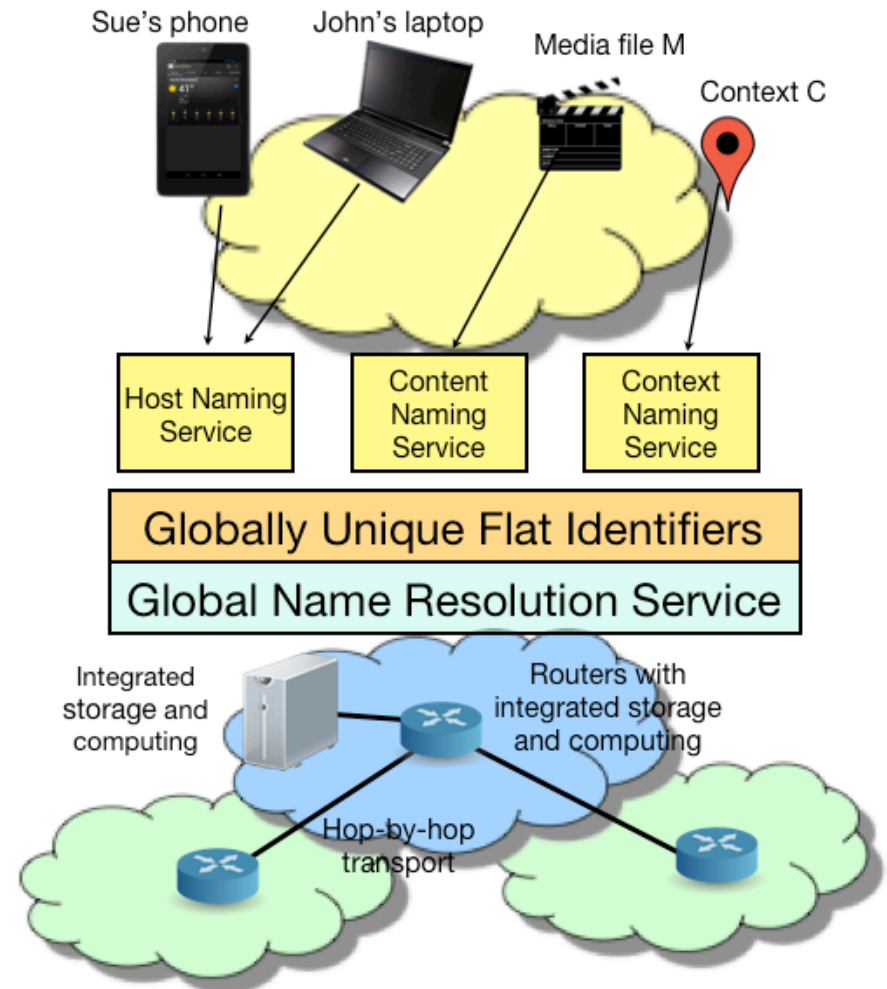
Motivation - continued

- Characteristics of Information Centric Networks



MobilityFirst architecture overview

- Names (GUID) to identify network objects (e.g. source, sink, content, context)
- A locator (network address) separate from GUID for addressing
- A Global Name Resolution Service to track GUID to NA mappings

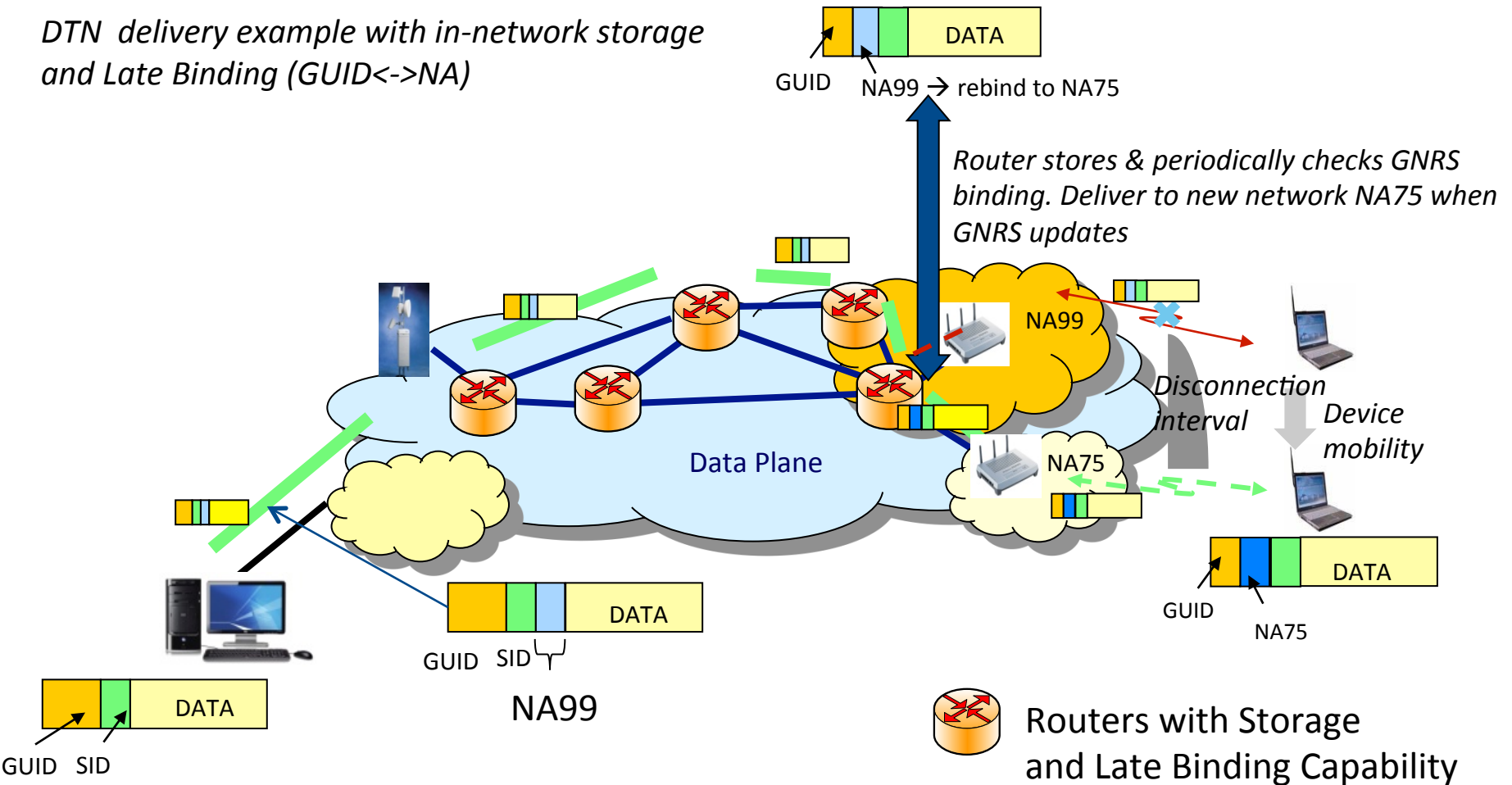


MF vs. NDN comparison

- Naming:
 - MF: flat, globally unique
 - NDN: hierarchical
- Routing:
 - MF: hybrid name + address routing supported by GNRS and late binding of NAs to names where needed
 - NDN: Interests routing based on FIB entry; Data forwarding according to PIT states
- Mobility support
 - MF: via name resolution with optional late binding (needed for in-flight data to moving dst.)
 - NDN: consumer and producer mobility supported by different mechanisms

MF data delivery example

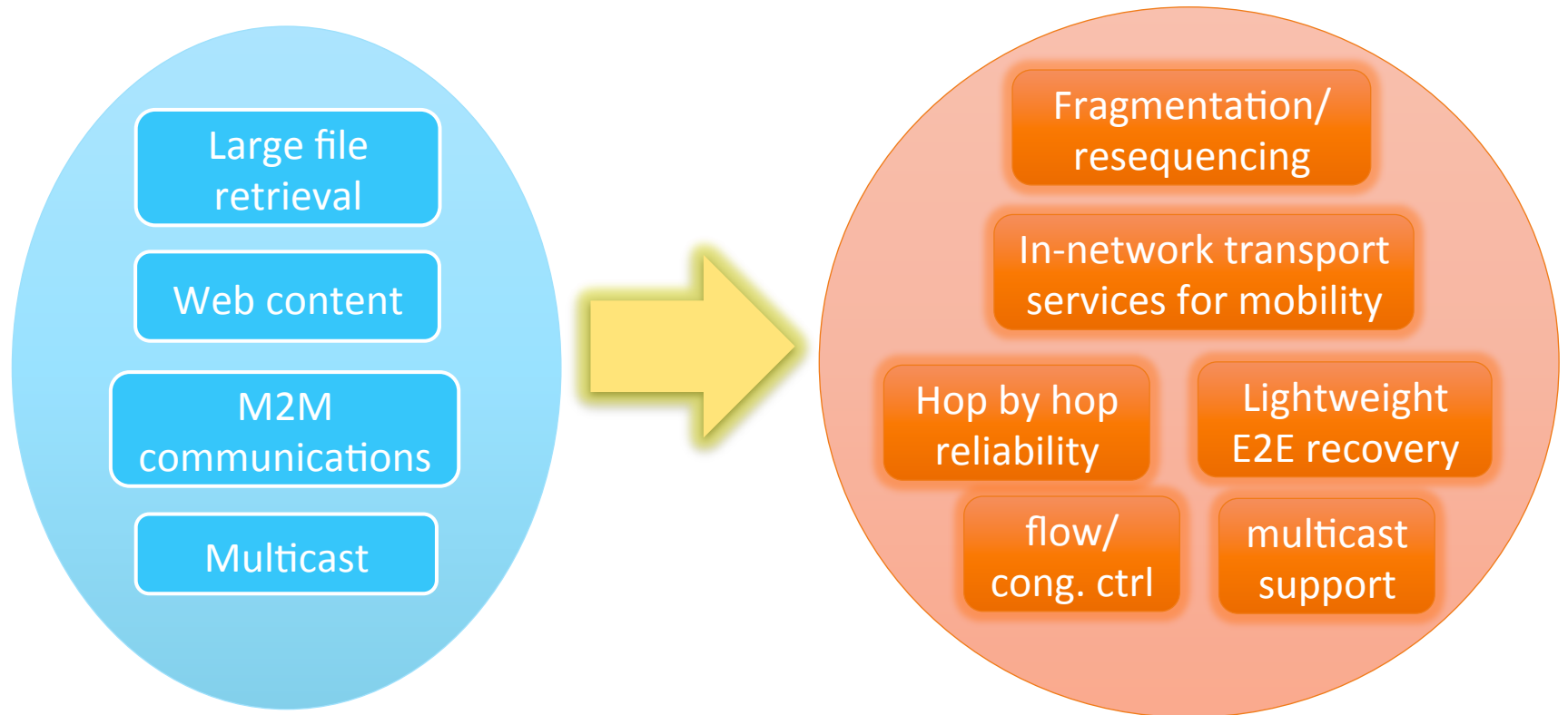
DTN delivery example with in-network storage and Late Binding (GUID \leftrightarrow NA)



Send data file to "John Smith22's laptop", SID= 11 (unicast, mobile delivery)

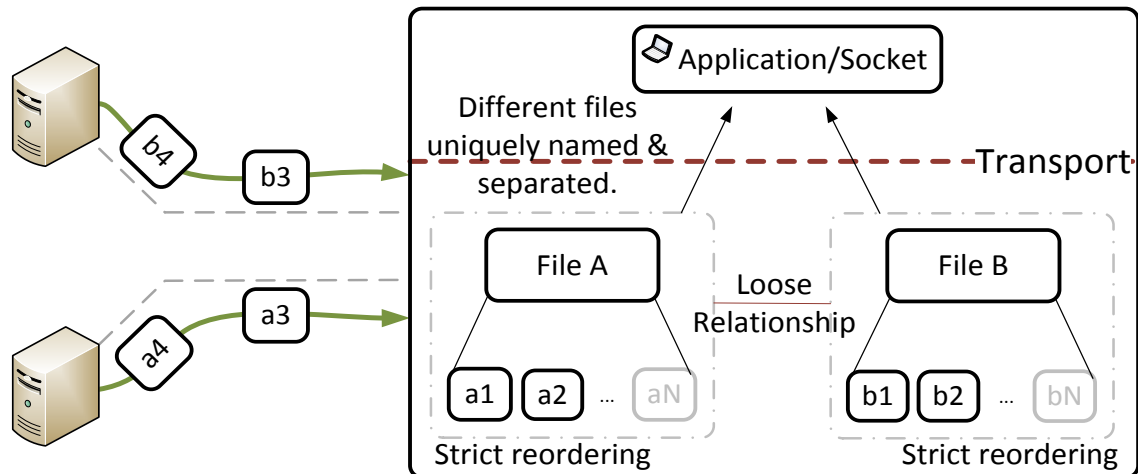
From service scenarios to transport requirements

Goal: have a flexible transport such that every service doesn't have to reinvent the wheel



MFTP design – fragmentation and sequencing

- Application data fragmented into large chunks
- In-order delivery for chunks of each content
- Because each content is named, no head-of-line blocking if concurrently requesting more than 1 content



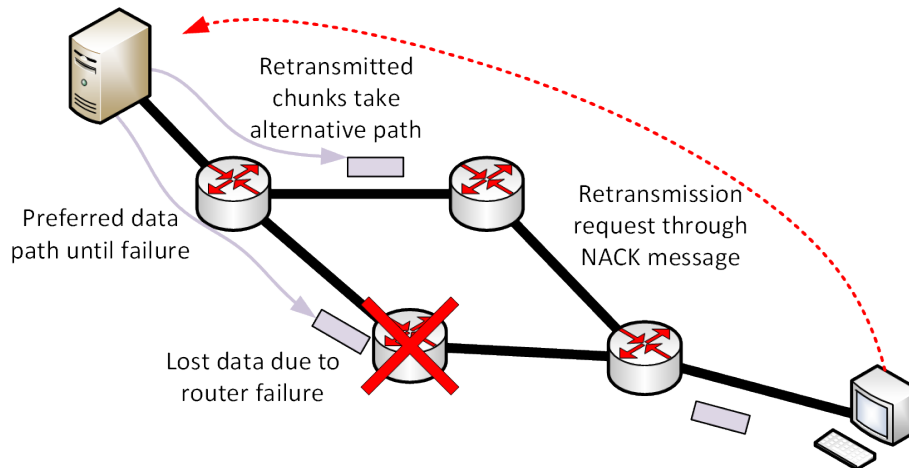
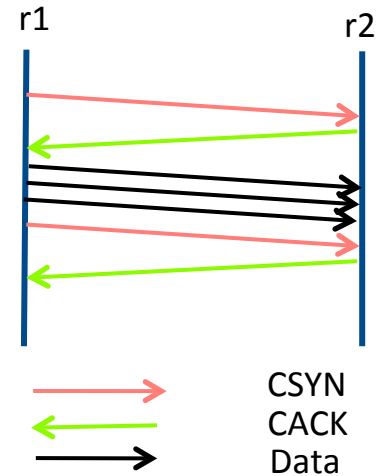
MFTP design – per-hop and end-to-end error recovery

- Per-hop recovery

- Based on HOP: chunk transfer w./ block ACK
- Optimizations:
 - multiple chunks can be transferred concurrently
 - for short transfers: CSYN/Data sent all at once

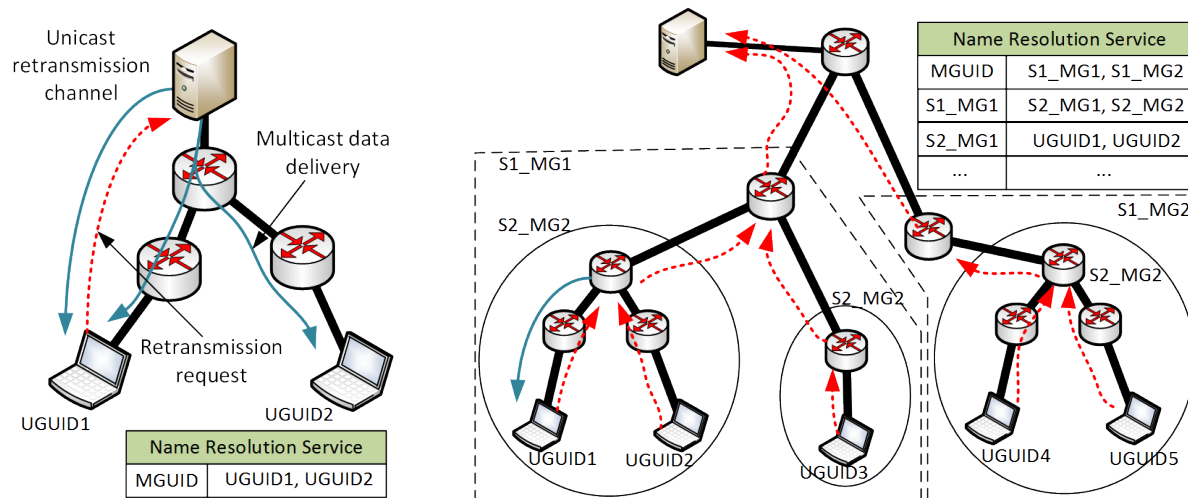
- End-to-end mechanisms

- To deal with delivery failure at the chunk level
- Flexible: NACK, ACK, *don't care* (UDP) options
- Most commonly: NACKs - dst sends retx requests when anticipated data doesn't arrive after a long period of time



MFTP design – on congestion control and multicast

- Congestion control:
 - Unlike TCP combining congestion control and flow control, MFTP separates the two tasks
 - Window based flow control
 - ECN + source rate adaptation for congestion control
- Multicast
 - Use a group GUID to identify a multicast group
 - NACK for requesting undelivered data
 - In-network aggregation of NACKs



Evaluation methodology

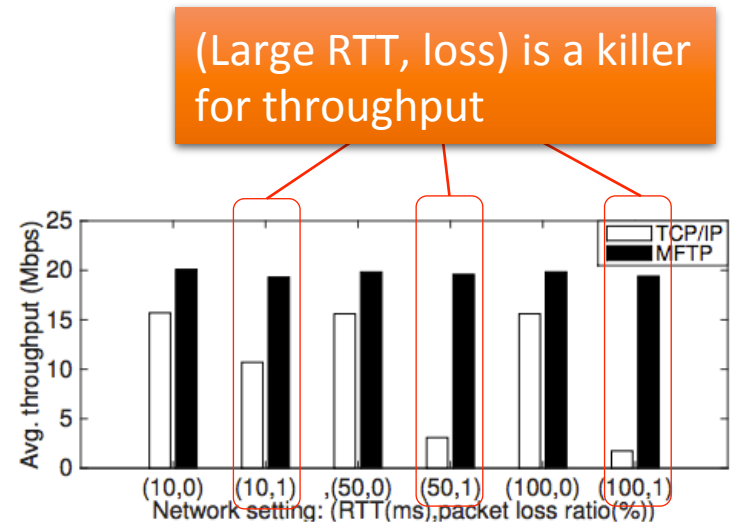
- Prototype implementation
 - MFTP Endhost stack
 - In-network transport services at Click routers
- Experiments:
 - ORBIT topology set-up



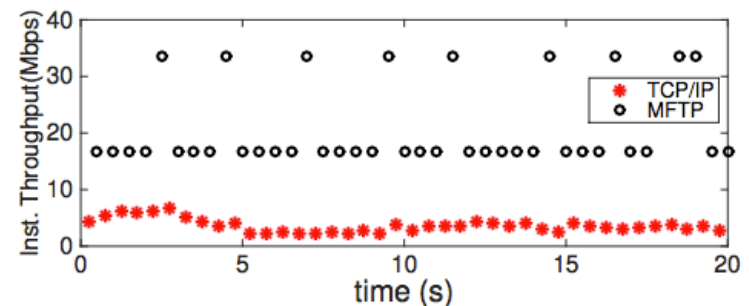
- Compare MFTP vs. TCP
- Evaluated use cases:
 - Large file transfer under wireless
 - Content retrieval at intermittently connected client
 - To evaluate transport proxy for disconnection
 - Web content retrieval

Use case evaluation – large content retrieval

- Client requests and retrieves a 400MB file from the server
- Better throughput, as a result of
 - Loss recovered locally, instead of on end-to-end basis
 - No misinterpretation of random loss on wireless links as congestion, i.e. no reduction of rate upon wireless loss



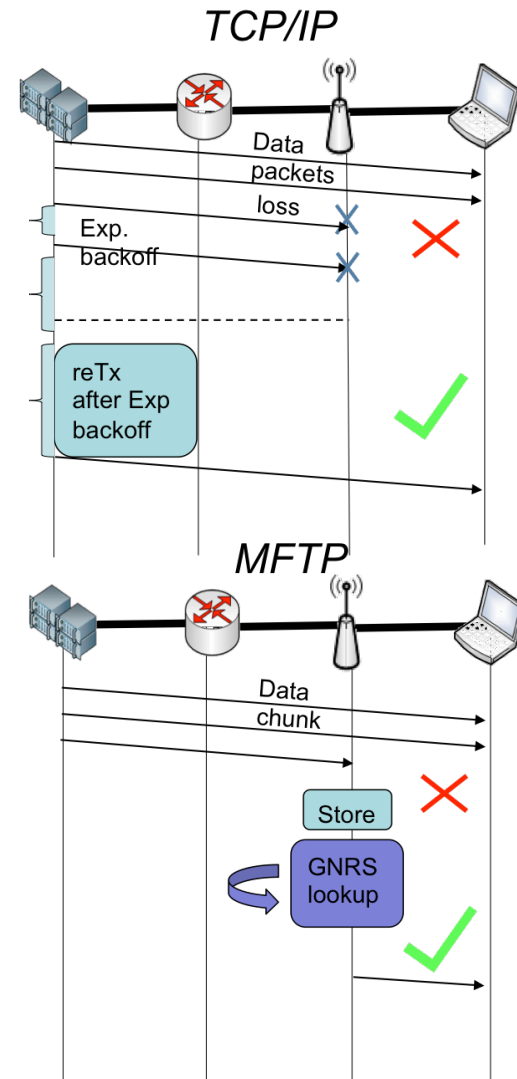
(a) Average throughput comparison for 6 different (RTT, loss rate) profiles.



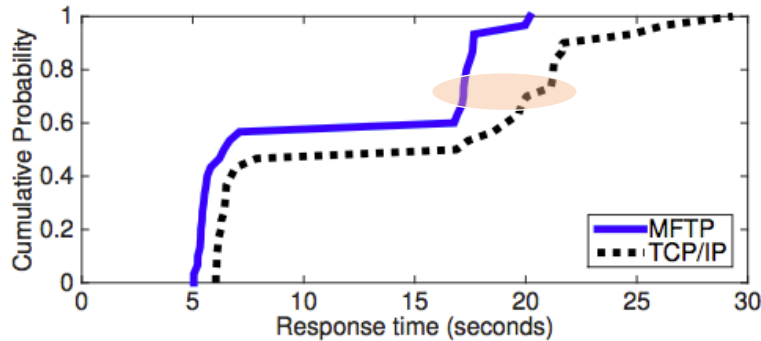
(b) Instantaneous throughput (per 500ms) for 50ms RTT and 1% loss.

Use case evaluation – transport proxy for disconnection

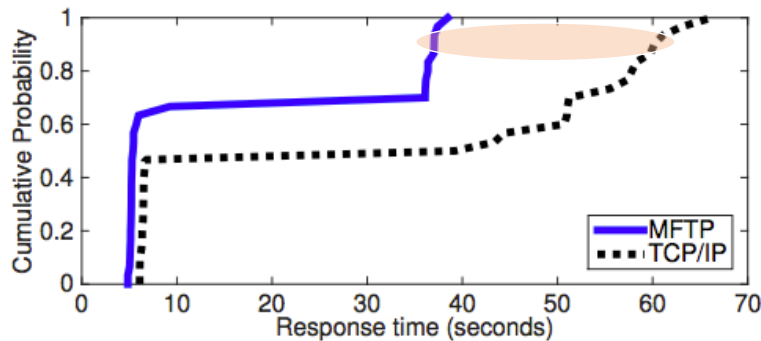
- Experiment: retrieve 10MB files
 - WiFi conn. is on for 10s, and is off for 10s. Always on afterwards.
 - Client requests the file at random time in first 10s
 - If transfer is not complete in 10s, it experiences disconn
 - Repeat the experiments 30 times, with MFTP/TCP
- In the presence of disconnection:
 - TCP: reTx based on a timer w./ exponentially increasing timeout
 - MFTP: stores data in case of disconn., NA resolution is triggered when the storage timer expires and it reTx only when conn. is recovered.



Use case evaluation – transport proxy for disconnection



(a) With 10s disconnection



(b) With 30s disconnection

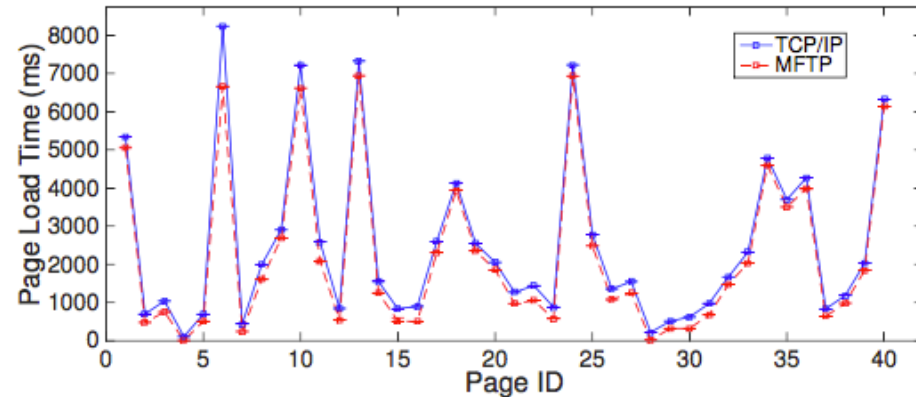
Improvement in response time proportional to length of disconnection.

MFTP:

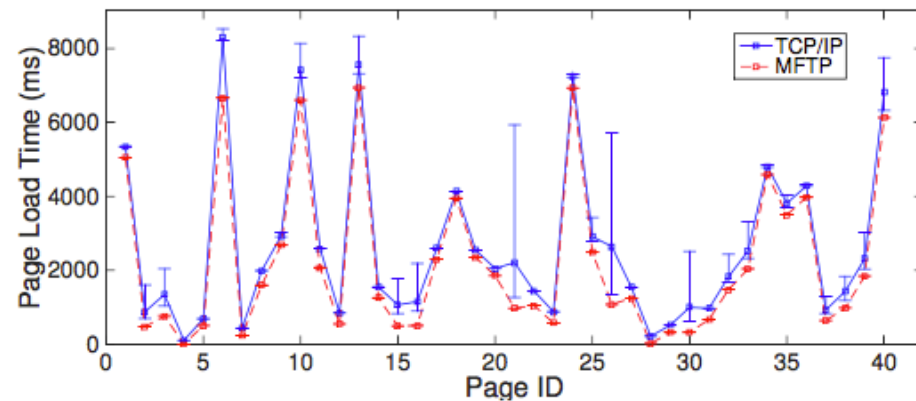
- Better performance due to better accuracy in the knowledge of end-to-end connectivity
- Better manageability of end-to-end timers

Use case evaluation – web content retrieval

- 40 webpages and constituent objects placed on server
- Client use a web browser emulator to sequentially retrieve all pages
- MFTP uses HTTP-MF-HTTP proxies at the two ends
- TCP uses default browser settings:
 - 6 concurrent connections
- MFTP has shorter PLT, due to
 - No head-of-line blocking (inherent in HTTP/TCP suite)
 - No heavy set-up for short transfers
 - Better loss tolerance



(a) 50ms RTT, no loss



(b) 50ms RTT, 1% residual loss

Conclusions

- ICN needs new class of transport distinct from TCP/UDP
- Designed MFTP to operate on top of MF protocol stack
 - Fragmentation and sequencing
 - In-network transport service
 - Per-hop and end-to-end error correction
- Experimental evaluation shows improved throughput/latency in content deliveries, and better mobility support.
- Many of the principles can be adapted to work for other ICN protocols

Thank you

MobilityFirst website: mobilityfirst.winlab.rutgers.edu