Advanced Aerospike Features

Tim Faulkes
Director of Solutions Architecture
Aerospike

Caching in Aerospike

 All rights reserved. © 2019 Aerospike Inc

Caching in Aerospike

“Aerospike has no caching” – Not True!

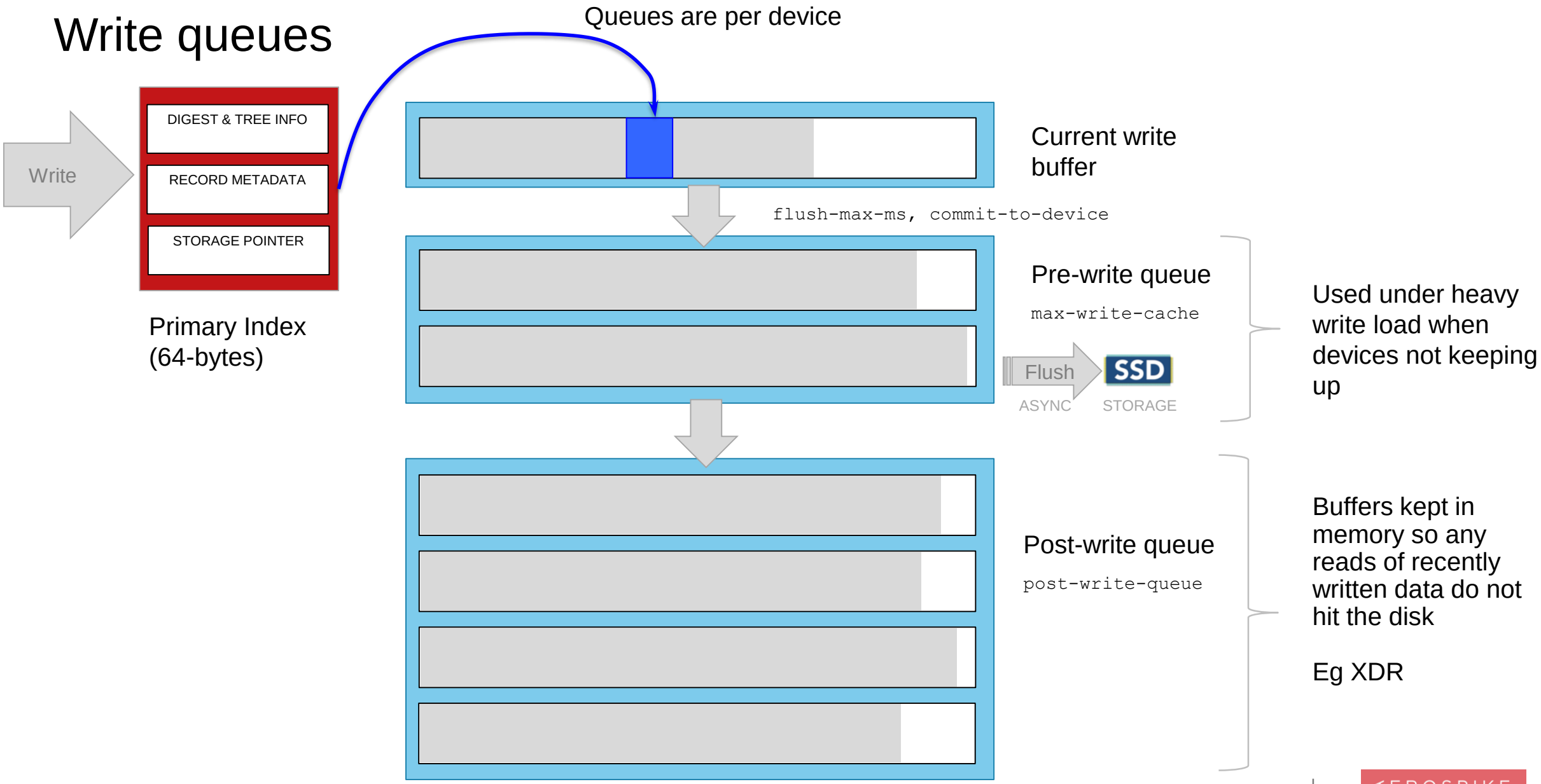
“Aerospike has no NEED FOR caching” – Often True!

Actually, there are several levels of caching in Aerospike

- **Pre-write queue**
- **Post-write queue**
- **Read cache**
- **Page caches + hardware caches**

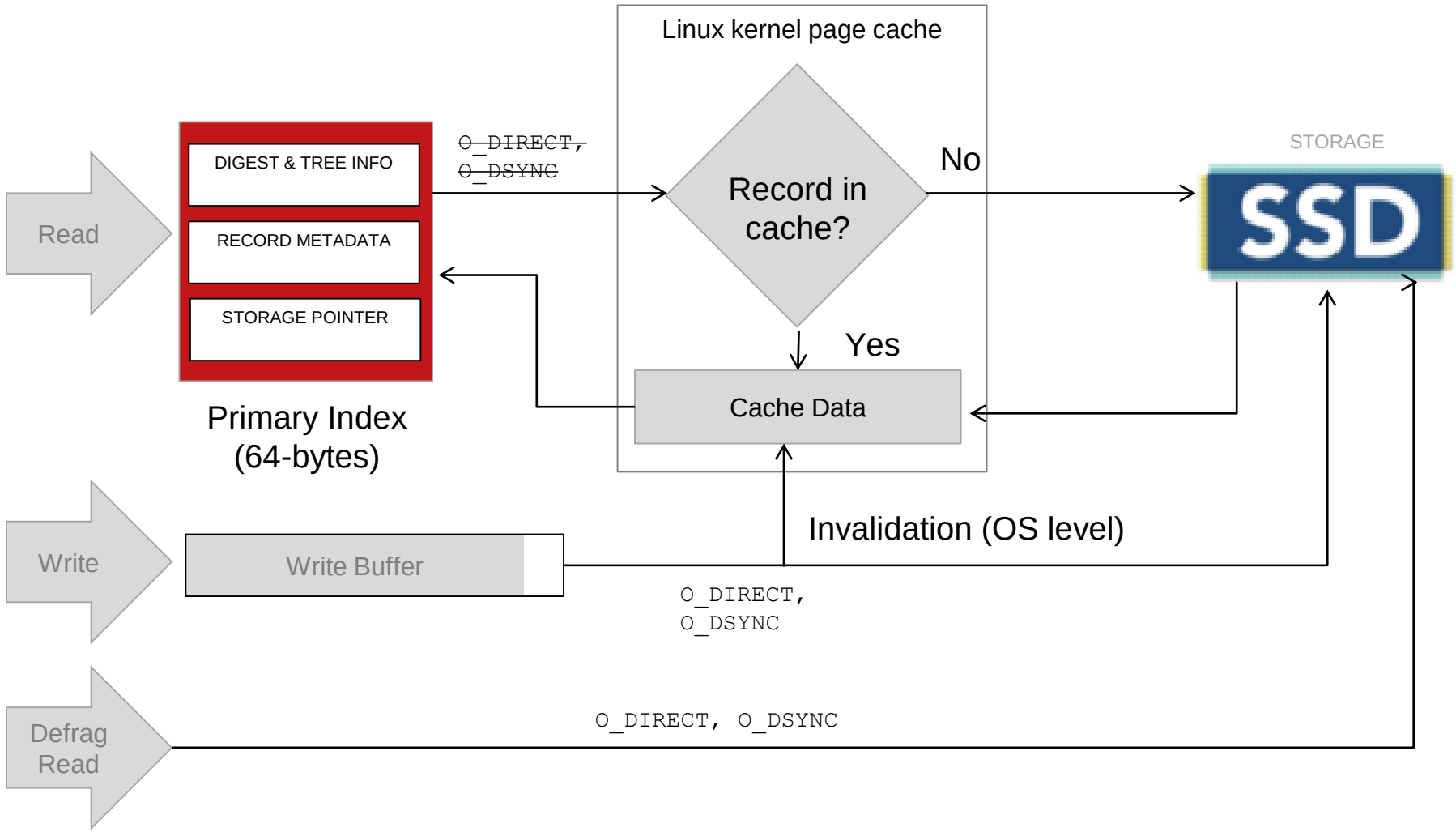
 All rights reserved. © 2019 Aerospike Inc

Write queues



 All rights reserved. © 2019 Aerospike Inc

Read cache (v 4.3.1+)



- By default, O_DIRECT & O_DSYNC are on for raw devices, off for files
- These can be turned on for files using `direct-files` flag
- `read-page-cache` turns O_DIRECT and O_DSYNC off for application reads

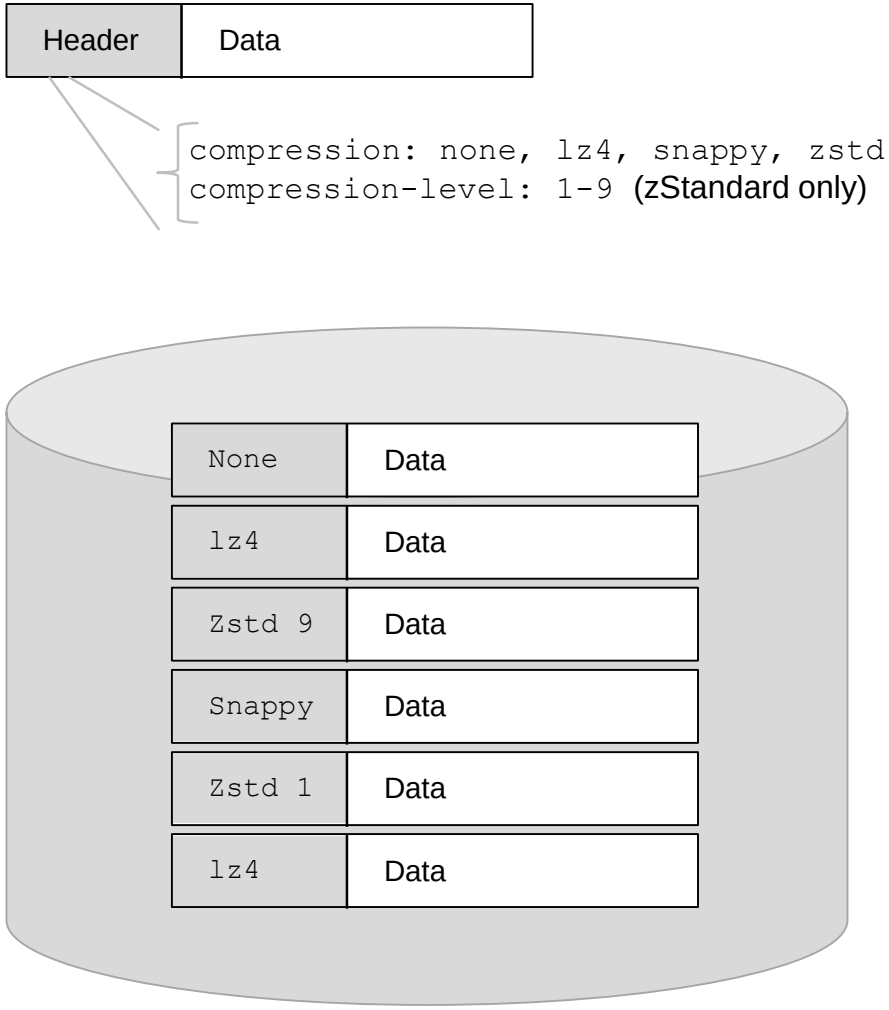
 All rights reserved. © 2019 Aerospike Inc

Compression at rest

 All rights reserved. © 2019 Aerospike Inc

Data Compression

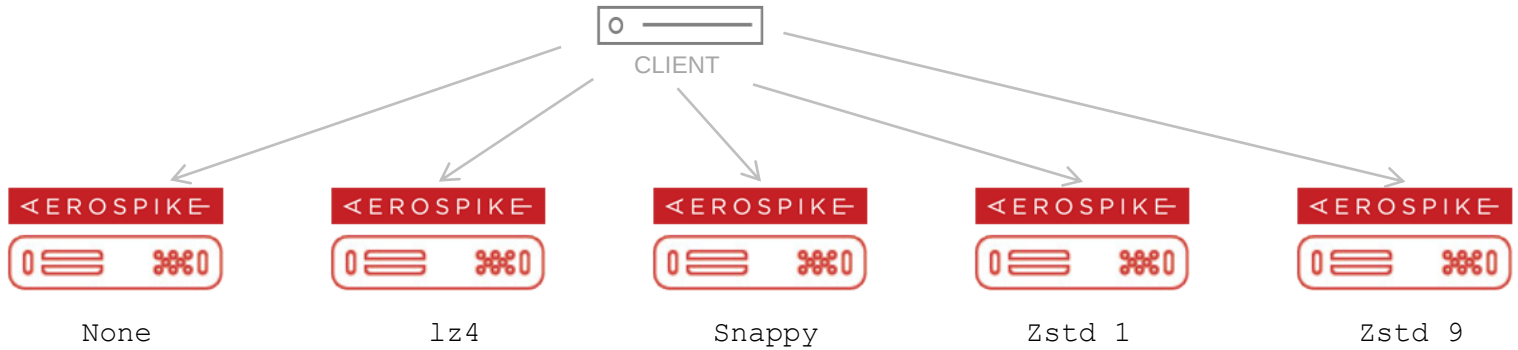
- Available in EE v 4.5.0+, requires separate license key
- Compression exchanges CPU for storage space
- Transparent to application: secondary indexes, CDT operations, etc continue to work.
- The compression scheme is now stored in the header of the record.
- If a compressed record is larger than the uncompressed one, uncompressed one is written.
- Different records on the same drive can have different compression schemes, whatever is active when the record is written.
- Both `compression` and `compression-level` are dynamic and specified in the `storage-engine` section
- New metric to see the compression ratio on disk: `device_compression_ratio`. This measures the compression of recently written records (100k – 1M)



 All rights reserved. © 2019 Aerospike Inc

Determining optimal compression

- Compression ratio is data specific – must test in your use case!
- Determine desired algorithms to test – don't forget to do `none` as the baseline!
- Set up cluster with N nodes and different compression algorithms on each of them, ideally in staging environment
- Write N x 1,000,000 records to the cluster
- Monitor CPU, compression ratio, latency, disk utilization.



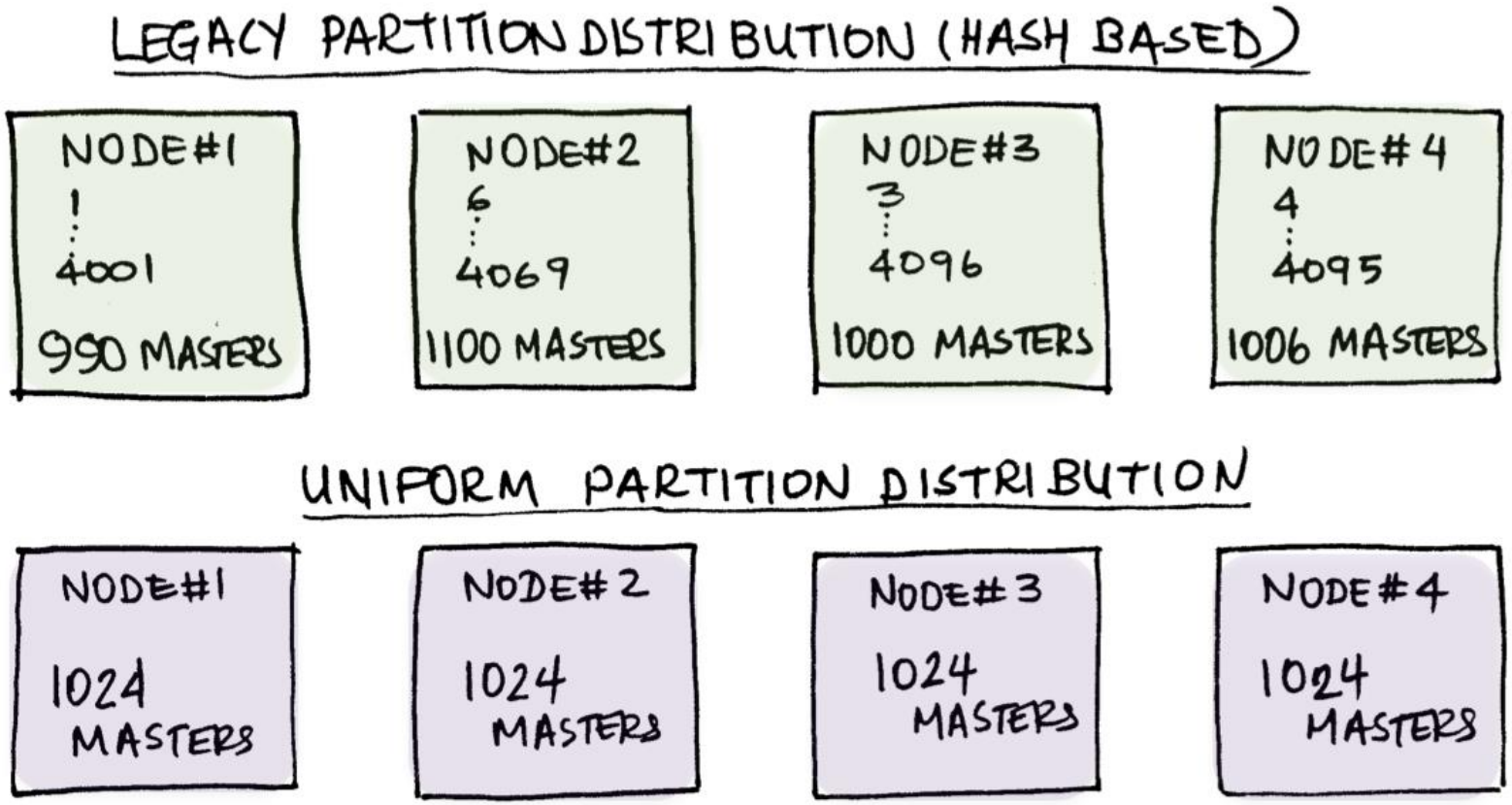
 All rights reserved. © 2019 Aerospike Inc

Uniform Balance

 All rights reserved. © 2019 Aerospike Inc

Uniform Distribution of Partitions across Cluster Nodes

Aerospike Server Version 4.3.0.10 introduced option to uniformly balance partition distribution across the nodes of a namespace.



 All rights reserved. © 2019 Aerospike Inc

Uniform Balancing of Partitions across Cluster Nodes

Configuration Option: prefer-uniform-balance *true*.

Uniform Balancing:

- **Can be invoked dynamically, on a per-namespace basis.**
- **Yields better capacity utilization by even distributing records across all nodes of the cluster.**
- **Results in**
  - better distribution of data
  - Better utilization of IOPs at each node if workload is uniform.

➢ **Recommended for Clusters greater than 10 nodes in size. Smaller clusters will also benefit.**

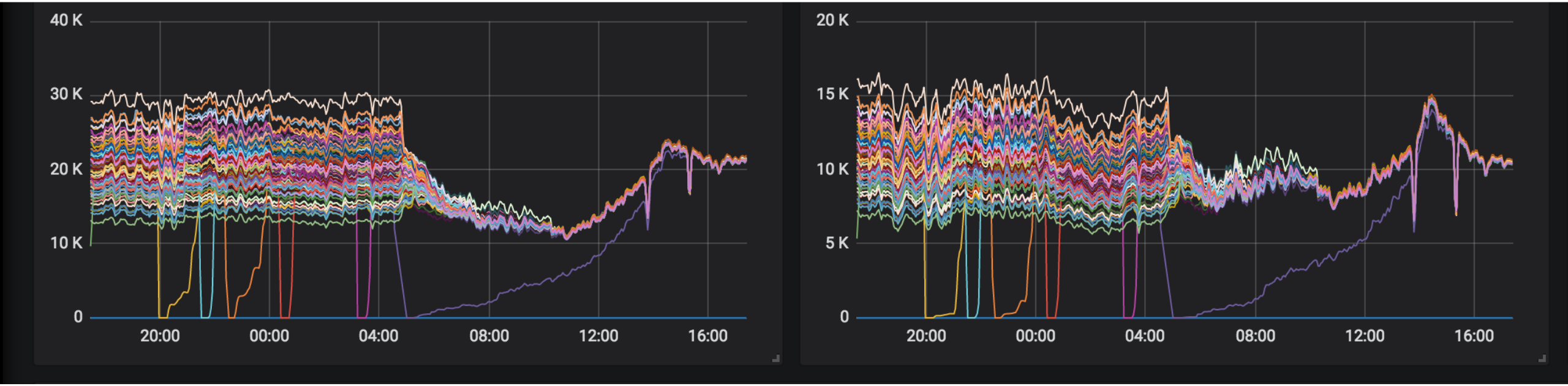
➢ **Maintains uniform distribution of partition masters even when cluster size changes.**

➢ **Lower Total Cost of Operations!**

 All rights reserved. © 2019 Aerospike Inc

Prefer uniform balance results – real customer

Large cluster data for transactions per second (TPS).



Spread was 50% of Nominal TPS, tightened to < 5% after Uniform Balance.

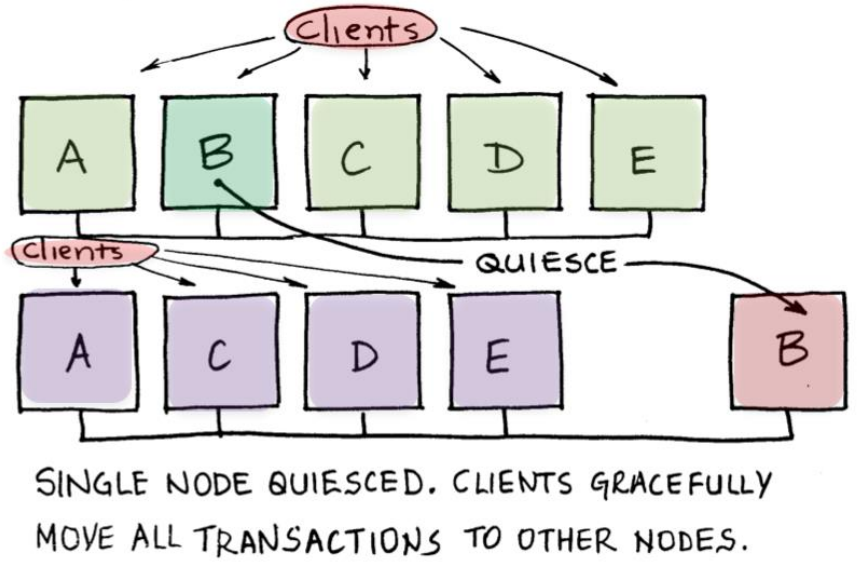
 All rights reserved. © 2019 Aerospike Inc

Quiescing and Delay Fill Migrations

 All rights reserved. © 2019 Aerospike Inc

Node Quiescing

- **Node quiescing allows us to gracefully sideline a node in a cluster.**
- **Quiesced Node is still part of the cluster, but no longer the master or replica of any partition.**

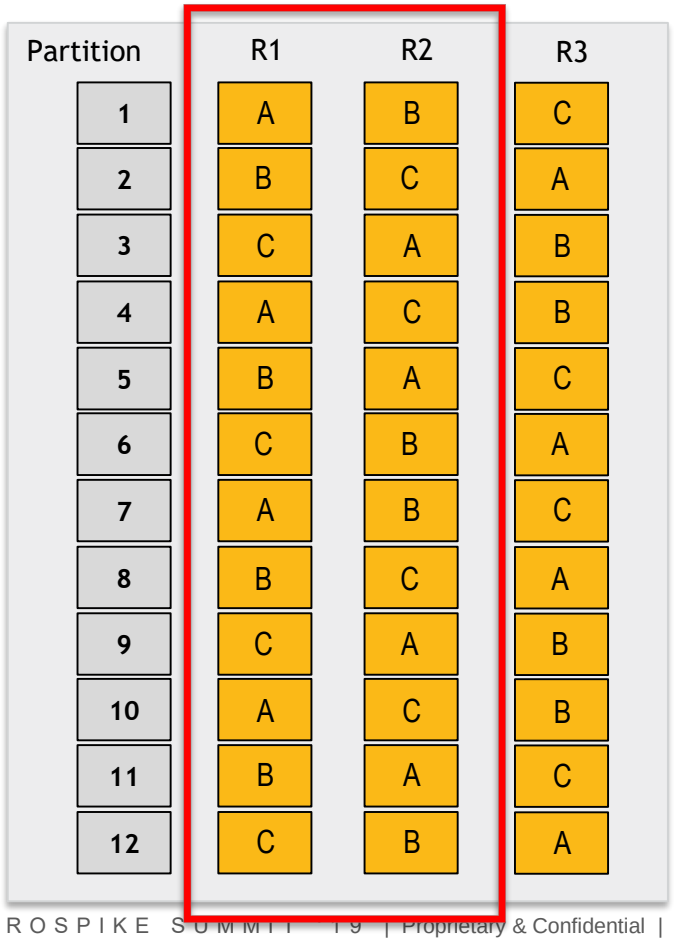


- **Data will migrate to make sure we again have replication factor number of copies of the data.**
- **Clients discover the new master & replica. Quiesced node will continue to proxy client requests to the new master.**

 All rights reserved. © 2019 Aerospike Inc

Recall: Partition Map

Partition Table is generated by a deterministic algorithm using node ids. Given 3 nodes, each node gets assigned replica id for each partition. Based on replication factor, Master and Replica(s) are determined.



Replication Factor = 2
Use R1 & R2 to get Partition Map

Partition Map

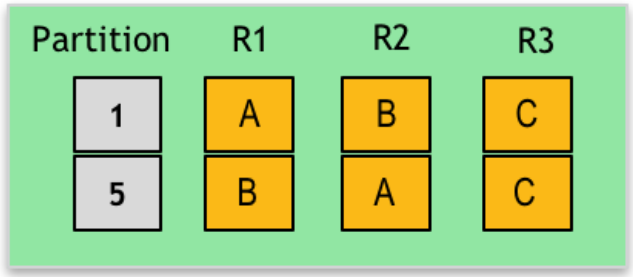
Partition	Master	Replica
1	A	B
2	B	C
3	C	A
4	A	C
5	B	A
6	C	B
7	A	B
8	B	C
9	C	A
10	A	C
11	B	A
12	C	B

 All rights reserved. © 2019 Aerospike Inc

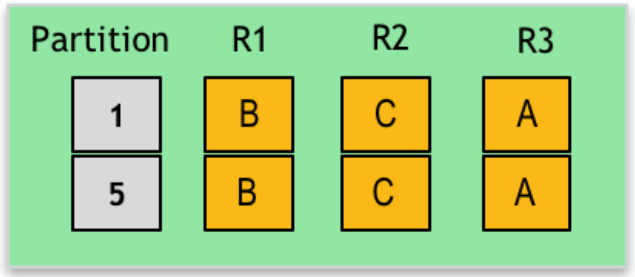
Quiescing a Node

- **A node, when quiesced, is moved to the far right position in Partition Table.**

For example: If we quiesce Node A, it is moved to the right most column.



On Quiescing Node A:
Partition 1: Replica B promotes to master, C becomes new replica.
Partition 5: C becomes the new replica



What ensues (we are using RF=2 in our discussion):

- **Node A will no longer be the designated master for any partition.**
- **"Fill" Migrations will start:**
  - From promoted master (B) to new replica (C) (eg: Partition 1) OR
  - From current master (B) to new replica (C) (eg: Partition 5)
- **Node A is still present, so requests to A will proxy => No timeouts!**

 All rights reserved. © 2019 Aerospike Inc

Migrations Nomenclature

- **Fill Migrations: When records in a partition must be copied over to an empty, new replica node.**
  - In SC Mode, fill migrations happen only to **non-roster replicas**.

- **Delta Migrations: When a prior master or replica rejoins the cluster, migrate only the records that were changed during its absence.**
  - **Lead Migrations:** These are delta-migrations to a non-empty replica in AP mode or to a *roster-replica* in SC mode.

New feature in 4.3.1+ : migrate-fill-delay

- **It allows us to intentionally delay the onset of fill migrations by prescribed number of seconds.**
- **It provides an operational advantage, e.g.: when doing rolling upgrades.**
- **Allows us to manage capacity during short term outages.**

 All rights reserved. © 2019 Aerospike Inc

All Flash

 All rights reserved. © 2019 Aerospike Inc

ALL FLASH Configuration

Aerospike Server Version 4.3.0.2+ introduces ALL FLASH storage option.

- **Allows user to store the PRIMARY INDEX (PI) on device (NVMe SSD).**

Edge Systems

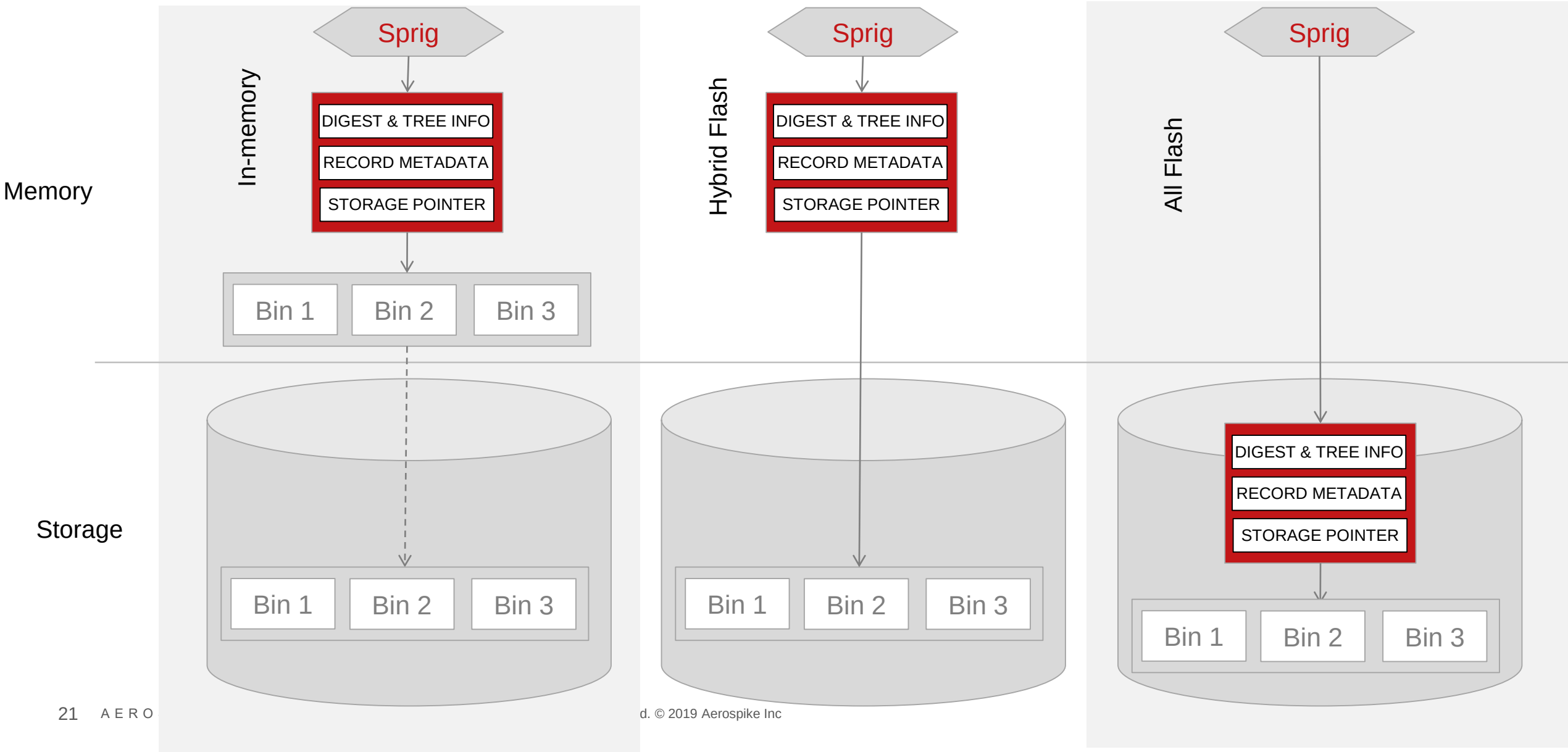
- **For large number of very small size records with relaxed latency needs.**
- **RAM vs SSD storage space ratio approaches 1:1 causing server sprawl.**
- **Significant cost savings by using ALL FLASH storage.**
- **No need to modify data model with a reverse lookup implementation to improve RAM:SSD ratio.**

System of Record

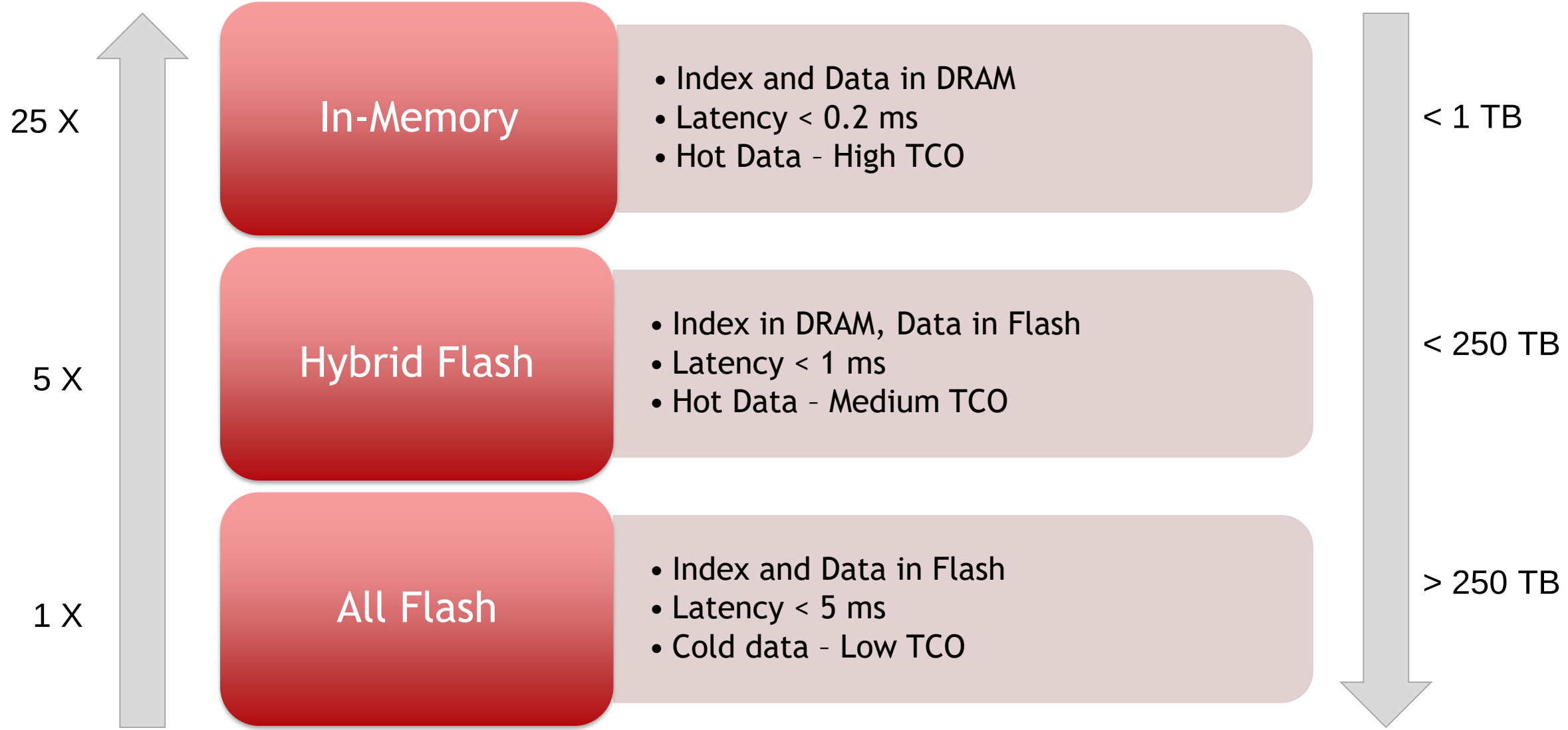
- **Cost savings with very large data stores. (> 100 TB)**

 All rights reserved. © 2019 Aerospike Inc

All Flash

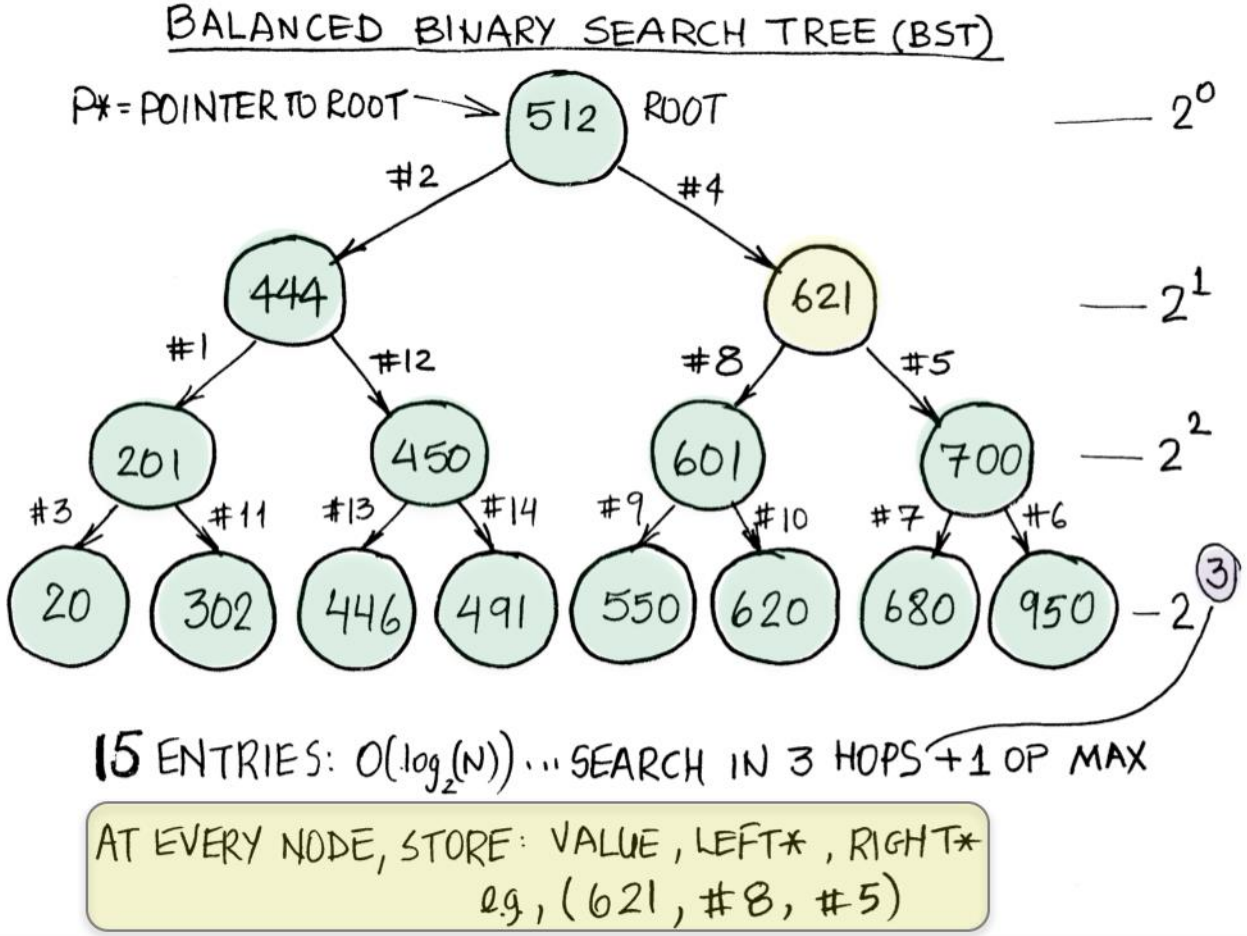


d. © 2019 Aerospike Inc



 All rights reserved. © 2019 Aerospike Inc

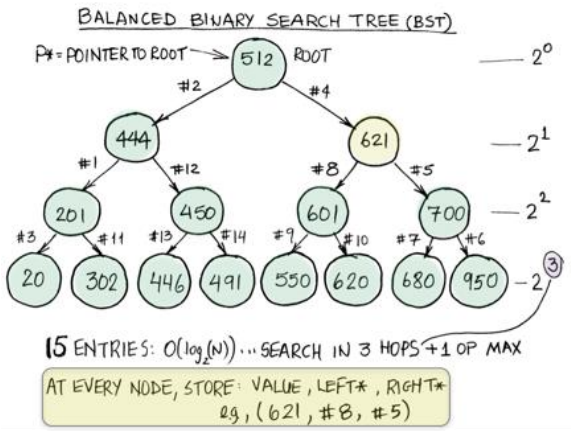
Aerospike Indexes as Red-Black Tree



- Nodes (Entries) **have keys (Values).** Every parent node can have two child nodes.
- **Left child node is always lower in value. Right child node is always higher in value.**
- Search $O(\log_2(N))$, N is the number of nodes or entries in the tree.

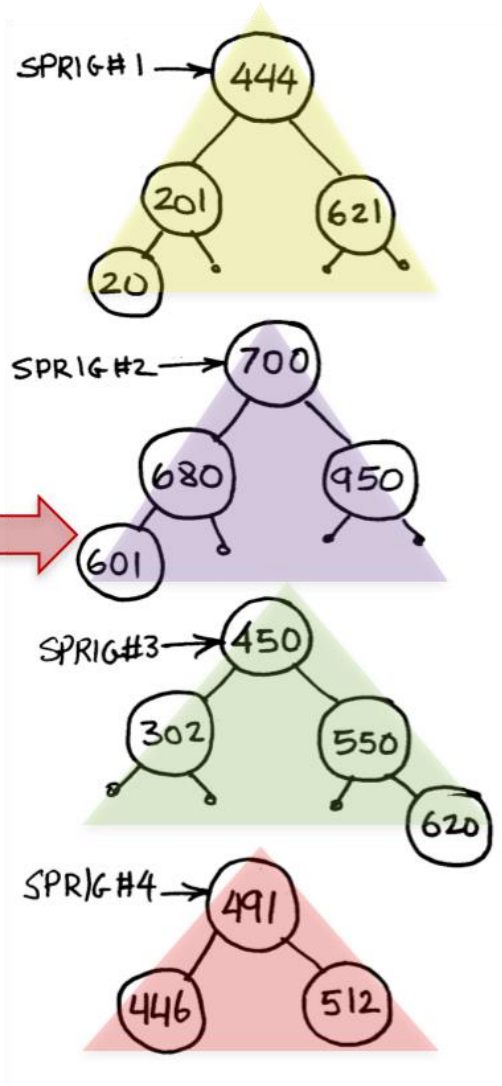
 All rights reserved. © 2019 Aerospike Inc

Break large Tree into smaller SPRIGs



BALANCED BST - NODE ENTRIES

ENTRY	VALUE (0-1023)	LEFT*	RIGHT*
#1	201	#3	#11
#2	444	#1	#12
#3	20	–	–
#4	621	#8	#5
#5	700	#7	#6
#6	950	–	–
#7	680	–	–
#8	601	#9	#10
#9	550	–	–
#10	620	–	–
#11	302	–	–
#12	450	#13	#14
#13	446	–	–
#14	491	–	–
#15	512	#2	#4



- **Single R-B Tree can grow 30 deep, PI look up time in RAM goes up.**
- **SPRIGs reduce tree height.**
- **Using all available SPRIGs (2^{28}), 30 OPS will drop to 2 OPs**
- **Using min 2^8 sprigs, 30 OPs→ 22 OPs.**
- **Number of Sprigs: 256 – 256M, RAM: 8 bytes + 5 bytes for EE**

 All rights reserved. © 2019 Aerospike Inc

Some Maths! (Oh no!)

- **SSDs tend to read 4kB blocks, so 1 byte or 4kB take the same time**
- **Aerospike's Primary Index is 64 bytes**
- **Want to read the whole sprig in 1 disk IO**
  - Maximum number of records/sprig is 4096/64 = 64
  - Better to leave some overhead => use 32 records/sprig

- **Consider storing 100 billion objects on a 20 node cluster, RF=2**
  - DRAM (Hybrid memory): (64 x 2 x 100B) / 20 / 0.8 ~= 860GB / node
  - Under all flash:
    - Sprigs per partition needed ~= 819,200 (1,048,576 rounded up)
    - DRAM needed / node ~= 64K + (8M x 2 + 4096 x 1M x 2 x 13) / 20 = 5.6GB / node
  - Cost savings at US$9/GB of DRAM: ~= $154,000

 All rights reserved. © 2019 Aerospike Inc

Questions?

 All rights reserved. © 2019 Aerospike Inc