LRC Erasure Coding in Windows Storage Spaces

Cheng Huang

Microsoft Corporation

Joint work with Parikshit Gopalan, Erik Hortsch, Jin Li, Karan Mehra, Shiv Rajpal, Surendra Verma, Sergey Yekhanin

Outline

- Storage Spaces Overview
- Resiliency and Availability Mechanics
- LRC Erasure Coding
- Cost and Performance Benefits

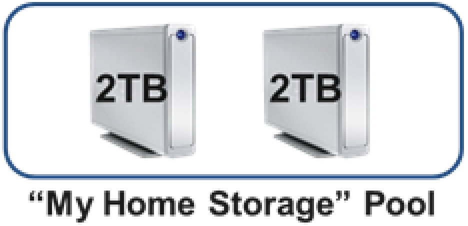
Windows Storage Spaces Overview

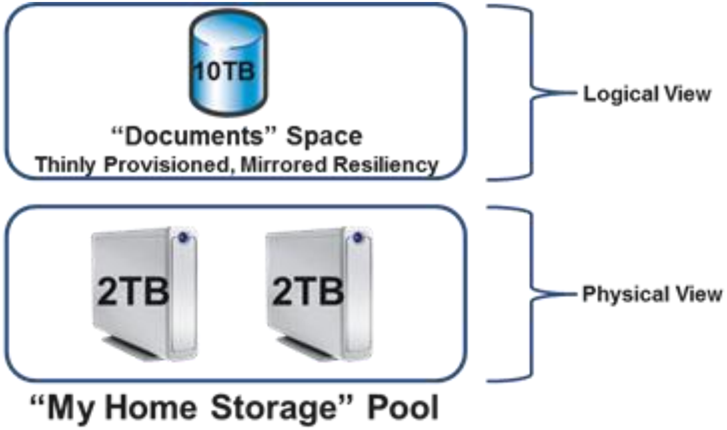
Storage Spaces Overview

- Storage Spaces: storage virtualization platform in Windows 8 and Windows Server 2012
  - Greatly enhanced in Windows Server 2012 R2 and Windows 8.1
- Flexible, resilient, scalable and highly available storage for both consumers and enterprises

Home Example

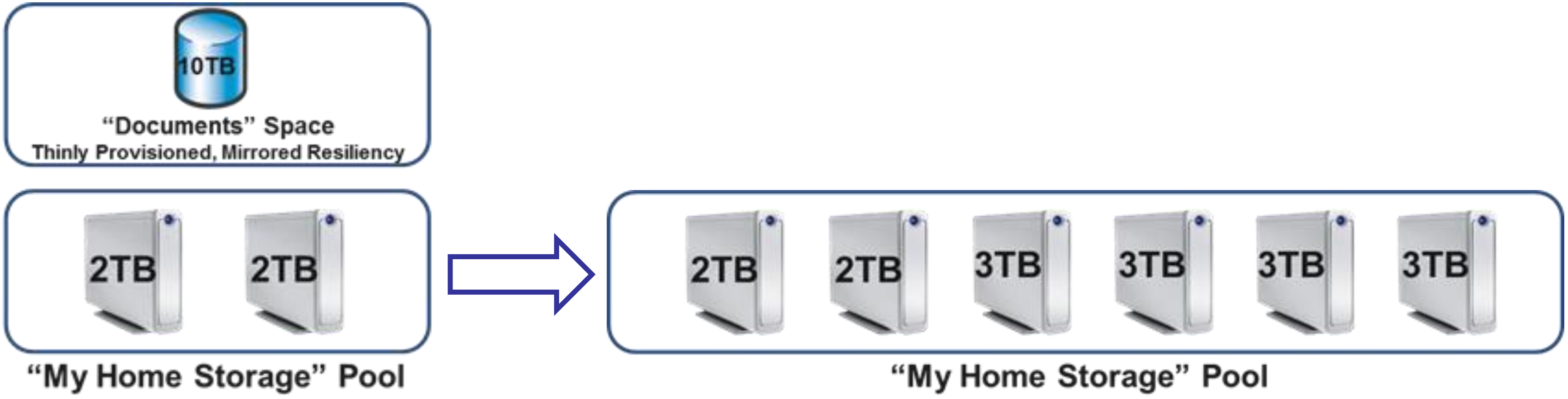
- Storage Pool: a collection of physical drives
- Storage Space: virtual drive created from free space in a storage pool





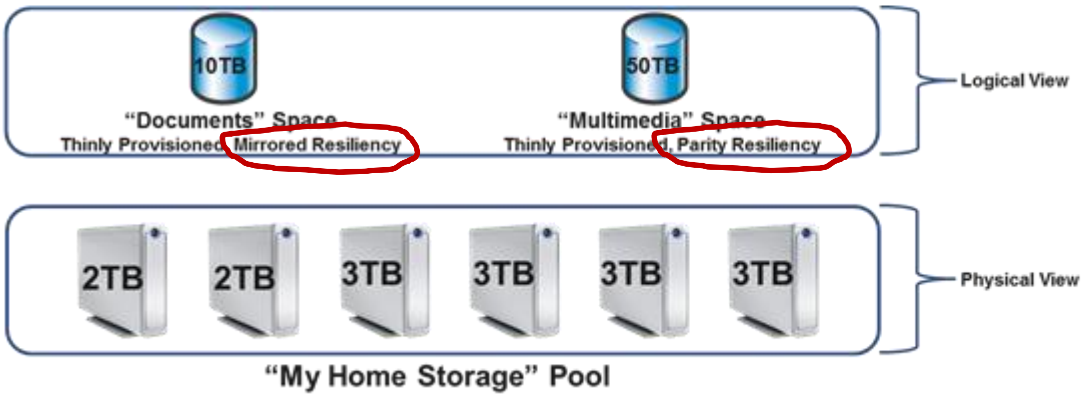
Thin Provision

- Thin Provision: actual capacity is not consumed by the space until used

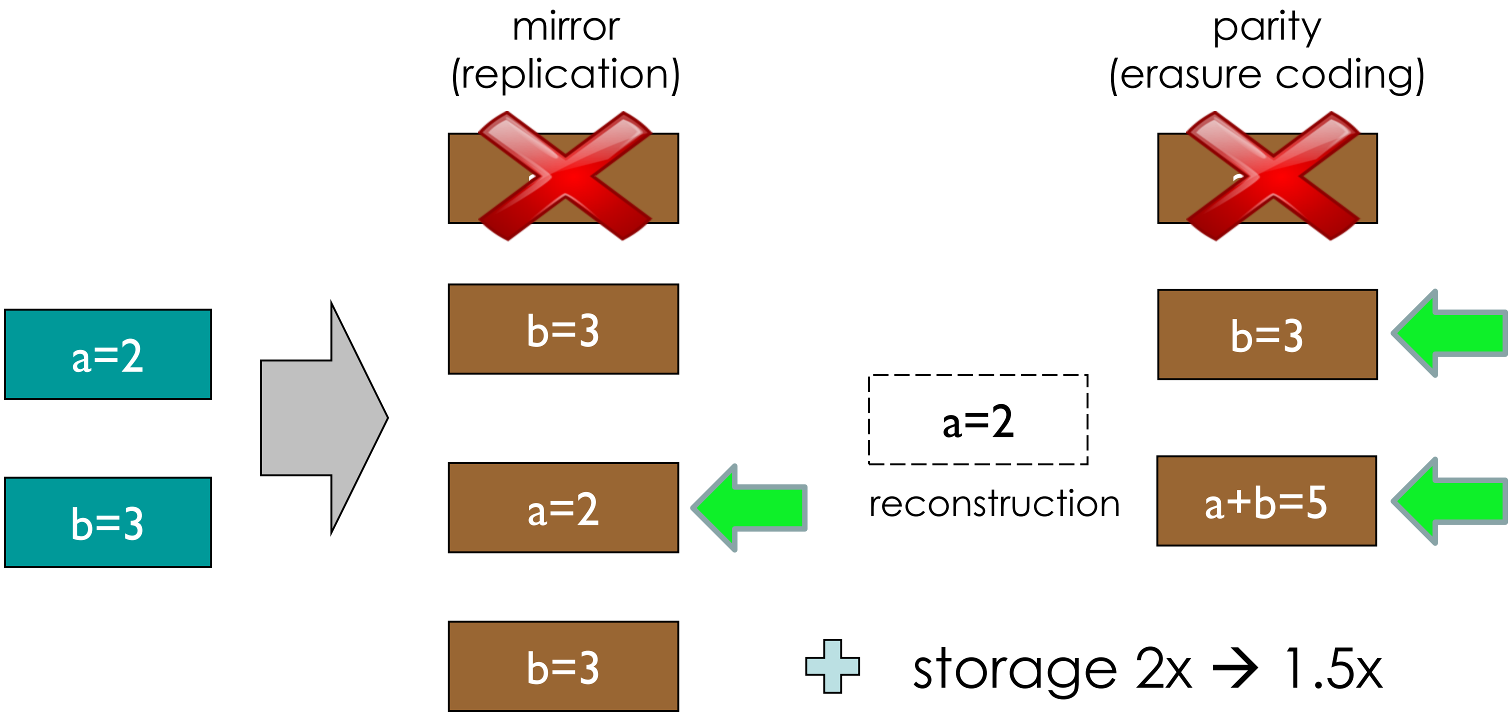


Flexibility

- Multiple spaces from the same pool
- Each space chooses its own resiliency scheme

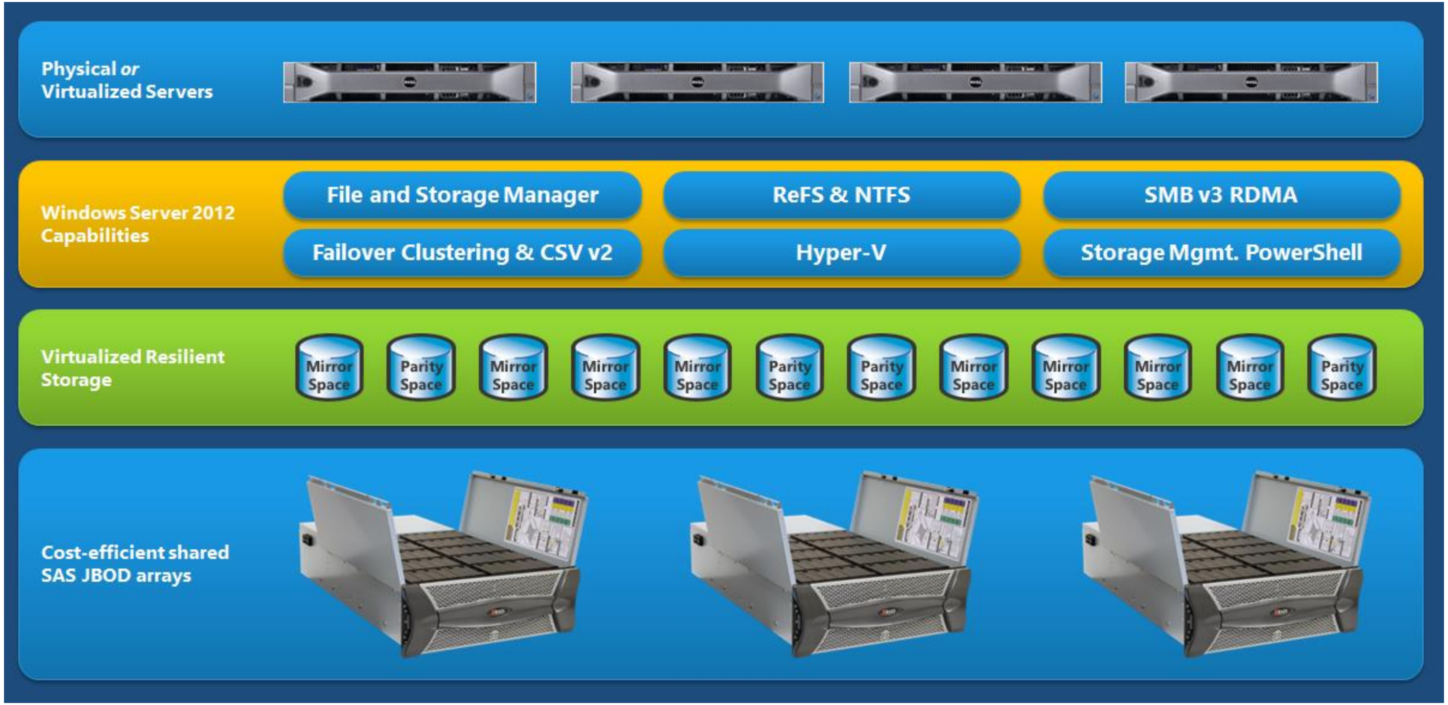


Mirror vs. Parity Resiliency



- storage 2x → 1.5x
- reconstruction 1 → 2

Enterprise Example



From Single Server to Cluster of Servers with Multiple JBOD Enclosures

Clustered Storage Spaces

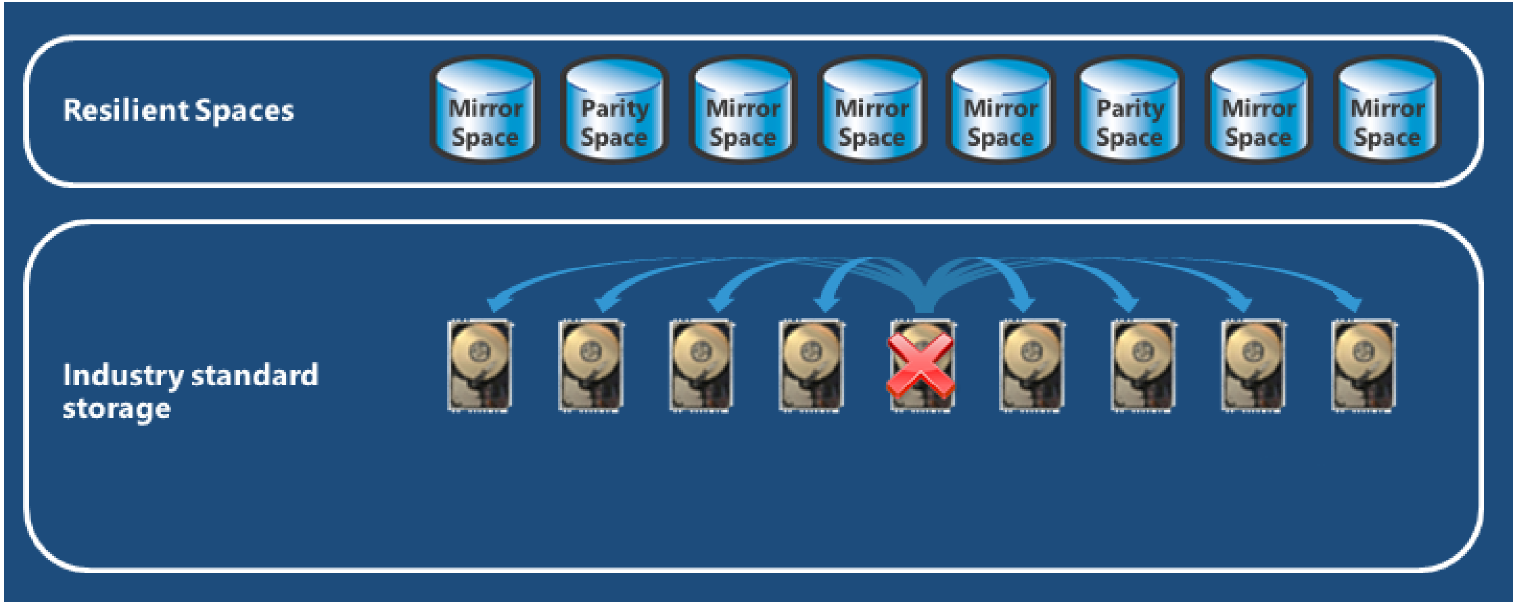
- Use Storage Spaces together with Failover Clustering feature in Windows Server
  - Create storage pool across multiple JBOD enclosures
  - Read/Write storage space from any server in the cluster
  - Automatic failover during failures of hard drive, JBOD enclosure and server

Resiliency & Availability Mechanics

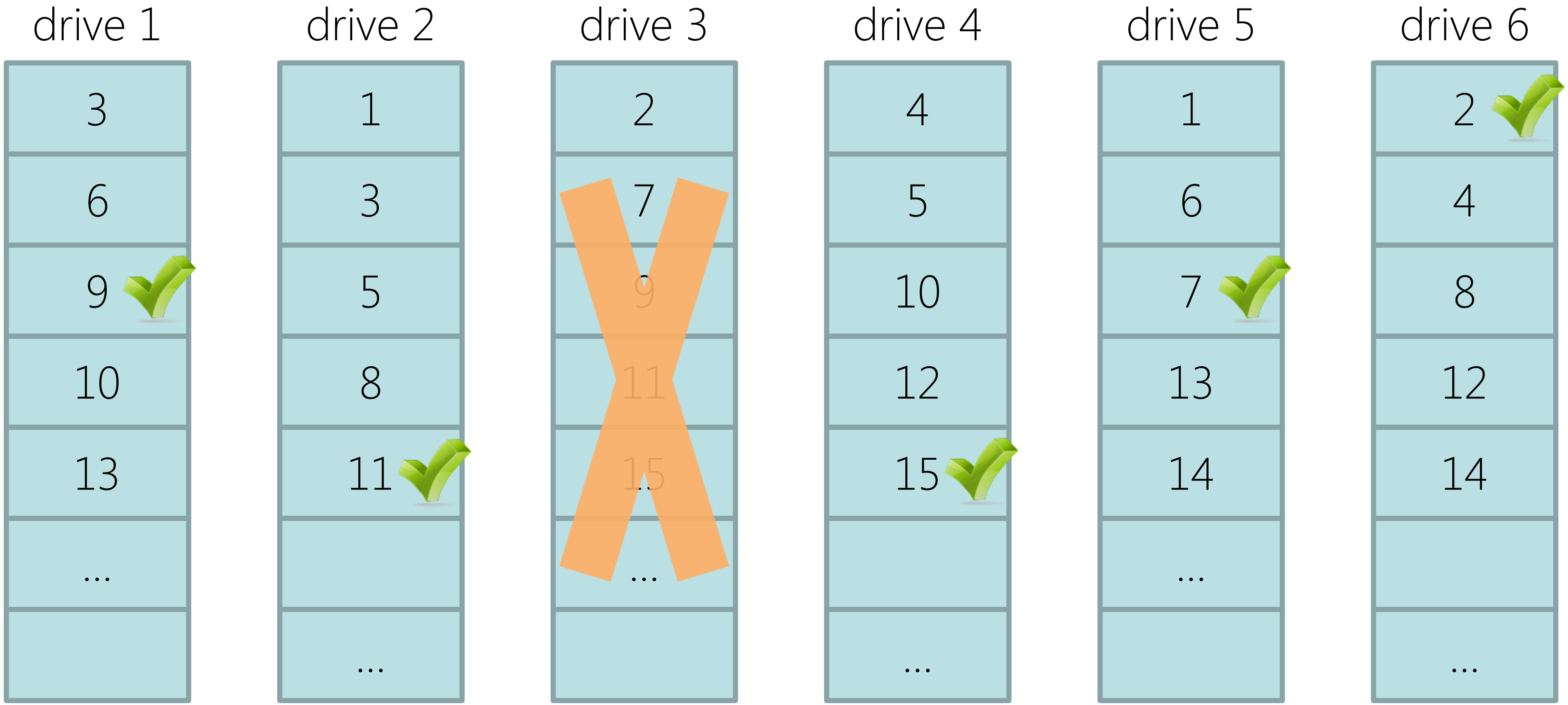
Capacity Allocation

- Storage Space allocates physical capacity in “slabs”
  - slab size = 256MB
- Mirror Space
  - Each slab is mirrored on 2 separate drives
- Parity Space
  - Slabs across multiple drives form erasure coding groups

Parallel Failure Rebuild



Parallel Failure Rebuild



rebuild uses all remaining drives as both source and destination

slabs:

Journaling

- Space is mutable → slabs can be overwritten
- Integrity of Space against power loss or drive failure is protected by journaling
  - Journal mirrored
    - 2-way or 3-way based on resiliency scheme
  - Incoming writes journaled before applied to target slabs
  - SSD as journal most effective in absorbing random writes

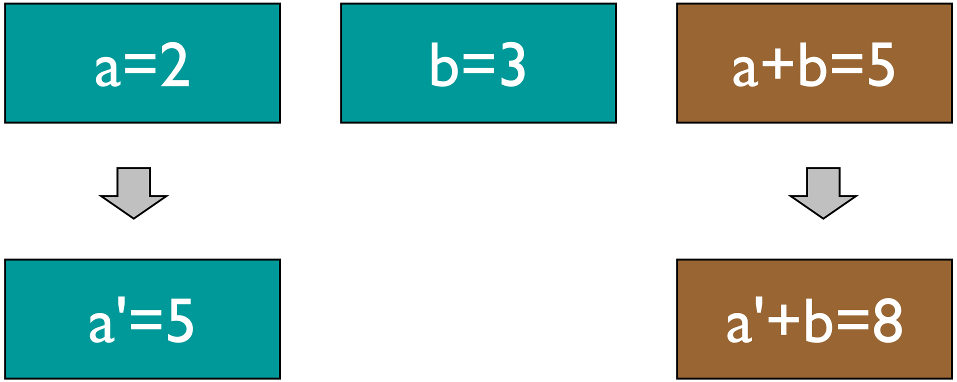
Write IO Example

- On critical path
  - Incoming write sent to mirrored journal
  - Flush sent to journal
  - → Write completed and acked

- In background
  - De-stage from journal to target slabs

Write IO Example

- De-stage overwrite IO in Parity Space
  - overwite IO changes a=2 → a’=5



- Read
  - new data (a’=5) from journal
  - old data (a=2) from disk
  - old parity (a+b=5) from disk
- Calculate new parity
  - (a’-a) + (a+b) = 8
- Flush new parity to journal
- Flush new data and parity to disk

Handling Data Corruption

- Storage Spaces even more powerful in handling data corruption together with ReFS
  - ReFS keeps checksum for every block and srubs data on rest in background
  - Storage Spaces automatically repair slabs when data corruption is detected

LRC Erasure Coding

Classic Erasure Codes

- Reed-Solomon (RS) codes most widely used
  - basis of RAID
- Example – RAID6_{4+2}
  - 2 parities calculated from 4 data blocks
  - tolerates up to 2 failures

a	b	c	d	p	q

Why New Erasure Codes?

- Classic erasure codes were designed and optimized for communication, not storage.
- Naively applying classic erasure codes in storage system is okay, but missing enormous opportunities!

Opportunity I – Space Saving

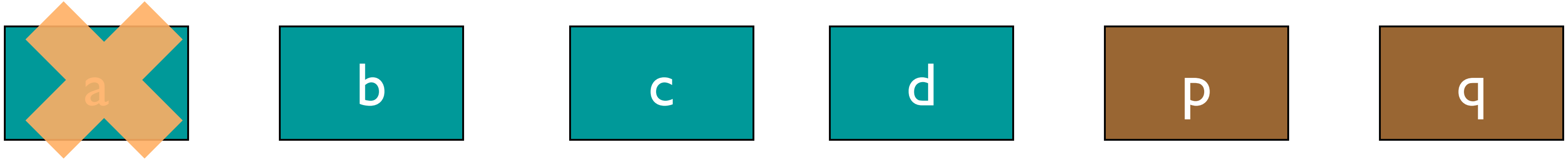
- Storage systems are often hierarchical, bringing multi-level durability requirements
- Consider a Storage Pool with 6 JBODs
  - How to tolerate failures of 1 JBOD + 1 HDD?
    - Note: no need to tolerate 2 JBOD failures

Opportunity I – Space Saving

- How to tolerate failures of 1 JBOD + 1 HDD?
  - RAID6_{4+2} is an option, but it tolerates 2 JBOD failures
  - Excessive durability → storage space waste!
- New erasure codes designed targeting multi-level durability requirements can reduce storage space

Opportunity II – Performance Gain

- Failures do happen, but storage systems continue to operate
- Missing data need to be reconstructed to
  - serve read IO targeting missing data
  - bring resiliency back to desired level



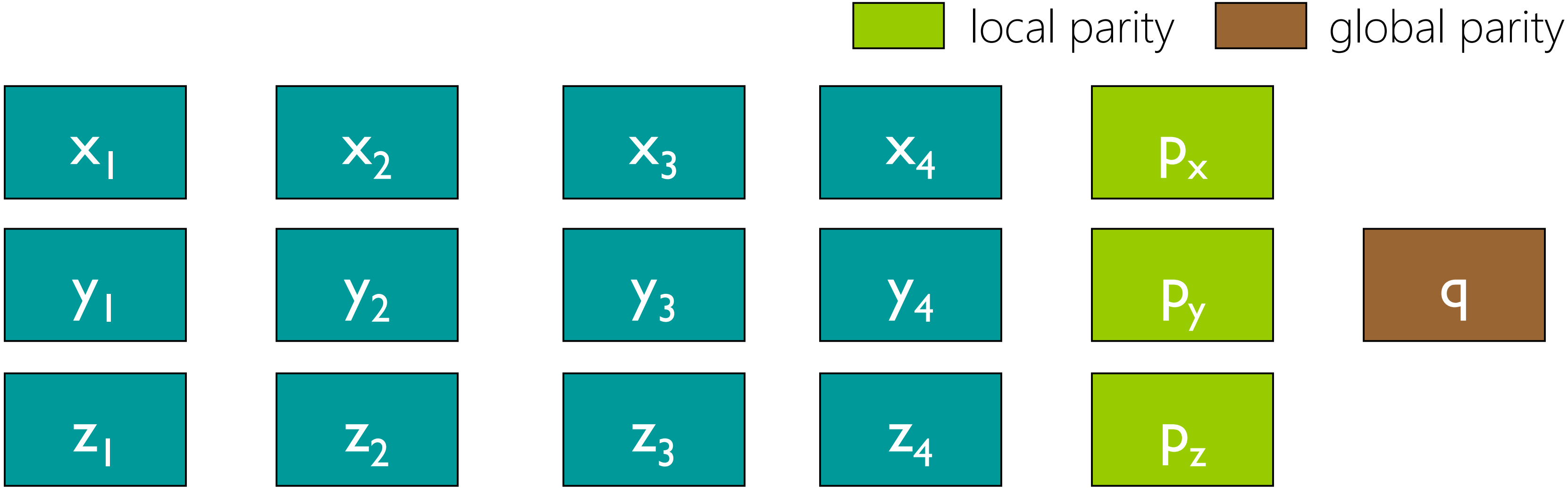
Opportunity II – Performance Gain

- Reconstruction bears IO cost
  - In classic erasure codes, reconstruction cost is the same despite of the number of failures
    - RAID6_{4+2}: reconstruction of 1 and 2 failures both cost 4 IOs
  - In storage systems, single failure way more common than multiple failures
- New erasure codes optimized for single failure can reduce reconstruction cost for common case

LRC Erasure Coding

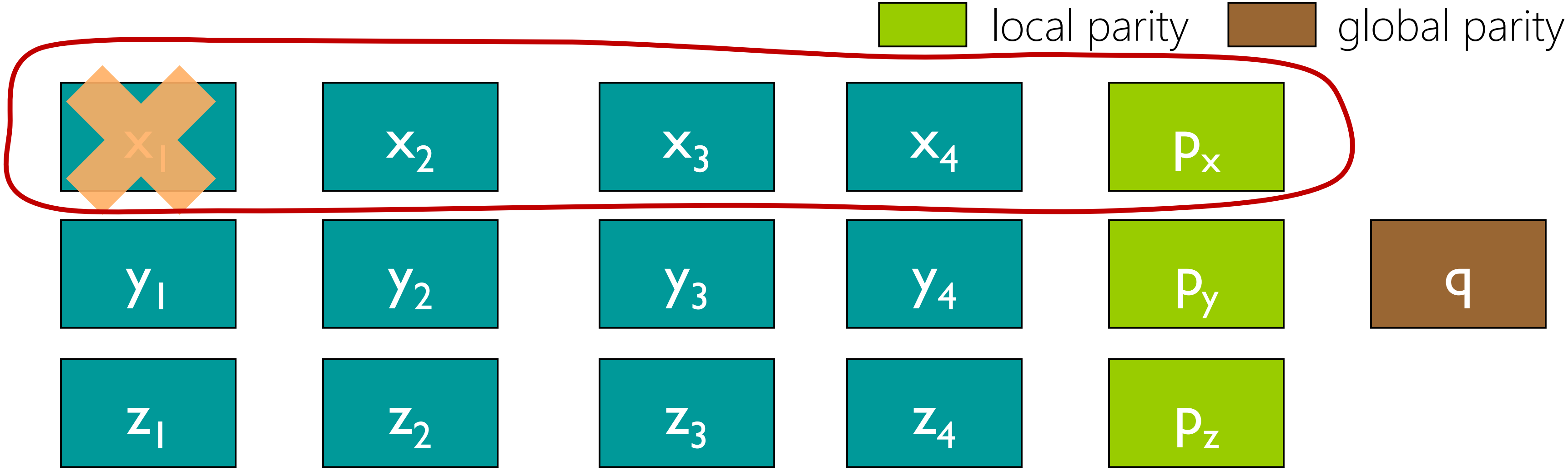
- LRC: erasure codes optimized for storage
  - designed targeting multi-level durability requirements
    - space saving over classic erasure codes
  - optimized for single failure reconstruction
    - performance gain over classic erasure codes
  - introduces parity locality
    - LRC stands for Local Reconstruction Codes

LRC Erasure Coding Example



- LRC specified by # of data, local parity and global parity
  - LRC_{12+3+1}: 12 data, 3 local parities and 1 global parity

LRC Erasure Coding Example

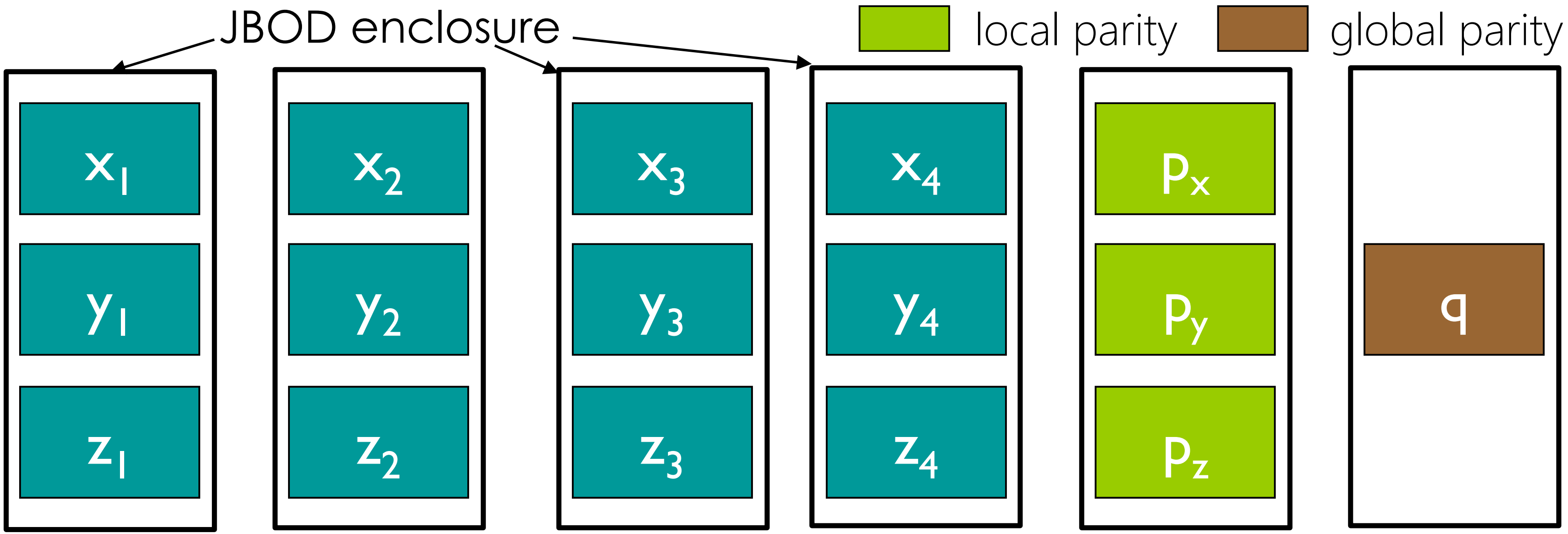


- LRC specified by # of data, local parity and global parity
  - LRC_{12+3+1}: 12 data, 3 local parities and 1 global parity
  - local reconstruction: $x_1 = p_x - (x_2 + x_3 + x_4)$

Cost and Performance Benefits

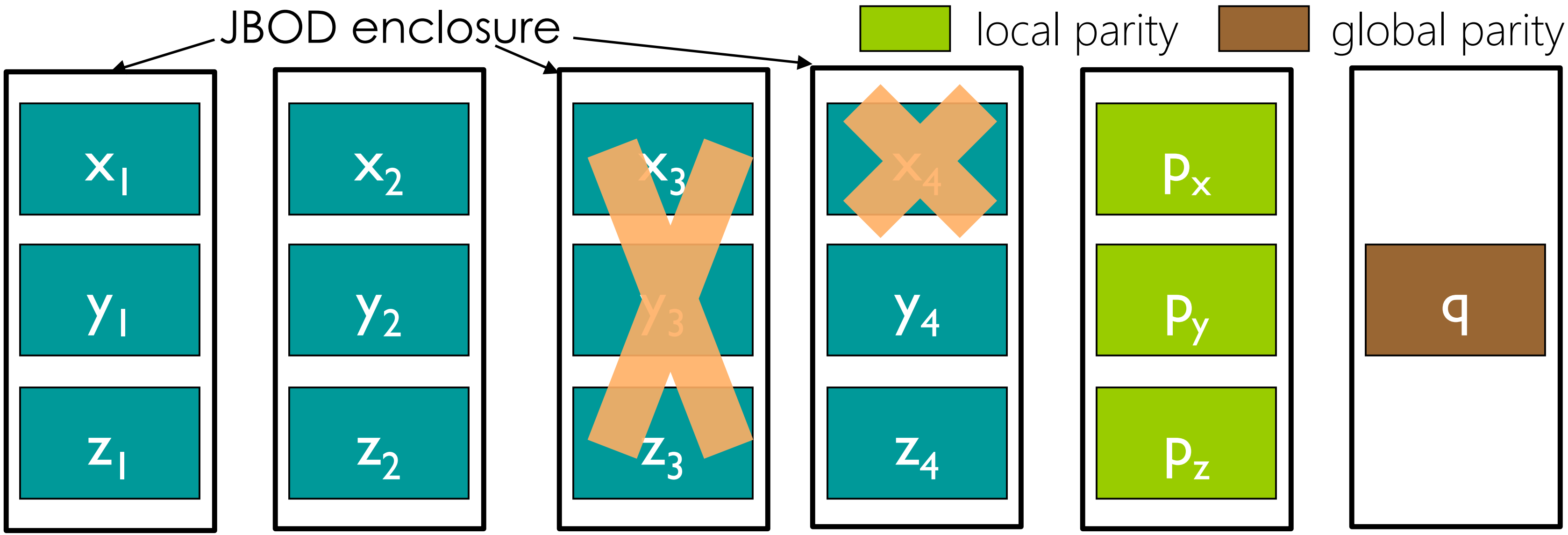
Space Saving over RAID

10 YEARS SDC STORAGE DEVELOPER CONFERENCE SNIA SANTA CLARA, 2013



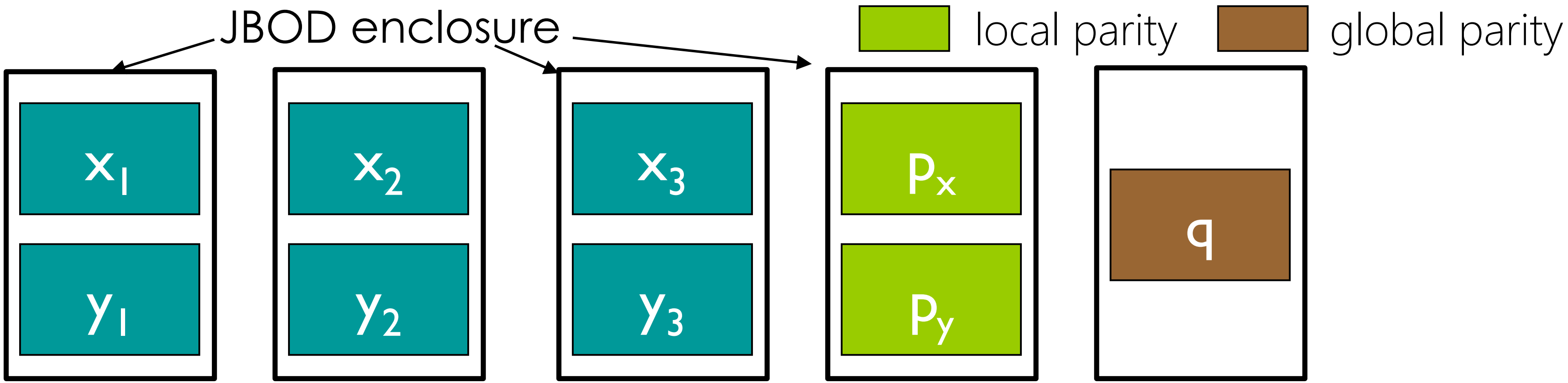
- storage overhead: 1.33x (LRC_{12+3+1}) < 1.5x ($RAID6_{4+2}$)
- But, does LRC indeed tolerate failures of 1 JBOD + 1 HDD?

Space Saving over RAID



- y_3 and z_3 are reconstructed using local parity p_y and p_z
- x_3 and x_4 are then reconstructed using p_x and global parity q

Performance Gain over RAID



- LRC_{6+2+1}: 6 data, 2 local parities and 1 global parity
- storage overhead: 1.5x (LRC_{6+2+1}) = 1.5x ($RAID6_{4+2}$)
- reconstruction IO: 3 (LRC_{6+2+1}) < 4 ($RAID6_{4+2}$)

LRC vs. RAID Summary

	$RAID6_{4+2}$	LRC_{12+3+1}	LRC_{6+2+1}
storage overhead	1.5x	1.33x	1.5x
reconstruction IO	4	4	3

tolerating failure of 1 JBOD + 1 HDD

- LRC offers better trade-offs for storage
  - same storage overhead → fewer reconstruction IOs
  - same reconstruction IO → less storage overhead

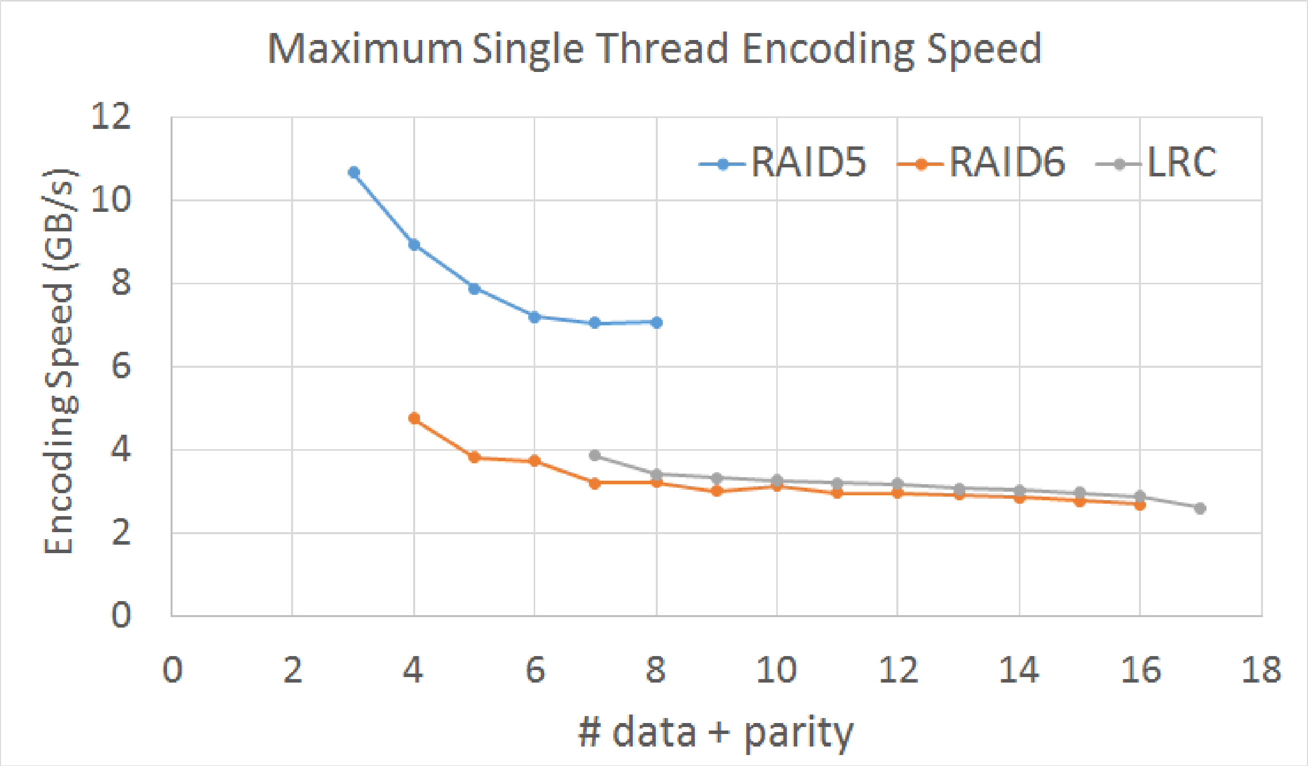
LRC vs. RAID Measurements

	$RAID6_{4+2}$	LRC_{12+3+1}	LRC_{6+2+1}
storage overhead	1.5x	1.33x	1.5x
reconstruction IO	4	4	3
reconstruction read (IOPS)	1333	1328	1695

measures from a 16-drive deployment

- LRC offers better trade-offs for storage
  - same storage overhead → 27% more IOPS
  - same reconstruction IO → 11% less storage overhead

Blazingly Fast Computation



Summary

- LRC: erasure codes optimized for storage
  - designed targeting multi-level durability requirements
  - optimized for single failure reconstruction
- LRC offers better space and performance trade-offs than classic erasure codes (RAID)
- Available now in Windows 8.1 and Windows Server 2012 R2