Towards Web-based Delta Synchronization for Cloud Storage Services

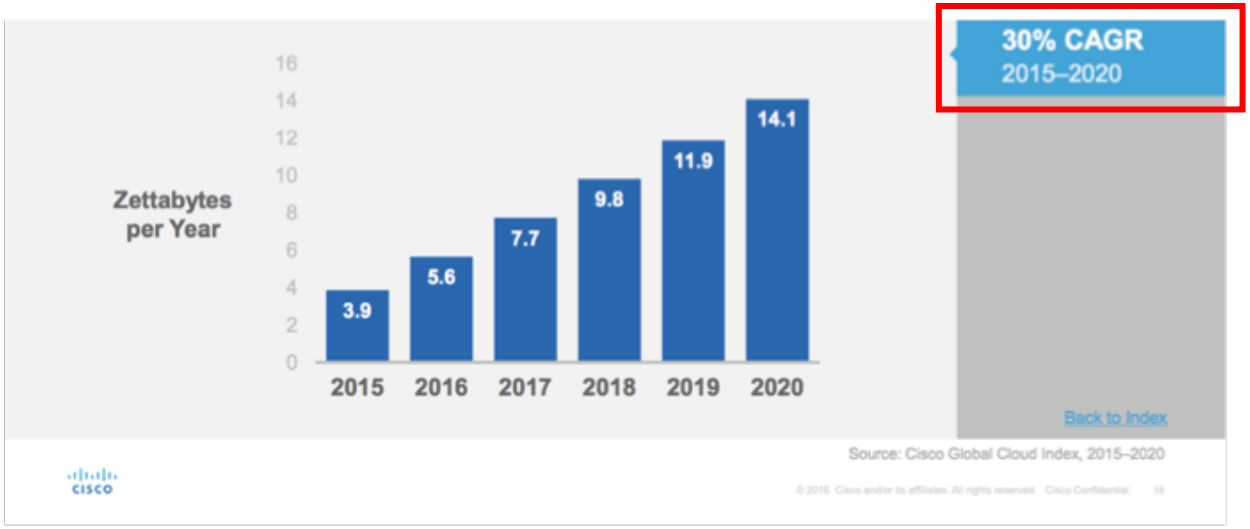
He Xiao and Zhenhua Li, **Tsinghua University**; Ennan Zhai, Yale University;
Tianyin Xu, UIUC; Yang Li and Yunhao Liu, Tsinghua University;
Quanlu Zhang, Microsoft Research Asia; Yao Liu, SUNY Binghamton

xiaoh16@gmail.com

Fast'18 Feb 14, 2018

Network Traffic is Overwhelming in Cloud Storage

Cloud Traffic has 30% CAGR (**C**ompound **A**verage **G**rowth **R**ate)



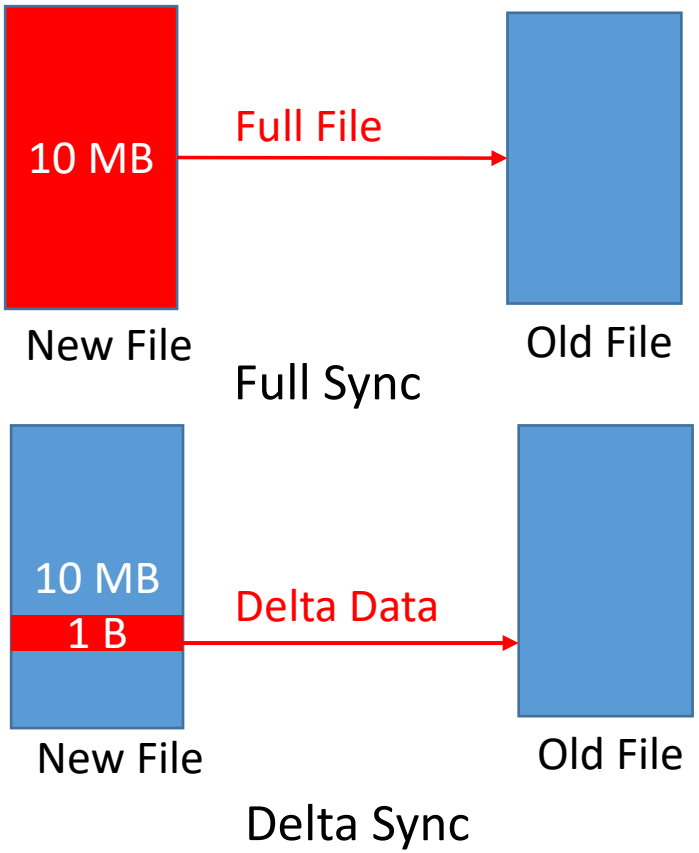
Client

File Synchronization(Sync)

Network Traffic

Cloud Sever

Delta Sync Improves Network Efficiency



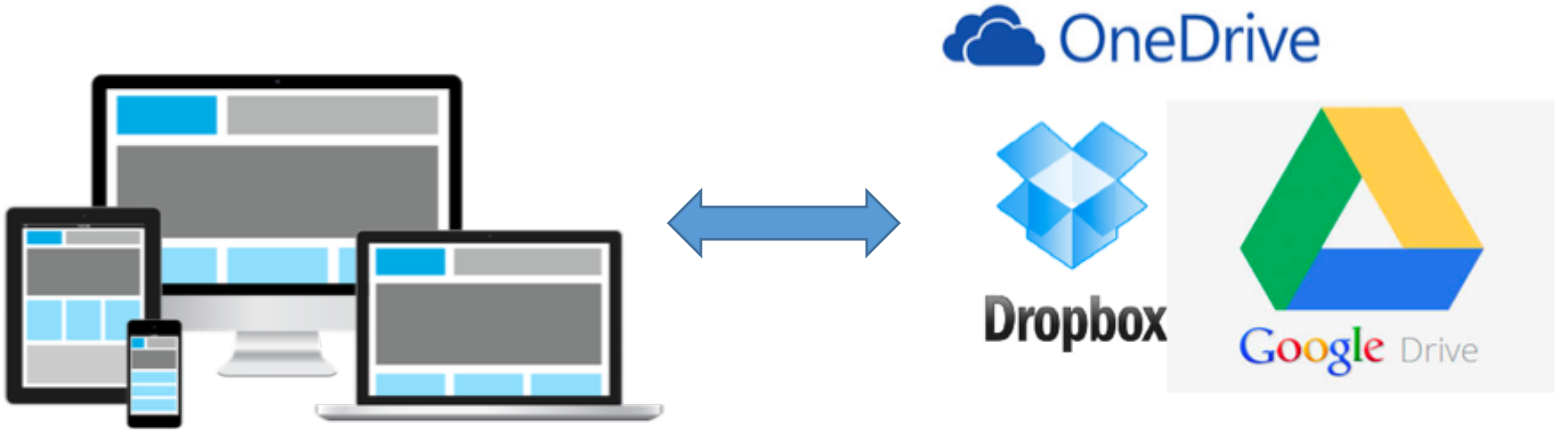
Delta sync support in nine state-of-the-art cloud storage services

Service	PC Client	Mobile App	Web Browser
Dropbox	Yes	No	No
Google Drive	No	No	No
OneDrive	No	No	No
iCloud Drive	Yes	No	No
Box.com	No	No	No
SugarSync	Yes	No	No
Seafile	Yes	No	No
QuickSync	Yes	Yes	No
DeltaCFS	Yes	Yes	No

Delta Sync is crucial for reducing cloud storage network traffic.

No Web-based Delta Sync

Web-based delta sync is essential for cloud storage web clients and web apps

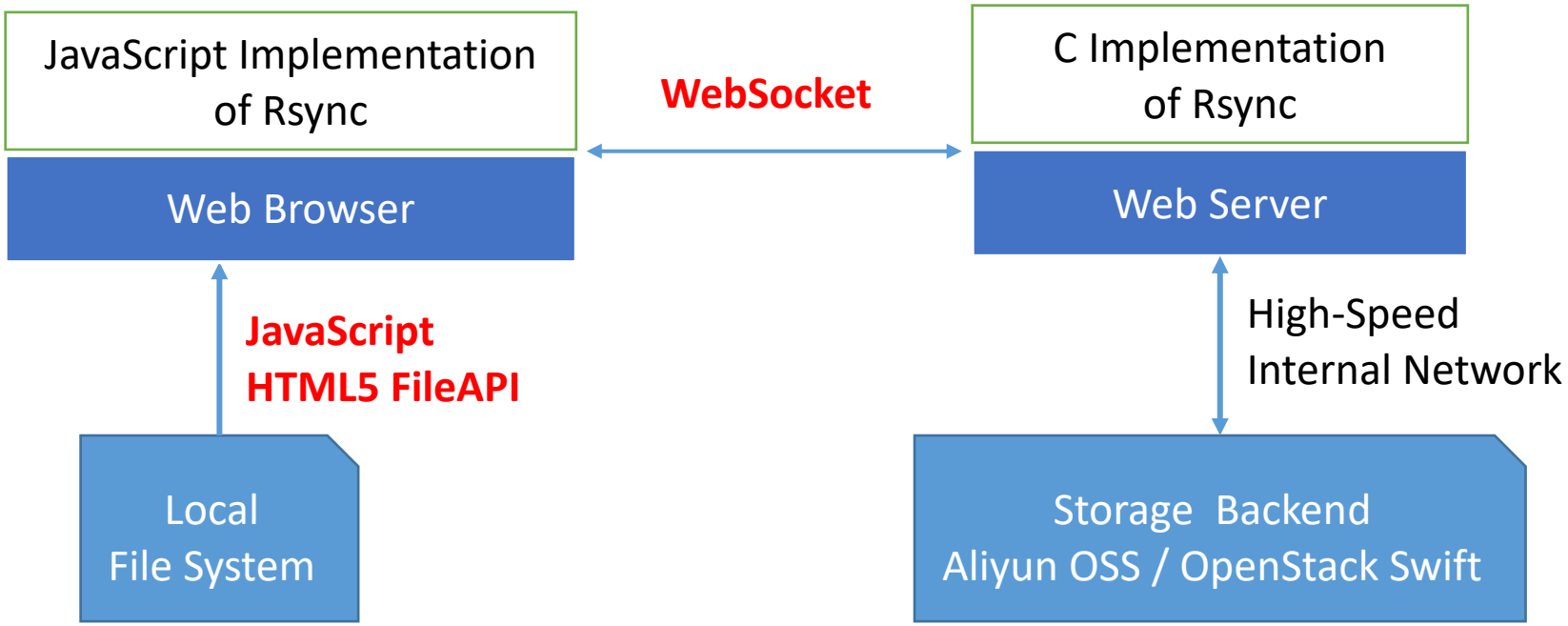


Web is the most pervasive and OS-independent cloud storage access method

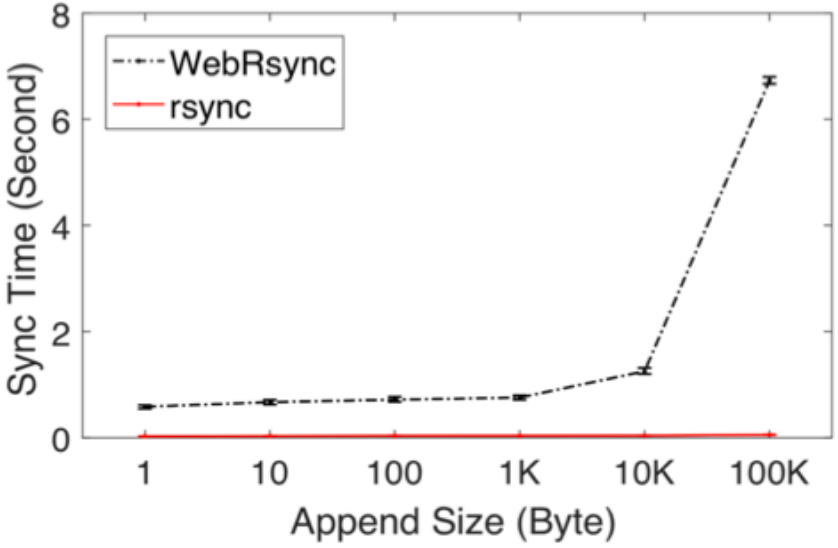
Why web-based delta sync is not supported by today's commercial cloud storage services ?

WebRsync: First Workable Web Delta Sync

- Implement rsync on web framework with pure web tech:
  JavaScript + HTML5 + WebSocket
- Points out the Challenges of supporting delta sync on web.

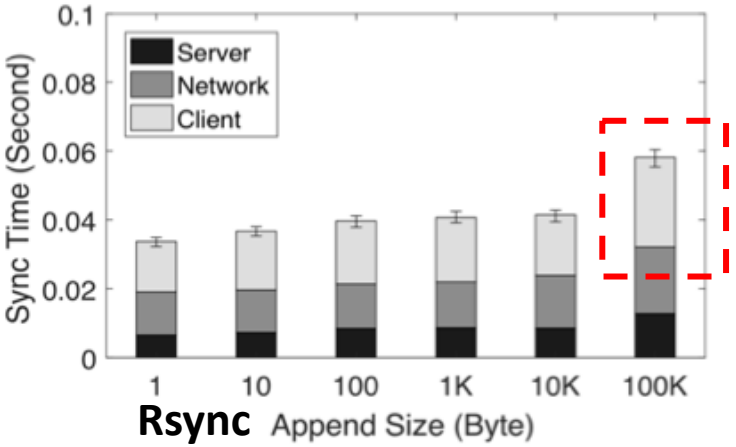


WebRsync benchmarking: poor client performance

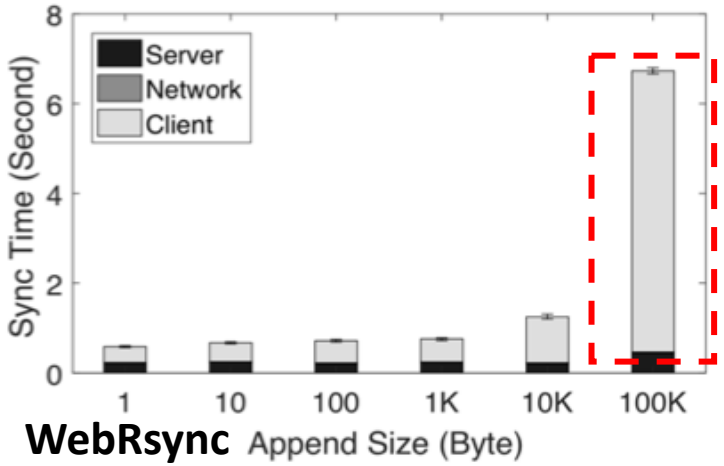


Sync time of WebRsync vs Linux rsync

14–25 times slower



~40%



60-92%

StagMeter Tool

Timing tasks: Printing timestamps every 100ms:

x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x x

Stagnation: single-thread is occupied by some backend tasks

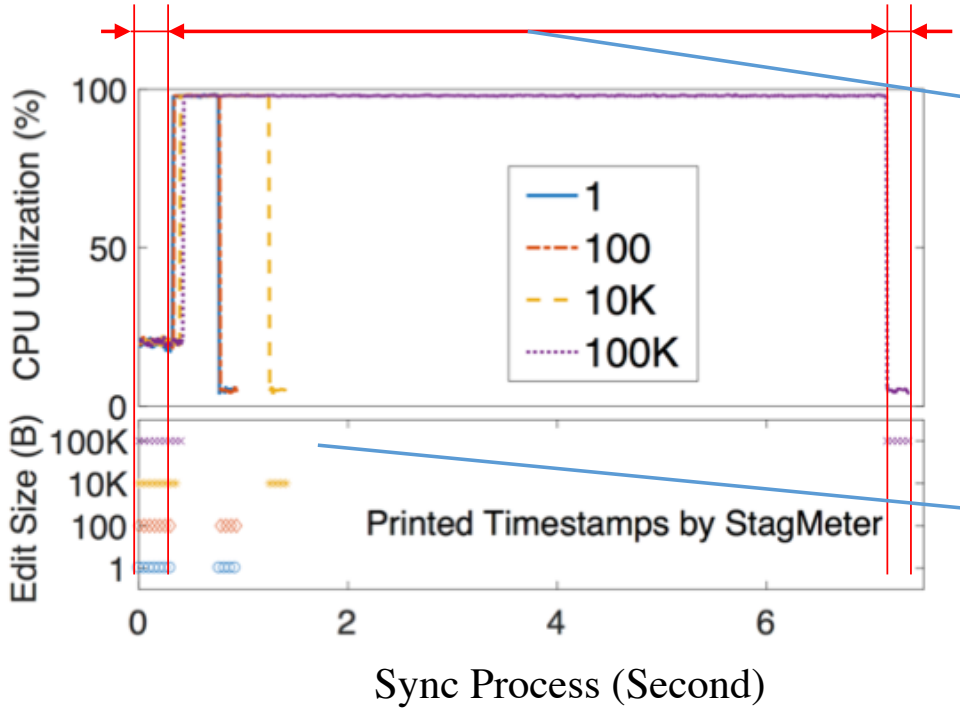
User’s operation cannot get response timely.

Measuring Stagnation with StagMeter

1. Send meta data
   Wait server

2. Checksum Search and Comparison

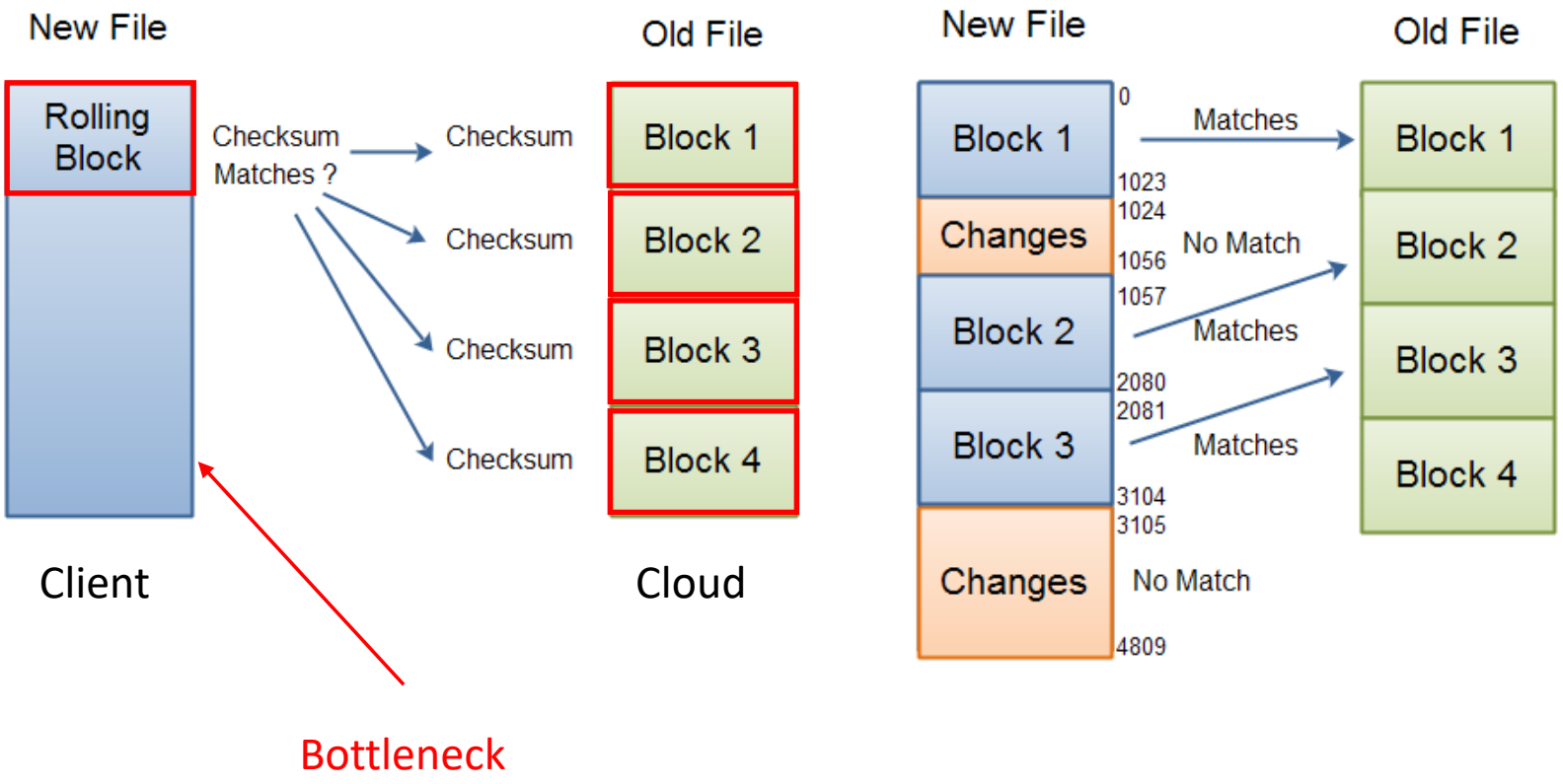
3. Send tokens and literal bytes



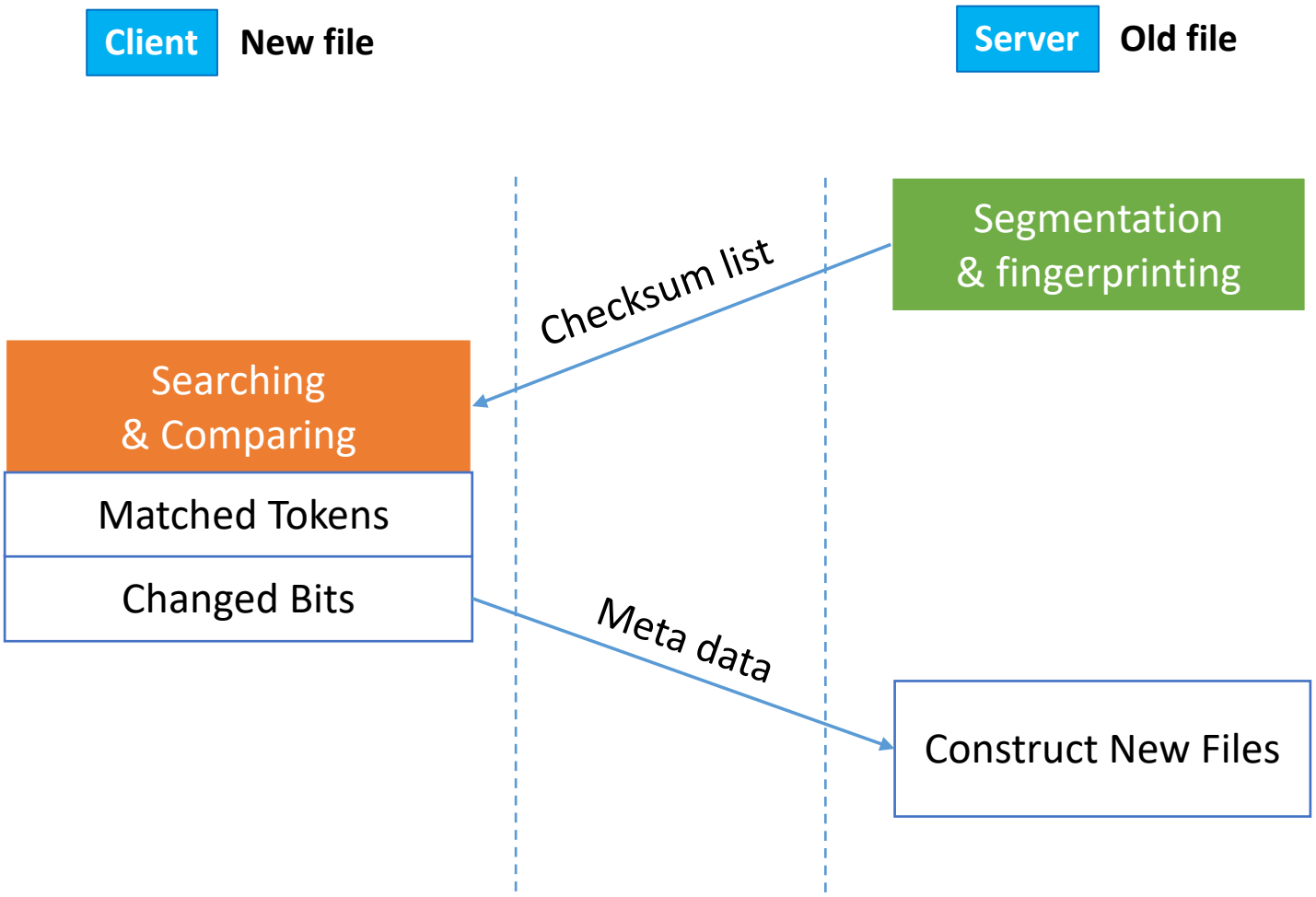
High CPU Utilization when computing

Timestamp Printing is suspended
Web is under stagnation state

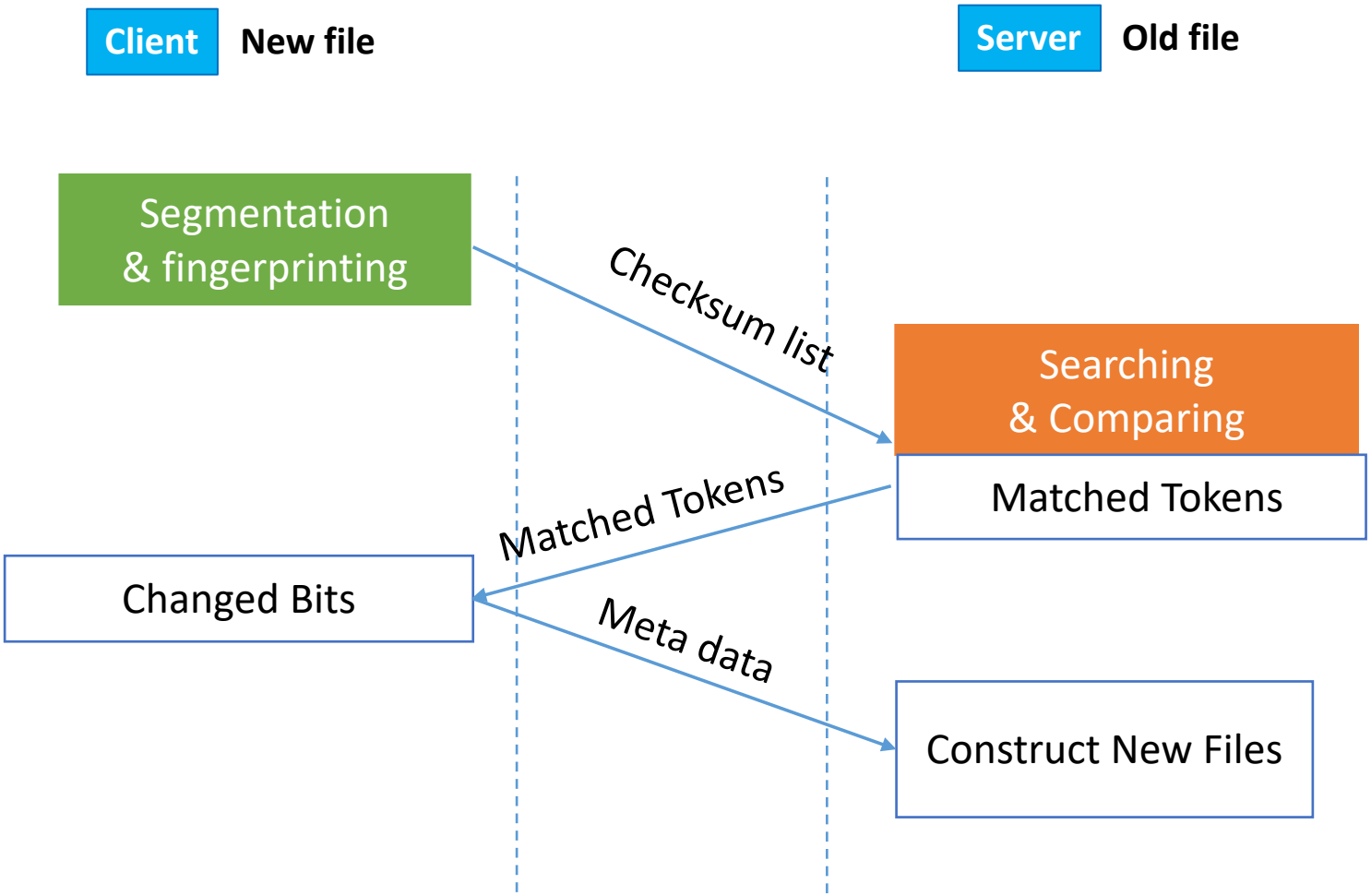
Why poor client : slow searching and comparing



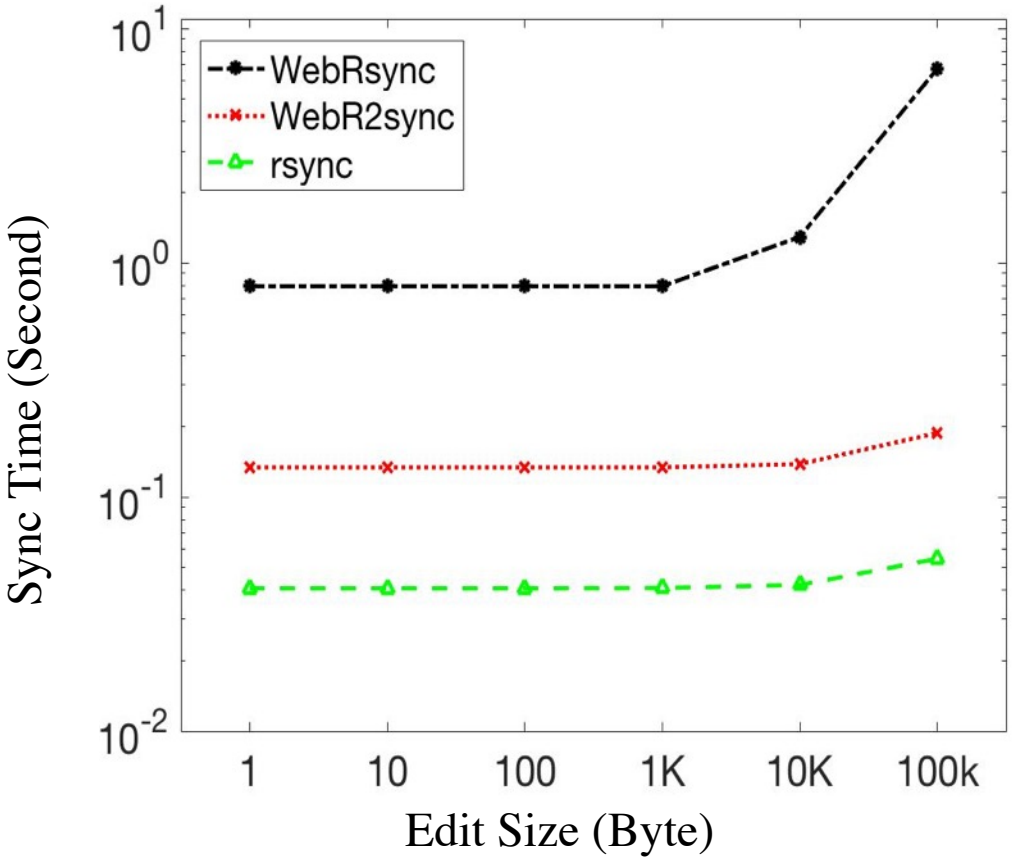
WebR2sync: Reverse Computation Process



WebR2sync: Reverse Computation Process



Performance of WebR2sync

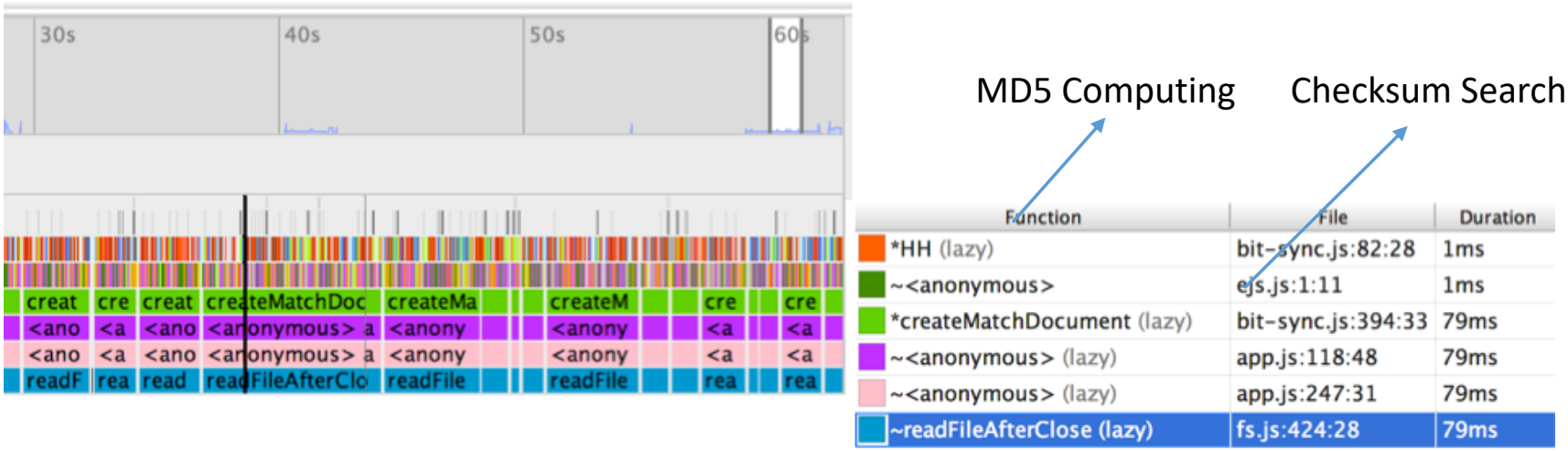


Sever side is 2-3 time slower

Issue: Server takes severely heavy overhead.

Server-side Overhead Profiling

Checksum searching and **block comparison** occupy **80%** of the computing time



- Use faster hash functions to replace MD5
- Reduce checksum searching overhead

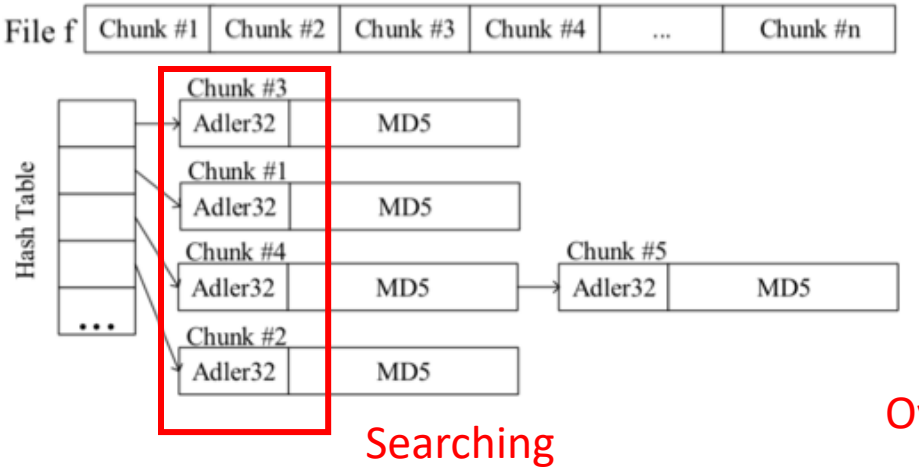
Replacing MD5 with SipHash in Chunk Comparison

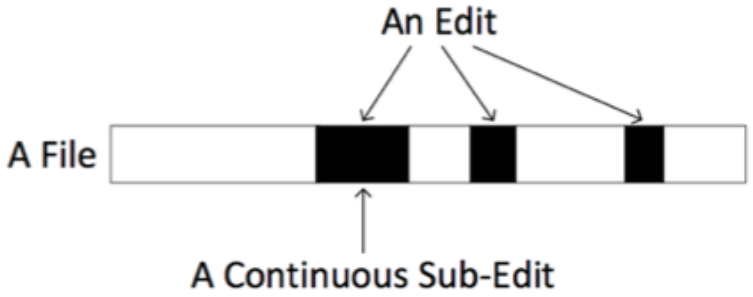
A comparison of pseudorandom hash functions

Hash Function	Collision Probability	Cycles Per Byte
MD5	Low ($< 10^{-6}$)	5.58
Murmur3	High ($\approx 1.05 \times 10^{-4}$)	0.33
CityHash	High ($\approx 1.03 \times 10^{-4}$)	0.23
FNV	High ($\approx 1.09 \times 10^{-4}$)	1.75
Spooky	High ($\approx 9.92 \times 10^{-5}$)	0.14
SipHash	Low ($< 10^{-6}$)	1.13

SipHash remain low Collision Probability at much faster speed

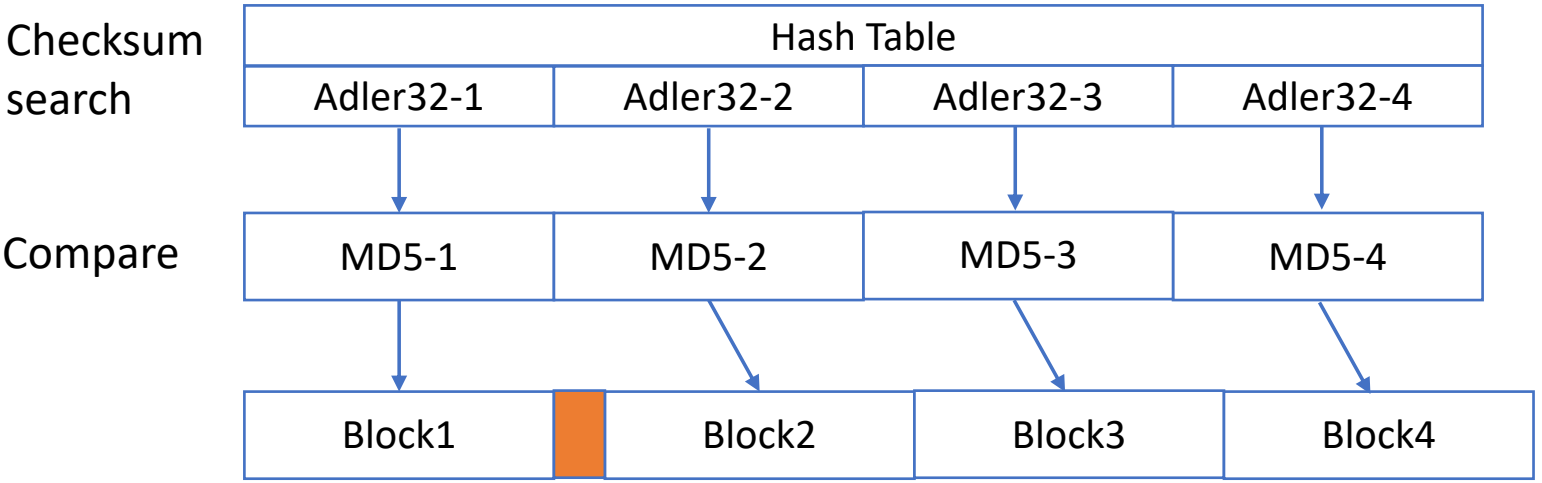
Reduce Checksum Searching by Exploiting Locality of File Edits.



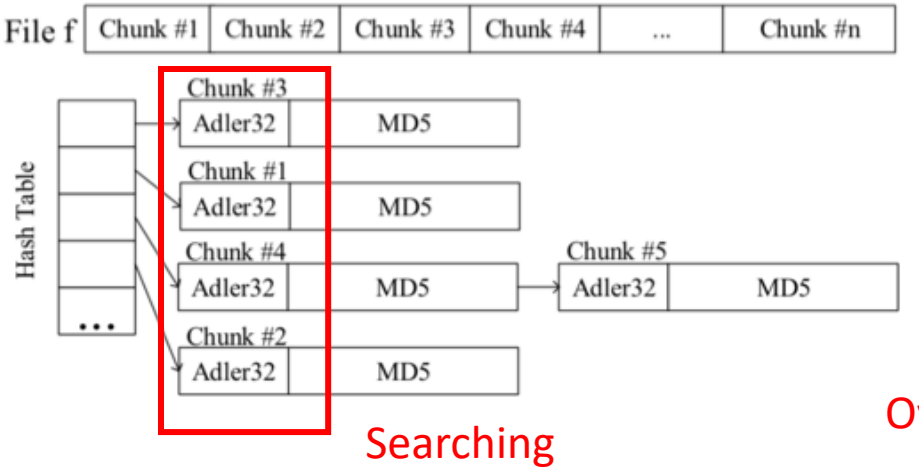


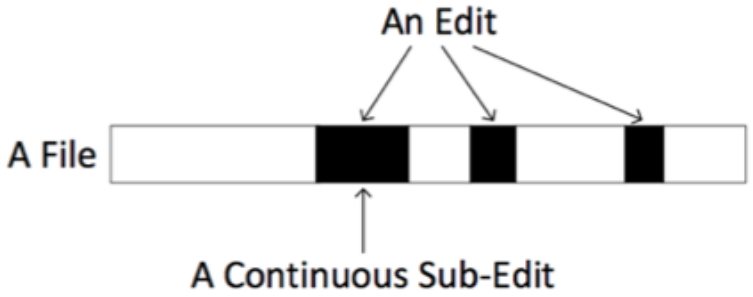
(a) An edit consists of several continuous sub-edits.

Over 95% modified files have less than 10 edits.



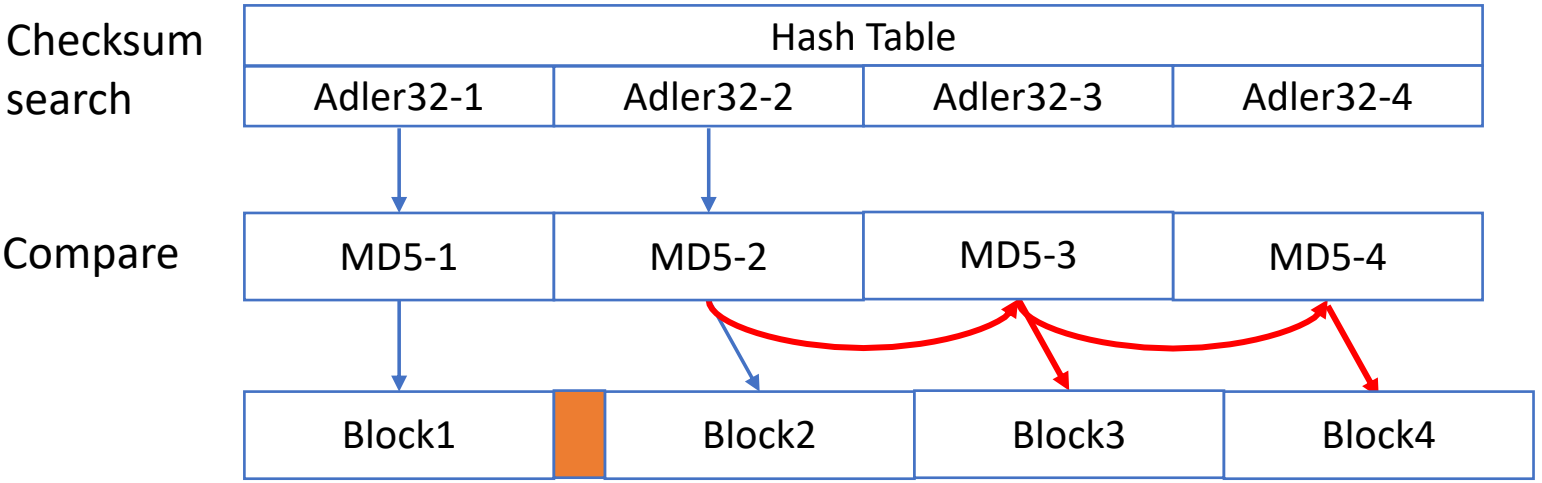
Reduce Checksum Searching by Exploiting Locality of File Edits.





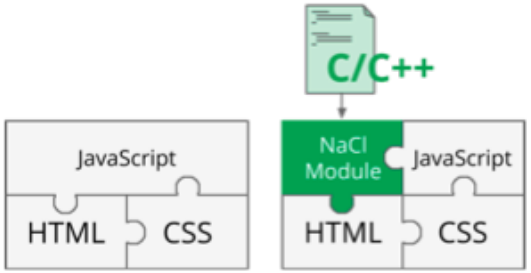
(a) An edit consists of several continuous sub-edits.

Over 95% modified files have less than 10 edits.

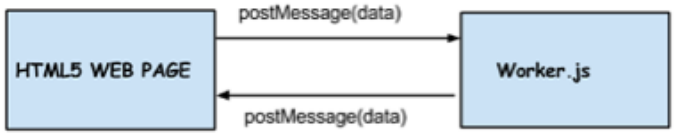


A Series of attempts of other techs:
Native Extension, Parallelism

- Native Extension: leverage the native client for web browsers. -> as quick as native rsync , supported platforms limited (e.g. Mobile web)



- WebRsync-Parallel: using HTML5 *web workers to avoid stagnations.* *-> avoid stagnation but not on sync time*

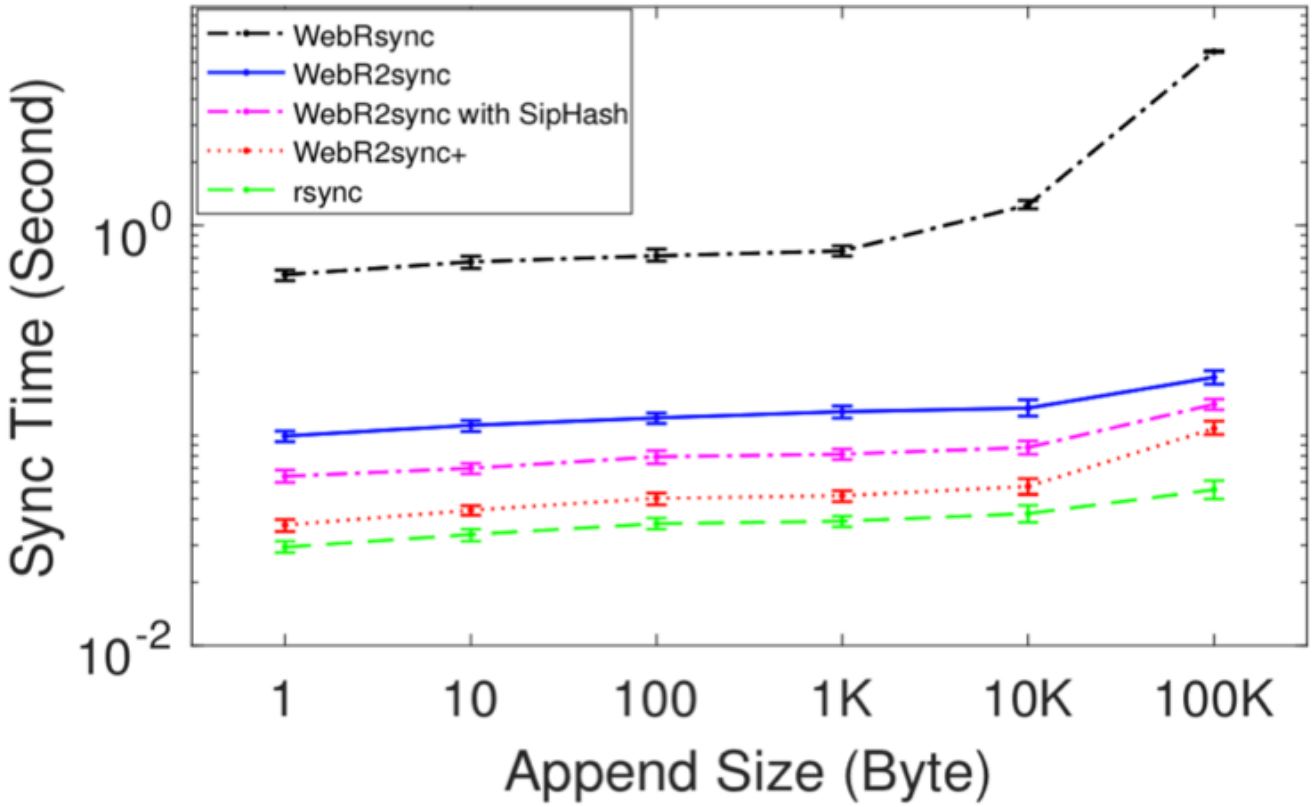


- The drawback of WebRsync cannot be fundamentally addressed through above optimizations

Evaluation Setup

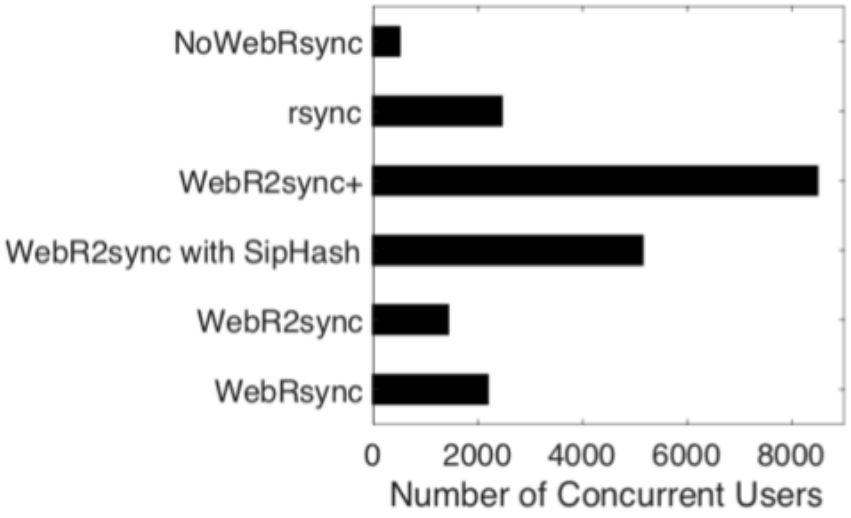
Basic experiment setup visualized in a map of China

Sync Time

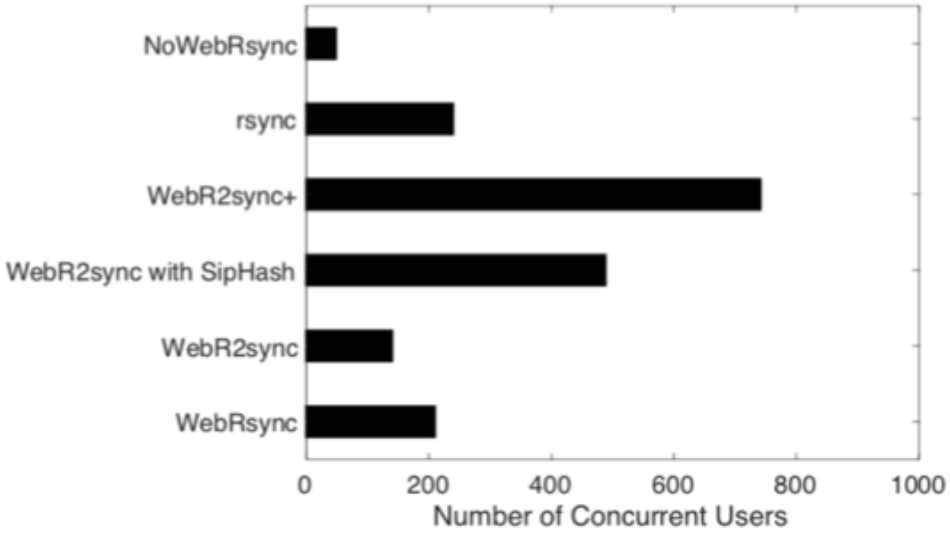


WebR2sync+ is 2-3 times faster than WebR2sync and 15-20 times faster than WebRsync

Throughput



Regular Workload



Intensive Workload

This throughput is as **4** times as that of WebR2sync/`rsync` and as **9** times as that of NoWebRsync.

Conclusion

- Implement a workable web-based delta sync named WebRsync using JavaScript and Html5, then **quantifying** the stagnation on browser by StagMeter.

- WebR2sync: **Reverse** the rsync process by moving computation-intensive operations from client with JavaScript to server side with efficient native C code.

- WebR2sync+: By exploiting the edit **locality** and trading off **hash** algorithms, we make the computation overhead affordable at the server side.

Future Work

- A **seamless** way to integrate the server-side design of WebR2sync+ with the back-end of commercial cloud storage vendors (like Dropbox and iCloud Drive).

- Explore the benefits of using more fine-grained and complex delta sync protocols, such as **CDC** and its variants.

- We envision to expand the usage of WebR2sync+ for a **broader range** of web service scenarios.

Q&A

Thanks!