SDC
STORAGE DEVELOPER CONFERENCE
SNIA SANTA CLARA, 2016

Unified Deduplication

Louis Imershein
Permabit Technology Corporation

Overview

- Fundamentals of deduplication
- Primary Deduplication vs Backup
- Unifying deduplication across all storage
- Case Study: VDO Optimizer

Fundamentals of deduplication

- GOAL: Identify duplicate byte streams and store them once on write
  - Segment data into byte streams
  - Use hashing to name the byte streams
  - Lookup the byte streams in an index
  - Update metadata, deposit unique data as needed and acknowledge the write

Key areas where implementations differ

- Segmentation
- Hashing
- Indexing

Segmentation

- Fixed block size
  - Split data streams into fixed size chunks
  - Small chunks yield best results, larger chunks may provide better performance
- Variable block size
  - For known types of data (e.g tar, zip), parse on known boundaries.
  - For unknown types of data, use a rolling hash to identify logical boundaries.

Variable block segmentation

- Known types approach
  - Identify potential file type from header info
    - Segment based on header information
    - Segment based on boundary tags in a stream
    - Segment based on footer data (undesirable)
- Rolling hash approach
  - Identify potential segments by using a hash technique on a sliding window of a predefined size
  - Calculate a hash on the window and see if the hash matches a pre-defined fingerprint

Hashing

- Strong cryptographic hashes
  - Can be used as canonical names for chunks
  - Require more CPU resources
- Fast non-cryptographic hashes
  - Identify potential matching chunks
  - Require read-verify of existing chunks, more IO resources

Indexing

- Performance
  - RAM is orders of magnitude faster than storage media
- Resource efficiency
  - Storage media is substantially less expensive than RAM
- Indexes require persistence

Key assumptions for primary vs backup

- Backup deduplication workflows
  - can anticipate higher redundancies
  - can use asynchronous IO paths
  - require high sequential throughput
- Primary deduplication workflows
  - see less redundancy
  - must provide synchronous IO guarantees
  - require low latency and high IOPS

What's special about backup?

- Characteristics of the backup workflow allow for simplifications
  - Buffering for large look-back window
  - Locality knowledge to individual sources
  - Large block similarities
- Without latency restrictions, backup solutions are able to devote more time to identifying intelligent boundaries for data segmentation

Backup doesn't apply to primary

	Backup	Primary
Data Flow	Stream-Oriented	Random Access
Latency Critical	No	Yes
Typical Chunk Size	128 KB and up	4 KB to 16 KB
Index Lookups	Thousands/sec	Millions/sec
# Objects	100s Millions	100s Billions

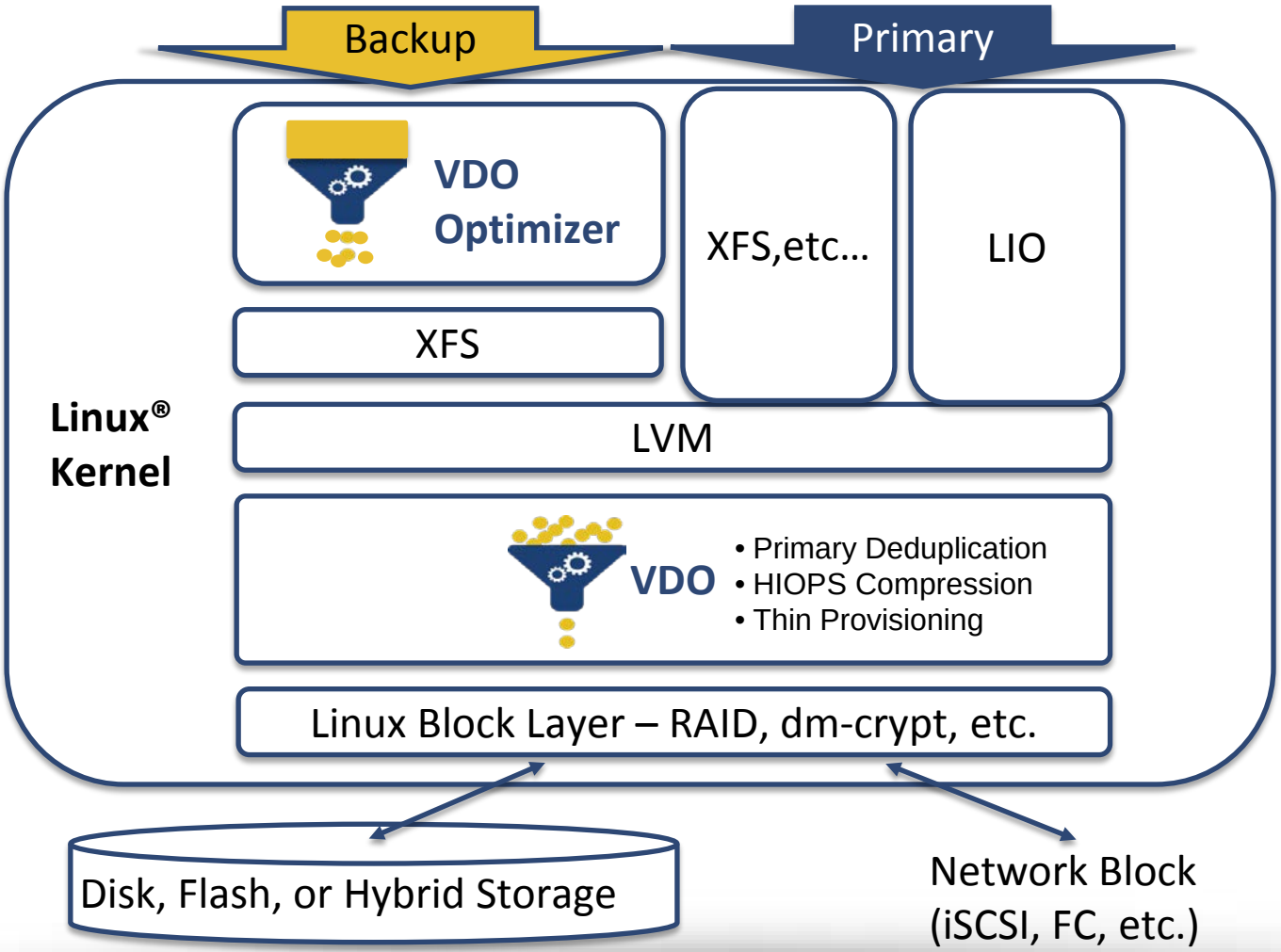
Can primary dedupe adapt to backup?

- To be useful for backups, a primary deduplication solution must:
  - Fit in the resource footprint available
  - Maintain temporal and spatial locality to maximize sequential performance
  - Be able to segment data along intelligent boundaries to get comparable data reduction rates

Case Study: VDO Optimizer

- Permabit VDO Optimizer File System
  - Segments content on intelligent boundaries and pads to maximize deduplication rates for backup
  - Simple pass-through file system mounted on top of a standard file system
  - Primary (fixed-block) deduplication is handled
    - by primary deduplication in the file system; or
    - through primary deduplication implemented in the block layer (e.g. Permabit's VDO device mapper target)

VDO Optimizer with VDO



Optimizer Architecture

- Scanners – parse and realign/pad content to increase fixed-block deduplication rates
  - Tar, Zip today
  - API for 3rd party scanners (future)
  - Generic – rolling hash based
- Metadata – stored in each processed file
  - Header – contains file-level information
  - Mapping blocks – identify where actual data was written

Optimizer implementation choices

- VFS in kernel vs FUSE in user space
  - VFS proved more robust, better performance
- In file metadata vs central database
  - In-file approach ensures consistent operations
- In-house rolling hash, similar to adler-32
  - Delivered excellent performance without compromising efficiency

Lessons Learned from Optimizer

- Short chunks in a fixed-deduplication system have the potential to waste space because we're zero-padding the short blocks, but with some tuning, in real-world data sets you still consistently come out ahead
- When properly implemented dual-purpose solutions require slightly more resources, but offer greater flexibility for users
- Hybrid solutions can address the efficiency, performance and scalability requirements expected for both primary and backup use cases