Internet Storage Sync Problem Statement

draft-cui-iss-problem

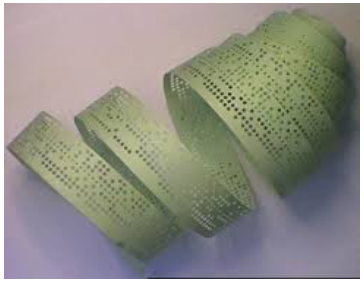
Zeqi Lai

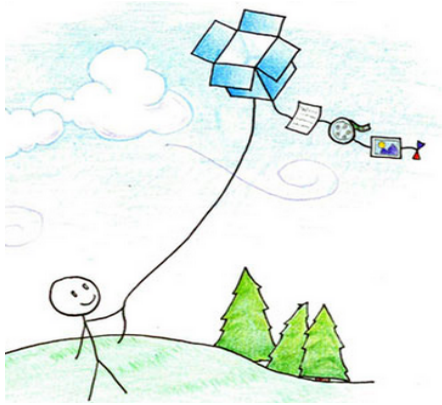
Tsinghua University

Outline

- Background
- Problem Statement
  - Service Usability
  - Protocol Capabilities
- Our Exploration on Protocol Capabilities
- Summary

The way we store our data…





Internet Storage Sync Services

- New data entrance of the Internet
  - Basic function: storing, sharing and synchronizing data
  - Large user base: Dropbox has more than 400 million users
  - Significant traffic: Dropbox accounts for approximately 4% of the total Internet traffic [IMC 2012]

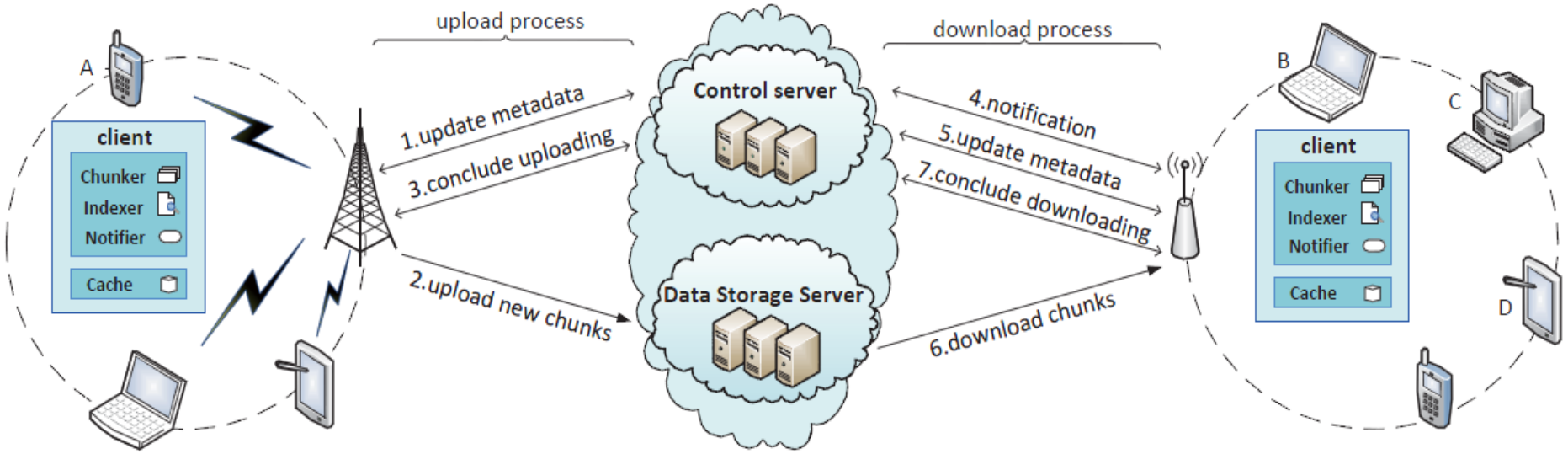
Internet Storage Sync Services

- New data entrance of the Internet
  - Major players: Dropbox, Google Drive, One Drive, Box.com, Apple …
  - Combining other services via APIs: photo sharing, email attachment, social apps




Typical Architecture & Flow

- Typical architecture of ISS services
  - Control flow: exchanging metadata
  - Storage flow: exchanging contents
  - Sync process with your multiple clients



Capabilities in Sync Protocol

- Key storage capabilities [IMC' 13]
  - **Chunking:** splitting a large file into multiple units
  - **Bundling:** multiple small chunks as a single one
  - **Deduplication:** avoiding the retransmission of content already available in the server
  - **Delta-encoding:** updating the modified portion

Capabilities	Windows			
	Dropbox	Google Drive	OneDrive	Seafile
Chunking	4 MB	8 MB	var.	var.
Bundling	√	×	×	×
Deduplication	√	×	×	√
Delta encoding	√	×	×	×
Data compression	√	√	×	×

Outline

- Background
- Problem Statement (draft-cui-iss-problem)
  - Service Usability
  - Protocol Capabilities
- Our Exploration on Protocol Capabilities
- Summary

Outline

- Background
- Problem Statement
  - Service Usability
  - Protocol Capabilities
- Our Exploration on Protocol Capabilities
- Summary

Using Multiple ISS Services

- Users may use multiple services
  - Performance or functionality diversity
  - Dropbox works better for synchronizing docs
  - Google Drive connects to Gmail and Google Doc
  - BaiDu cloud provides 2TB free space

However that is not easy …

- For users
  - Users may install multiple similar clients
  - It is unable to synchronize data across services (e.g. sync between a Dropbox user and a Google Drive user)
- For application developers
  - A developer has to deal with many different APIs in order to connect his app with multiple sync services

Using a Private ISS Service

- Enterprise may want their own storage
  - Public ISS services may not be trusted
  - Like what email is doing
- It is difficult to build and use a private ISS service
  - There is no standard sync protocol
  - Need to start from scratch

Outline

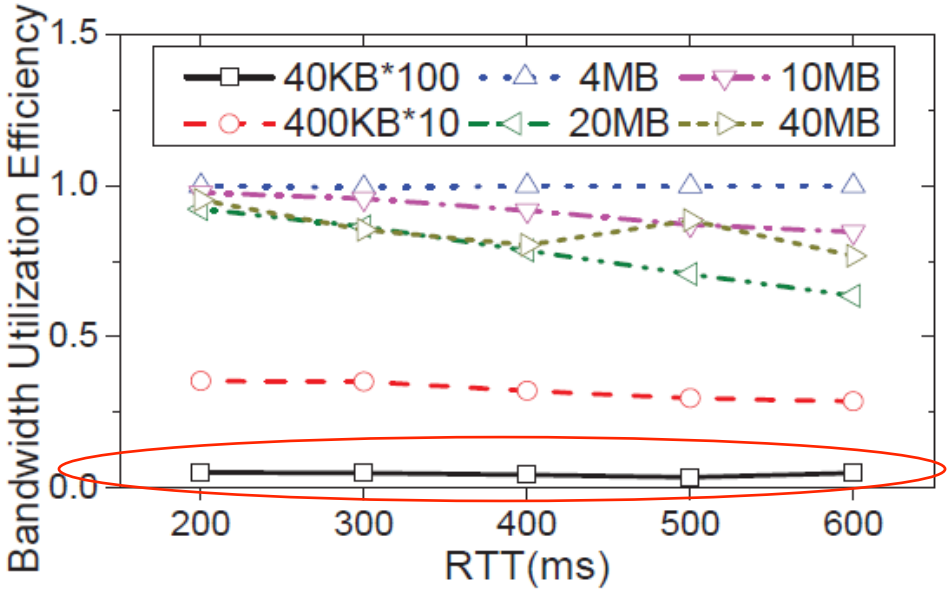
- Background
- Problem Statement
  - Service Usability
  - Protocol Capabilities
- Our Exploration on Protocol Capabilities
- Summary

Rethinking about Capabilities

- Ideal
  - With these capabilities, sync services can efficiently synchronize our data
- Reality
  - The **sync time is still much longer than expected** with various network conditions!
- Measurement study
  - We measured several sync services to identify and analyze the sync inefficiency problem

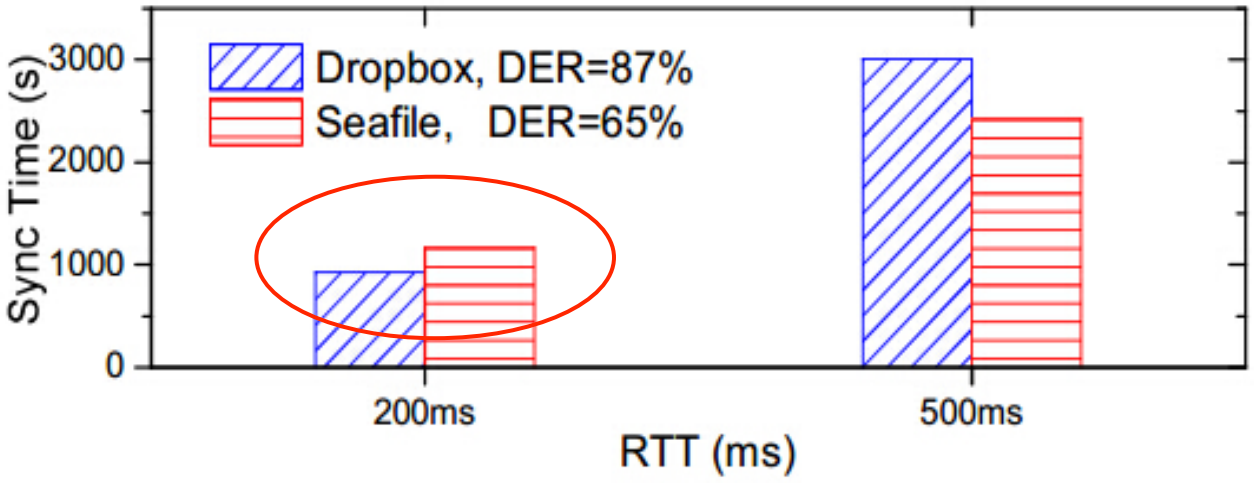
Impact of Missing Capabilities

- Bandwidth inefficiency
  - Sync is not efficient for large # of small files in high RTT conditions because the client waits for an app-level ACK before sending next chunk
  - Bundling is quite important!



Impact of Misusing Capabilities

- Deduplication is NOT always efficient
  - More effective dedup does not work well in good network conditions because of its high computation overhead
  - Network-aware dedup may be important

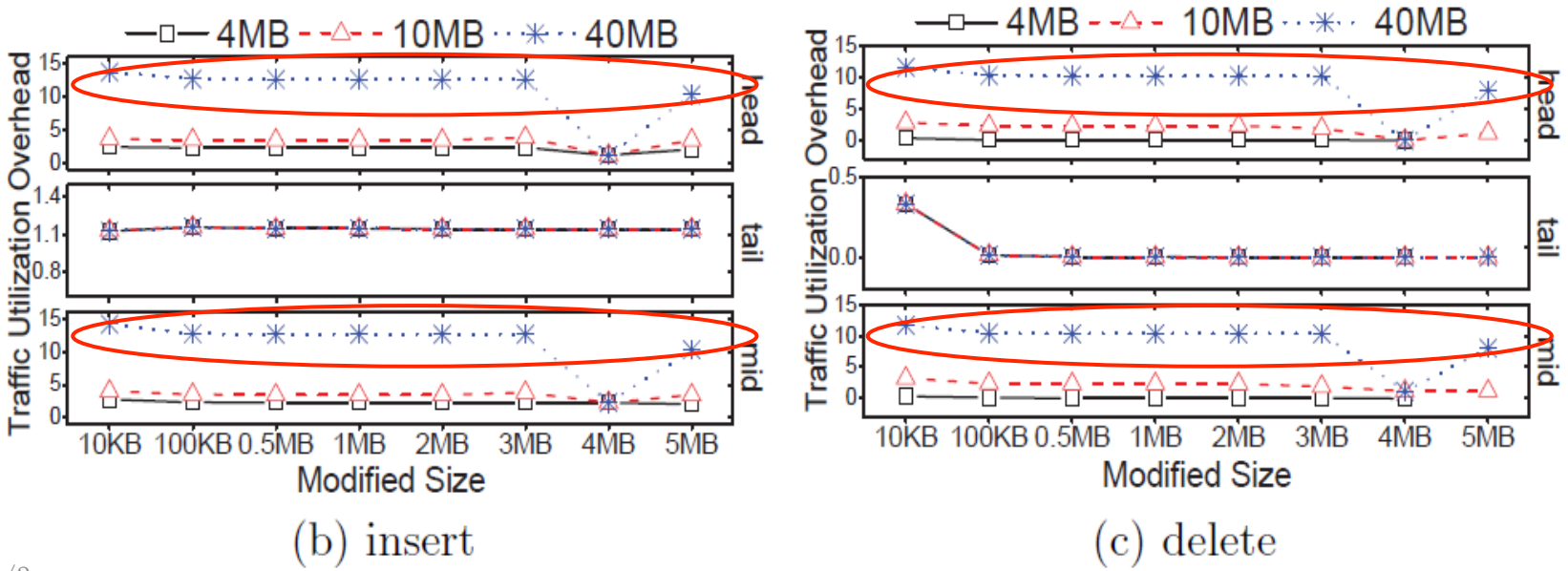


DER: the ratio of the deduplicated file size to the original file size

Impact of Misusing Capabilities

- Delta-encoding fails with fixed-size chunking
  - 3 basic file operations (flip bits, insert, delete)
  - Changing 2MB of a 10MB file leads to more than 6MB sync traffic

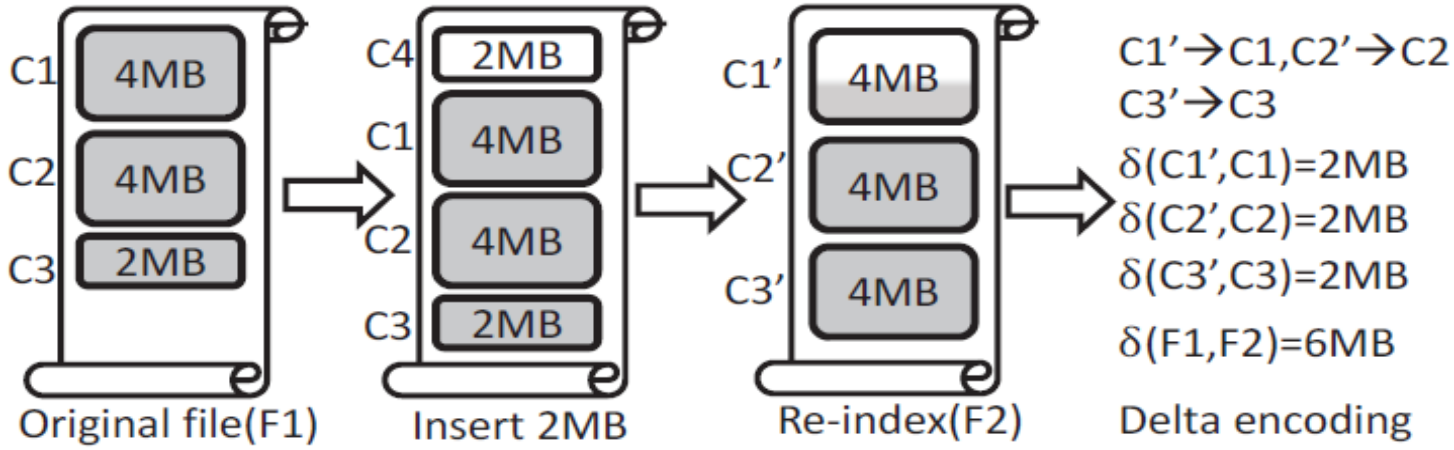
TUO: Traffic data / modified data



(b) insert (c) delete

Impact of Misusing Capabilities

- Why the delta-encoding fails?
  - A large file is split into multiple chunks
  - Delta-encoding is performed between chunks
  - But modifications will move cut points!



Measurement Conclusion

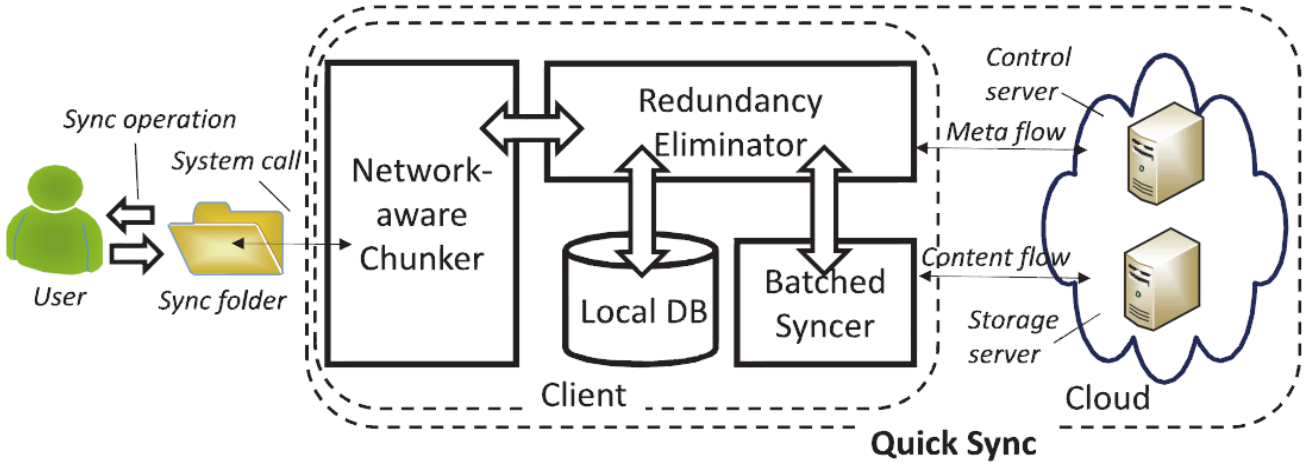
- Missing or Misusing these key capabilities leads to the sync inefficiency problem
- Challenges of improving sync efficiency
  - Are these capabilities enough?
  - Should we combine these storage techniques with network parameters (e.g. delay, loss and etc.)?
  - And how?

Outline

- Background
- Problem Statement
  - Service Usability
  - Protocol Capabilities
- Our Exploration on Protocol Capabilities
- Summary

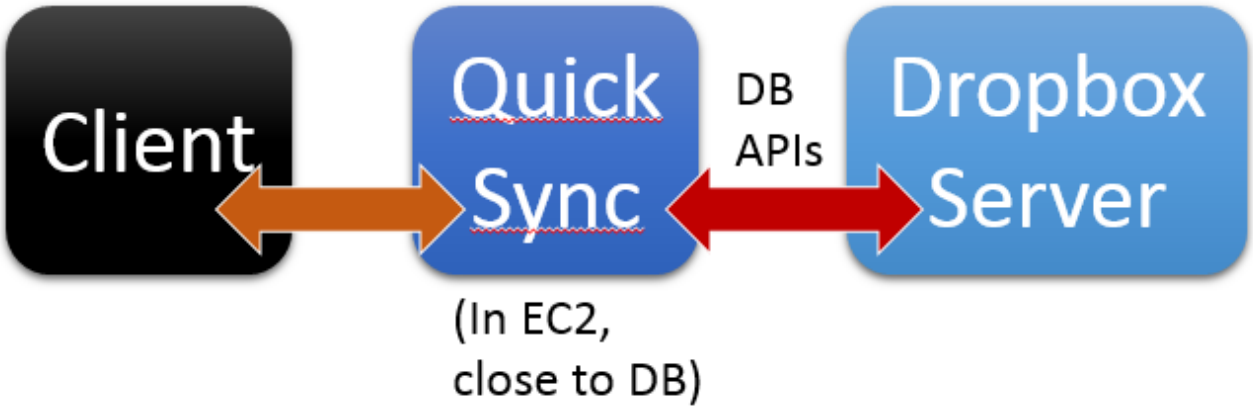
Exploration on Capabilities

- QuickSync [MobiCom15] with 3 techniques
  - Propose network-aware content-defined chunker to identify redundant data
  - Design improved incremental sync approach that correctly performs delta-encoding between similar chunks to reduce sync traffic
  - Delay-batched ACK to improve sync throughput



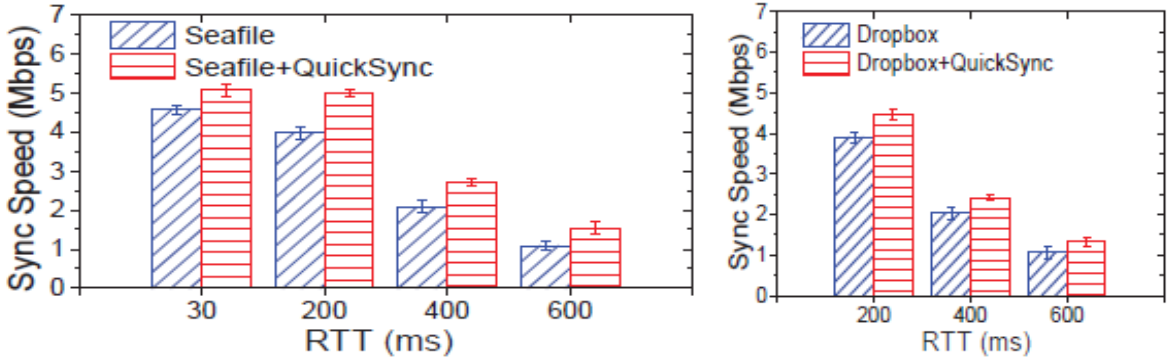
QuickSync Implementation

- Implementation over Dropbox
  - Unable to directly modify Dropbox, so we design a proxy-based architecture built on Amazon EC2
- Implementation over Seafile
  - The proxy-based architecture adds overhead
  - Full implementation with Seafile (open source)



Impact of Network-aware Chunker

- Network-aware Chunker
  - Larger chunks in good network conditions, make aggressive chunking in slow networks
- Performance results
  - 200GB backup; up to 31% speed improvement
  - Network-aware chunker works well



(a) Seafile (b) Dropbox

Integrated System Performance

- Setup
  - Practical sync workloads on Windows / Android
- Performance results (Win / Android)
  - Traffic size reduction: up to 80.3% / 63.4%
  - Sync time reduction: up to 51.8% / 52.9%

Workload (Platform)	# of Events (C/M/D)	Traffic Size (Origin/Ours/Reduction%)	Sync Time (Origin/Ours/Reduction%)
QuickSync Paper (W)	74/0/0	4.67MB/4.32MB/7.4%	27.6s/17.9s/35.1%
Seafile Source (W)	1259/0/0	15.6MB/14.2MB/8.9%	264.1s/127.3s/51.8%
Document Editing (W)	12/74/7	64.3MB/12.7MB/80.3%	592.0s/317.3s/46.4%
Data Backup (W)	37655/0/0	2GB/1.4GB/30.6%	68.7m/43.1m/37.4%
Document Editing (A)	1/4/0	4.1MB/1.5MB/63.4%	24.4s/14.3s/41.4%
Photo Sharing (A)	11/0/0	21.1MB/20.7MB/1.9%	71.9s/54.6s/24.1%
System Backup (A)	66/0/0	206.2MB/117.9MB/42.8%	612.3s/288.7s/52.9%

Outline

- Background
- Problem Statement
  - Service Usability
  - Protocol Capabilities
- Our Exploration on Protocol Capabilities
- Future work

Related Work

- WebDAV [RFC 4918], Git
  - These efforts focus on authoring and versioning
  - Can not well support large files
- Rsync
  - Delta-encoding algorithm only works well in file granularity
- Different from ISS
  - ISS focuses on the sync operation
  - Other important capabilities are closely related and required (e.g. chunking, deduplication)

Future Work

- Goal: usability & capabilities
  - Easier to use multiple storage sync services
  - Easier to build a private sync service
  - Achieve interoperability
  - Reasonably configure capabilities
- Possible solution: standard sync protocol
  - Standardize the sync process and capabilities
  - Want to apply IETF Transport and Security expertise

References

- Problem Statement: http://datatracker.ietf.org/doc/draft-cui-iss-problem/
- Wiki: https://github.com/iss-ietf/iss/wiki/Internet-Storage-Sync
- QuickSync [MobiCom2015]: http://www.4over6.edu.cn/cuiyong/cindex.html
- A First Look at Mobile Cloud Storage Services [IEEE Network Magazine]: http://www.4over6.edu.cn/cuiyong/cindex.html