COMPUTERWORLD SNIA

SNW

April 2010 | Rosen Shingle Creek Resort | Orlando, Florida

April 2010
Rosen Shingle Creek Resort
Orlando, Florida

Data Reduction and File Systems

Jeffrey Tofano
Chief Technical Officer,
Quantum Corporation

Today's Agenda

- File Systems and Data Reduction Overview
- File System and Data Reduction Integration Issues
- Reviewing Data Reduction Technologies
- Reduction and Data Management
- Data Reduction Technologies "On-the-Wire"
- Summary and Questions

About Quantum

- **The global leader in backup, recovery and archive**
- **Pioneer in disk backup**
  - First with VTL for open systems
  - First with integrated D2D2T system
  - Pioneering patent in variable-length deduplication
- **Dedupe solutions trusted throughout the world**
- **Over 800 PB protected by Quantum deduplication technology**

Overview

- Data storage requirements are growing wildly
  - Fastest growth in unstructured data
  - File/object-based storage dominates
  - Data reduction becoming a required feature to deal with growing volume of data
- Many file system and reduction techniques available
  - Lots of options and tradeoffs for integration
  - What makes sense going forward
- Many ways to integrate reduction techniques into file systems:
  - Basic integration (on-disk reduction)
  - Data management integration (on-the-move reduction)

File Systems and Reduction

- **Key issues to consider when integrating reduction technology and file systems:**
  – What are available reduction techniques
    - Compression
    - Single instancing
    - Dedupe
  – How do selected methods get integrated
    - Layered on top of file system
    - Integrating into file system
    - Build new file system around selected reduction methods(s)
  – What are tradeoffs of each reduction method with respect to selected integration model
  – How does select model affect data management

Data Reduction Technologies

- **We are reduction technologies available to integrate?**
  - Compression technologies
    - Tend to operate at file or object scope, and on relatively small "chunks".
    - Typically good "intrinsic reduction".
    - Little or no cross-object benefits.
  - Single Instance technologies
    - Tend to operate across files/objects/blocks and on relative large "chunks"
    - Poor "intrinsic reduction"; best at removing copies
    - Very sensitive to small changes
  - Dedupe technologies
    - Also operate across files/objects/blocks but at a lower level of granularity than SI
    - Can generate good intrinsic reduction.
    - Can handle "small" changes well
    - Reduction rate often scales with size of repo (to a point).
  - Hybrids and mixed technologies
    - Tend to be mixes of above schemes
    - The path to the future

April 2010
Rosen Shingle Creek Resort
Orlando, Florida

Compression: An Overview

- **Lot's of different compression schemes exist**
- **At high level, most share common processing model:**
  - Build a dictionary (static/dynamic/probabilistic)
  - Encode symbols that represent larger entities in dictionary
  - Replace data extends in stream with encoded symbols
- **Most schemes are local in scope: reduce individual objects well but doesn't recognize copies of objects**
- **Some are general, some are data set specific; reduction rate varies from 2x to 8x**
- **Starting to see compression schemes that are opening scope of reduction across objects**

Single Instancing: An Overview

- **Again, many different solutions in market, but most operate at file/object scope**
- **At high level, operation is simple:**
  - File or object is fingerprinted (checksum/hash/signature)
  - File or object fingerprint is indexed
  - When objects move in our out, index is queried to see if matching fingerprint exits
  - Either original object or a reference to existing object is stored
- **Single instancing is typically global in scope (limited by index)**
- **Reduction rates vary depending on level of object level copies**
- **File and block solutions exist**
- **Small changes often result in entirely new object**

Dedupe: An Overview

- **Although several different solutions available, there are three major variants:**
  - Hash-based variable schemes
  - Similarity/Byte differential schemes
  - Fixed block schemes (essentially same as block SI)
- **At high level, operation is a bit more complex, especially for variable schemes:**
  - Some criteria (e.g., simple rolling hash) is used to determine data boundaries and form “chunks”.
  - Chunks are fingerprinted (complex hash/checksum).
  - As chunks move in and out, index is queried to see if matching fingerprint exists
  - Either unique chunk is stored or a reference to existing chunk
  - Map data must be created to “rehydrate” the data properly

Dedupe: Additional Details

- **As simple as dedupe process may seem, there are lots of variations:**
  - ***"Where"* dedupe gets done**
    - Client-side, target side or co-operative?
    - In the application, at the file level or at the block level?
  - ***"When"* dedupe gets done**
    - Inline, post-process or in adaptive/hybrid fashion?
  - ***"How smart"* is the dedupe method**
    - Application or data format aware?
    - One parsing model fits all?

Data Reduction: Rules for All of Us

- **The bigger the data set, the more costly to process (IO, CPU, memory bandwidth)**
- **Index-based technologies are difficult to scale**
  - Lots of little chunks (good for reduction) means lots to search
  - Less big chunks (less reduction) means less to search
  - Index has to be stored somewhere
  - References can introduce fragmentation.
  - Consider math behind a mutli-terabyte store
- **Global scope means many objects with potentially many chunks**
- **Many reduction methods only work on sequential data; workloads matter…..a lot**
- **There is no "free lunch" – somehow, somewhere you have to dehydrate and re-hydrate the data!**

Data Reduction: Consequences

- **Each reduction method makes tradeoffs to balance performance cost against reduction ratio:**
  - Just because we can reduce wonderfully, doesn't mean customer is willing to pay performance cost.
- **In each class of reduction technology, individual solutions make additional/different tradeoffs**
  - Can't mix and match most without "bottlenecks".
  - Since SW is often bound to HW, this can lead to a lot of deployment complexity
- **Each reduction method tends to have a "sweet spot" with respect to data types it can reduce well**
  - Solutions must increasingly deal with low and high entropy data
  - Solutions must increasingly deal with pre-compress/encrypted data
  - Customers don't want to manage pools of "like" data

Integrating Reduction and File Systems

- **There are tradeoffs when integrating different reduction methods:**
  - Compression schemes
    - Desired reduction rate can drastically affect performance and trigger excessive RMW operations
    - CPU utilization varies dramatically; offload options still increase access latencies
  - Single Instancing schemes
    - Often troublesome for hot-data that changes a lot
    - Often requires namespace enhancements (i.e., lookup by hash)
  - Dedupe schemes
    - Fixed schemes often far easier than variable schemes
    - Indexing can become central access bottleneck

Data Management and Reduction

- **How does data reduction affect typical data management tasks:**
  - What are typical data management features
    - Snapshots
    - Replication
    - Migration and ILM/HSM
  - How do various data reduction schemes map on data management features
    - Cost/benefits in on-disk footprint
    - Cost/benefits when on the wire
    - Cost/benefits when data is tiered and retrieved

Snapshots and Data Reduction

- **Are snapshots and described reduction schemes compatible and/or beneficial?**
  - Snapshots and dado are both reference-based technologies
    - Should they be layered or designed together?
    - What are benefits/downsides of each
  - Can deltas be reduced with value in single server and distributed environments

April 2010
Rosen Shingle Creek Resort
Orlando, Florida

Replication and Data Reduction

- **Since replication typically involves on-the-wire transfers, data reduction benefits are obvious, but:**
  - Not all reduction schemes are equivalent
  - Benefits often depend on topology
    - Simple DR setups
    - Edge-to-core setups
    - Distribution and migration setups
  - Distance can dramatically affect benefits of each reduction scheme

April 2010
Rosen Shingle Creek Resort
Orlando, Florida

Data Reduction “On-the-Wire”

- **Multiple considerations when moving data over-the-wire:**
  - Is data being moved between a data-reduced repo and traditional “raw” system
  - Is data being moved between two systems with same reduction technology
  - When using similar data reduction systems, is existing data being replicated or “copied”
  - Can multiple data reduction technologies be employed at each stage of movement
- **Mixing file and block level solutions is problematic – often, mixing NAS and VTL demonstrate similar problems**
- **What media must the data be moved over: high-latency or low-latency?**
- **Each data reduction scheme has benefits and downsides in each of above scenarios**

Compression “On-the-Wire”

- **Data compression is most ubiquitous on-the-wire solution**
- **Many solutions available….often they don’t need to be matched (smart compress/decompressors)**
- **Benefits are obvious, but so are costs**
  - Less data moves (directly related to reduction rate), which is good, but….
  - System resources are consumed on one (or both) sides depending on the need and model
  - Two identical files being moved are each reduced, but there are still two files transferred; very limited (if any) copy protection afforded
  - Although rare, some schemes required significant static dictionary communications before data can be shipped

Single Instance "On-the-Wire"

- **Single instance technologies also widely available for on-the-wire reduction**
- **Solutions must often be matched – many variations in what gets fingerprinted and how; both ends must match**
- **Overall scheme is simple**
  - Client obtains or calculate an object fingerprint
  - Client sends fingerprint to server
  - Server queries object index and responds
  - Client only sends object if unique
- **Benefits and cost are also obvious**
  - File level SI can completely eliminate the transfer of a copy with one back-and-forth negotiation
  - Block level SI often goes through a series of fixed size negotiations to accomplish same thing
  - But things work best when fingerprints are known ahead (e.g. replication)
  - When fingerprints are not known ahead of time, they must be calculated; CPU load and costly file buffering can be introduced

Dedupe "On-the-Wire"

- **Most dedupe vendors offer dedupe-enabled replication , buts there is a lot of variance**
- **Most are somewhat complex forms of a simple model**
  - Client batch up a group of sequential chunk fingerprints
  - Client send batch to smart target that can query existence of each fingerprint
  - Target sends back results and client pushes unique data
- **Above scheme only works when client/server both can form identical chunks and fingerprints**
- **Collaborative dedupe schemes are less common; these schemes provide a method that allows client to chunk and fingerprint data to enable the negotiation**
- **Collaborative schemes don't work over the old legacy protocols (NAS); that's starting to change (OST/XAM/pNFS)**

Dedupe "On-the-Wire"

- **Benefits and cost are more subtle:**
  - Most dedupe solutions send file/object level hash of hashes to prune copies similar to SI technologies
  - Some solutions provide hierarchical hash-of-hashes to obviate the transfer of large ranges
  - Most solutions can negotiate individual chunks
  - For solutions that negotiate all (or most) chunks, a large number of hash negotiations can result
    - Results can be excellent when much of actual data transfer is obviated
    - Results can add to transfer overhead when dedup ratios are low
    - Cost of hash negotiations serializes data transfers; this can be invisible on low-latency wires but cause significant slow downs on high-latency wires

Data Reduction and ILM/HSM

- **Similar to replication, data reductions benefits seem obvious, but:**
  - How do different reduction schemes affect movement from disk-to-disk and/or disk-to-tape?
  - How do different schemes affect read-only copies and version?
  - How do different schemes affect or complement searches and lookups?

Options: How does a Customer Choose?

- **How do you know if a solution works for your type of data?**
  - Ask the vendor?
  - Rough math?
  - Try it?
  - Data analysis tools?
  - Sizing tools?
  - Other customer references?
- **Whatever you do, start to understand:**
  - What performance level you need when pushing data to/from the repo
  - What are your data protection/replication needs; do you need to implement on high-latency or low-latency networks (or both)