



COM Verification

Alan G. Yoder, Ph.D.
SNIA Technical Council
Huawei Technologies, LLC



Outline

- COM overview
- How they work
- Verifying the COMs

COM overview

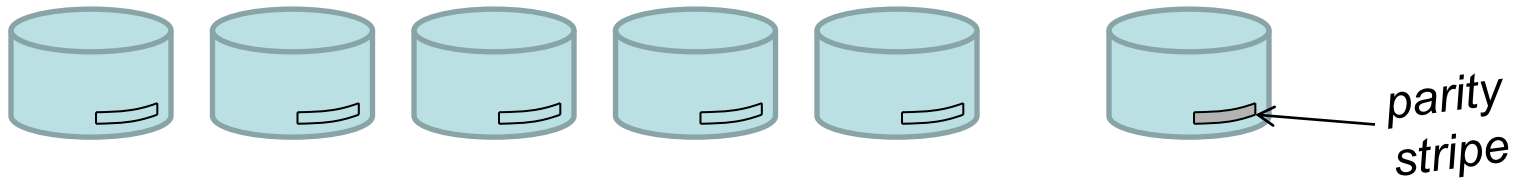
- COM = Capacity Optimization Method
- Basic idea: figure out a way to store more data in less space
 - ◆ energy use is theoretically proportional to space used
- Reference point is RAID 1 (mirroring)
 - ◆ this has been best practice in enterprise data centers through about 2005

Currently acknowledged COMs

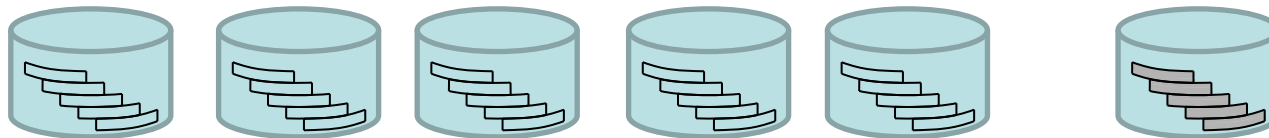
- Parity RAID
- Thin Provisioning
- Read-only Delta Snapshots
- Writeable Delta Snapshots
- Data Deduplication
- Compression

How they work – Parity RAID

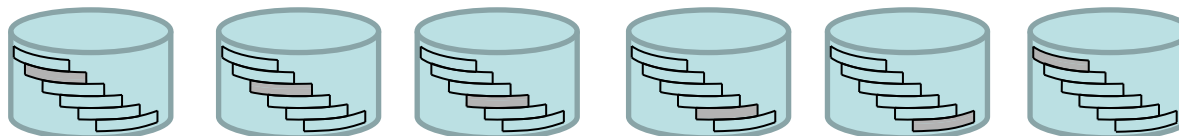
➤ Parity RAID = striping + parity



➤ RAID 4 – NetApp – dedicated parity drive



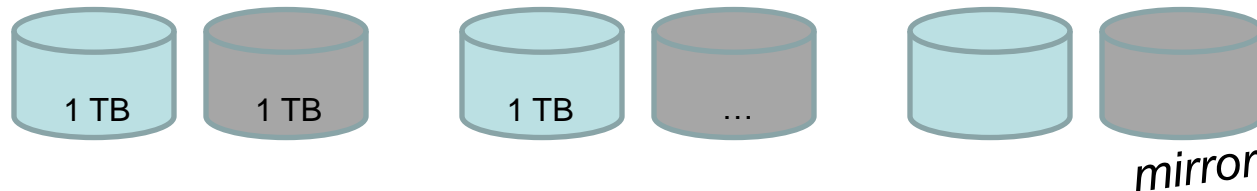
➤ RAID 5 – Other vendors – parity striped across RAID set



*"virtual"
parity drive*

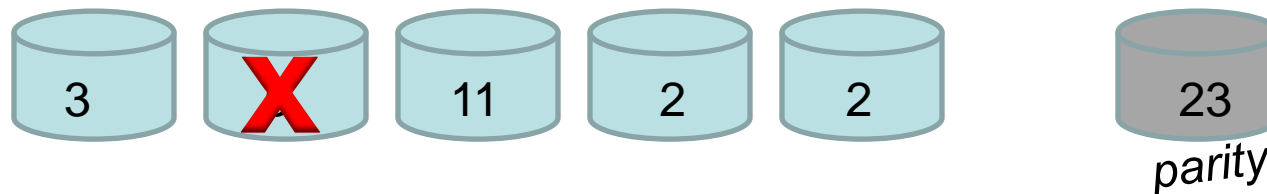
How they work – Parity RAID cont.

- RAID 1 (mirroring) – the gold standard for enterprise data protection



3 TB
addressable
50% overhead

- Parity RAID (level 4)



5 TB
addressable
17% overhead

- Uses XOR – works like ordinary addition

$$3 + 5 + 11 + 2 + 2 = 23$$

$$23 - 3 - 11 - 2 - 2 = 5$$

**In this example:
67% reduction in overhead**

How they work – Parity RAID cont.

➤ Many types of RAID

RAID 0	simple striping	not really RAID	
RAID 1	mirroring	NOT parity RAID	
RAID 4	parity on a separate drive	okay for ES	<i>only good for smaller drives</i>
RAID 5	parity striped across drives	okay for ES	
RAID 6	double parity	okay for ES	<i>protection against failures during RAID reconstruct</i>
“erasure codes”	non-XOR parity	okay for ES	
distributed parity	multiple parity, widely distributed ¹	okay for ES	
RAID 0+1, 1+0, RAID 10	striping+mirroring	NOT parity RAID	
replication	e.g. Hadoop, AWS	NOT parity RAID	

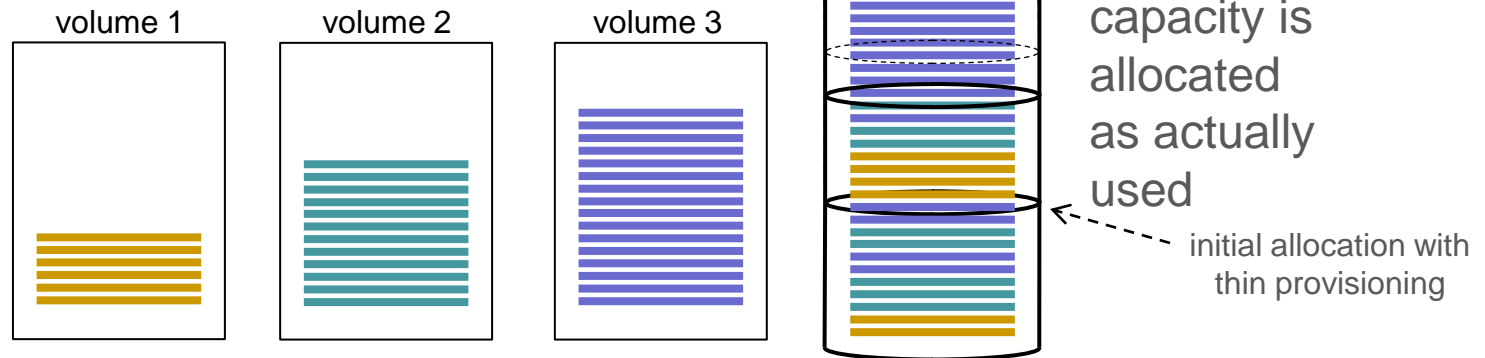
How they work—Thin Provisioning

➤ Traditional: pre-allocation of storage space

- ◆ Storage is dedicated in advance of application usage
- ◆ Much wastage due to multiple levels of over-provisioning

➤ Thin provisioning: allocation on demand

- ◆ Admins track total storage used by all users of the system and expand as needed
- ◆ More of a storage utility model



How they work—Delta Snapshots

➤ Snapshot

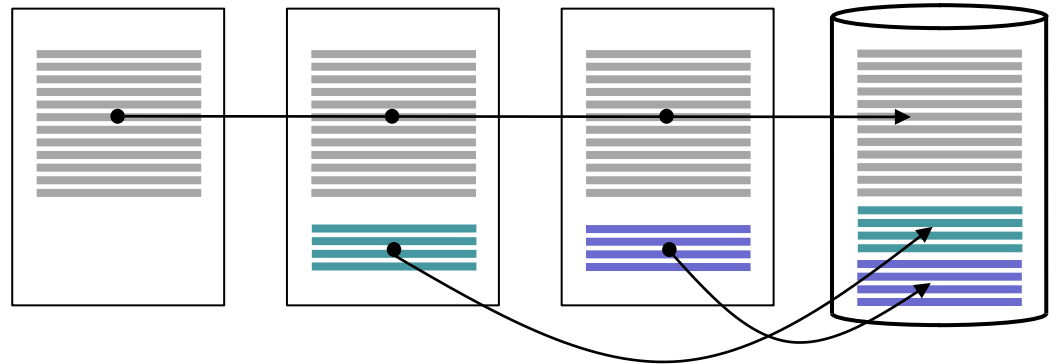
- ◆ A Point-in-time (PIT) copy of some data
- ◆ Usually at a volume or filesystem level

➤ Traditional method

- ◆ Full copy
- ◆ Lock volume, suspend or log writes, make copy, write log, unlock

➤ Delta method

- ◆ Copy on write
- ◆ Snapshots share blocks



How they work—Delta Snapshots cont.

➤ Read only

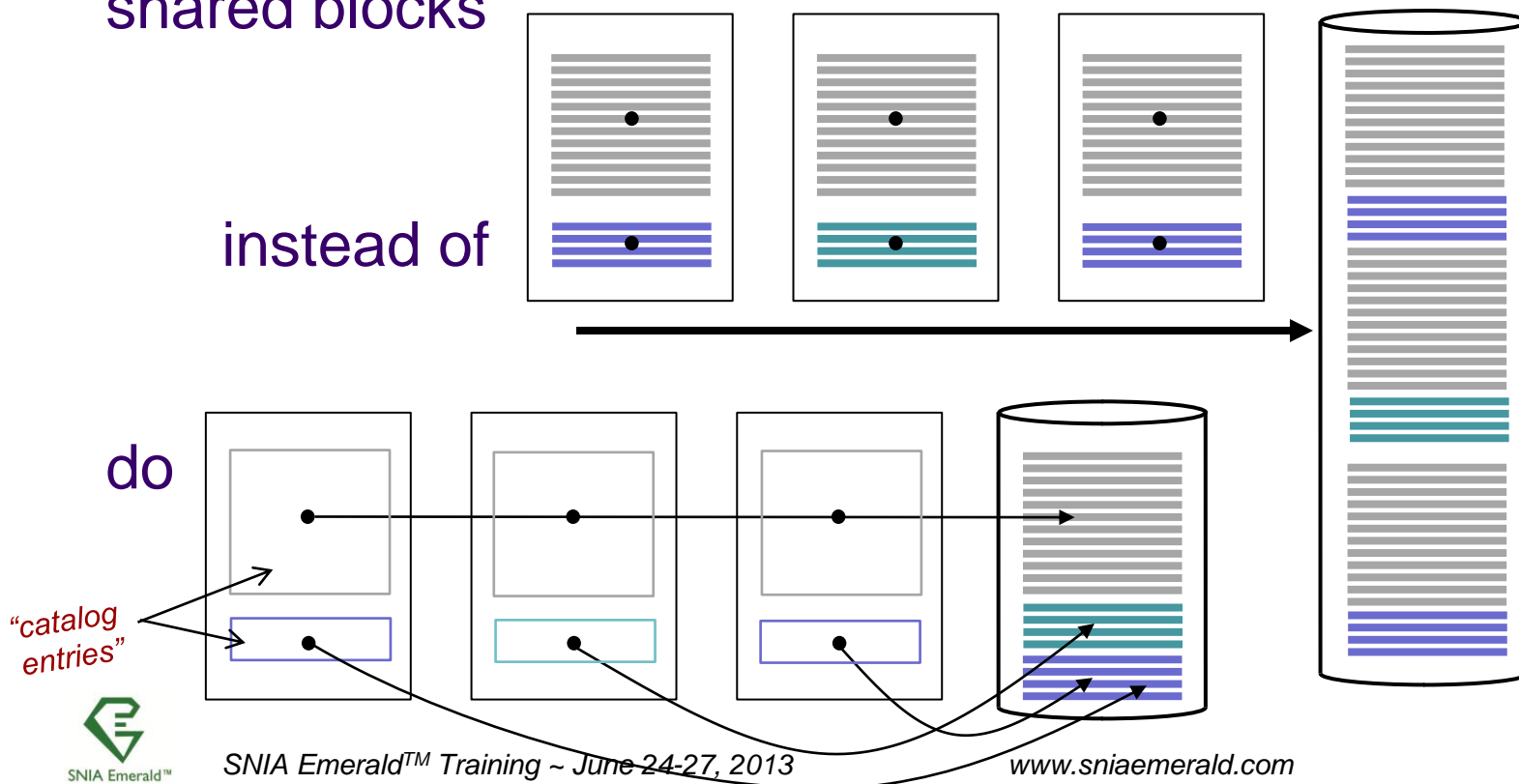
- ◆ Live copy continues as before
- ◆ PIT copy cannot be written
- ◆ Useful for backups

➤ Writeable

- ◆ Live copy continues as before
- ◆ PIT copy can be written
- ◆ Useful for “what if” scenarios, test runs on live data
- ◆ Example: NetApp “flexclones”

How they work—Data Deduplication

- A.K.A. “dedup”
- Basic idea: replace duplicate blocks with pointers to shared blocks



How they work—Data Deduplication

➤ Two fundamental types

- ◆ “Inline” – Dedup at wire speed, before writing to disk
 - Usually used for streaming backup systems
 - Note: streaming backup systems are not “online” systems in the SNIA taxonomy, so are not covered by the ES spec
- ◆ “Post process” – Dedup performed after initial write to non-volatile media

➤ Either type is acceptable

- ◆ So long as it’s an Online system by the SNIA taxonomy

How they work—Data Deduplication

➤ Many variations

- ◆ File or object level
 - › Coarsest grained, least overhead
- ◆ Block level
 - › Granularity at 4K or larger
- ◆ Variable-size
 - › Finer granularity, but more overhead
- ◆ Yada yada
 - › Whatever

➤ All are acceptable for ES on online systems

- ◆ Must meet performance criterion
 - › It's not a high bar – most systems will pass

How they work—Compression

- Old and venerable technology
- Well understood
- Zip, pkzip, WinRAR, others
- Finer grained than dedup
 - ◆ Byte level dedup inside typically a 32K sliding window
- Difficult, but possible, to combine with dedup
- Which is “better” depends on dataset

Validating the COMs

- Step 1: select COM(s) to validate
- Step 2: generate data set if required
- Step 3: validate the COM(s)

Validating the COMs -- selection

➤ Parity RAID is REQUIRED

- ◆ A bar to entry, not a COM to validate
- ◆ Must be a customer-selectable option
- ◆ No validation requirement—check the equipment data sheet and cross-check against the list of acceptable RAID types

➤ One COM is required for Online 3 and 4 systems

- ◆ Pick the easiest one to do first
 - › Thin provisioning and delta snapshots are good candidates
 - › Dedup and compression are a bit more work
- ◆ Validate all that are available for extra credit

Validating the COMs – generating the data sets

- Download code from
<https://sourceforge.net/projects/sniadeduptest/>
- Compile code with gcc or Visual Studio
 - ◆ NOTE: this code is relatively unbaked at this point – bugs will be fixed as we uncover them
- Run the executable on the host to generate the datasets
 - ◆ 5GB free space required
- Generate datasets
 - ◆ irreducible.dat
 - ◆ compressible.dat
 - ◆ dedupable.dat

Verifying the COM– Parity RAID

➤ Parity RAID is REQUIRED for ENERGY STAR

- ◆ Must be at least a customer-selectable option
 - Preferred: ships with it turned on by default
- ◆ Check the equipment data sheet

Verifying the COM– Thin Provisioning

1. Determine the amount of free space FS_{tot} available on the SUT as seen by the storage admin
 - ◆ Satisfy yourself that the admin is using a standard tool for this
2. On the SUT, allocate a container 15GB in size
 - ◆ The admin will do this
 - ◆ 15GB is the “nominal” size
 - › The SUT should actually allocate considerably less than this
 - › Check by determining the amount of free space using the same tool as in step 1
3. Mount the container on the host test machine
 - ◆ If the SUT supports CIFS, that will be easiest
 - ◆ Otherwise use iSCSI
 - ◆ *The host system should indicate that it is a 15GB container*

Verifying the COM– Thin Provisioning

4. On the host, write the irreducible data set to the container
5. After a suitable amount of time, not to exceed 1 hour
 - ◆ determine the amount of free space FS_{eot} available on the SUT as seen by the storage admin
 - ◆ Use the same free space determination as in step 1
6. Calculate the amount of formatted capacity I_{com} used by the test
 - ◆ $I_{com} = FS_{sot} - FS_{eot} \quad \%$. Note: $S_{ds} = \text{sizeof}(\text{dedupable.dat})$
 - ◆ If I_{com} is less than 3GB, then the SUT passes the test.

Verifying the COM—RO Delta SSs

➤ 7.4.5.4.1 Heuristic 1: Readonly delta snapshots

- The method varies according to where the SUT places snapshots.
 - ◆ Follow the appropriate procedure to set up and take the snapshot
 - ◆ Correct sequence is step1 then step 3, or step 2 then step 3

Verifying the COM—RO Delta SSs

1. For a SUT which places snapshots in separate containers:
 - a. On the SUT, create two containers, each 15GB in size.
 - a. One in the live data container/partition, one in the snapshot container
 - b. Mount the first container on a host, via any chosen protocol.
 - a. Since it's block storage, either iSCSI or FC
 - c. Determine the amount of free space FS_{sot} available on the SUT as seen by the storage administrator.
 - a. Satisfy yourself that the admin is using a standard tool for this
 - d. Write the irreducible data set to the first container.
 - e. Perform a read only delta snapshot of the first container and expose it through the second container, disabling any optional background copying mechanism. As an example, the snapshot of lun1 may be exposed as lun2.
 - f. Perform whatever steps are necessary to mount the second container as a file system. Open a small file on this file system (i.e. one of the files in the irreducible data set and read some data from it, and close the file. Confirm that the file has been successfully read.

Verifying the COM—RO Delta SSs

2. For a SUT which places snapshots on the originating container
 - a. On the SUT, create a container of 15GB in size.
 - b. Mount the container on the host via any chosen protocol.
 - a. Since it's block storage, either iSCSI or FC
 - c. Determine the amount of free space FS_{sot} available on the SUT as seen by the storage administrator.
 - a. Satisfy yourself that the admin is using a standard tool for this
 - d. Write the irreducible data set to the container.
 - e. Perform a read-only snapshot of the container, disabling any optional background copying mechanism.
 - a. The admin will do this
 - f. Perform whatever steps are necessary to mount the container as a file system. Open a file in the snapshot (i.e. one of the files in the irreducible data set), read some data from it, and close the file. Confirm that the file portion has been successfully read.

Verifying the COM—RO Delta SSs

➤ 3. Finally (after performing (1) or (2))


- a. Determine the amount of free space, FS_{eot} , available on the container containing the snapshot, as seen by the storage administrator.
- b. Calculate the space required for the snapshot
 - a. $I_{com} = FS_{sot} - FS_{eot}$
- c. If I_{com} is less than 2.5 GB then the SUT passes the test.

Verifying the COM—R/W Delta SSs

➤ 7.4.5.4.2 Heuristic 2: Writeable delta snapshots

- The method varies according to where the SUT places snapshots.
 - ◆ Follow the appropriate procedure to set up and take the snapshot
 - ◆ Correct sequence is step1 then step 3, or step 2 then step 3

Verifying the COM—R/W Delta SSs

- 
- 1. For a SUT which places snapshots in separate containers:
 - a. On the SUT, create two containers, each 15GB in size.
 - a. One in the live data container/partition, one in the snapshot container
 - b. Mount the first container on a host, via any chosen protocol.
 - a. Since it's block storage, either iSCSI or FC
 - c. Determine the amount of free space FS_{sot} available on the SUT as seen by the storage administrator.
 - a. Satisfy yourself that the admin is using a standard tool for this
 - d. Write the irreducible data set to the first container.
 - e. Perform a writeable snapshot of the first container and expose it through the second container, disabling any optional background copying mechanism. As an example, the snapshot of lun1 may be exposed as lun2.
 - f. Perform whatever steps are necessary to mount the second container as a file system. Open a small file on this file system (i.e. one of the files in the irreducible data set), write a few characters to it, and close the file. Confirm that the file has been successfully written with its new contents.

Verifying the COM—R/W Delta SSs

2. For a SUT which places snapshots on the originating container

- a. On the SUT, create a container of 15GB in size.
- b. Mount the container on the host via any chosen protocol.
 - Since it's block storage, either iSCSI or FC
- c. Determine the amount of free space FS_{tot} available on the SUT as seen by the storage administrator.
 - Satisfy yourself that the admin is using a standard tool for this
- d. Write the irreducible data set to the container.
- e. Perform a read-only snapshot of the container, disabling any optional background copying mechanism.
 - The admin will do this
- f. Perform whatever steps are necessary to mount the container as a file system. Open a file in the snapshot (i.e. one of the files in the irreducible data set), write a few characters to it, and close the file. Confirm that the file has been successfully written.

Verifying the COM—R/W Delta SSs

3. Finally (after performing (1) or (2))

- a. Determine the amount of free space, FS_{eot} , available on the container containing the snapshot, as seen by the storage administrator.
- b. Calculate the space required for the snapshot
 - a. $I_{com} = FS_{sot} - FS_{eot}$
- c. If I_{com} is less than 2.5 GB and the small file was successfully written onto the read-only delta snapshot destination, then the SUT passes the test.

Verifying the COM—Dedup

- a. On the SUT, create a container 15GB in size.
- b. For a SUT exporting block storage, perform whatever steps are necessary to make the container visible on the host from which tests are being run, and create and mount a local file system on that container.
- c. For a SUT exporting file storage, mount the container on the host via any chosen file protocol such as NFS or CIFS.
- d. Determine the amount of free space FS_{sot} available on the container as seen by the storage administrator.
- e. Write the deduplication-reducible data set (dedupable.dat) to the container.
- f. Wait a suitable amount of time as specified by the TEST SPONSOR for non-inline deduplication processes to have completed.
- g. Determine the amount of free space FS_{eot} available on the container as seen by the storage administrator.
- h. Calculate the amount of formatted capacity saved by data deduplication
 - ♦ $I_{com} = 1 - [(FS_{eot} - FS_{sot}) / S_{ds}] * 100\%$. Note: $S_{ds} = \text{sizeof}(\text{dedupable.dat})$
- i. If I_{com} is greater than 10%, then the SUT passes the test.

Verifying the COM—Compression

- a. On the SUT, create a container 15GB in size.
- b. For a SUT exporting block storage, perform whatever steps are necessary to make the first container visible on the host from which tests are being run, and create and mount a local filesystem on that container.
- c. For a SUT exporting file storage, mount the container on the host via any chosen file protocol such as NFS or CIFS.
- d. Determine the amount of free space FS_{sot} available on the container as seen by the storage administrator.
- e. Write the compression-reducible data set (compressible.dat) to the container.
- f. Wait a suitable amount of time as specified by the TEST SPONSOR for non-inline compression processes to have completed.
- g. Determine the amount of free space FS_{eot} available on the container as seen by the storage administrator.
- h. Calculate the amount of formatted capacity saved by compression
 - ♦ $I_{com} = 1 - [(FS_{eot} - FS_{sot}) / S_{ds}] * 100\%$. %. Note: S_{ds} = sizeof(dedupable.dat)
- i. If I_{com} is greater than 10%, then the SUT passes the test.

Proposed new COM (info only)

➤ Auto-tiering

- ◆ A technology that automatically moves cold data to fat, slow (more energy efficient) drives
- The hot banding test already rewards auto-tiering
- No final decision on whether to classify it as a COM
 - ◆ How to measure it?

Q & A

