Scalability and high volume performance of indexer clustering at Splunk.

What's on your bucket list?

Scalability and high volume performance of indexer clustering at Splunk.

Brent Davis
Principal Performance Engineer
Splunk

Justin Lin
Principal Performance Engineer
Splunk

Cher-Hung Chang
Principal Software Engineer
Splunk

Forward-Looking Statements

During the course of this presentation, we may make forward-looking statements regarding future events or plans of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results may differ materially. The forward-looking statements made in the this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, it may not contain current or accurate information. We do not assume any obligation to update any forward-looking statements made herein.

In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only, and shall not be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionalities described or to include any such feature or functionality in a future release.

Agenda

1. Indexer Clustering Overview
2. Clustering improvements in 8.0
3. Scaling Clustering in Splunk Labs

Indexer Clustering Overview

splunk> .conf19

Why Indexer Clustering?

High Availability and Disaster Recovery

- Ability to withstand loss of one or more indexers, or an entire site.

Search Affinity

- Strategically locate search heads to reduce long-range network traffic.

Consistent Shared Configuration

- Ensure that all indexers share a common set of configuration files

Clustering Components

Cluster Master

- A single **master node** to manage the cluster
- Stateless: maintains in-memory state of all the peers and buckets
- Coordinates the replicating activities of the peer nodes
- Tells the search head where to find data

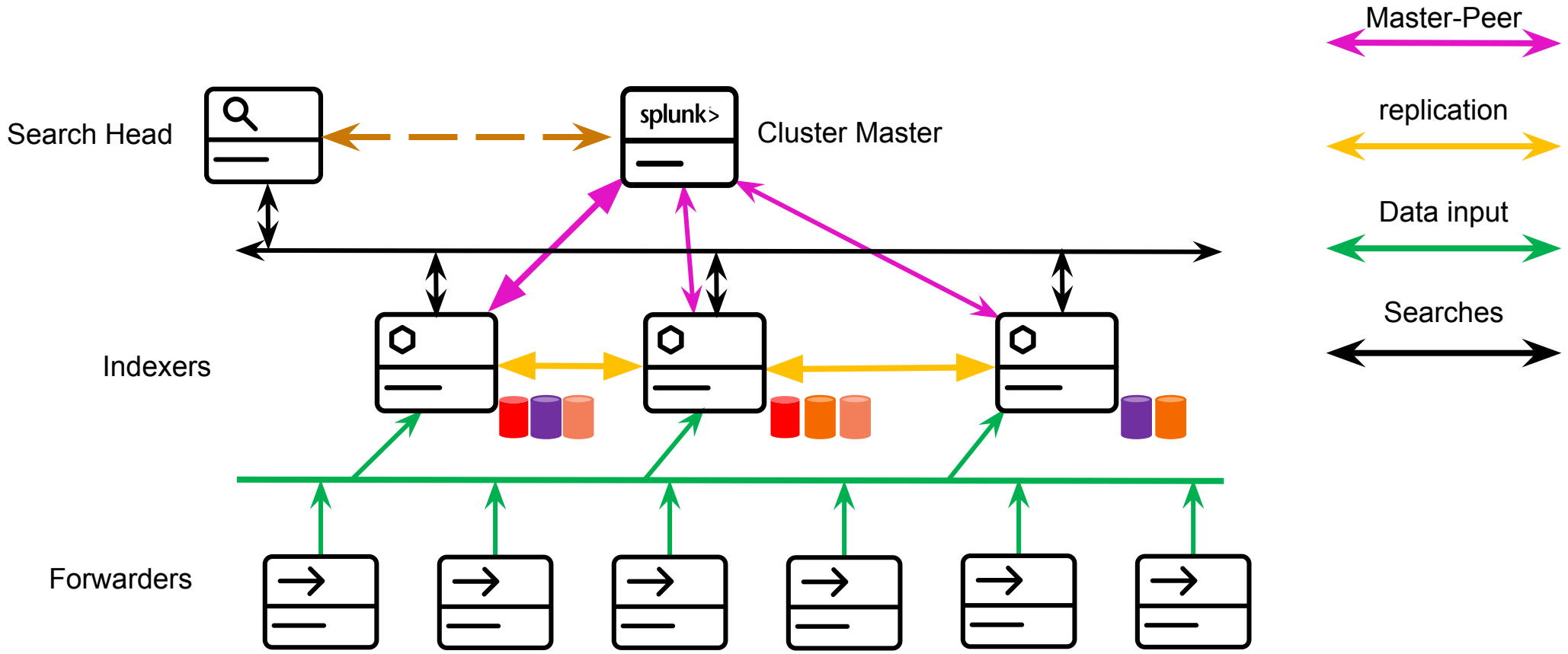
Indexer Nodes (Peers)

- **Index** and maintain multiple copies of the data and run **searches** across that data
- Reports its state and all its buckets to Cluster Master

Search Head(s)

- One or many **search heads** coordinate searches across all the peer nodes.

Index Clustering Topology



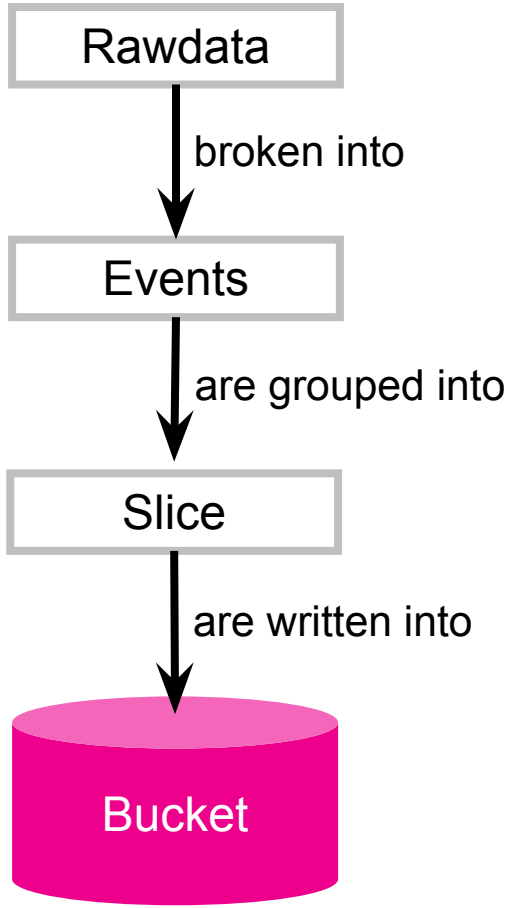
Key Indexer Clustering Concepts

- Buckets
- Replication, Search Factor, and Multisite
- Bucket Fixups
- Heartbeats
- Rolling Restart

Buckets

Unit of data the cluster is aware of

- Created on the indexer
  - Indexer notifies Cluster Master upon every state transition of its bucket
- Configurable size
- More data, more buckets



Key Indexer Clustering Concepts

- Buckets
- Replication, Search Factor, and Multisite
- Bucket Fixups
- Heartbeats
- Rolling Restart

Replication Factor/Search Factor

Replication Factor

The number of copies of data that the cluster maintains. A cluster can tolerate a failure of (replication factor - 1) peer nodes

Search Factor

The number of searchable copies of data that an indexer cluster maintains

As Replication factor increases, the Cluster Master has to manage more buckets

Multisite

Cluster Master can explicitly configure indexer clusters on a site-by-site basis. Multisite provides:

- **Improved disaster recovery:** Store buckets at multiple locations, to maintain access a if a disaster strikes at one location
- **Search affinity:** A separate search head on each site can limit its searches to local peer nodes, reducing network overhead

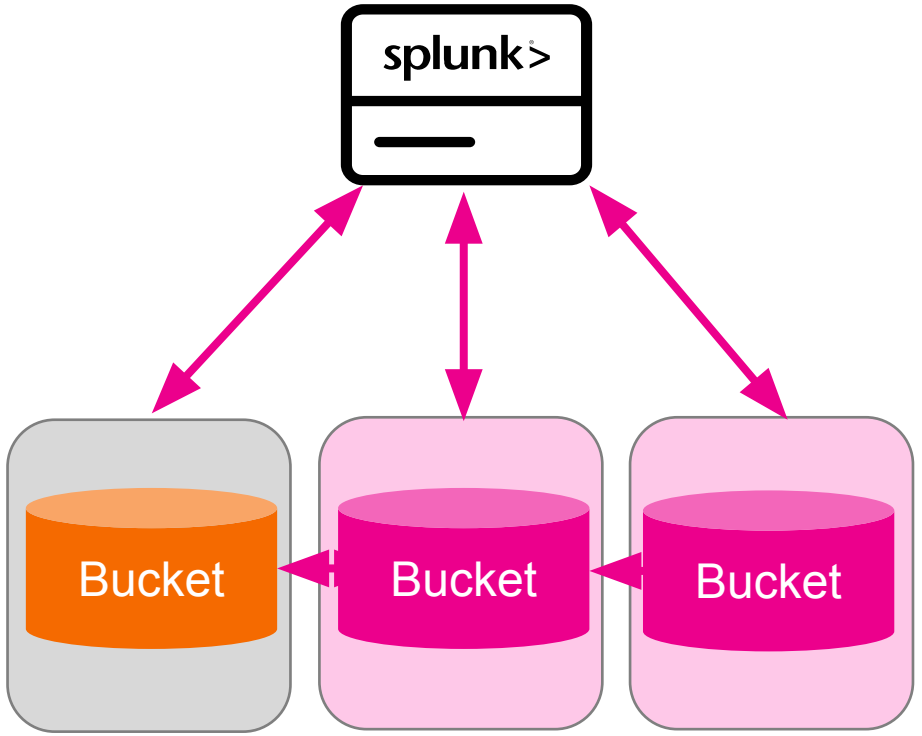
Key Indexer Clustering Concepts

- Buckets
- Replication, Search Factor, and Multisite
- **Bucket Fixups**
- **Heartbeats**
- **Rolling Restart**

Bucket Fixing

A **Bucket Fixup** is the remedial activity that occurs when a peer node goes offline.

The Cluster Master orchestrates the remaining peers in replicating buckets and indexing non-searchable bucket copies, to return the cluster to a valid state.



Key Indexer Clustering Concepts

- Buckets
- Replication, Search Factor, and Multisite
- Bucket Fixups
- Heartbeats
- **Rolling Restart**

Heartbeats

One of the mechanisms the Cluster Master uses to communicate with indexers

Status synchronization

Once a peer registers to the master, it starts to heartbeat to master every **heartbeat_period** seconds

CM utilizes **heartbeat_timeout** to consider if peer is offline and perform fixup if necessary

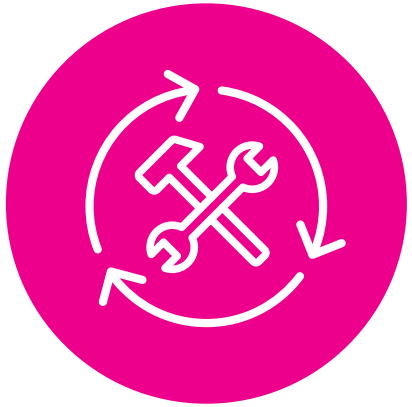
Key Indexer Clustering Concepts

- Buckets
- Replication, Search Factor, and Multisite
- Bucket Fixups
- Heartbeats
- **Rolling Restart**

Rolling Restart

A rolling restart performs a phased restart of all peer nodes

- the indexer cluster as a whole can continue to perform its function during the restart process
- ensure that load-balanced forwarders sending data to the cluster always have a peer available to receive the data



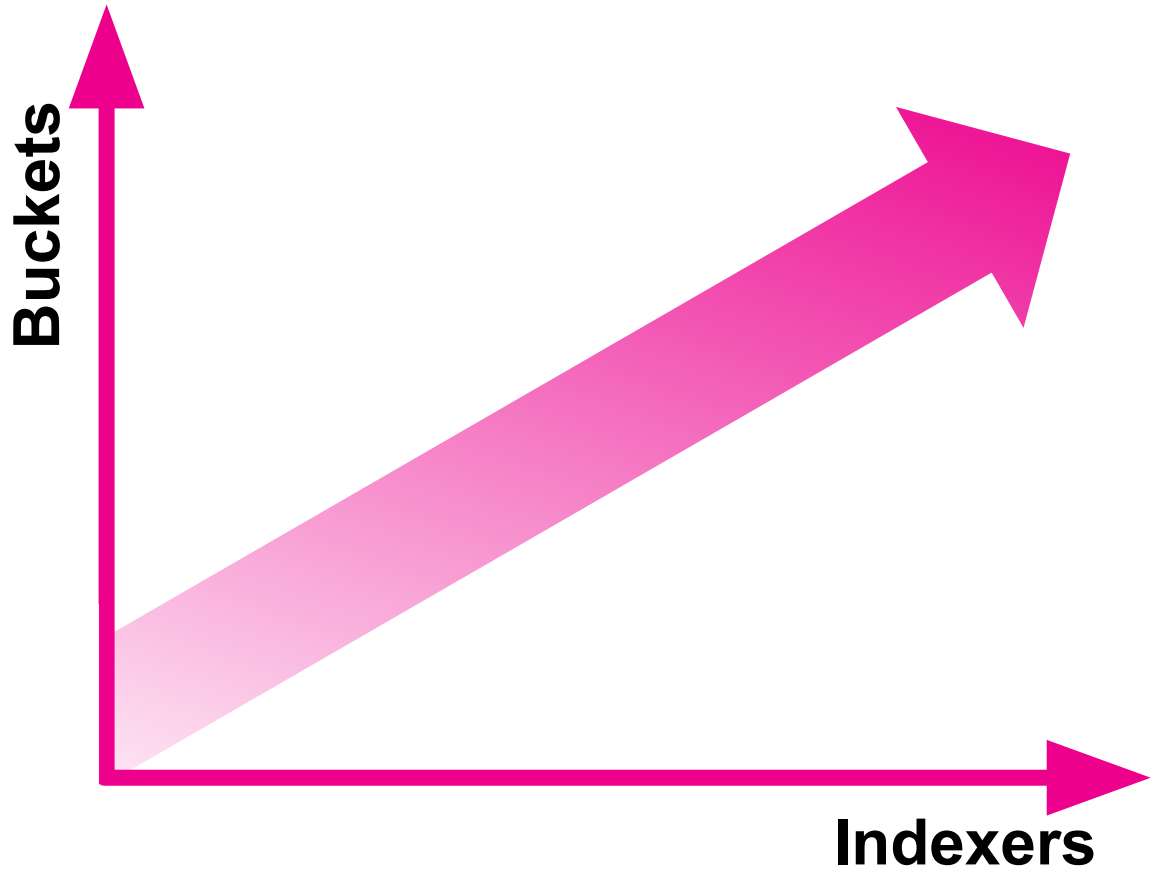
Initiated when a new configuration needs to be distributed to the peer nodes

Specify the percentage of peers to restart at one time:
percent_peers_to_restart

Key Indexer Clustering Concepts

Scaling Factors:
Buckets and Indexers

Scaling Factors: Buckets and Indexers



More Data ⟶ More Buckets

More buckets means the Cluster Master has to do more work

- Iterates through each bucket, checking whether it needs to queue up any fixup jobs
  - Replication Jobs (to meet RF)
  - Search Jobs (to meet SF)
  - Primary Jobs (all buckets need to have a primary copy per site)
  - Other jobs (freezing, checksum, rolling, etc)

More Data ⟶ More Indexers

More Indexers (peers) means the Cluster Master has to do more work

- High number of peers means high number of heartbeats to the master
- If peer heartbeats start timing out, this can lead to “flapping” – where peers keep leaving and joining the cluster

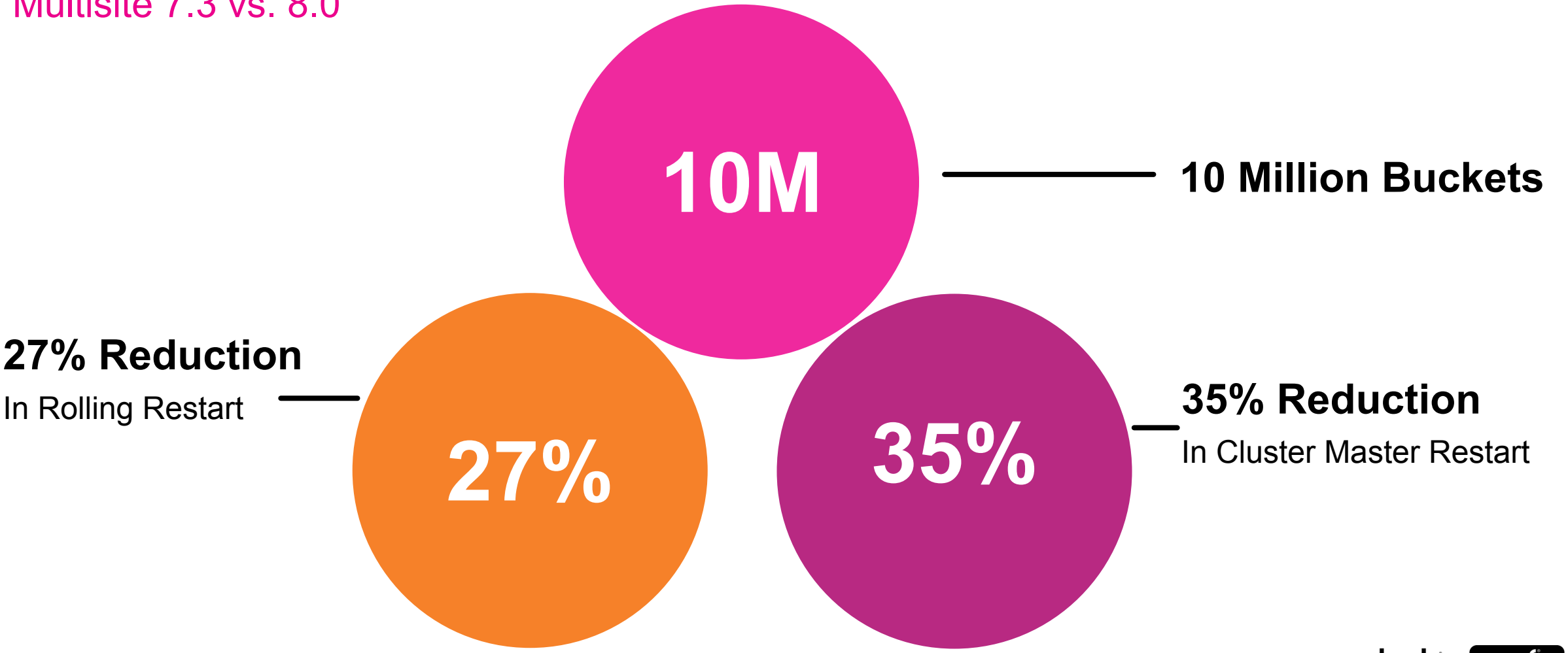
Scaling Clustering in Splunk Labs

splunk> .conf19

Large Scale Testing, Increasing Buckets

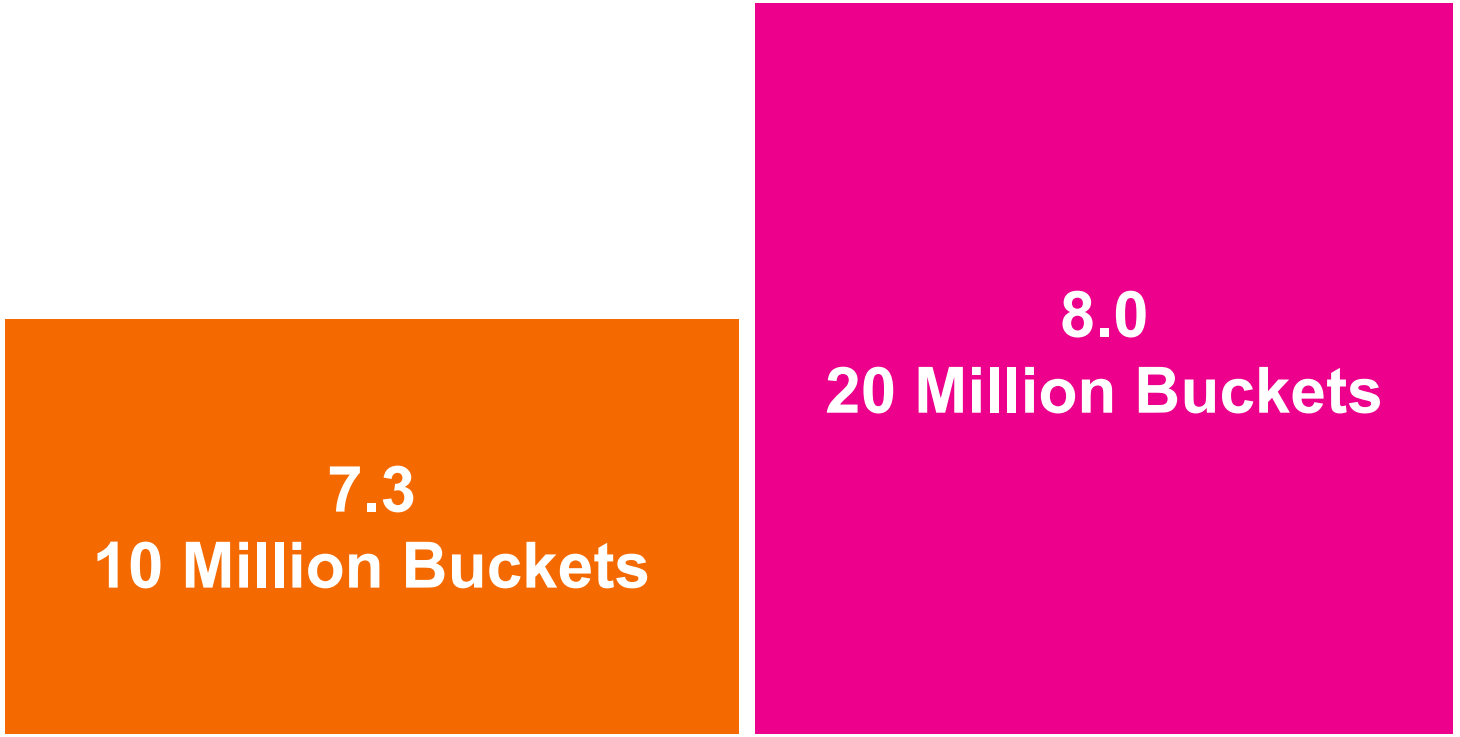
Cluster Maintenance Performance: Multisite

Multisite 7.3 vs. 8.0

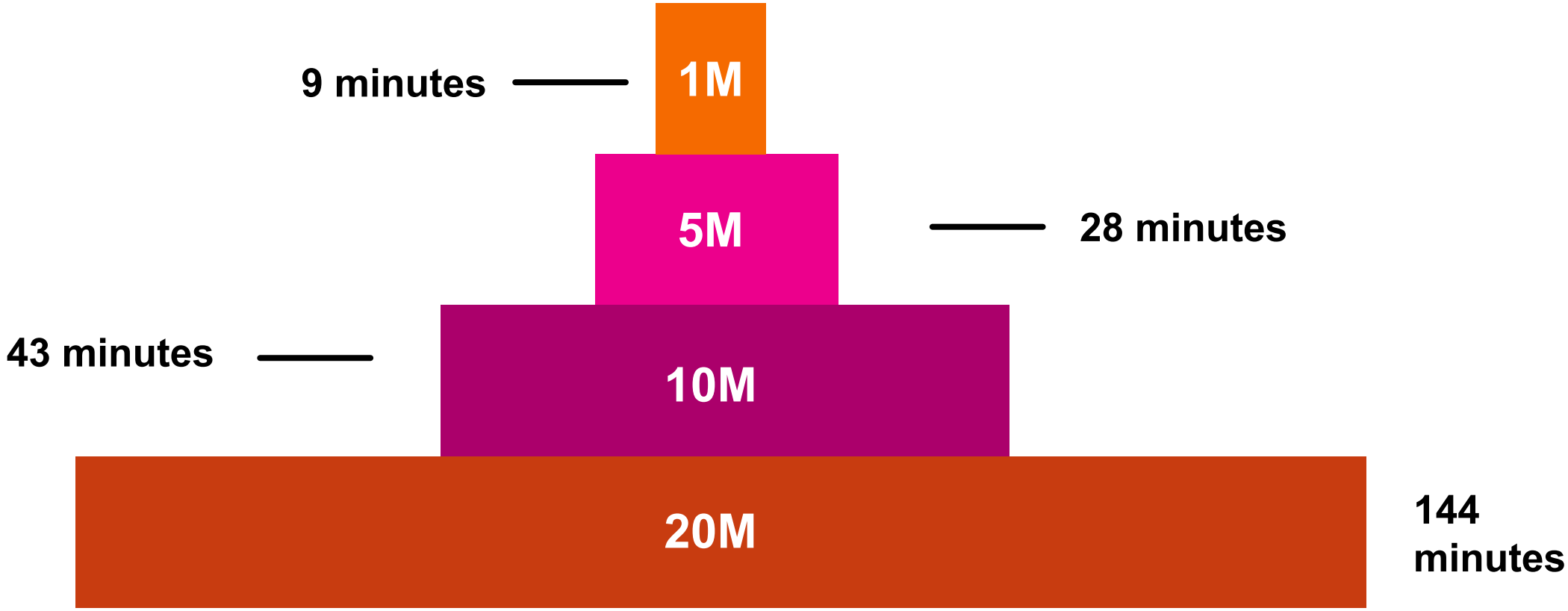


20 Million Buckets in 8.0

2x Increase over 7.3

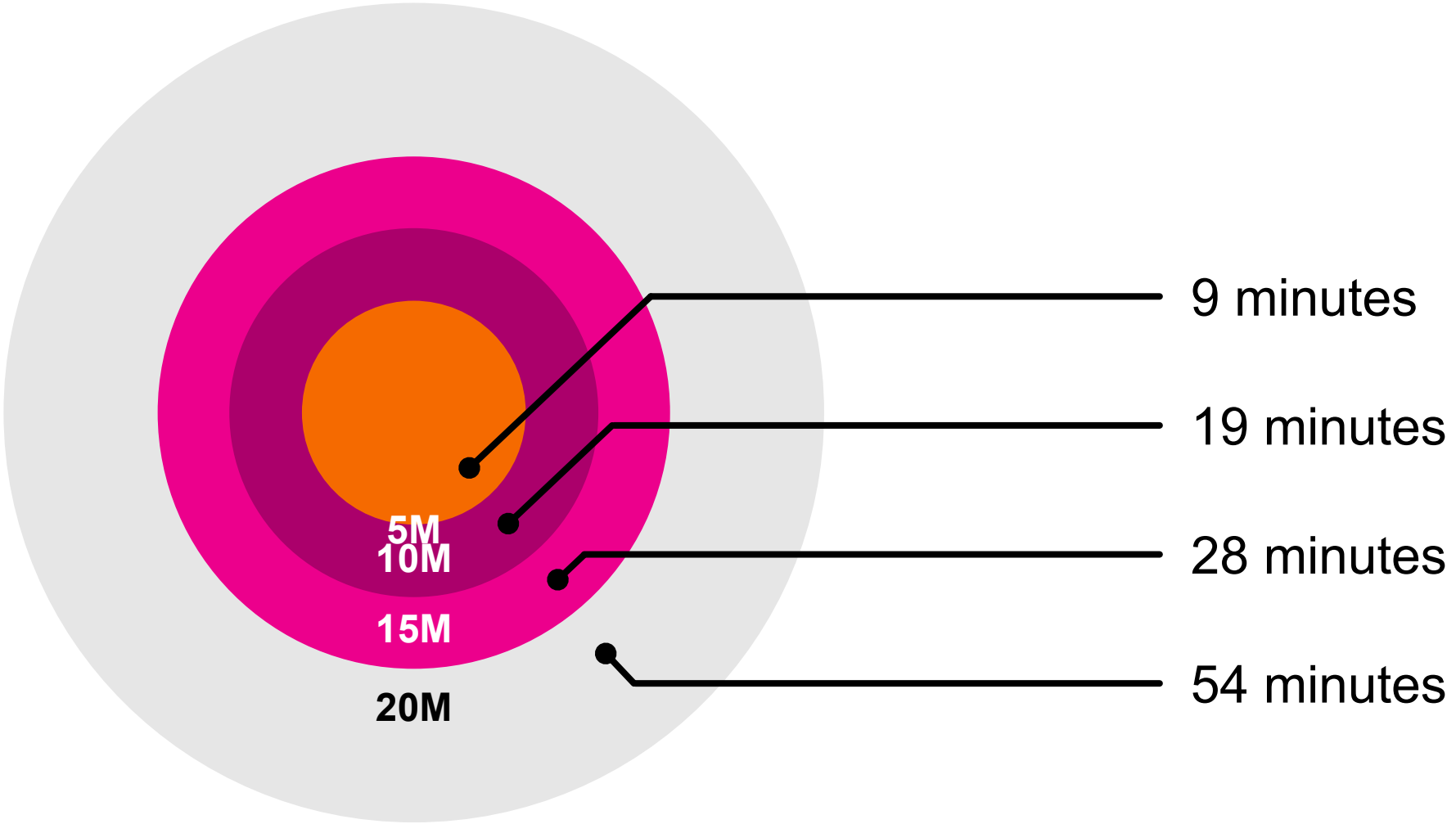


Scalability to 20 Million Unique Buckets in 8.0



Rolling Restart Time with Millions of Buckets in 8.0

Scalability to 20 Million Unique Buckets in 8.0



Master Restart Time with Millions of Buckets in 8.0

Clustering Improvements in 8.0

Performance Improvements

- Improved fixup prioritization
- More responsive Cluster Master
- Reduced maintenance window

Default configurations are tuned for scalability up to 10 million unique buckets

- Much less tuning efforts!

Configuration for 20M Buckets

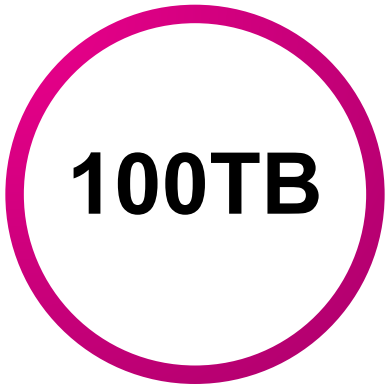
Parameter	Value Used	Default	Reason
heartbeat_timeout	900	60	Prevent cluster from missing the heartbeat if the cluster master is under load
restart_timeout	300	60	
max_fixup_time_ms	1000	5000	upper-bound on fixup time
rep_cxn_timeout	900	5	
rep_send_timeout	900	5	
rep_rcv_timeout	900	10	Increase timeouts for replicating data
rep_max_send_timeout	900	180	
rep_max_rcv_timeout	900	180	
cxn_timeout	60	60	
rcv_timeout	900	60	Increase timeouts between cluster nodes
send_timeout	900	5	

Large Scale Testing, Increasing Indexers

splunk> .conf19

High Volume Across Multiple Indexer Clusters

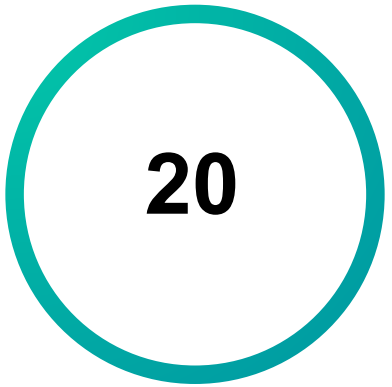
100TB/day ingestion volume



1000 Indexers

Indexes

Search heads



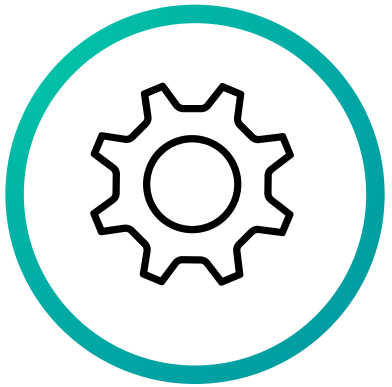
Getting to Indexer Scale

Multiple Indexer Clusters

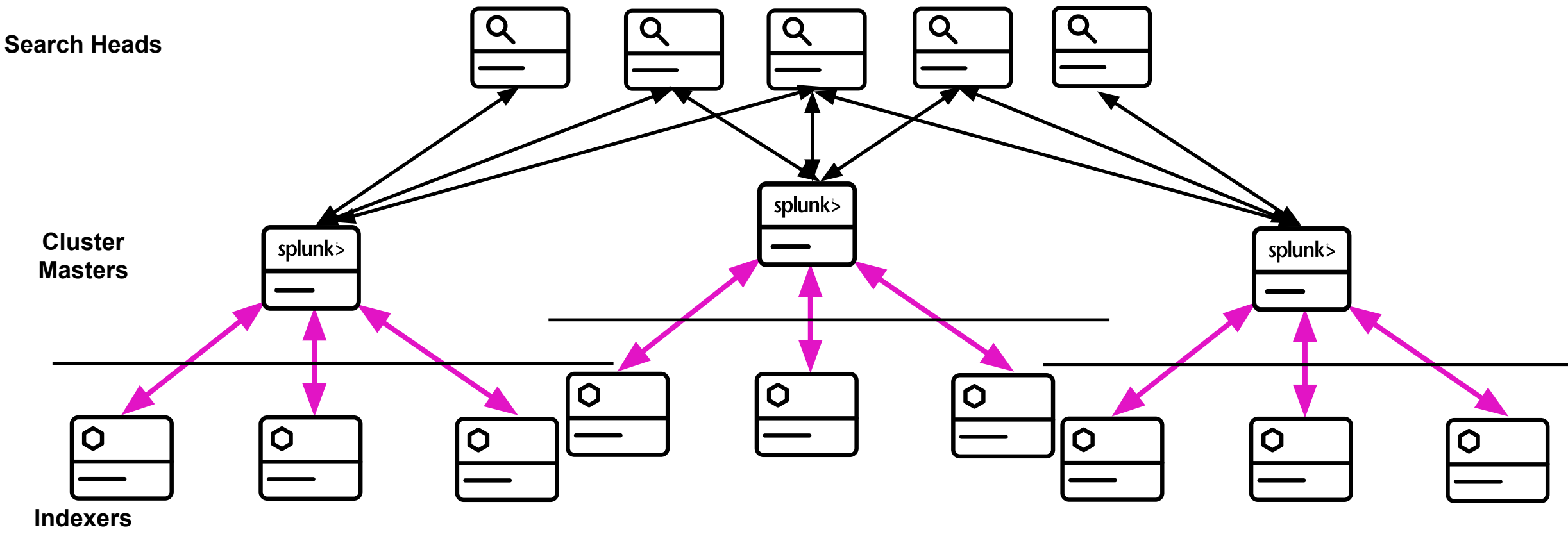
Managing Fixups

Cluster Master Memory

Configuration



Idea #1: Multiple Clusters



Idea #1 (cont): Multiple Clusters

Once a cluster reaches a high number of peers start a separate cluster and grow that to the limits.

- A tiered approach: multiple clusters up to some size before starting up another cluster
- A search head can be configured to point to each Cluster Master

Idea #2: Managing Fixups

Fixups by default **are throttled** when there are many fixups doing replicate/repair:

- we don't want to overload indexers and impact indexing and search.

Fixups are quick to go away when there is nothing to fix.

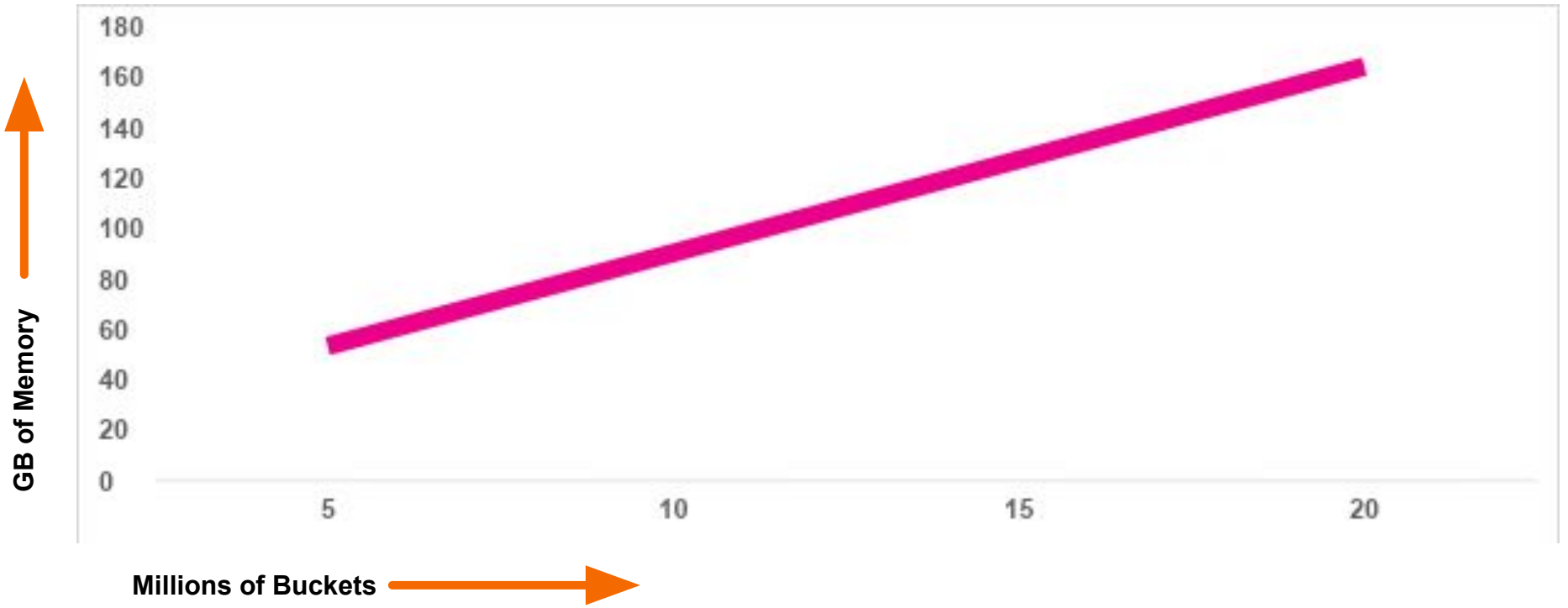
For faster fixups, increase these parameters:

- **max_peer_build_load**
- **max_peer_rep_load**
- **max_fixup_time_ms**

Use **maintenance mode** to avoid fixups when possible (during maintenance operations)

Idea #3: Size CM appropriately

Major resource requirement for Cluster Master is memory:



Idea #4: Configuration Parameters

Parameter	Value Used	Default	Reason
heartbeat_timeout	900	60	Prevent cluster from missing the heartbeat if the cluster master is under load
heartbeat_period	10	5	
max_fixup_time_ms	1000	5000	upper-bound on fixup time
rep_cxn_timeout	600	5	Increase timeouts for replicating data
rep_send_timeout	600	5	
rep_rcv_timeout	600	10	
rep_max_send_timeout	900	180	
rep_max_rcv_timeout	900	180	
cxn_timeout	900	60	Increase timeouts between cluster nodes
rcv_timeout	900	60	
send_timeout	900	5	
quiet_period	180	60	On master startup, do not initiate any action, just wait for peers to register

Other Best Practices

Balanced ingest among indexers and clusters of indexers, as much as possible

Avoid creating small buckets

- **maxDataSize** = "auto_high_volume" (non-SmartStore deployments)
- Ensure event **timestamps** are set correctly
- Redirect out-of-order events to the quarantine index

percent_peers_to_restart

- **% peers to restart** = 6 / # of peers
  - i.e.: 5-7 peers adding at any given time

Key Takeaways

1. Scale to 20 Million Unique Buckets in 8.0
2. Reduced maintenance window with faster CM restarts and rolling restarts
3. Partitioned clusters at very large scale

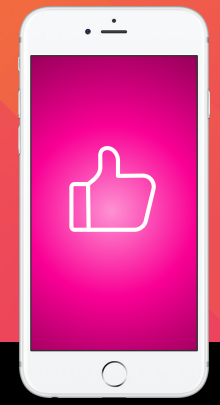
.conf19

splunk>

Thank You !

Go to the .conf19 mobile app to

RATE THIS SESSION



Q&A

Cher-Hung Chang | Principal Software Engineer, Splunk
Brent Davis | Principal Performance Engineer, Splunk
Justin Lin | Principal Performance Engineer, Splunk

splunk> .conf19