Network Coding Distributed Storage

Patrick P. C. Lee

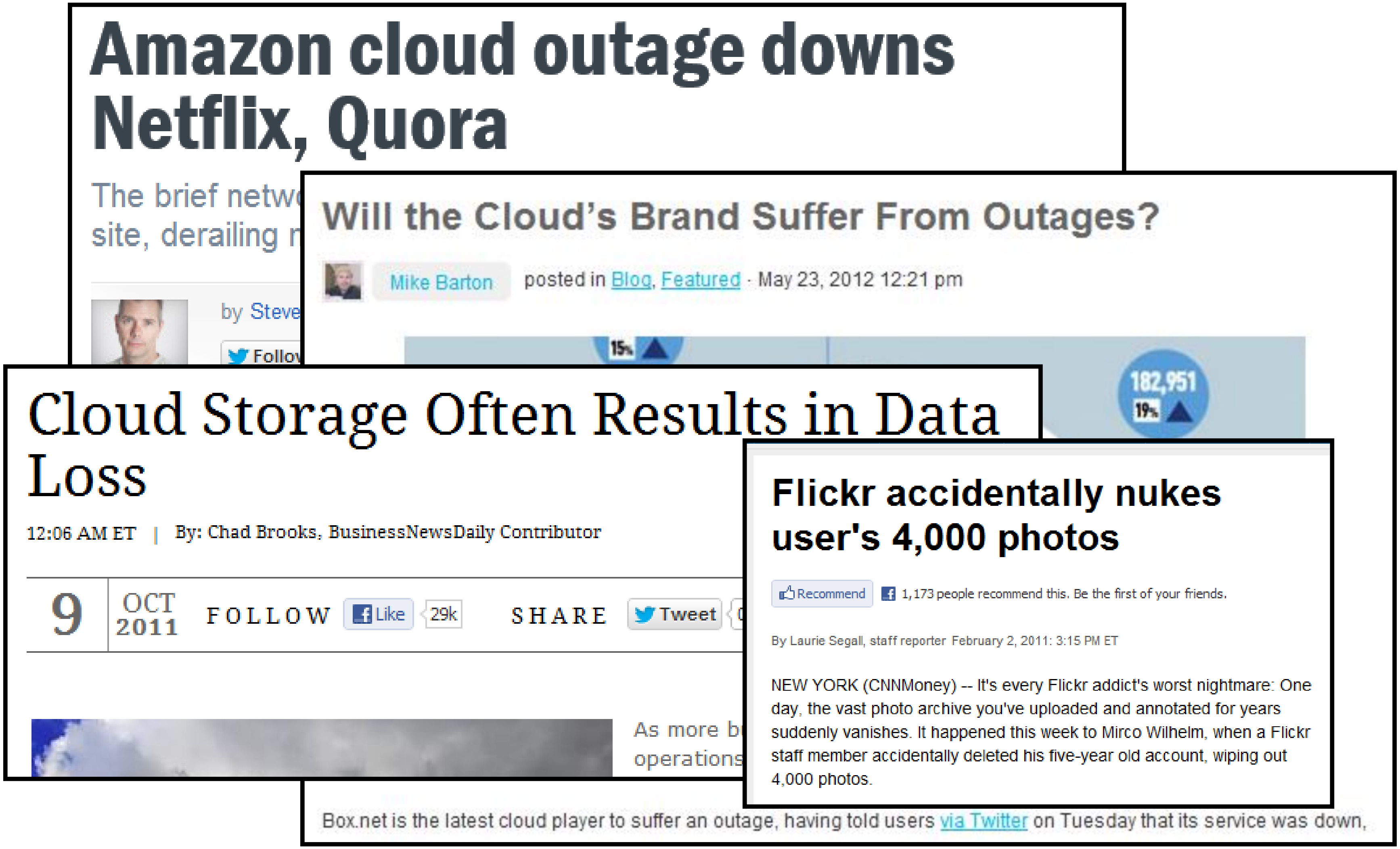
Department of Computer Science and Engineering
The Chinese University of Hong Kong

Cloud Storage

- Cloud storage is an emerging service model for remote backup and data synchronization

- *Is cloud storage fully reliable?*

Problems in the Cloud



Amazon cloud outage downs Netflix, Quora

The brief netw
site, derailing r

by Steve

Follow

Will the Cloud's Brand Suffer From Outages?

Mike Barton posted in Blog, Featured · May 23, 2012 12:21 pm

Box.net is the latest cloud player to suffer an outage, having told users via Twitter on Tuesday that its service was down,

Cloud Storage Often Results in Data Loss

12:06 AM ET | By: Chad Brooks, BusinessNewsDaily Contributor

9 OCT 2011 FOLLOW Like 29k SHARE Tweet

As more b
operations

Flickr accidentally nukes user's 4,000 photos

Recommend 1,173 people recommend this. Be the first of your friends.

By Laurie Segall, staff reporter February 2, 2011: 3:15 PM ET

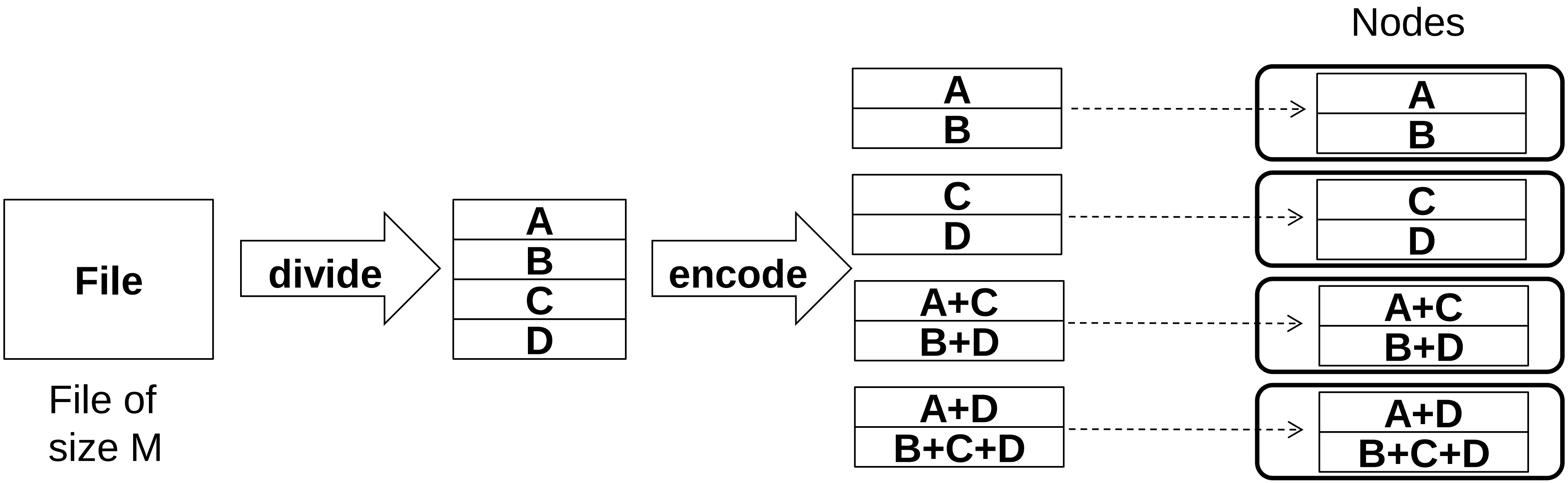
NEW YORK (CNNMoney) -- It's every Flickr addict's worst nightmare: One day, the vast photo archive you've uploaded and annotated for years suddenly vanishes. It happened this week to Mirco Wilhelm, when a Flickr staff member accidentally deleted his five-year old account, wiping out 4,000 photos.

Cloud Storage Requirements

- Data integrity protection
  - Detect any corrupted data chunks stored on cloud servers
- Fault tolerance
  - Tolerate any cloud server failures
- Efficient recovery
  - Recover any lost/corrupted data chunks with minimal overhead

(n, k) MDS codes

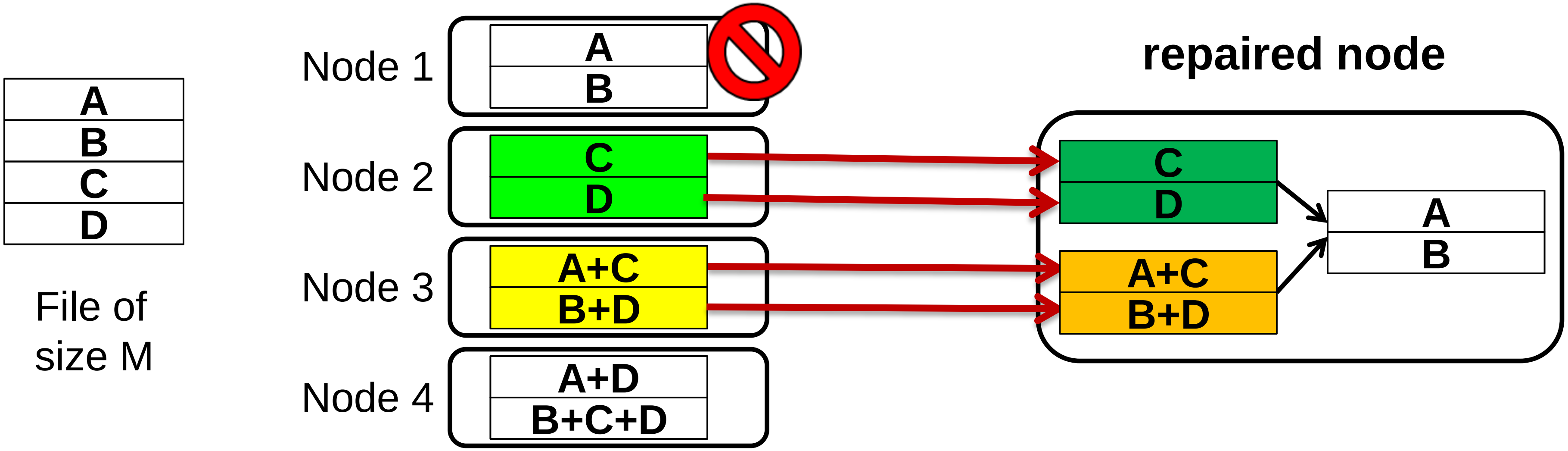
- Encode a file of size **M** into chunks
- Distribute encoded chunks into **n** nodes
- Each node stores **M/k** units of data
- **MDS property**: any k out of n nodes can recover file



n = 4, k = 2

Repairing a Failure

- **Conventional repair**: download data from any k nodes



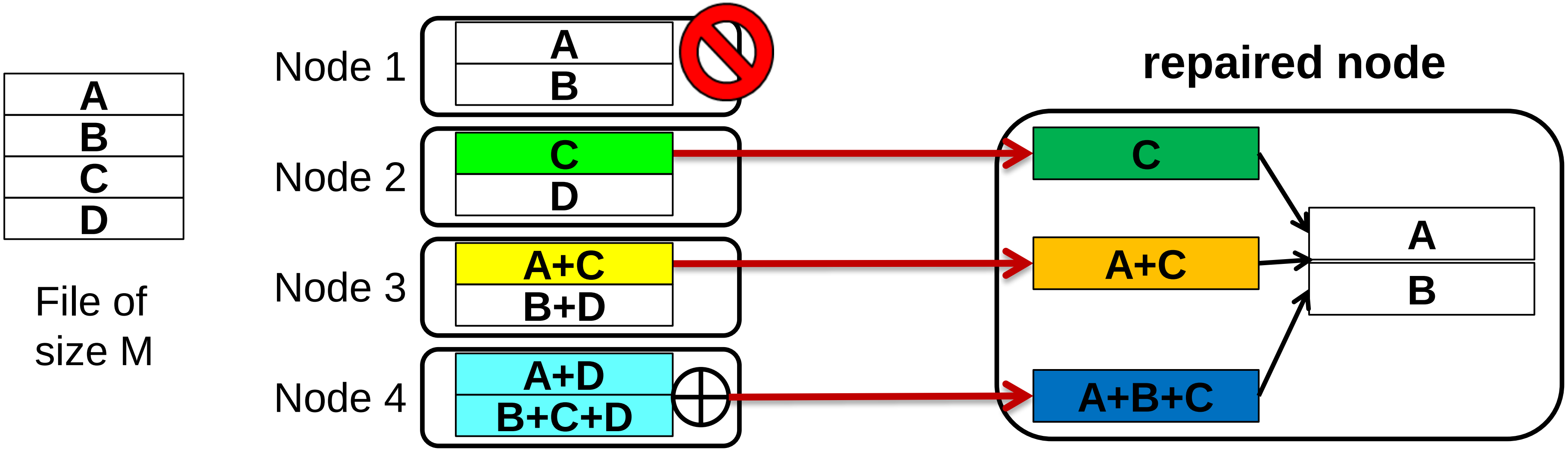
Repair Traffic = 🟩 + 🟨 = M

- *Q: Can we minimize repair traffic?*

Regenerating Codes

[Dimakis et al.; ToIT'10]

- Repair in regenerating codes:
  - Surviving nodes encode chunks (network coding)
  - Download one encoded chunk from each node



Repair Traffic = 🟩 + 🟨 + 🟦 = 0.75M

- *Minimizing repair traffic → minimizing system downtime*

Goals

- Challenges:
  - Mostly theoretical studies; limited empirical studies
  - Practical deployment remains unknown
- ***Goals**: Study practicality of network coding storage*
  - To realize network coding data storage in practical implementation
  - To conduct extensive experimental studies and evaluate the performance in a real storage environment
  - To provide insights into deploying network coding data storage in practice

Projects

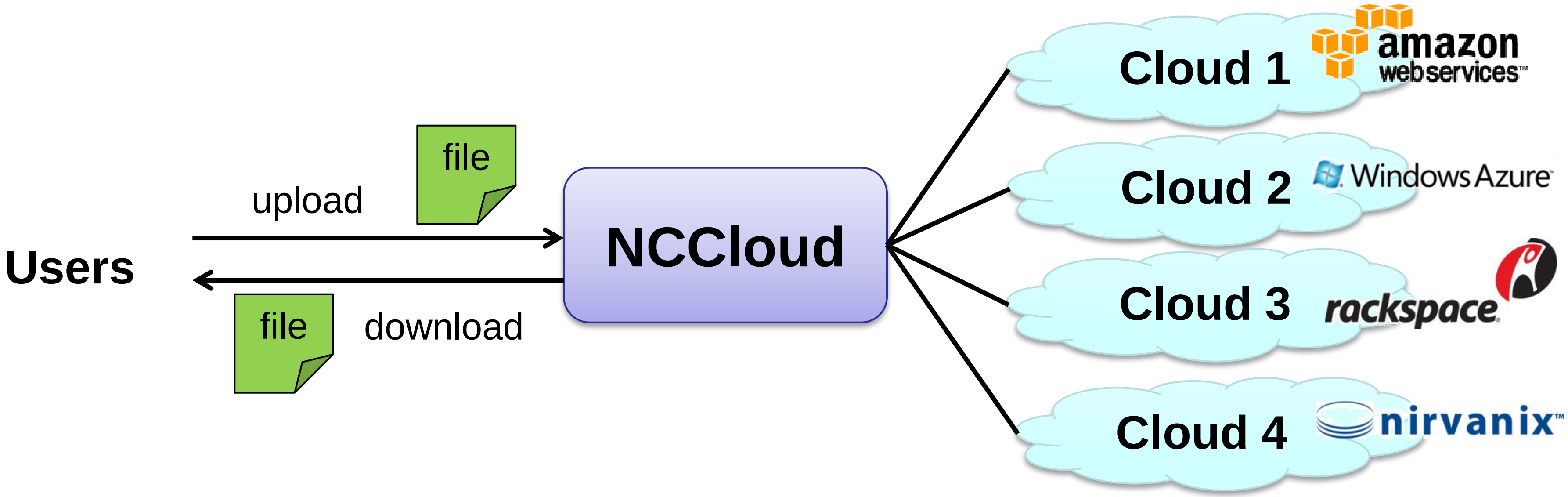
- **NCCloud** **[FAST’12, INFOCOM’13, TC]**
  - Network coding archival storage for public clouds
- **FMSR-DIP** **[SRDS’12, TPDS]**
  - Data integrity protection for network coding archival storage
- **CORE** **[MSST’13]**
  - Network coding primary storage for Hadoop file system
- **NCVFS**
  - Network coding video file system

NCCloud

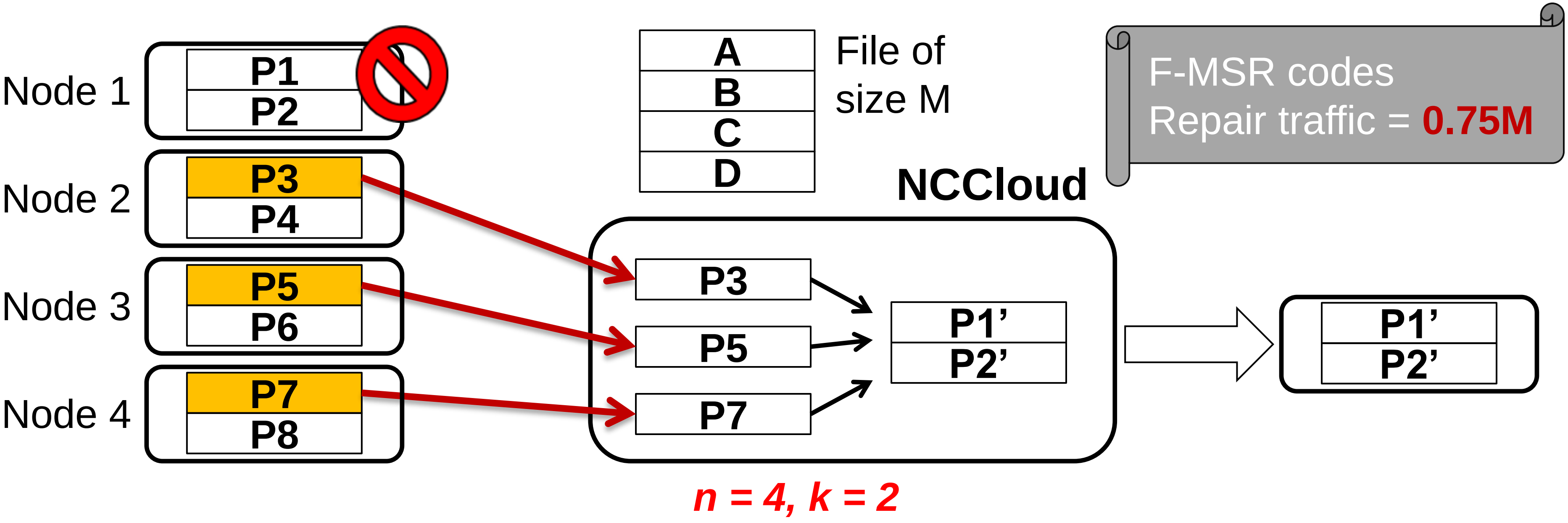
- **NCCloud** is a proxy-based storage system that applies regenerating codes in multiple-cloud storage
- Design properties:
  - Build on functional minimum-storage regenerating (FMSR) codes
  - Double-fault tolerance
  - Optimal storage efficiency
  - Minimum repair bandwidth for single-node failure recovery
    - Up to 50% saving compared to conventional repair
  - Uncoded repair
- Trick: non-systematic codes
  - Suited to **long-term archival storage** whose data is rarely read

NCCloud: Overview

- Multiple cloud storage:
  - Provide fault tolerance against cloud unavailability
  - Avoid vendor lock-ins



NCCloud: Key Idea

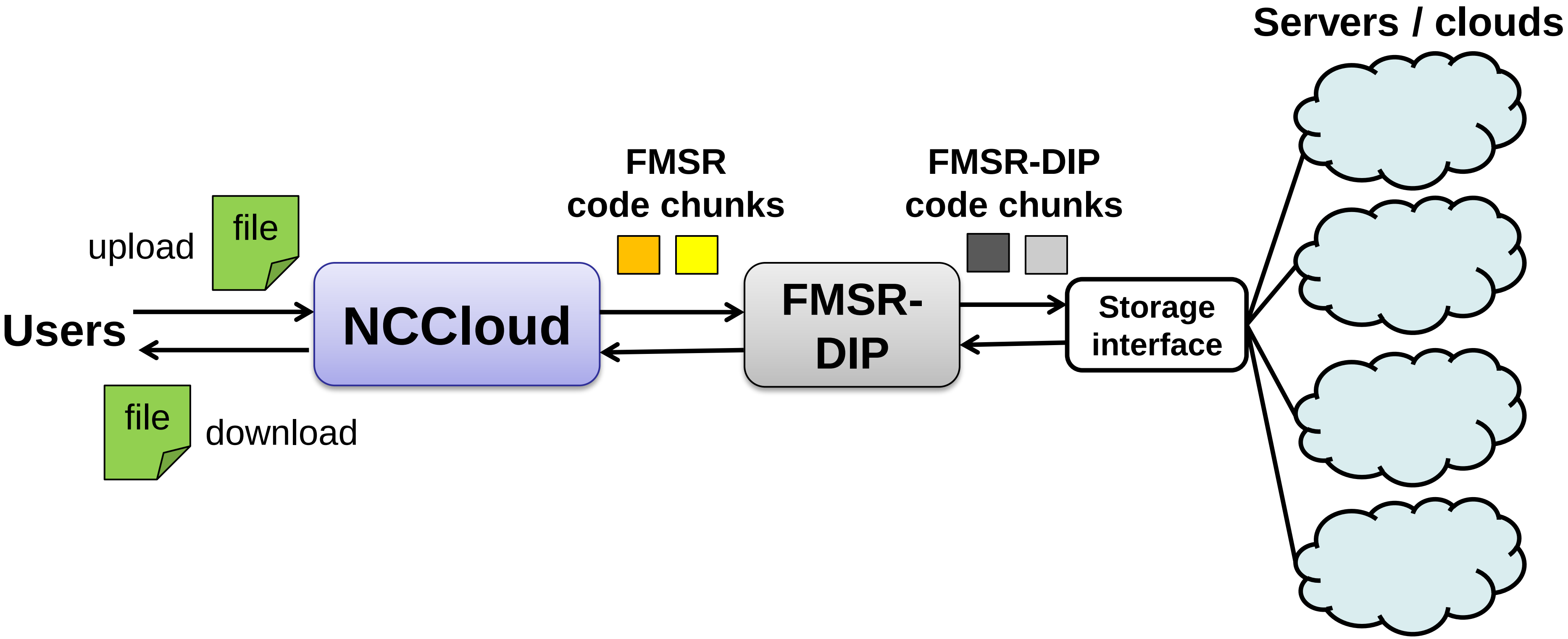


- Code chunk P_i = linear combination of original data chunks
- Repair:
  - Download one code chunk from each surviving node
  - Reconstruct new code chunks (via linear combination) in new node

FMSR-DIP

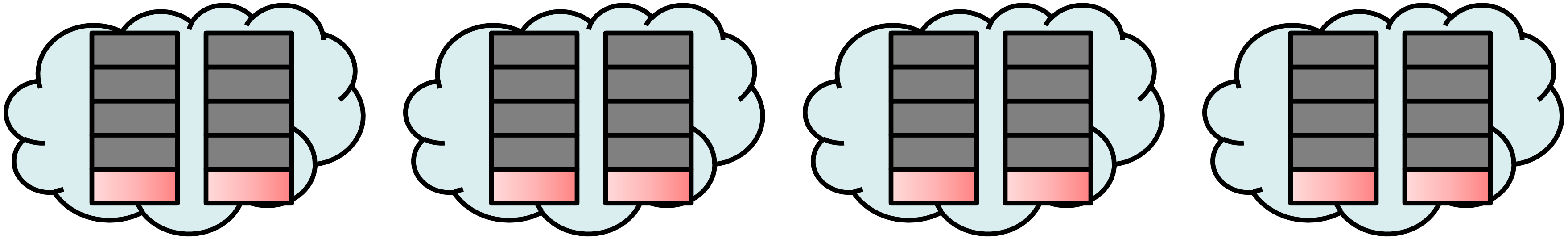
- **FMSR-DIP** enables data integrity protection, fault tolerance, and efficient recovery for NC storage
- Threat model: **Byzantine, mobile adversary** [Bowers et al. '09]
  - exhibits arbitrary behavior
  - corrupts different subsets of servers over time
- Design properties:
  - Preserve advantages of FMSR codes
  - Work on thin clouds (i.e., only basic PUT/GET assumed)
  - Support byte sampling to minimize cost

FMSR-DIP: Overview



Four operations: ***Upload, Check, Download and Repair***

FMSR-DIP: Key Idea

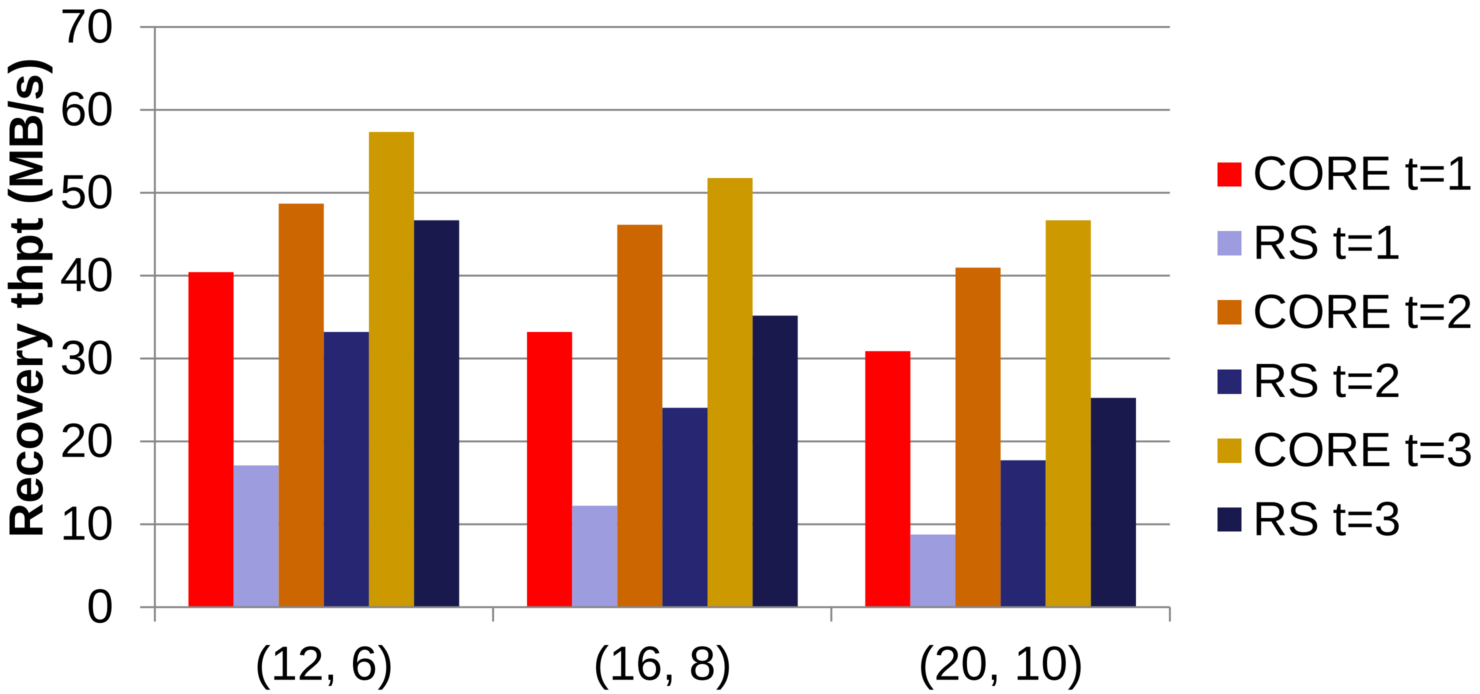


- Two-level protection:
  - Fault tolerance (horizontal) protection by FMSR codes
  - Integrity (vertical) protection by adversarial error correcting code
- Apply adversarial error-correcting codes to each FMSR code chunk
- Enable tunable parameters to trade between performance and security

CORE

- **CORE** augments existing optimal regenerating codes to support both single and **co**ncurrent failure **re**covery
  - Achieves minimum recovery bandwidth for concurrent failures in most cases
  - Retains existing optimal regenerating code constructions
- Implement CORE atop Hadoop HDFS
- Enable fault-tolerant, storage-efficient MapReduce

CORE: Performance



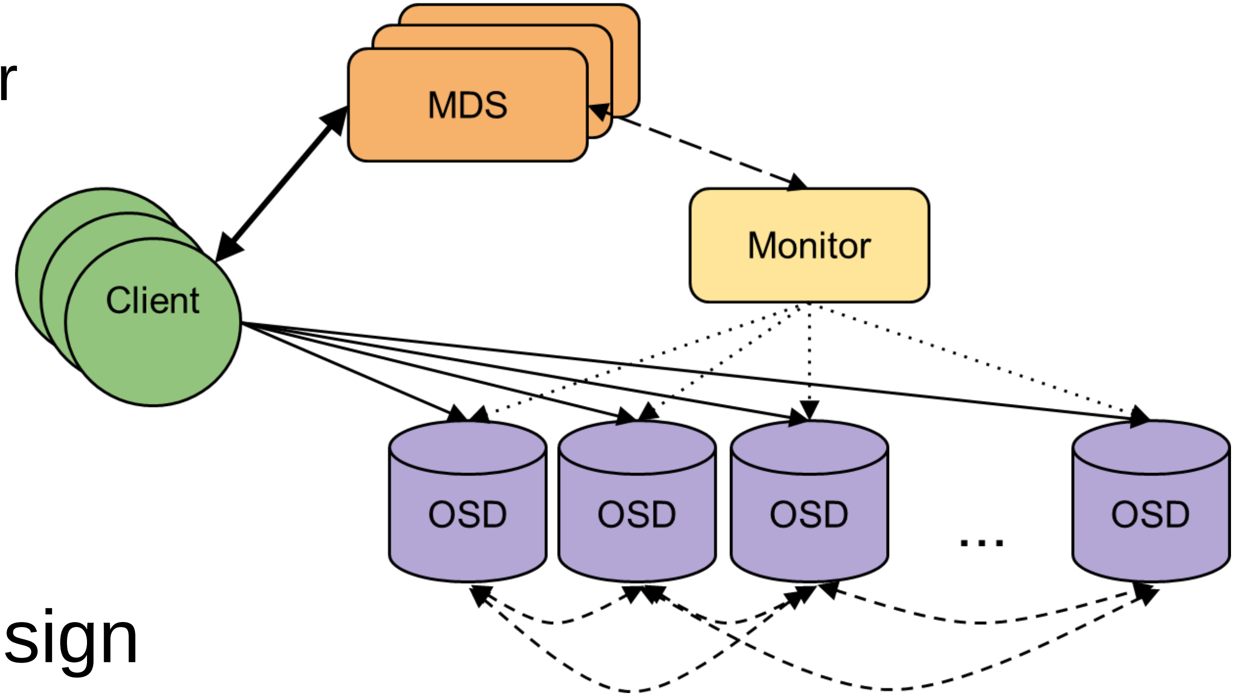
- CORE shows significantly higher throughput than Reed Solomon codes
  - e.g., in (20, 10), for single failure, the gain is **3.45x**; for two failures, it's **2.33x**; for three failures, is **1.75x**

NCVFS

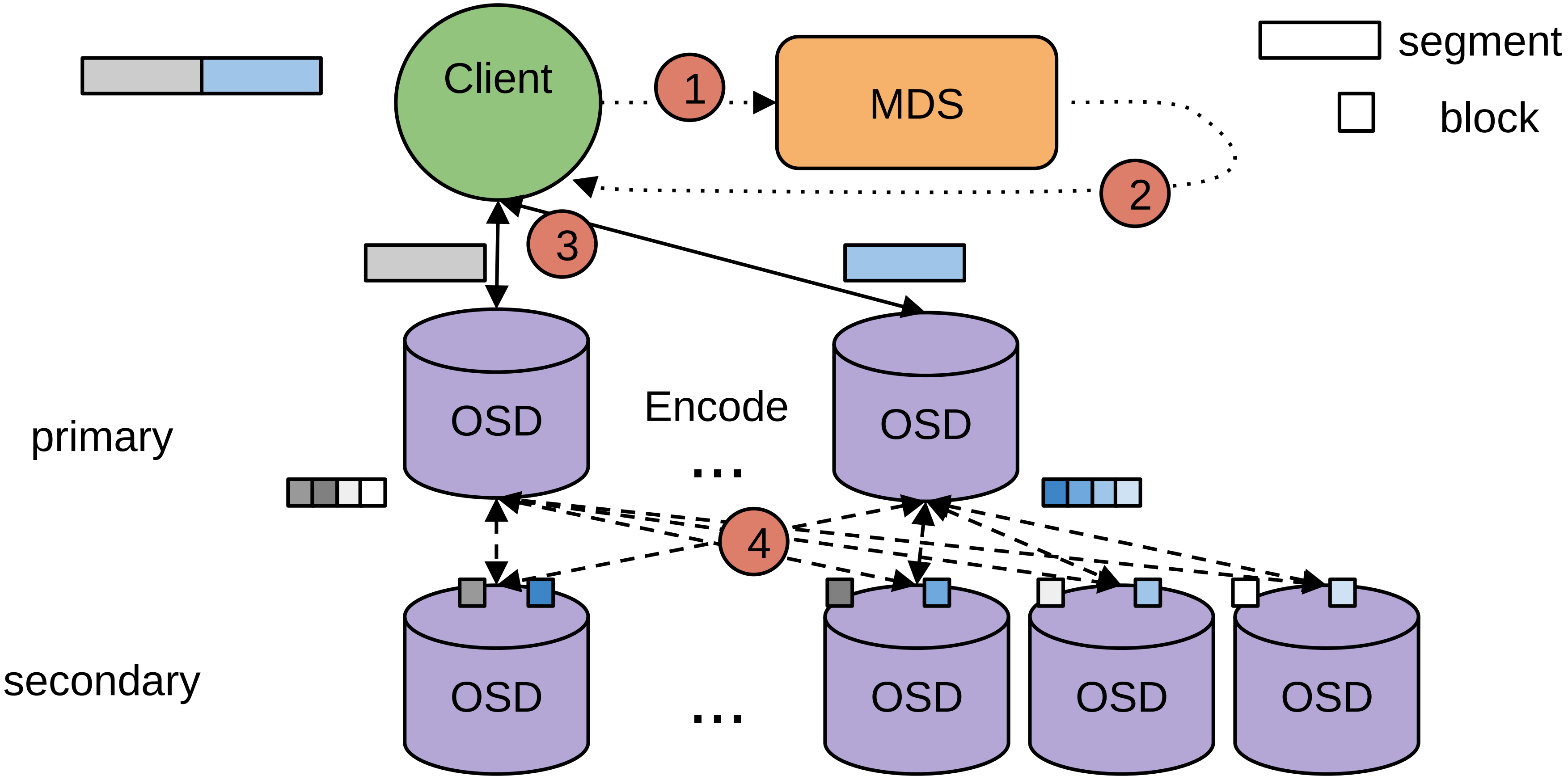
- **NCVFS**, network coding video file system
  - Splits a large file into smaller segments that are striped across different storage nodes
- Flexible coding
  - Each segment is independently encoded with erasure coding or network coding
- Decoupling metadata management and data management
  - Metadata updates off the critical path
- Lightweight recovery
  - Monitor health of storage nodes and trigger recovery if needed

NCVFS: Architecture

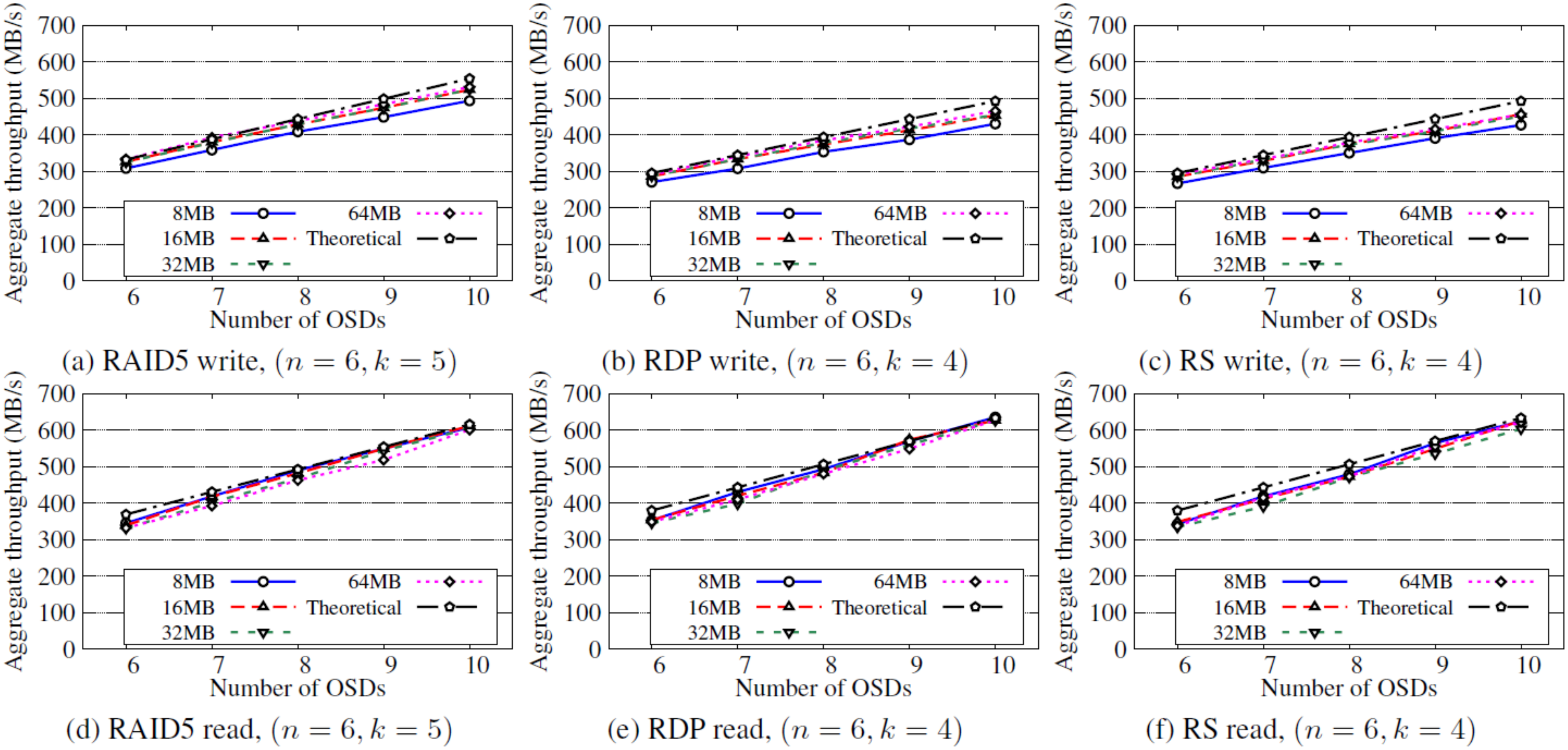
- Key entities:
  - Metadata server (MDS)
  - Object Storage Device (OSD)
  - Clients
  - Monitors
- Master-slave design



NCVFS: I/O Path



NCVFS: Performance



(a) RAID5 write, $(n = 6, k = 5)$

(b) RDP write, $(n = 6, k = 4)$

(c) RS write, $(n = 6, k = 4)$

(d) RAID5 read, $(n = 6, k = 5)$

(e) RDP read, $(n = 6, k = 4)$

(f) RS read, $(n = 6, k = 4)$

- ***Aggregate read/write throughput***
  - ***Achieve several hundreds of megabytes per second***
  - ***Network bound***

Research Philosophy

- Emphasis spans on wide range of theoretical and applied topics

- Research topics need to be:
  - Novel and useful
  - Addressed by both
    - Rigorous algorithmic design and analysis
    - Extensive system implementation, prototyping and experiments

- Our measure of success:
  - Visibility in international research community
  - Conference/journal papers + software tools

Network Coding Storage Research

- Research results published in top systems journals/conferences
  - *e.g., TC, TPDS, FAST, DSN, INFOCOM, MSST, SRDS*
- Open-source software released
- Publications and source code:
  - **http://www.cse.cuhk.edu.hk/~pclee**

Thank you!