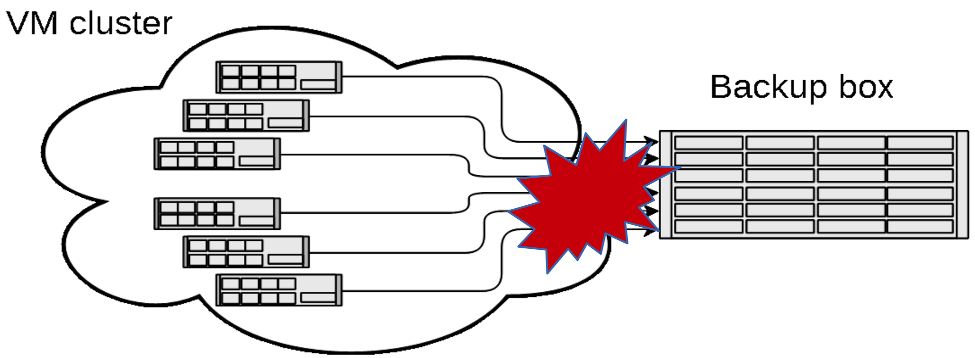
Low-Profile Source-side Deduplication for Virtual Machine Backup

Daniel Agun, Tao Yang
University of California at Santa Barbara
Wei Zhang
Pure Storage

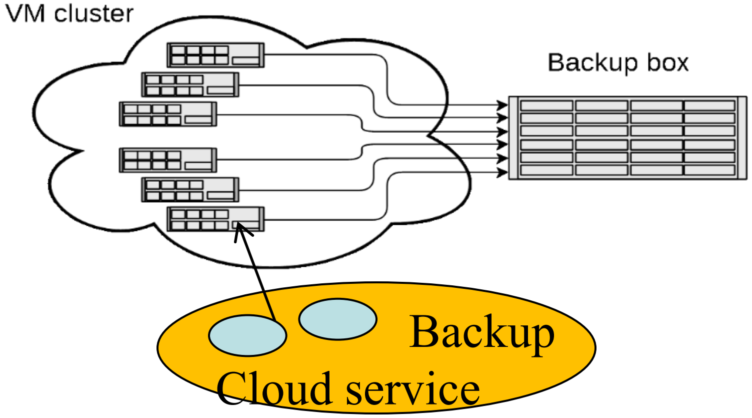
Cloud Platform and VM Snapshot Backup

- Public and private IaaS cloud systems have become industry standard
- Frequent virtual machine snapshot backups improves system reliability
- Backup traffic of VM snapshots with limited source-side deduplication is huge.
  - 100,000VMs with 76% dirty bit detection still requires ~1 petabyte of networking with 40GB per VM snapshot



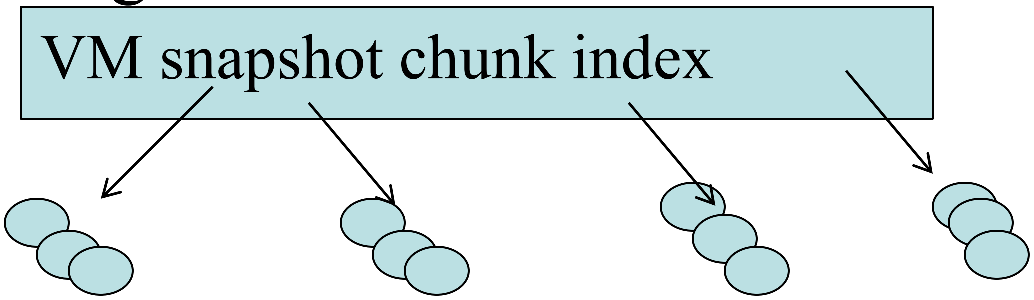
Objective: Aggressive Source-Side Deduplication With low-profile computing

- Backup data daily for tens of thousand VMs within a few hours each day.
- Minimize network traffic via aggressive source-side deduplication
  - State-of-art deduplication algorithms are memory/compute-intensive
- Resource friendly – small memory footprint and CPU usage, minimum impact to primaryservices



Strategies for Scalable/Low-cost Aggressive Source-Side Deduplication

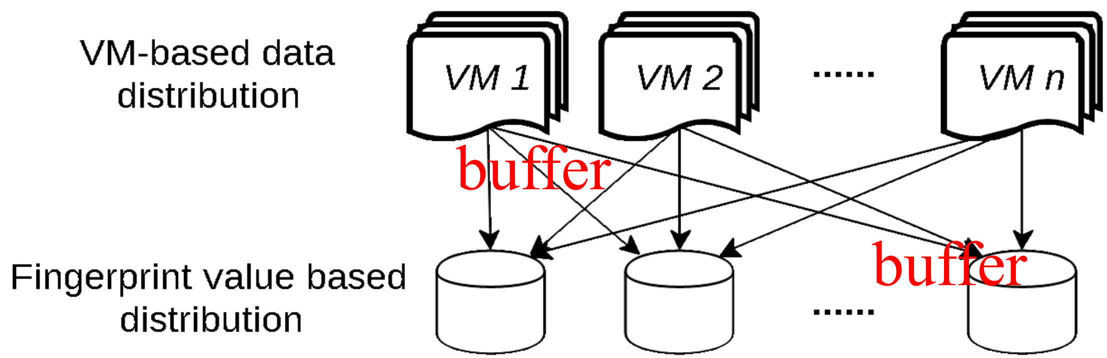
- Focus on popular data chunks shared among snapshots
  - Zipf distribution. Top 2-4% of most popular items (plus inner-VM dedup) accomplishes ~98% deduplication efficiency.
- Cluster-based deduplication
  - Distribute VM chunk signatures to cluster machines



  - Minimize job completion time instead of individual chunk backup time.
- Approximated snapshot deletion

Low-cost source-side cluster-based deduplication

- Given a set of VMs to be backed up, find if their block signatures are duplicates of the existing snapshot blocks.

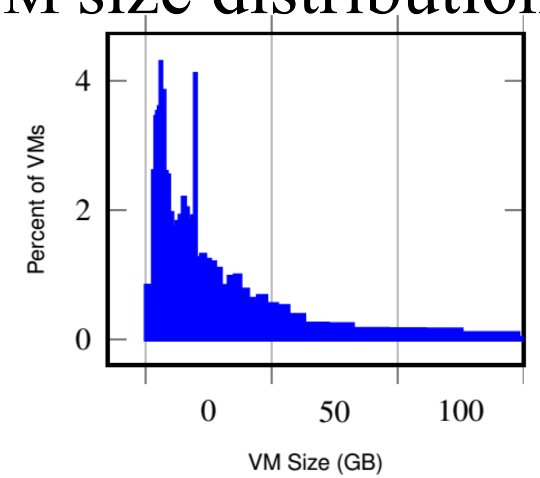


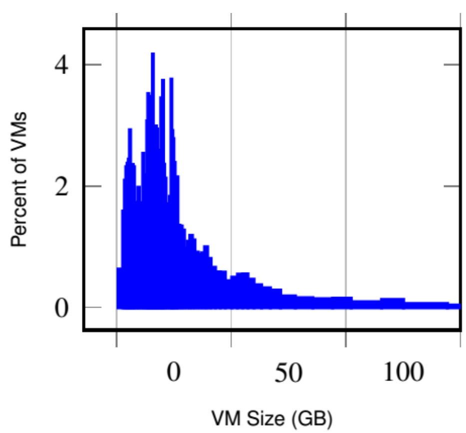
- Challenge in control buffer size during data shuffling
  - Complicated by uneven VM size distribution.

Example datasets from Alibaba.

Left: 4200 VMs with max/average VM size= 20.

Right 8000 VMs with max/avg=45



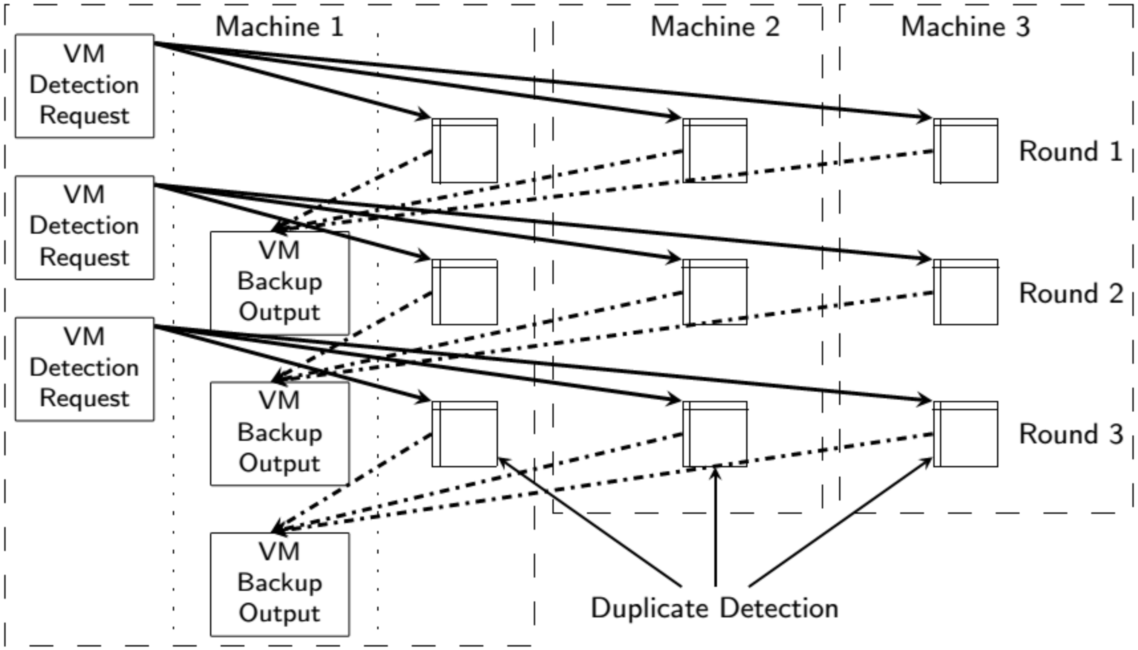


Multi-round Collaborative Deduplication

- Major stages of each duplicate detection round

Stage 1: Collect fingerprints in parallel
Stage 2: Detect duplicates in parallel
Stage 3: Perform actual VM backup in parallel

- k rounds
- k too small – more buffering needed
- k too large – more dedup overhead



Choose k so that buffer memory ≤ 100MB

How many rounds of backup batches?

- Estimate # of rounds *k* based on memory usage per node

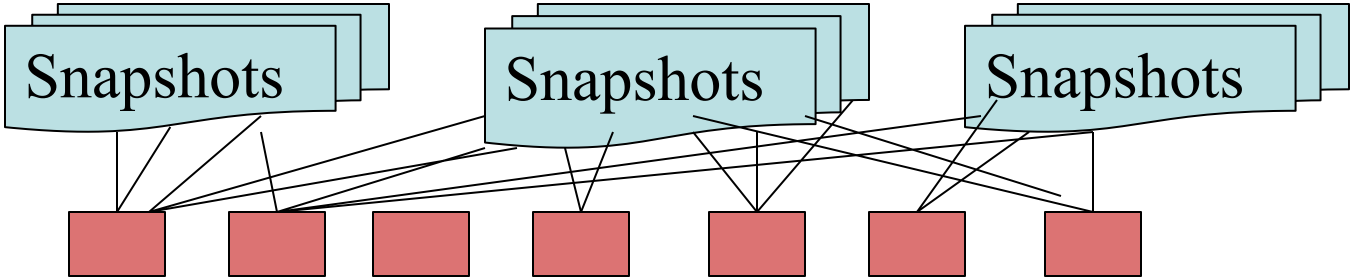
$$\frac{D*V}{r*p}\left[\frac{1}{k*q} + \frac{b\mu}{q} + \frac{1}{k}\right] \leq 100\text{MB}$$

- p is the number of physical machines.
- V is number of VMs hosted per machine
- q is the number of fingerprint partitions per machine
- D is size of modified data per VM
- μ is percentage of unique chunks among dirty data accumulated
- b is the average number of snapshot versions per VM.
- r is the ratio of chunk size over index entry size

p=100, V=25, D=8.8GB, μ =22.8%, b=10, r=136, q=400
→k=12. 9% of VMs is handled per batch

Low-cost Design for Snapshot Deletion

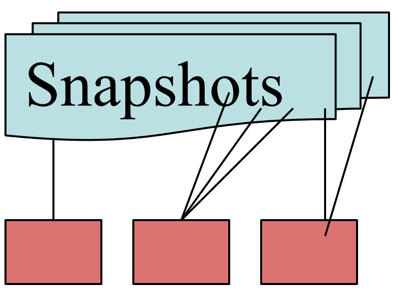
- **Snapshot deletion is as frequent as creation**
- **Identifying unused chunks with reference counting is costly**
  - Grouped Mark-and-sweep [Guo et. al , ATC'11]: A block can be deleted if its reference count is zero

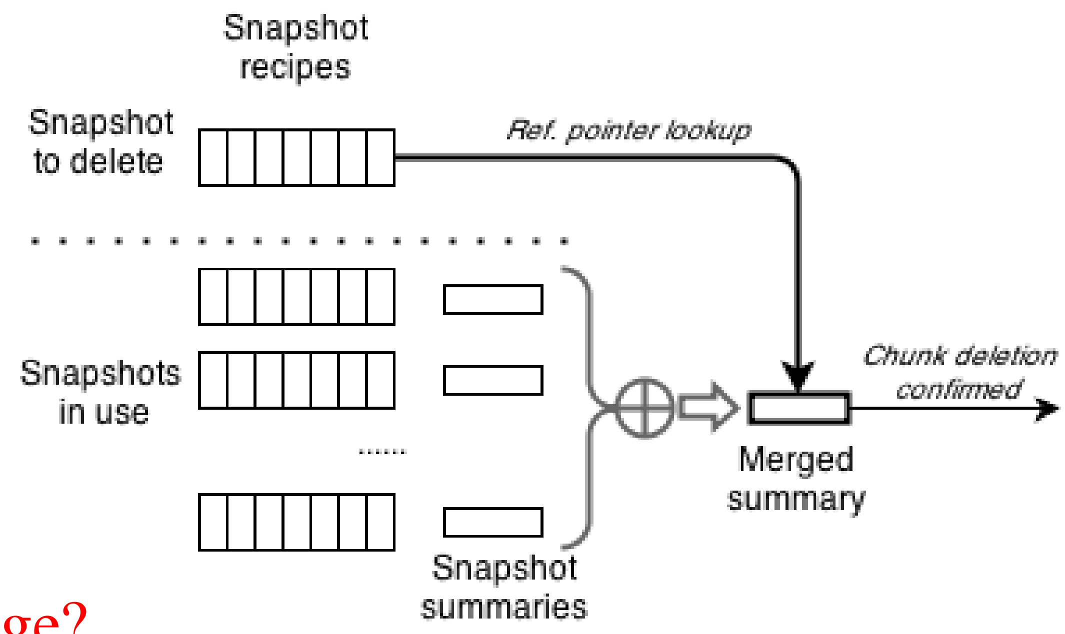


- **Our approximate approach**
  - Separate strategies for popular chunks (2-4%) and non-popular inner VM chunks.
  - Approximate deletion for VM-specific chunks with bloom filter

Approximate Deletion for VM-specific chunks

- **Summary vector to detect the usage of a chunk within a VM.**
  - Use bloom filter to summarize snapshots of VM
    - Summary vectors of live snapshots represent the chunks in use
  - Checking the existence of a chunk reference is fast
    - Tolerate small percentage of storage leak to allow fast deletion with approximation





How often to repair leakage?

Leakage Analysis: How Often to Repair?

- Periodically repair with mark-and-sweep to remove false negatives (those with 0 reference, but not removed)
- u : the initial size of a snapshot
- Δ*u: average VM change* between consecutive snapshots.
- Total chunks stored after h snapshots per VM:

$$U=u+(h-1)\,\Delta u$$

- Total leakage after R rounds: L=R $\varepsilon\Delta u$

ε is the misjudgement rate of bloom-filter summary vector

- How often to repair?

$$\frac{L}{U}=\frac{R\Delta u\varepsilon}{u+(h-1)\Delta u}>\tau\Longrightarrow R>\frac{\tau}{\varepsilon}\times\frac{u+(h-1)\Delta u}{\Delta u}$$

With daily backup, *Δu/u=2.5%, h=10, t=0.1,* → *R=19.6 days*

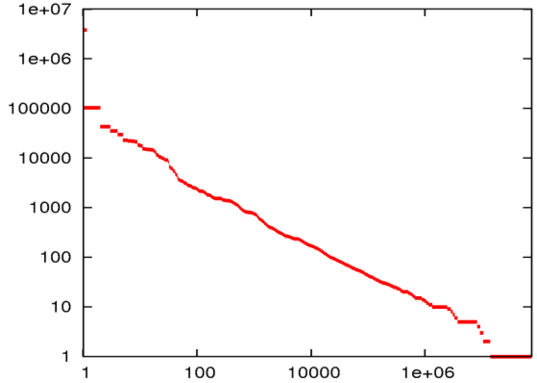
Evaluation

- Prototype implementation in C. Evaluated on a Linux cluster of 8-core 3.1 GHz, AMD FX-8120. 16GB memory
- **Test data from Alibaba Aliyuan cloud**
  - 41TB. 10 snapshots per VM for 2500 VMs
  - Segment size: 2MB. Avg. chunk size: 4KB.

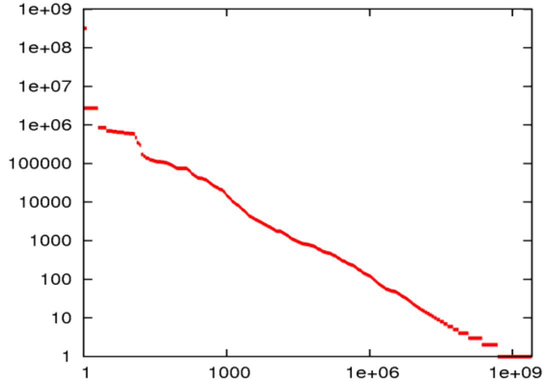
  SHA-1 fingerprint hash.
- **Evaluation objectives**
  - Compare resource usage of three source-side deduplication methods: 1) dirty bit. 2) Synchronous method. 2) Collaborative multi-round with k=12.
  - Impact of multi-round scheduling on backup job span
  - Compare exact deletion with approximate deletion on resource usage, time, and space leakage.

Data Characteristics

- **Each VM uses 40GB storage space on average**
- **OS and user data disks: each takes ~50% of space**
  - OS data : Debian, Ubuntu, Redhat, CentOS, Win2003 32bit, win2003 64 bit and win2008 64 bit.
- **Zipf-like distribution of VM OS/user data:**
  - frequency of any chunk is inversely proportional to its rank in the frequency table



(a) Data blocks from OS disks



(b) Data blocks from data disks

Resource Comparison

- Resource usage comparison per snapshot.
- Local disk IO and memory costs are per machine.
- Storage and network cost are for 100 physical machines after deduplication.

Algorithm	Mem (MB)	Local IO (GB)	Storage (GB)	Network (GB)
Dirty Bit	<10	220	22000	22000
Synchronous	40	453	4840	5500
Collaborative	90	491	4840	5500

Aggressive source-side deduplication incurs 4.55x less space and 4x less network traffic

Job time comparison

- Job span in hours (total time for backup of all VM snapshots)
- Average per-VM backup time
- Even VM size distribution vs skewed distribution with max/average size=20.

Hours	Job span	Backup time per VM (even)	Backup time per VM (skew)
Dirty Bit	1.25	0.05	0.05
Synchronous	50.40	2.75	50.40
Collaborative	2.36	0.23	0.23

Multiround collaborative processing with k=12
is 21x faster than synchronous method for job span.
1.88x slower than dirty bit method but still finishes in 2.36 hours.

Effectiveness of Approximate Deletion

- Processing time and per-machine memory usage of four deletion methods.
- # of machines: p= 50 and 100 while # of VMs per machine=25

	Time p=50 (hours)	Time p=100 (hours)	Memory (GB)
Mark&sweep	35.9	84.3	1.2–3
Grouped mark&sweep	18.6	43.6	1.2–3
Local w/o sum.	0.7	0.82	0.05 – 1.96
Approx. local	0.012	0.014	0.015
Leak repair	0.7	0.82	0.05 – 1.96

•Approximate deletion is 3114x faster than the grouped mark&sweep method.

•Leakage repair is 53x faster with 35% to 96% less memory usage

Contributions & Conclusions

- Scalable low-profile multi-round source-side deduplication for frequent VM snapshot backup.
- **For the tested dataset,**
  - Network cost: 4x and storage cost is reduced by 4.55x compared to a dirty-bit based method.
  - Multi-round deduplication is an order of magnitude faster than a synchronous scheme in dealing a skewed load.
  - Approximate snapshot deletion only requires 15MB per machine
    - 3114x faster than the grouped mark&sweep method.
    - Leakage repair is 53x faster with 35% to 96% less memory usage.

Thank You!

Questions?