



Education

# Strategies for Single Instance Storage

Michael Fahey

Hitachi Data Systems

## Single Instance Strategies for Storage

Single Instance Storage has become a very popular topic in the industry because of the potentially large reduction in cost and increase in efficiency it offers. Various technologies are being promoted for different tasks including backup, archiving, and transactional storage. Each of these storage uses represents a unique set of challenges and the single instancing technology being implemented has major implications for scale, performance, and functionality. This session will review various technologies available including object level de-duplication, fixed block de-duplication, and variable length block de-duplication and the implications of each for backup, archive, and transactional storage.

# SNIA Legal Notice

- The material contained in this tutorial is copyrighted by the SNIA.
- Member companies and individuals may use this material in presentations and literature under the following conditions:
  - ◆ Any slide or slides used must be reproduced without modification
  - ◆ The SNIA must be acknowledged as source of any material used in the body of any document containing material from these presentations.
- This presentation is a project of the SNIA Education Committee.

# What is Single Instance Storage (SIS)?

- Commonality Factoring
- Data De-duplication
- Data Coalescence
- Any technique which identifies multiple copies of identical content and reduces it to a single copy
  - ◆ Primarily for the purpose of reducing storage
  - ◆ It could be at the file or object level or it could be at a sub-file or block level
  - ◆ Should be totally transparent to an end user or application

# SIS Background

- For many years customers have known that many things are stored multiple times in
  - ◆ Primary Storage
  - ◆ Backups
  - ◆ Archives
- How can someone safely reduce the amount of common content across storage?
- In many ways SIS is another larger form of lossless compression
  - ◆ Common items (usually text) are stored in a repository (library) and referenced by a table
  - ◆ Some things compress well and others don't
  - ◆ SIS (or compression) is a trade of CPU processing and latency for storage capacity

- Snapshot
  - ◆ Simple point in time copies
- Content Awareness
  - ◆ Hash Evolution and the MIPS Explosion
- Common Object Model
  - ◆ Identical files or objects
- Common Fixed Block Model
  - ◆ Common fixed size blocks across files or objects
- Common Variable Length Block Model
  - ◆ Common bit patterns across files or objects

# Snapshots

- A file system's inodes and blocks are snapped at a point in time
- After the point in time there are two separate mount points
  - ◆ One for production and future changes
  - ◆ One for reference to the previous point in time
- If a file is changed, only the changed blocks are created
  - ◆ Most of the file system is static
  - ◆ Very good for backup copies
  - ◆ Dozens or even hundreds of snapshots can be made of a file system

# Snapshot tradeoffs

- If the same file is stored multiple times at different file system locations, snapshots won't detect them
- With many applications (such as MS Office) small changes to a file will shift the bulk of the file's content across blocks, reducing snapshot effectiveness
  - ◆ But remember that most files in most files systems don't change over time
- Snapshots don't require a database, content analysis, have very little CPU impact, and are well proven

# Beyond Snapshots: Content Awareness

- What if a storage system could analyze the content that it is storing?
- What if the storage system could identify identical files or portions of files?
- Why should I store the same thing twice?
- How do know if the content I am analyzing is really identical?
- What impact will that have on my storage system?



# Hash and CPU Development

- Hashing technology improved significantly in the 1990's
  - ◆ MD5 and SHA1 introduced
  - ◆ A 'fingerprint' of a file could validate that it is unchanged and be used to detect common content
- But Moore's Law continues unabated
  - ◆ How long before someone could spoof the hashing algorithm?
  - ◆ Generating collisions can be done today with MD5
  - ◆ How long before a second pre-image attack can occur?
- New longer hashes are just part of an arms race
  - ◆ Should be a consideration in an archive strategy

# Common Hash Algorithms

<i>Name</i>	<i>Bit length</i>
<i>MD5</i>	<i>128</i>
<i>SHA1</i>	<i>160</i>
<i>RIPEMD 128/256</i>	<i>128/256</i>
<i>SHA 256/224</i>	<i>256/224</i>
<i>SHA 512/384</i>	<i>512/384</i>
<i>WHIRLPOOL</i>	<i>512</i>

- A longer hash is more secure but...
- Longer hashes take more CPU resources
- A 512 bit hash will take 4 times longer to compute than a 128 bit hash



# Hashing Concerns

- Anytime hashes are used to represent larger objects than the hash itself there is a possibility of a hash collision
  - ◆ Two different items have the same hash result
- Two objects can be manipulated simultaneously to create an MD5 collision today
  - ◆ Other stronger hashes will fail eventually
- It is much more difficult to manipulate an object to resolve to particular hash (pre-image or second pre-image attack)
  - ◆ In a long term archive think of what might be possible as long as the archive is used
- Two objects can be compared in their entirety before SIS but this requires more latency and CPU processing
  - ◆ The hash is just used to identify candidates for SIS
  - ◆ Probably with less impact than a longer hash on everything

# Common Block File System

- A file is broken up into fixed arbitrarily assigned block sizes
- All blocks are hashed and the hashes are listed in a lookup table
- Blocks with identical hashes are targeted for duplicate elimination
- Files which have identical common content share identical blocks
- Bit pattern shifts still cause difficulty (just like with snapshots)

# Venti: The first common block Storage System

- Developed by Bell Labs in the 90's for the Plan 9 file system
- Files were broken up into 8KB blocks
- Blocks were hashed with SHA1 and listed in a lookup table
- Blocks with identical hashes were single instanced
- Heavy CPU and latency impact but Venti was used for the archive (compare to optical retrieval times)
- Generated about 30% reduction in storage by eliminating redundant blocks



# Common Object Model

- Instead of SIS on blocks just use SIS on objects or files
  - ◆ Much lower object counts and less overhead
  - ◆ Eliminates multiple copies of the same file in a repository
- But it will not detect common sub file level content across files
- Guarantee of Authenticity
  - ◆ Since every file is hashed anyway check it against its hash
- Some applications will get a big benefit and others won't
  - ◆ It should be selectively applied
  - ◆ Some applications can single instance as well (at the file level)
  - ◆ But how much can the application scale?

- ▶ Common bit patterns are detected across files
  - ◆ Solves the problem of the shifting bit pattern
  - ◆ Algorithms may have tunable parameters for the best effectiveness
  - ◆ Not every identical bit pattern will be detected (but most will)
- ▶ Most commonly are implemented with backups today
  - ◆ Variable length blocks are discovered within a subset of content (backups of the same file system or databases)
  - ◆ Efficiencies can be very large
  - ◆ Often implemented with a backup to disk strategy

# A Very Simplified Example

*example.doc*

Block 1	Block 2	Block 3	Block 4	Block 5	Block 6	Block 7	Block 8
---------	---------	---------	---------	---------	---------	---------	---------

*example modified.doc*

Block 1	Block 2		Block 3	Block 4	Block 5	Block 6	Block 7	Block 8
---------	---------	--	---------	---------	---------	---------	---------	---------

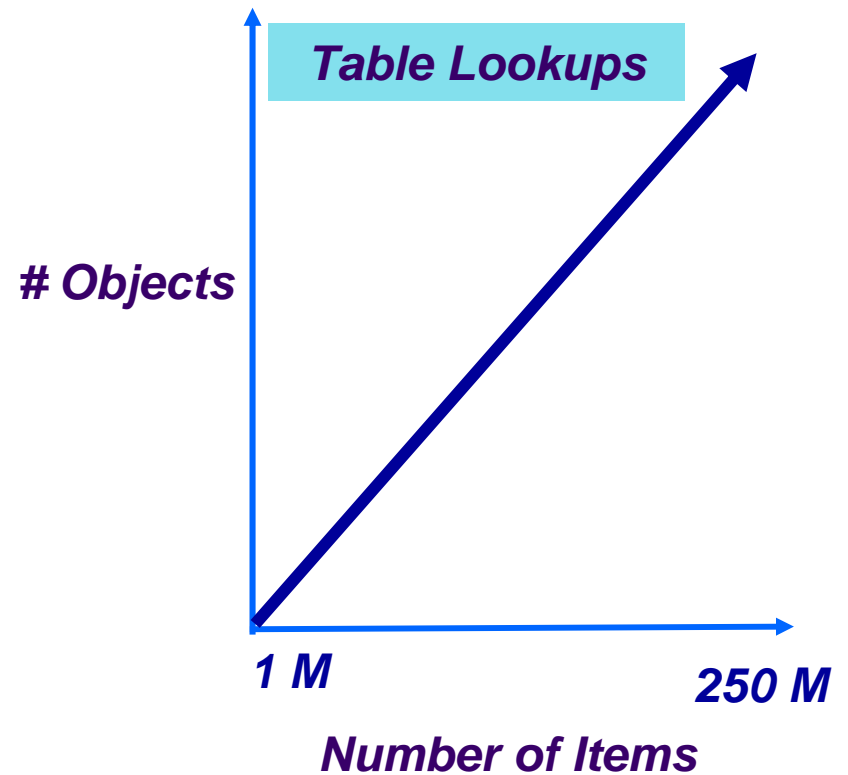
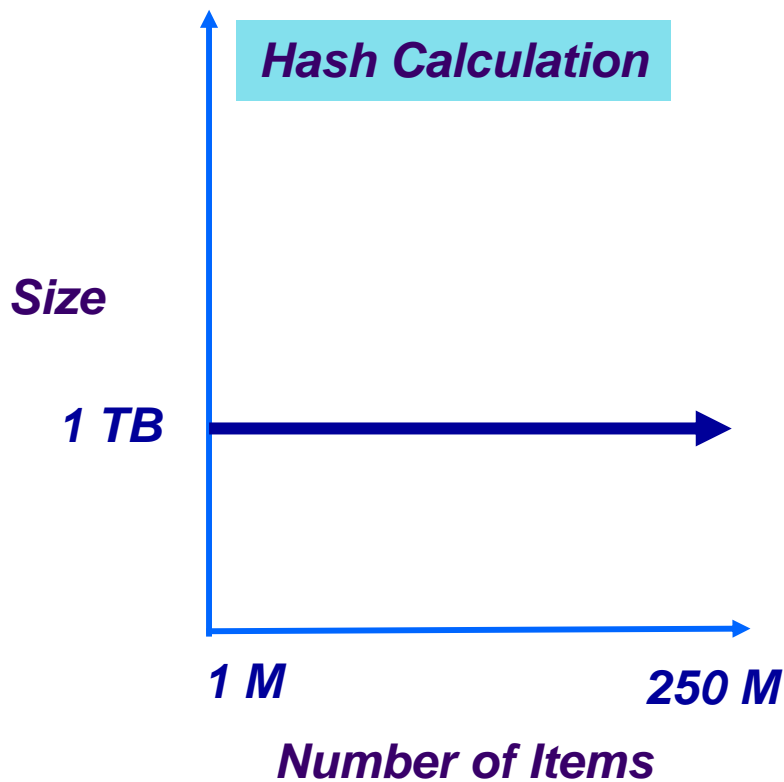
- Analysis is performed of content in the modified portion of a file looking for the common bit patterns from the original file
- When these are found the common blocks can be found across both files even though bit patterns have shifted

# How Many Objects?

- There is object handling overhead
- The two biggest impacts to performance are table lookups and hash calculation
- Hash calculation impact on CPU is linear with the size of the content being hashed
- Table lookups are linear with number of items being hashed
- How large do I want my SIS repository to be?
- Do I have enough compute power and can I tolerate the latency?

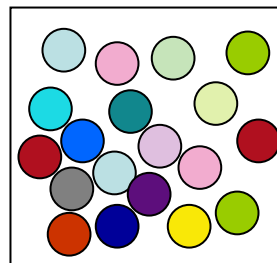
# An Example

**1M files of 1 MB in size = 1 TB**  
**250 M blocks of 4 kB in size = 1 TB**



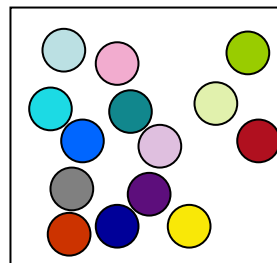
# Scalability and hashing

098f6bcd4621d373cade4e832627b4f6
098f6bcd4621d373cade4e832627b4f6
b9780ed9c1631acd32c234bcccd44c3a
↓
5d41402abc4b2a76b9719d911017c592
5d41402abc4b2a76b9719d911017c592
8ff32489f92f33416694be8fdc2d4c22
9f863753d6778f8d0c686ab4de352ea9



- Objects are listed in a hash table
- Identical hashes identify candidates for SIS
- For any given system, there is a limit to scale and performance

# Scalability and hashing

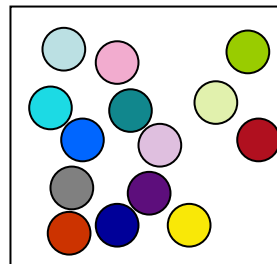


- Objects are listed in a hash table
- Identical hashes identify candidates for SIS
- For any given system, there is a limit to scale and performance

# Scalability and hashing

a 9

lookup table



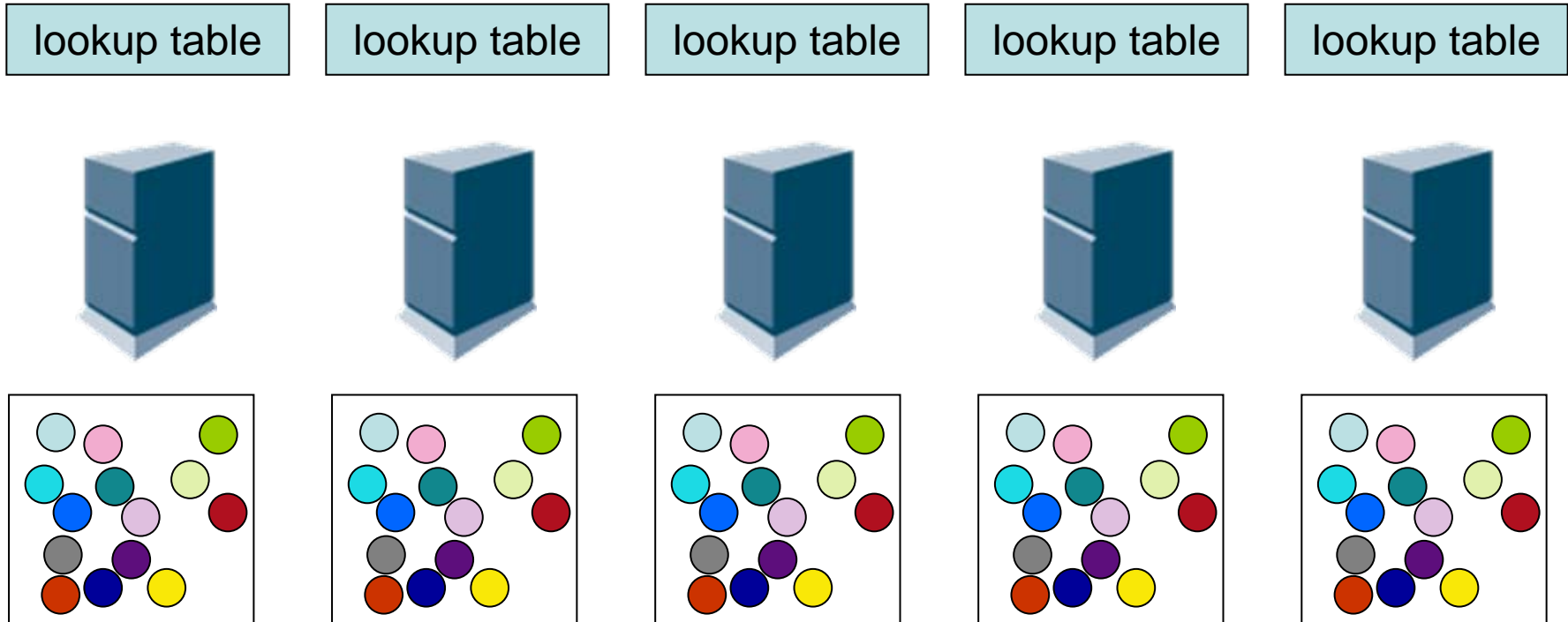
# One Table, Multiple Systems

- Any one physical system has a limit to object count and content quantity
- But a hash table can span many systems
  - ◆ The size of the table has very impact on the time to perform a lookup operation
  - ◆ This increases the likelihood of finding common content
- Systems can be co-located in a common cluster
  - ◆ Common in archiving
- Or distributed across an enterprise
  - ◆ Common in backup systems
- Much greater efficiencies result

# One Table, Multiple Systems

a

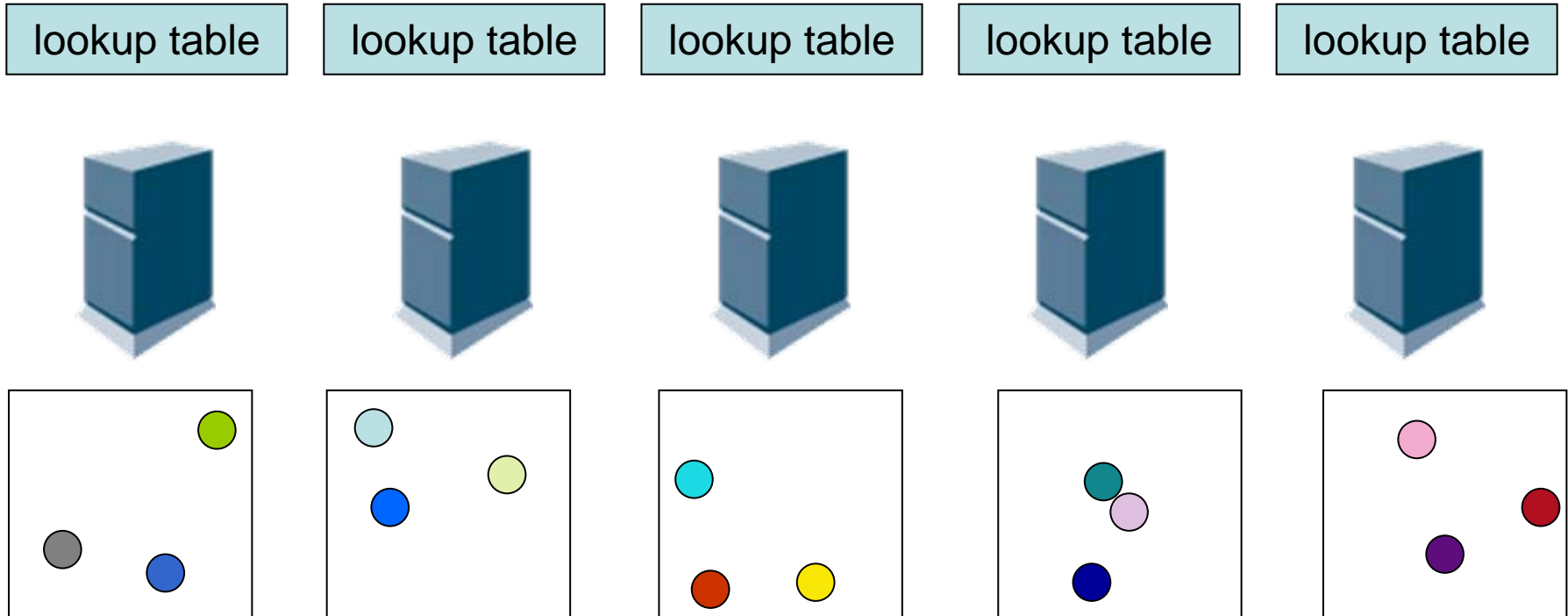
9



# One Table, Multiple Systems

a

9



# Where Do I Use SIS?

- (Easy Answer) Wherever identical content is being stored
- (Real Answer) Where it makes sense
- Considerations
  - ◆ Can my system tolerate the CPU overhead involved?
  - ◆ Will the content I am storing get a benefit from SIS?
    - › Test a sample of your data
  - ◆ What type of SIS should I implement?
  - ◆ How reliable is the SIS vendor?
    - › Remember, if you lose the lookup tables you lose your data!

# Other Considerations

- ▶ Some vendors may use more than one SIS process
  - ◆ Backup vendors
    - › May use a common object model, common fixed block, and identified common variable block model across all systems
  - ◆ Is the system in band?
  - ◆ Does it work with existing backup systems
  - ◆ Client based computing may detect common content before it ever goes across the wire
- ▶ How do I test for common content?
  - ◆ Locality of common content varies widely
    - › The same file may be stored many times by a user or a group of users located on the same system
    - › The same file may be stored by many users on many systems across an enterprise



# Guarantee of Authenticity

- If a file is going to be recreated from its constituent components (whether fixed blocks or variable length blocks) then it is a good idea to have a hash of the original file itself to measure that this file is unchanged
- If a file is stored in a common object model system then it already has a hash to offer a guarantee of authenticity

# Where Do I Use SIS?

## Production Systems

- Almost never a good fit for SIS
- Content is always changing, large latency and CPU hit

## Archive (or Tier 2)

- Good fit for Common Object or Common Fixed Block
- Strong future for Common Variable Block

## Backup/Restore

- Good fit for snapshots as long as it is on disk
- Good fit for Common Variable Block and Common Object as long as it is on disk
- Different topologies offered

# References

- Handbook of Applied Cryptography, Menezes, Oorschot, and Vanstone
- <http://www.cryptography.com/cnews/hash.html>
- <http://storageconference.org/2003/papers/25-Hamilton-Design.pdf>
- <http://www.microsoft.com/downloads/details.aspx?FamilyID=99f8ee58-4faf-4951-ba84-7237b5c639b5&DisplayLang=en>

- Please send any questions or comments on this presentation to SNIA: [trackdatamgmt@snia.org](mailto:trackdatamgmt@snia.org)

**Many thanks to the following individuals  
for their contributions to this tutorial.**

*SNIA Education Committee*

**Michael Fahey**

**Rob Mason**

**Jack Orenstein**

**Dave Pinkney**

**John Rousseau**

**Andres Rodriguez**

**Rich Rogers**

**Russ Neufeld**

**Ben Bernhard**

**Eric Hibbard**