Splunk Cluster Administration

Document Usage Guidelines

- Intended for enrolled students only; do not distribute
- Instructor required; not intended to be a self-paced document

Course Objectives

- Identify factors affecting large-scale Splunk deployments
- Set up Splunk indexer clusters
- Deploy and configure a Splunk search head cluster
- Add new nodes into an existing cluster
- Decommission nodes from an existing cluster
- Deploy apps and configuration bundles in Splunk clusters
- Manage KV store collections and lookups in Splunk clusters
- Monitor and identify clustering issues with Monitoring Console
- Scale Splunk indexer cluster with SmartStore

Course Outline

- Module 1: Large-scale Splunk Deployment Overview
- Module 2: Single-site Indexer Cluster
- Module 3: Multisite Indexer Cluster
- Module 4: Indexer Cluster Management and Administration
- Module 5: Forwarder Configuration
- Module 6: Search Head Cluster (SHC)
- Module 7: SHC Management and Administration
- Module 8: KV Store Collection and Lookup Management
- Module 9: SmartStore Implementation

Before Taking This Course

- Recommended prerequisite courses:
  - Splunk Enterprise System Administration
  - Splunk Enterprise Data Administration
- Recommended skills and knowledge
  - Working knowledge of Linux, command line and vi or nano
  - Minimum 3 months hands-on Splunk administration experience

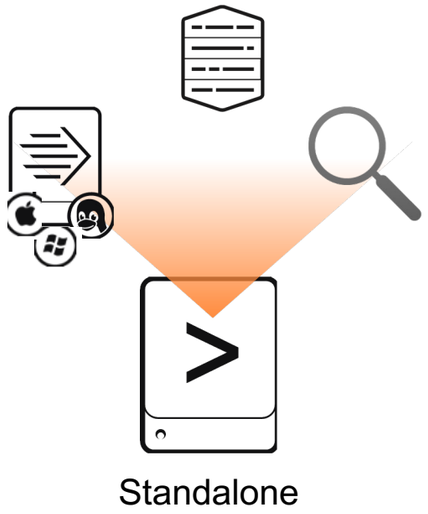
Module 1: Large-scale Splunk Deployment Overview

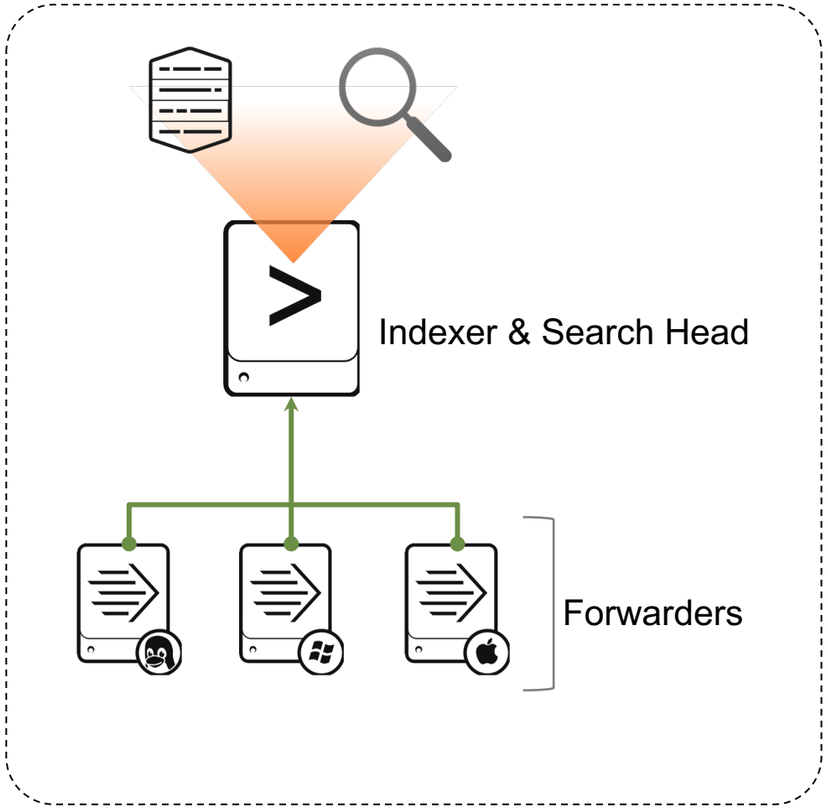
Module Objectives

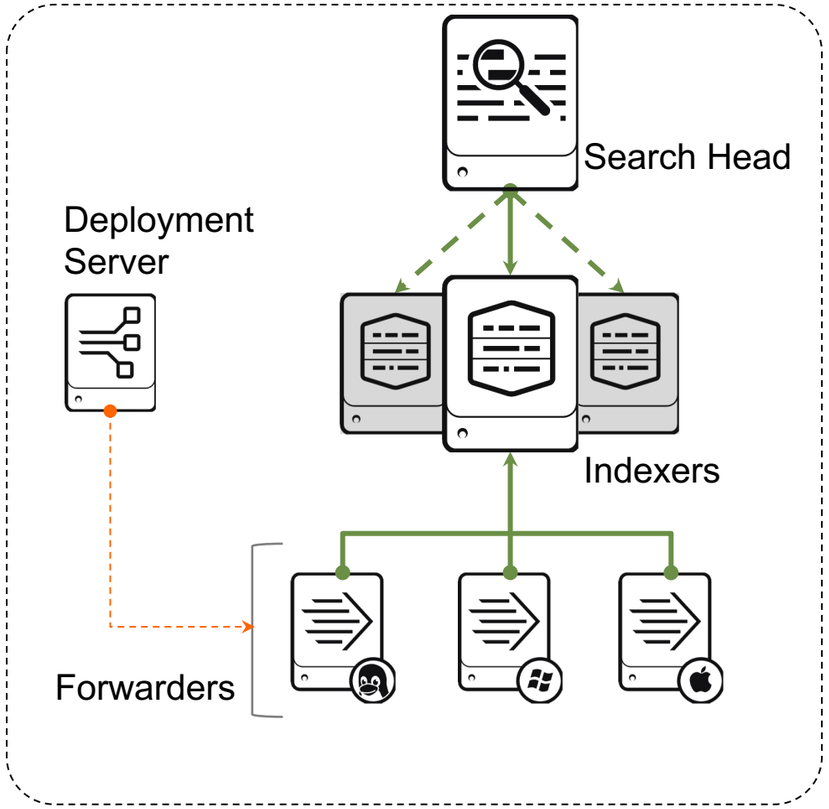
- Review Splunk deployment options
- Identify factors that affect large-scale deployment design
- Describe how Splunk can scale
- Configure a Splunk License Manager

Splunk Deployment Review

- Ingests data; stores and searches indexed data
- Scales by splitting functionality across multiple dedicated instances







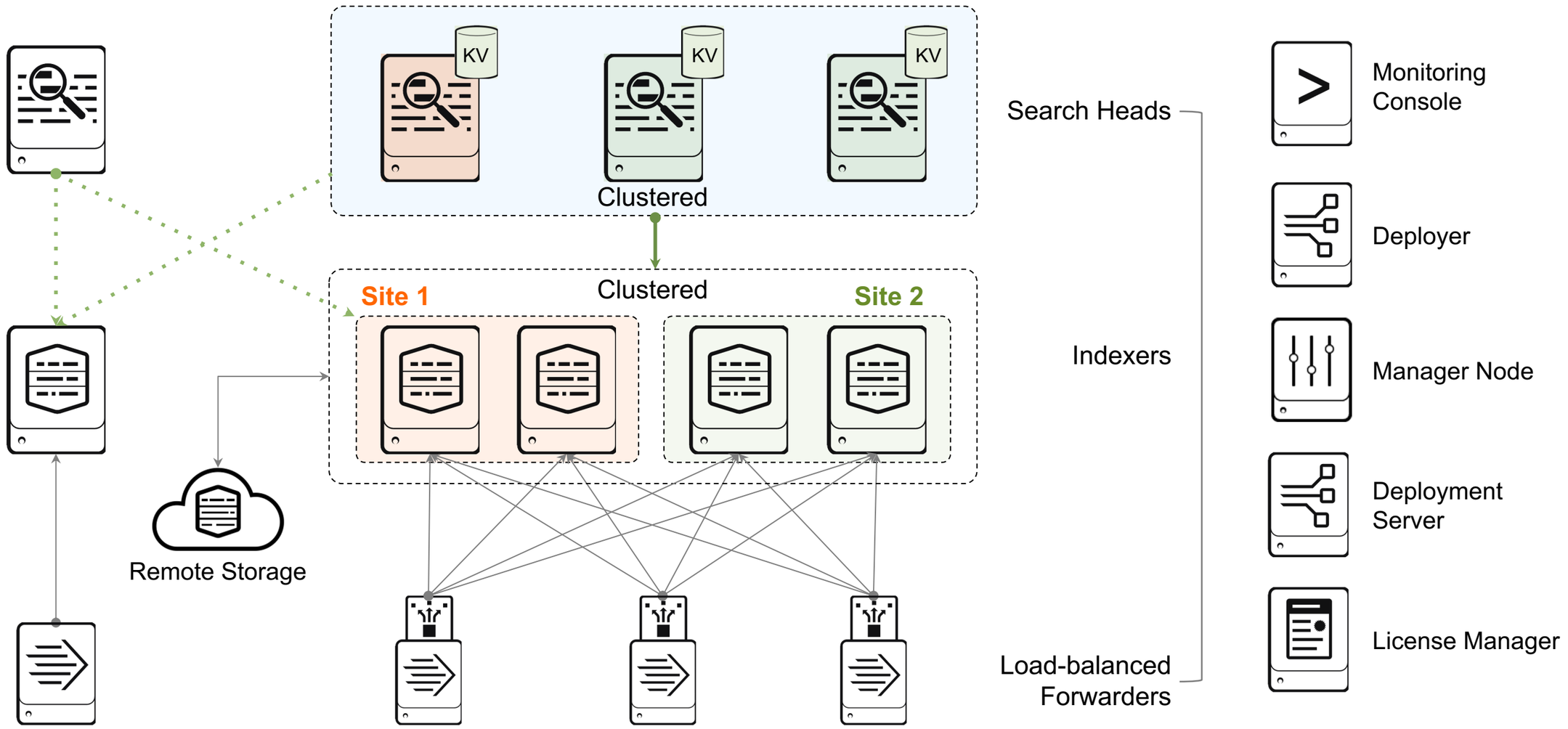
Splunk Clustering

Balance growth, speed of recovery, and disk usage by:

- Configuring indexers to replicate indexed data & maintain specified copies
- Grouping Search Heads to maintain consistent Knowledge Object & user access configuration and increased availability for scheduled searches

	High Availability (HA)	Disaster Recovery (DR)
Indexing Tier	Single-site cluster • Index replication • Flexible replication policies	Multisite cluster • Can withstand entire site failure • Supports active-passive and active-active configurations
	• SmartStore reduces the storage footprint while maintaining HA/DR	
Search Tier	• Search head or • Search head cluster	Search affinity (site-aware) • Search head or • Search head cluster

Server Roles in Splunk Clusters



Server Roles in Splunk Clusters (cont'd)

License Manager	Allocates license capacity and manages license usage of all cluster members
Manager Node	Regulates the functioning of an indexer cluster
Peer Node	Indexer (search peer) that participates in an indexer cluster
Search Head	Participates in clusters as stand-alone or search head cluster member
Deployment Server	Centralized configuration manager for forwarders
Deployer	Distributes configurations to search head cluster members
Monitoring Console	Allows admins to monitor performance and activity details regarding your Splunk environment

Note

Splunk recommends that you dedicate a host for each role. You can enable multiple Splunk server roles, with caveats. You will learn more about caveats of each server role throughout the course.

Reference Specs for Distributed Deployment

	Indexer	Search Head
OS	Linux or Windows 64-bit distribution	
Network	1GB+ Ethernet NIC (optional second NIC for a management network)	
Memory	12GB+ RAM	
CPU	Intel 64-bit chip architecture **12+ CPU cores** Running at 2+ GHz	Intel 64-bit chip architecture **16+ CPU cores, 32+ vCPU** Running at 2+ GHz
Disk	Disk subsystem capable of 800+ IOPS	SSD w/ 300GB+ free space

Note

Hardware sizing is discussed in detail in Architecting Splunk Course

License Manager Configuration

- Every Splunk instance is a **License Manager** by default
- All indexer cluster members must share:
  - The same licensing pool
  - The same licensing configuration
- Only incoming data counts against the license; replicated data does not
- Cannot use a free license for clustering
- Consider forwarding License Manager internal logs to the indexing layer
  - Internal logs do not count against the license
  - Run searches against the license logs
    - `index=_internal source=*license_usage.log`
  - Any unusual condition visible on all search heads using that set of indexers

License Manager Configuration (cont.)

Use SplunkWeb UI to add / manage licensing (*Settings* > *Licensing)*

From the command line:

- To add a license to a License Manager, run:

  `splunk add licenses {path_to_license_file}`

- To check the list of License Peer from the License Manager, run:

  `splunk list licenser-peers`

- To switch to a License Peer, run this command on each peer:

  `splunk edit licenser-localpeer –manager_uri https://{LM:Mport}`

- To check the peer configuration of a particular node, run:

  `splunk list licenser-localpeer`

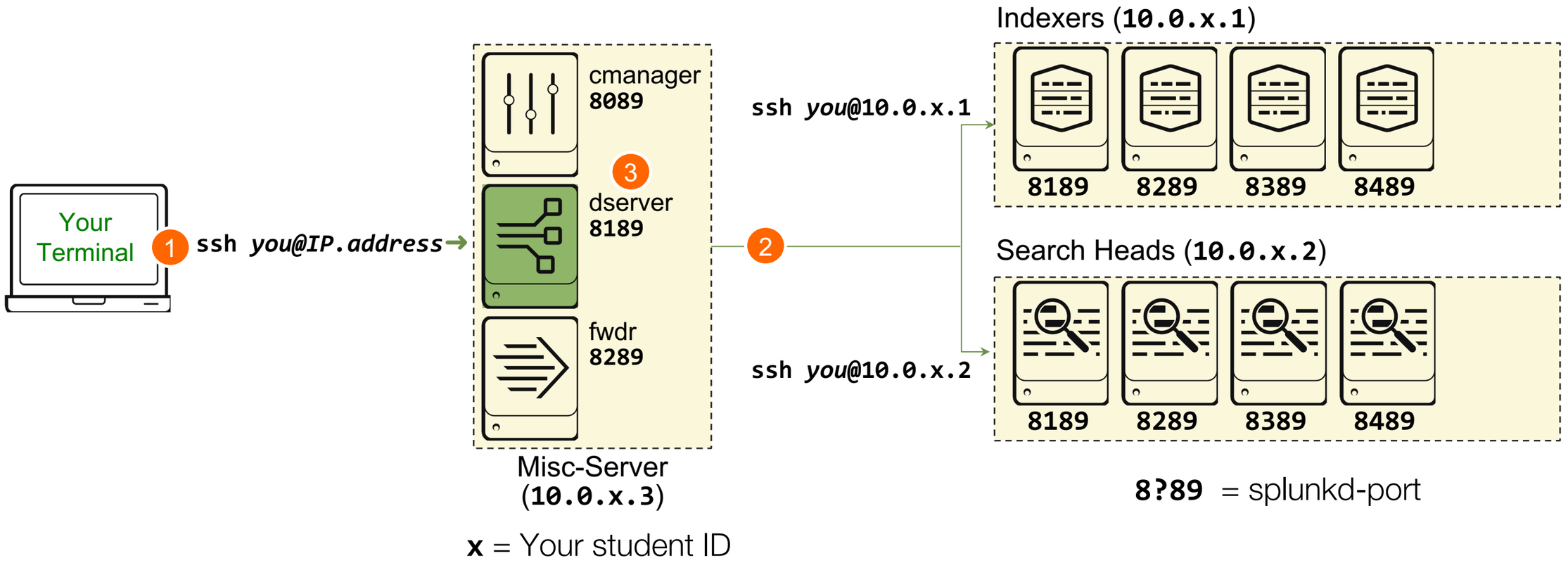
Further Reading: Large-scale Splunk Deployments

- Reference hardware
- Summary of performance recommendations
- Security updates

Lab Exercise 1: Configure Splunk License Manager

- **Time**: 30 minutes
- **Tasks**:
  - Access your designated Splunk environment
  - Set up password-less SSH connection
  - Configure your License Manager instance
  - Log into Splunk Web and verify the license information

Lab Exercise 1 – Configure Splunk License Manager (cont.)



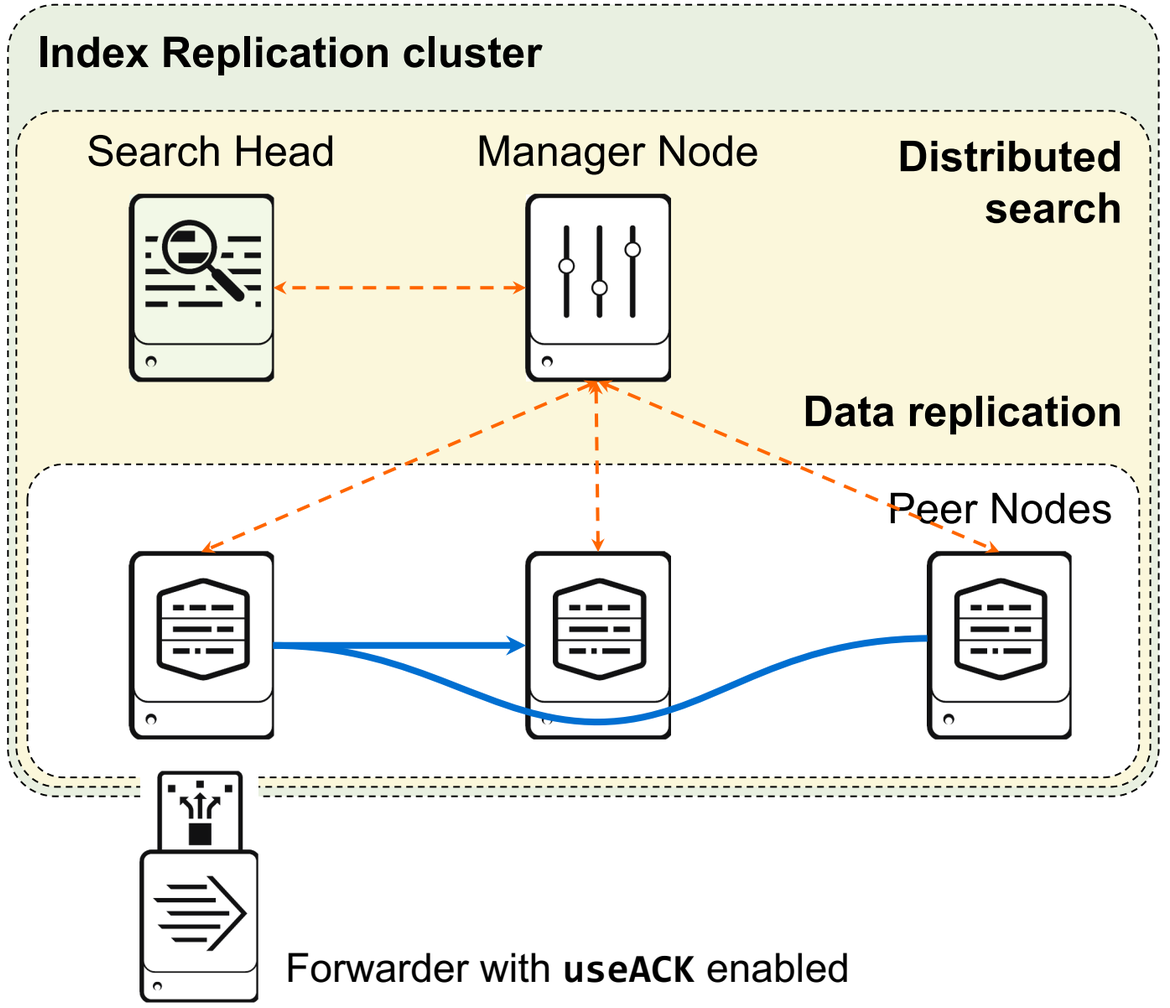
Module 2:
Single-site Indexer Cluster

Module Objectives

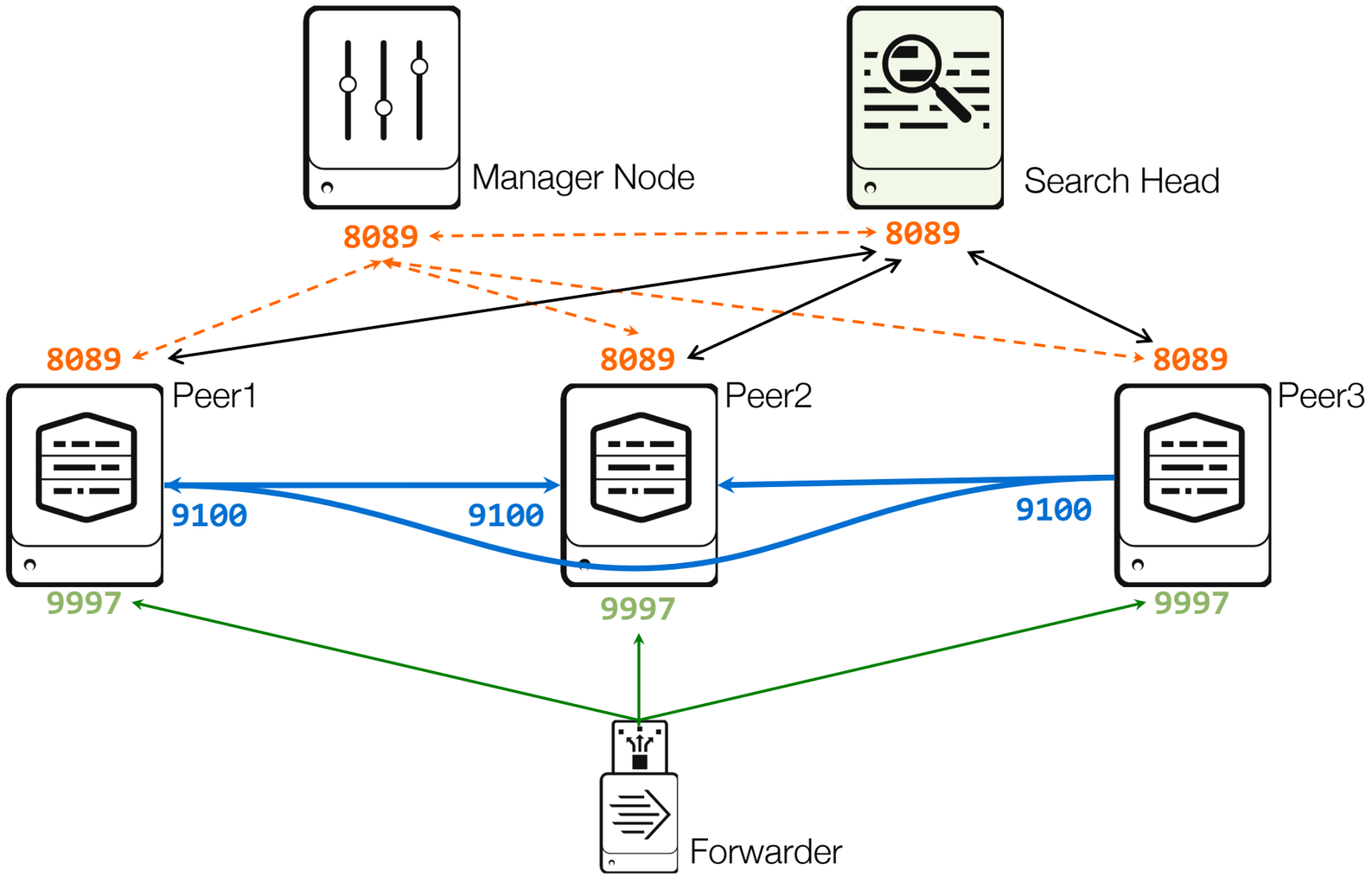
- Describe how Splunk single-site indexer clusters work
- Identify cluster components and terms
- Implement a single-site indexer cluster
- Search and review internal logs related to indexer clustering

Single-site Indexer Cluster Overview

- Manager node
  - Only one cluster manager permitted
  - Coordinates index replication
  - Distributes app bundles to peer nodes
  - Tells search head which peers to search
- Search head
  - Required component of indexer cluster
  - Relies on the manager for its target search peers
  - Works the same as any Splunk search head
- Peer nodes
  - Index data from inputs/forwarders
  - Replicate data to other peer nodes as instructed by the manager
- Forwarders
  - Send data to peer nodes



Ports for Indexer Clustering



Heartbeat, cluster management & searching (splunkd port)

Searching

Indexed data replication

Listening port for forwarded data input

Note

To participate in this indexer cluster, all nodes -- including the search head -- must use the same **`pass4SymmKey`**.

Indexer Cluster Considerations

Benefits

- Data availability and fast recovery
- Easier overall administration
  - Coordinated indexer configuration management
  - Automatic distributed search setup
  - Forwarder indexer discovery
- No additional cost for data replication

Trade Offs

- Increased storage requirements
- Increased processing load
  - Depending on the replication & search factors
- Requires additional Splunk instances
  - Minimum 1 separate instance for indexer cluster manager node
  - Minimum separate indexer for each desired copy (i.e: 3 copies <= 3 indexers)
- No support for heterogeneous indexers
  - Requires same OS and Splunk versions
- Requires cluster-specific deployment management

Indexer Cluster System Requirements

- Each node must run on its own host
- Manager node must run same or later version as peer nodes & search heads
  - At most three minor versions later than the peer nodes
- Search heads must run the same or later version as peer nodes
- All peer nodes must run EXACTLY the same version
- Peer node storage requirements:
  - Ability to sustain 800 IOPS for each peer node
  - Ratio of disks to disk controllers should mimic a database system requirement
- Cluster recovery depends upon system resources available on manager node

Indexer Cluster Deployment Overview

1. Identify clustering requirements
   - Replication policy, disk space, number of peer nodes, etc.
2. Install Splunk Enterprise and configure cluster instances
   - One manager, minimum two indexers, & one search head
   - Synchronize the system clocks on all machines
3. Enable clustering on each cluster instance
   - Use Splunk Web UI, CLI, or manually edit **`server.conf`**
4. Create and distribute configuration bundles to the peer nodes
5. Forward data to the peer nodes

Indexer Cluster Best Practices & Guidelines

- Share the same license pool w/ all members
- Dedicate hardware to manager node, search head, and each peer nodes
  - Do not share hardware across cluster instances
  - Allocate storage for each peer node
- Determine number of peer nodes by:
  - Expected availability requirements of your organization
  - Replication required, daily data rate, retention policy, & concurrent users
    - Note: Index replication does not increase your licensing usage

Note: Unable to use a deployment server to distribute configuration bundles directly to peer nodes

Manager Node Best Practices & Guidelines

- Install on a dedicated host
  - Cannot be shared with a peer node or search head instance
  - Automatically configured as a Search Head for indexers in the cluster
- Able to server additional roles under limited circumstances (e.g: CPU/memory):
  - License manager
  - Monitoring Console
  - Deployer

Single-site Cluster: Key Specifications

Key specifications are set under the **`[clustering]`** in **`server.conf`** on the Manager Node:

- Replication factor (**`replication_Factor`**)
  - Specifies how many total copies of **rawdata** the cluster should maintain
  - Sets the total failure tolerance level
- Search factor (**`search_Factor`**)
  - Specifies how many copies are searchable
    - A searchable bucket contains both **rawdata** and index files
    - Its **rawdata** is counted as a part of the replication factor
  - Cannot be larger than the replication factor
  - Determines how quickly you can recover the search capability
    - A trade-off between disk usage and search availability
- Security key (**`pass4SymmKey`**)
  - Authenticates communication between the cluster nodes
  - The key must be the same across all cluster instances

Configuring Splunk Clusters

- Use three approaches to configure clusters:
  - Splunk Web
  - CLI
  - **`server.conf`**
- Enable clustering on the instances in the following order:
  - Manager node > Peer nodes > Search heads
- Get help with Splunk cluster commands by running:
  - **`splunk help cluster`**
  - **`splunk help [list|edit] cluster-config`**

Configuring Splunk Manager Node

Enter the single command

```
> splunk edit cluster-config
-mode manager
-replication_factor 2
-search_factor 2
-secret <password#1>
```

Results in:

SPLUNK_HOME/etc/system/local/server.conf

```
[clustering]
mode = manager
replication_factor = 2
pass4SymmKey = <HASHED SECRET #1>
```

To see the decrypted **pass4SymmKey**, run:

splunk show-decrypted --value <HASHED SECRET#1>

- Splunk defaults to:
  - **replication_factor = 3**
  - **search_factor = 2**
- The **secret** parameter is encrypted and saved as **pass4SymmKey**
  - Required
  - **<password#1>** is the password for this cluster example

Configuring the Peer Nodes

```
> splunk enable listen 9997
> splunk edit cluster-config
-mode peer
-manager_uri https://10.0.1.3:8089
-secret <password#1>
-replication_port 9100
```

SPLUNK_HOME/etc/system/local/server.conf

```
[clustering]
mode = peer
manager_uri = https://10.0.1.3:8089
pass4SymmKey = <HASHED SECRET #1>

[replication_port://9100]
```

- Ports required on each peer:
  - Receiving port to listen to forwarders
  - Replication port to communicate with other peer nodes
- In this example:
  - **9997** = the forwarder listening port
  - **10.0.1.3** = manager node address
  - **8089** = manager node's splunkd-port
  - **<password#1>** = same cluster password
  - **9100** = index replication port

Configuring the Search Head

```
> splunk edit cluster-config
-mode searchhead
-manager_uri https://10.0.1.3:8089
-secret <password#1>
```

SPLUNK_HOME/etc/system/local/server.conf

```
[clustering]
mode = searchhead
manager_uri = https://10.0.1.3:8089
pass4SymmKey = <HASHED SECRET #1>
```

- To configure as a cluster search head:
  - **edit cluster-config**
- Functions as a regular search head
- For more help:

  splunk help [list|add|edit|remove] cluster-manager

Adding SH to an Additional Indexer Cluster

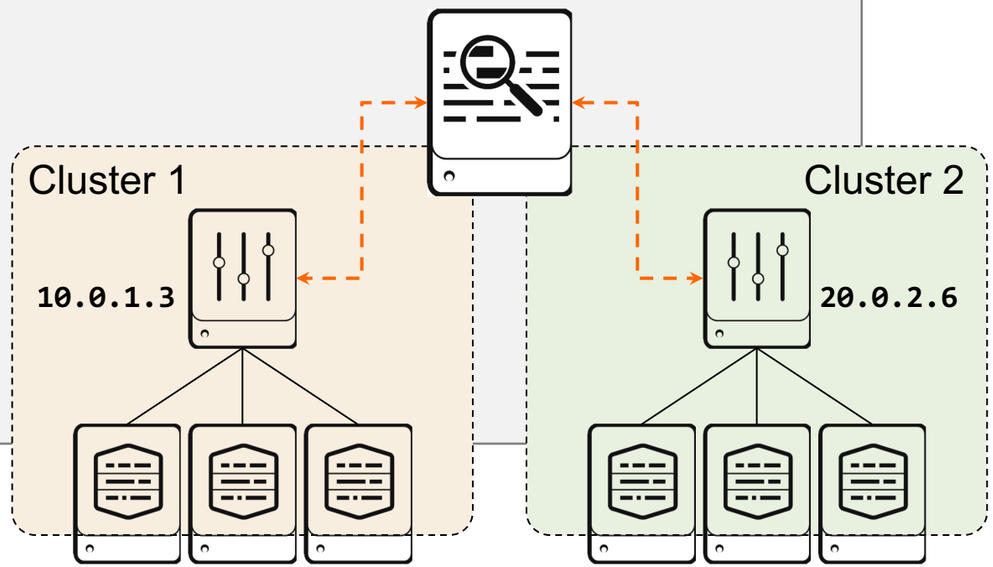
```
> splunk add cluster-manager
-manager_uri https://20.0.2.6:8089
-secret <password#2>
```

- SHs can belong to multiple clusters
- To allow SH to search additional clusters:

  splunk add cluster-manager

```
[clustering]
mode = searchhead
manager_uri = clustermanager:10.0.1.3:8089, clustermanager:20.0.2.6:8089

[clustermanager:10.0.1.3:8089]
manager_uri = https://10.0.1.3:8089
pass4SymmKey = <HASHED SECRET #1>

[clustermanager:20.0.2.6:8089]
manager_uri = https://20.0.2.6:8089
pass4SymmKey = <HASHED SECRET #2>
```

SPLUNK_HOME/etc/system/local/server.conf



Index Bucket Names

- Identify primary copies of a cluster's buckets to participate in search
  - Cluster Manager creates each new generation and assigns an ID
  - Peers track primary bucket copies in each generation
    - Non-searchable buckets will lack **`.tsidx`** files, metadata, etc.
- Change over time when one of the following occurs:
  - Peers leave and join the cluster
  - Primary copies are rebalanced
  - Primacy reassigned due to peer down (taken offline or crash)

File System Directory Names

Bucket type	Hot bucket	Warm/cold/thawed bucket
Non-clustered	`hot_v1_<localid>`	`db_<newest_time>_<oldest_time>_<localid>`
Clustered originating	`hot_v1_<localid>`	`db_<newest_time>_<oldest_time>_<localid>_<guid>`
Clustered replicated	`<localid>_<guid>`	`rb_<newest_time>_<oldest_time>_<localid>_<guid>`

Naming convention distinguishes type of copy:

- **db:** originating bucket
- **rb:** replicated copies
- **newest_time & oldest_time** = timestamps indicating age of bucket data
- **localid** = an ID for the bucket
- **guid** = globally unique identifier of source indexer

Note

The guid is located in the peer's:
`$SPLUNK_HOME/etc/instance.cfg`

Bucket location defined as homePath, coldPath, & thawedPath of index:

- Default: `$SPLUNK_HOME/var/lib/splunk/<indexname>/*`

 |

How the Cluster Manager Sees Buckets

Identifies buckets using:

- index name
- local ID
- guid of source indexer AKA “Bucket List”

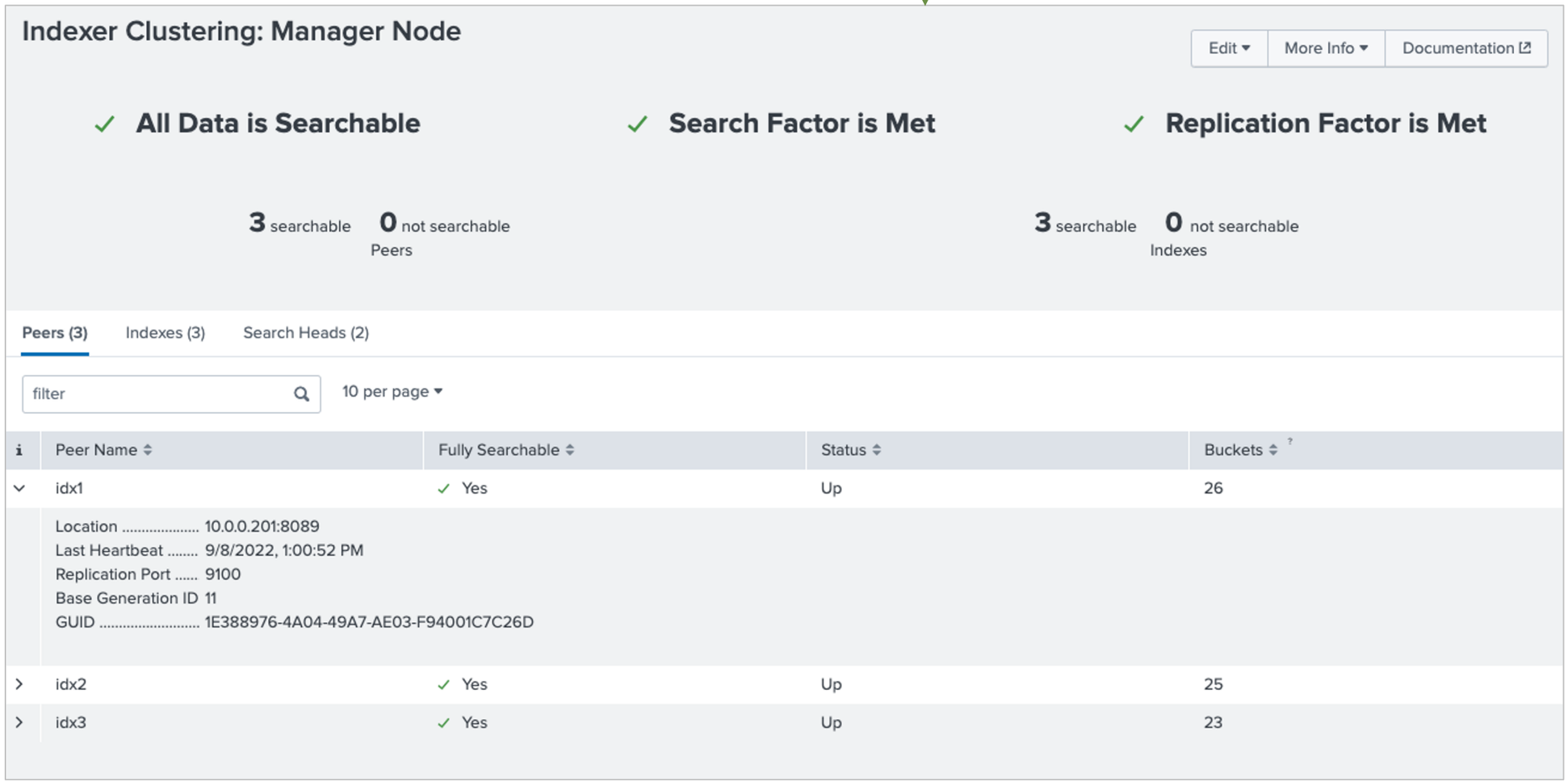
May identify buckets in two different ways:

- name: `_audit~3~1318C26A-D9FE-45F0-AF51-7A8038F5419C`
- separate field:

  index: `_audit,` ***bucket_id:*** `3_1318C26A-D9FE-45F0-AF51-7A8038F5419C`

Manager Dashboard – Single-site Cluster

On Manager Node SplunkWeb GUI: **Settings > Indexer clustering**



Index Replication Health

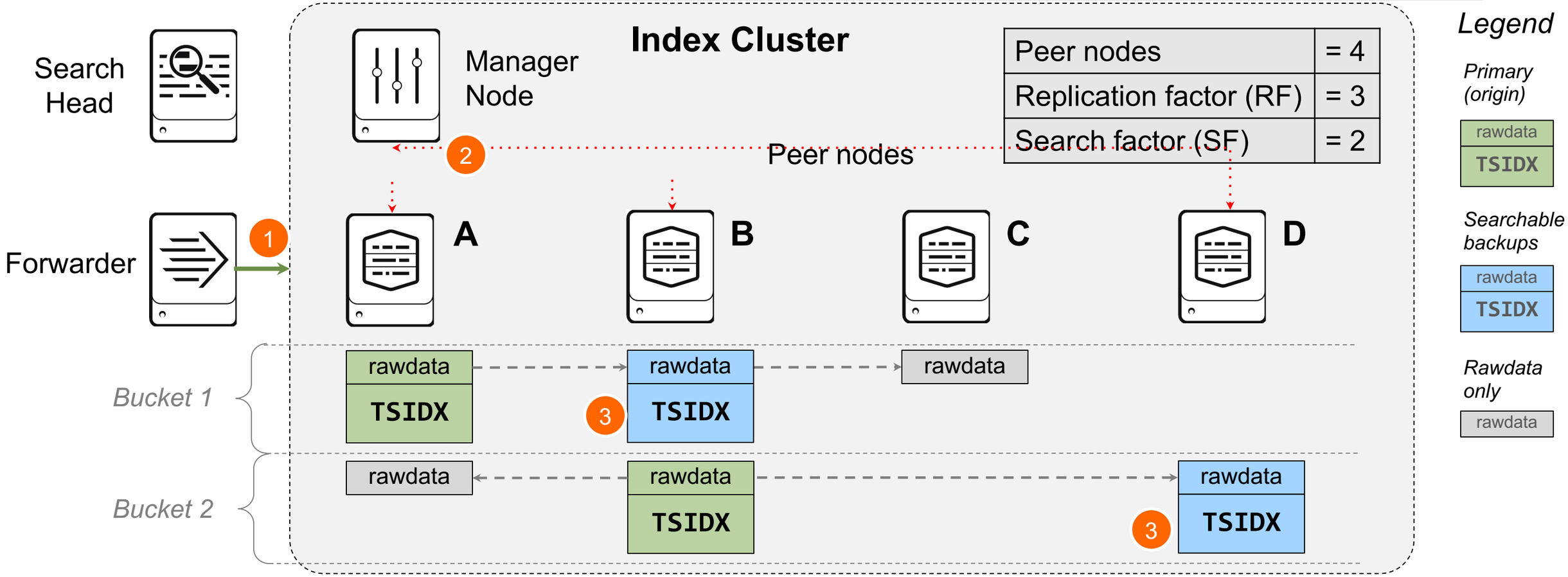
Cluster status is	When current operational RF is	When current operational SF is
Complete	Met as specified	Met as specified
Valid	Not required to be met	1 or greater

- With replication factor = 3 and search factor = 2:
  - A **complete** cluster has 3 copies of each bucket, 2 of which are searchable
  - A **valid** cluster has at least one searchable copy of all buckets
- With replication factor = 2, search factor = 2, and 2 peers:
  - A **complete** cluster has 2 searchable buckets, each having its copy of rawdata
  - A **valid** cluster has at least one searchable copy of all buckets

Factors in Action – Data Replication

Complete & Valid

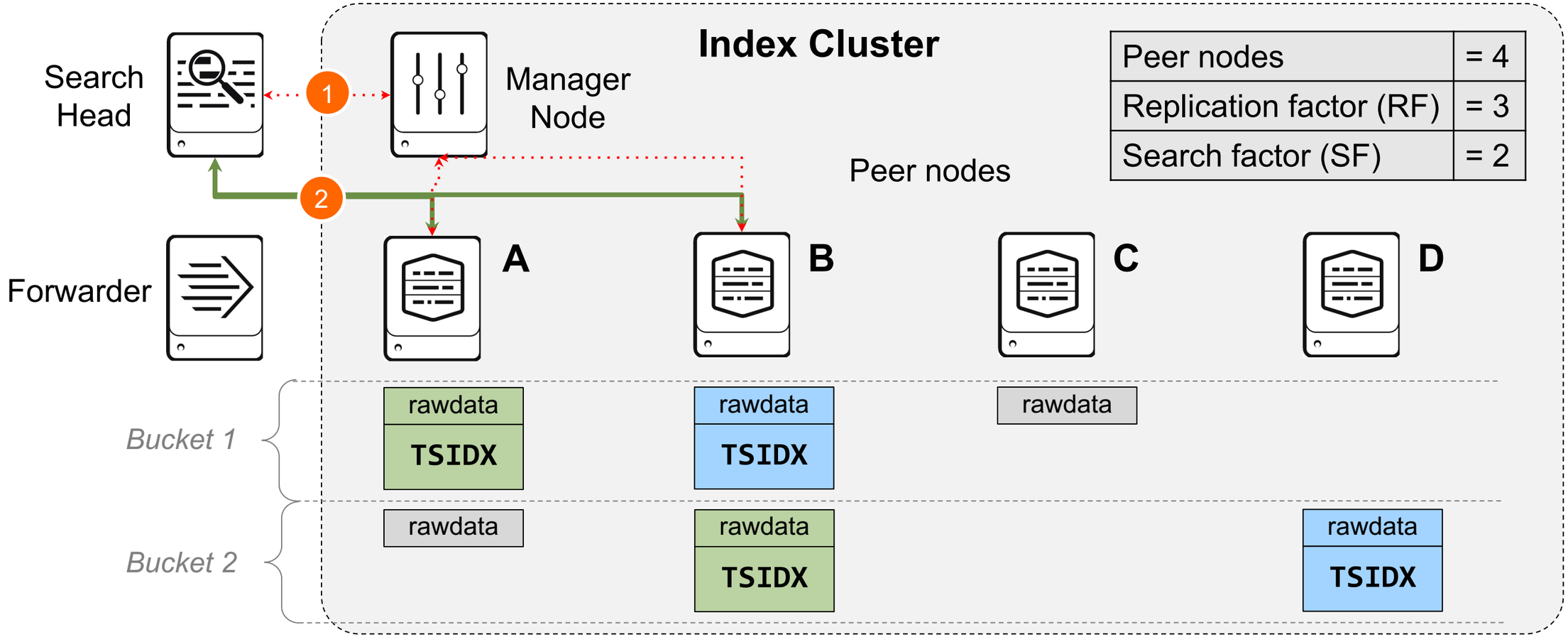
1. The peer receiving the data has the original bucket (Primary)
2. The Manager Node allocates replication jobs to the original peer to copy rawdata to randomly selected peers to meet RF
3. The Manager Node allocates jobs to random peers with a rawdata copy to build the `tsidx` file locally



Factors in Action – Search

Complete & Valid

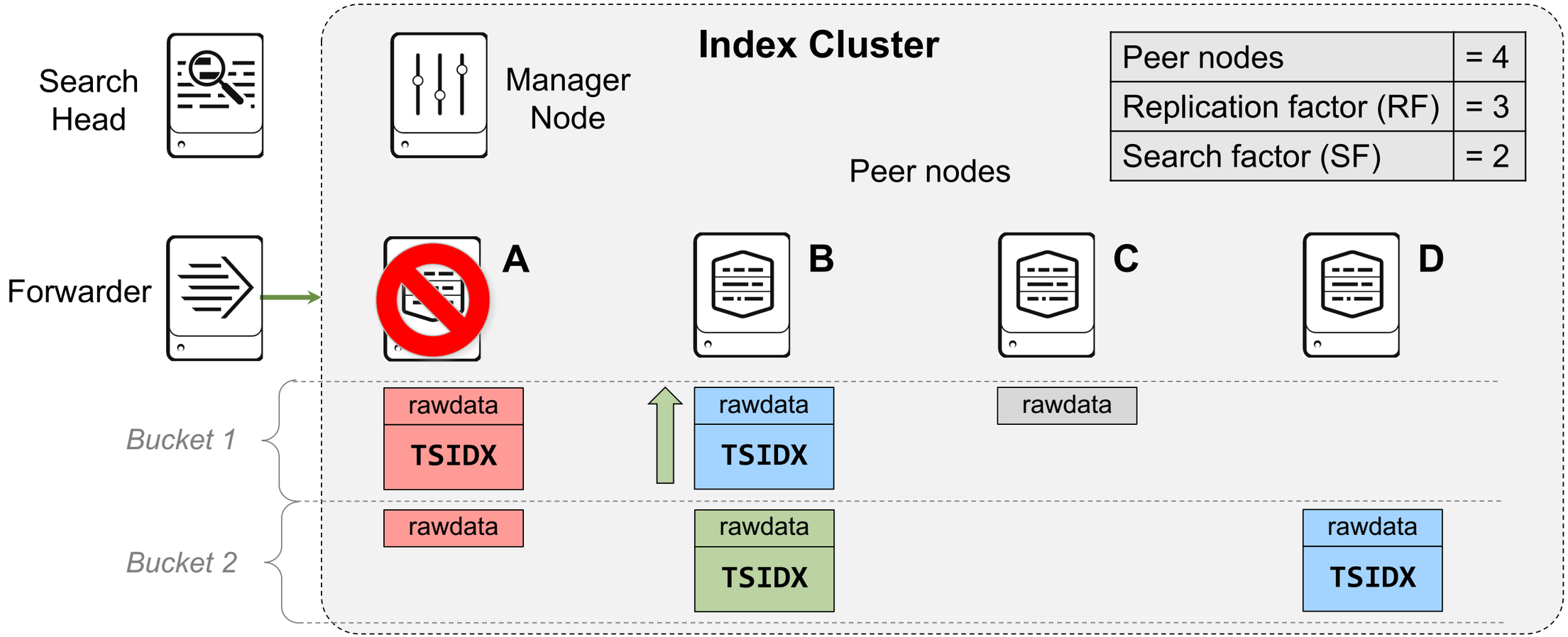
When a user initiates a search:
1. Manager Node responds with a manifest of current primary peers
2. Search head dispatches search job to the peers with primary buckets



Factors in Action – Initial Primary Loss

Valid but Not Complete

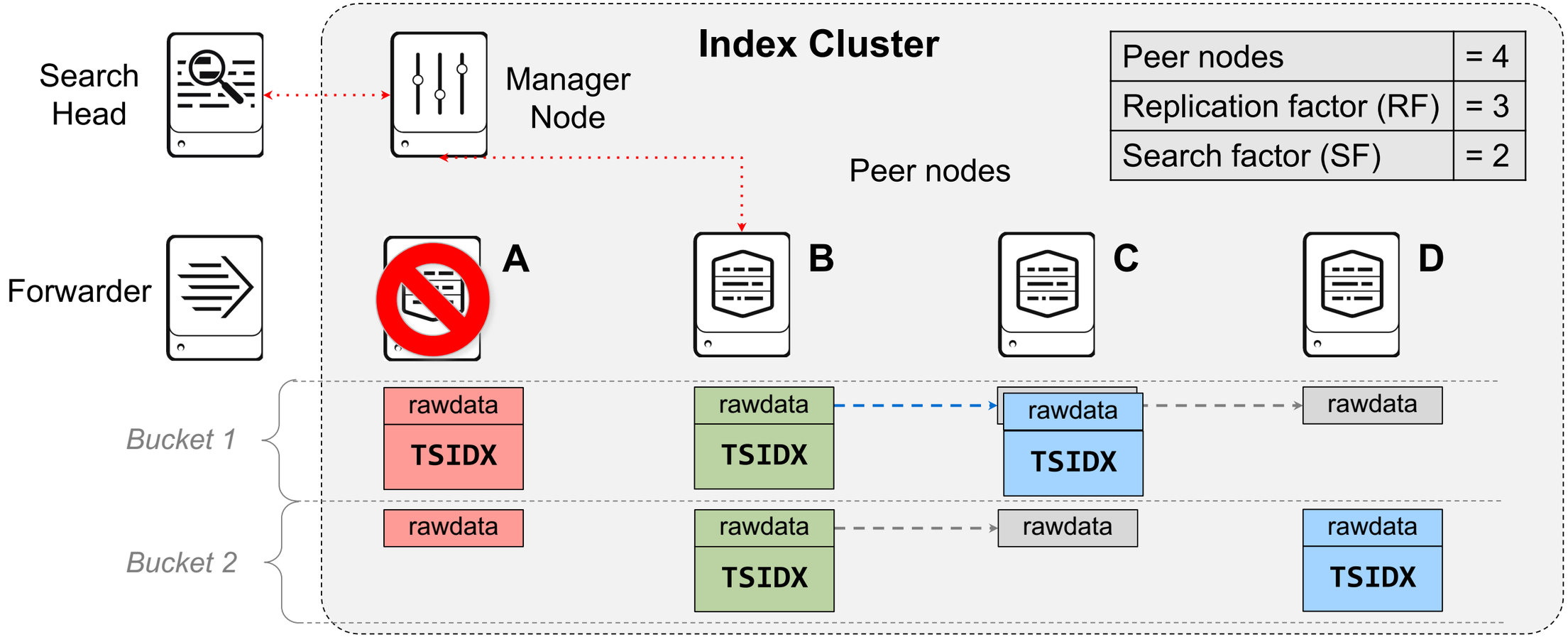
In the scenario of losing Peer A, the Manager Node randomly instructs the peers with Searchable Backups to promote them to Primary



Factors in Action – Initial Primary Loss (cont.)

Complete & Valid

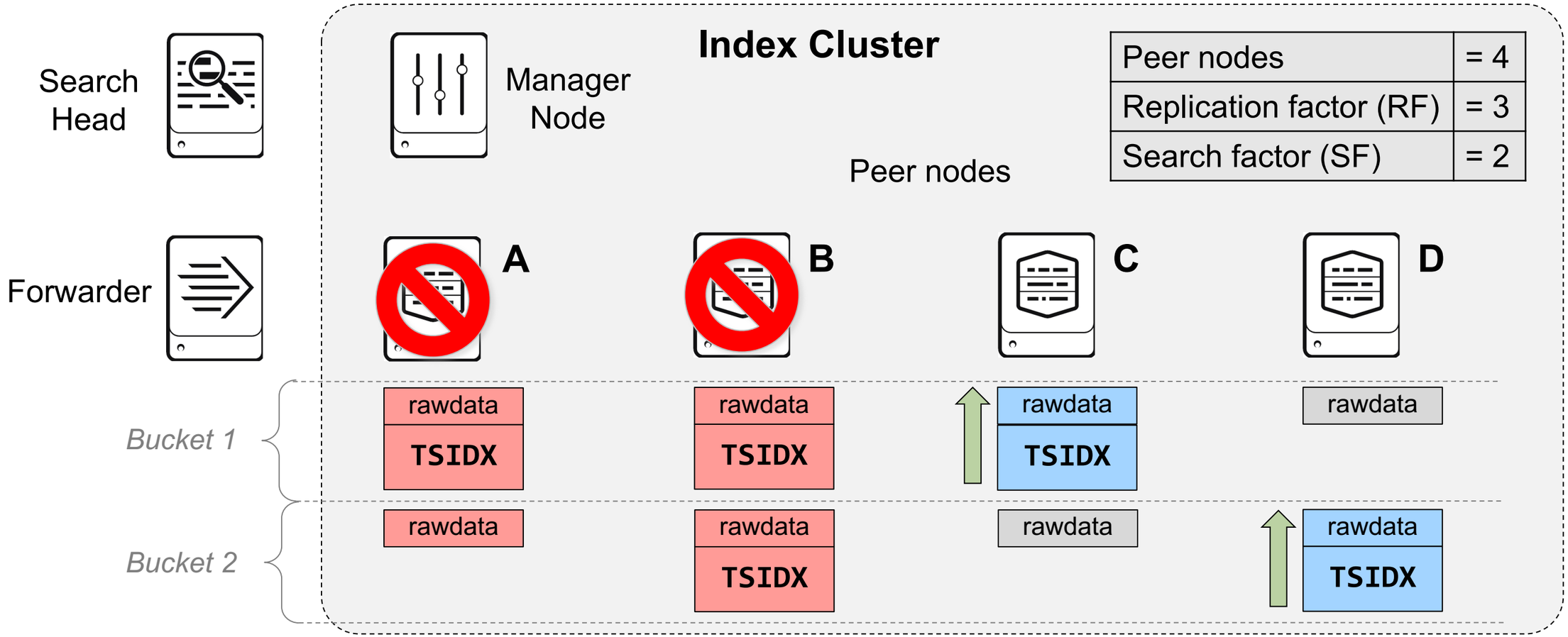
The cluster will attempt to establish the RF/SF by copying the **`.tsidx`** files to the **Rawdata Only** buckets (promote them to **Searchable Backups**) and copying rawdata to the remaining peers



Factors in Action – Second Peer Loss

Valid but Not Complete

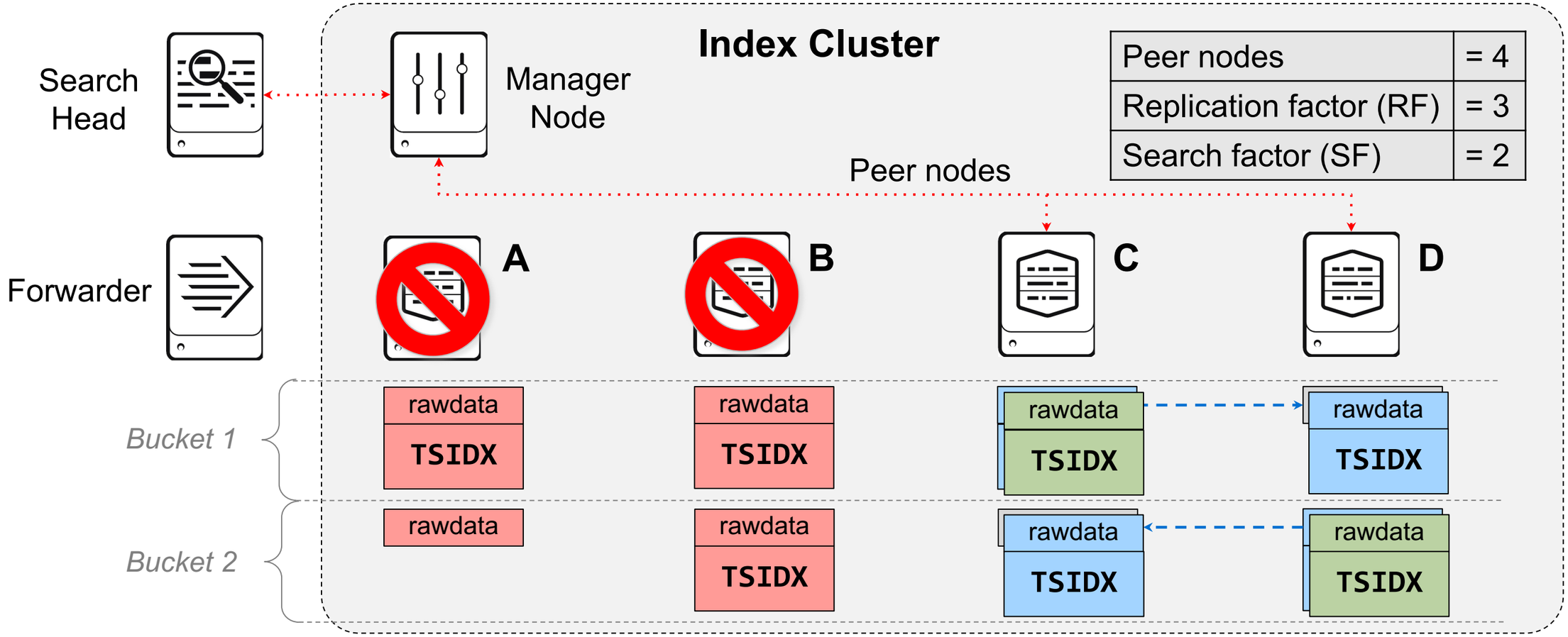
In the scenario of losing two peers, searchable backups are promoted to primary copies



Factors in Action – Second Peer Loss (cont.)

Valid but Not Complete

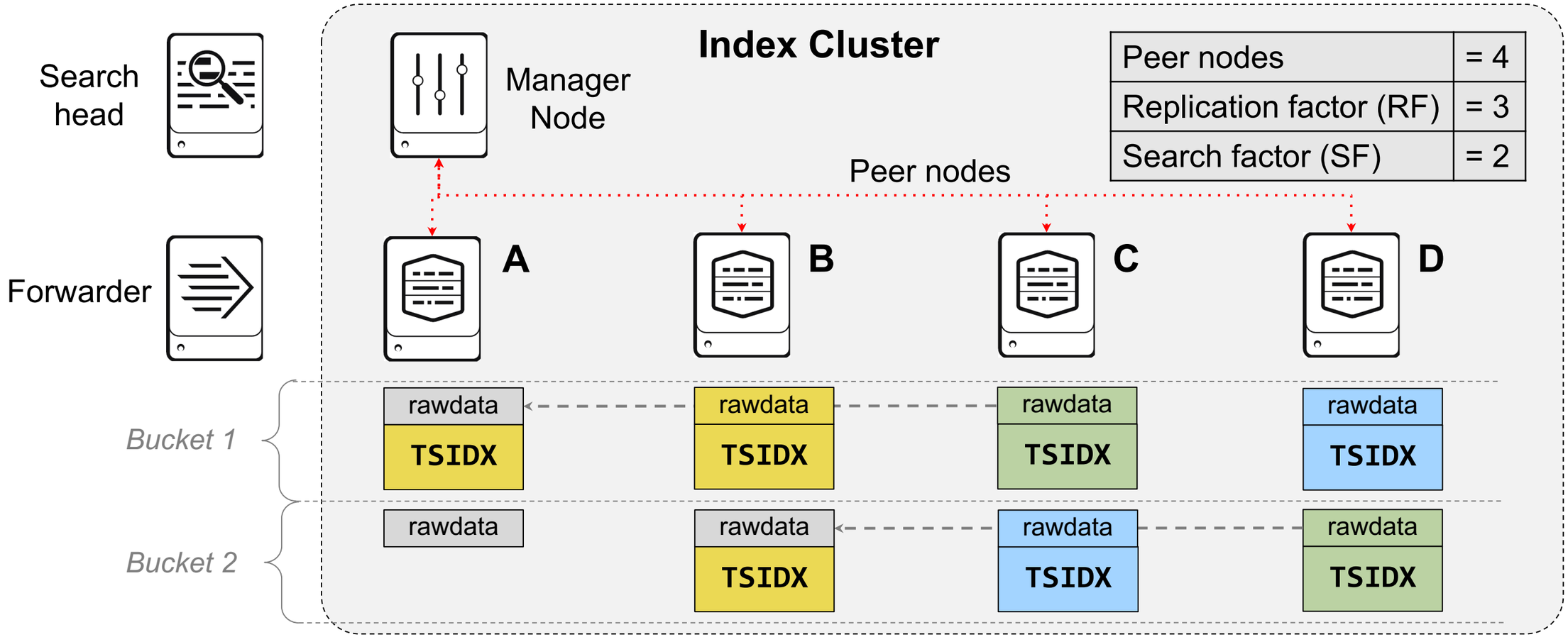
Rawdata copy promoted to searchable backup to meet SF, but RF cannot be met because the extra storage to maintain the copies does not exist



Factors in Action – Rebalancing

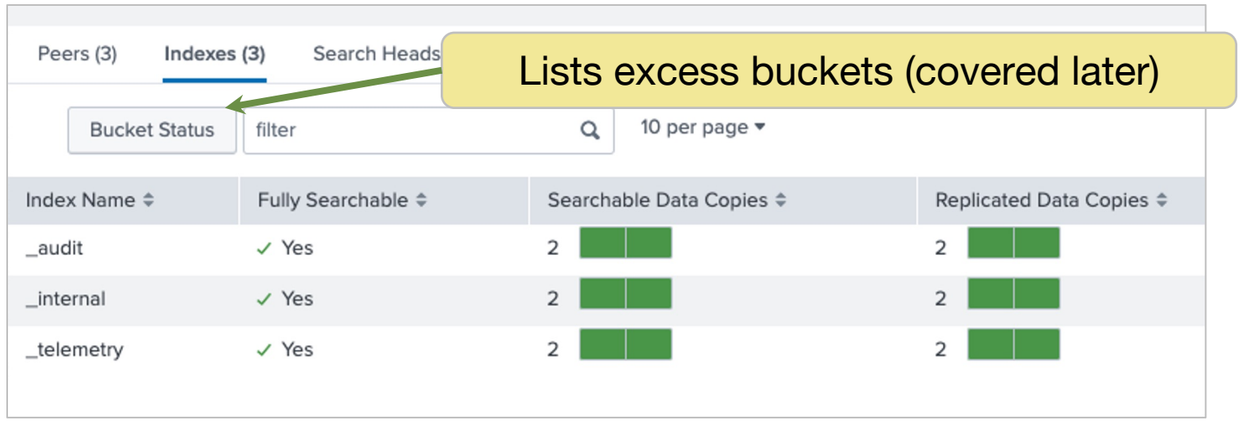
Complete & Valid After Rebalancing

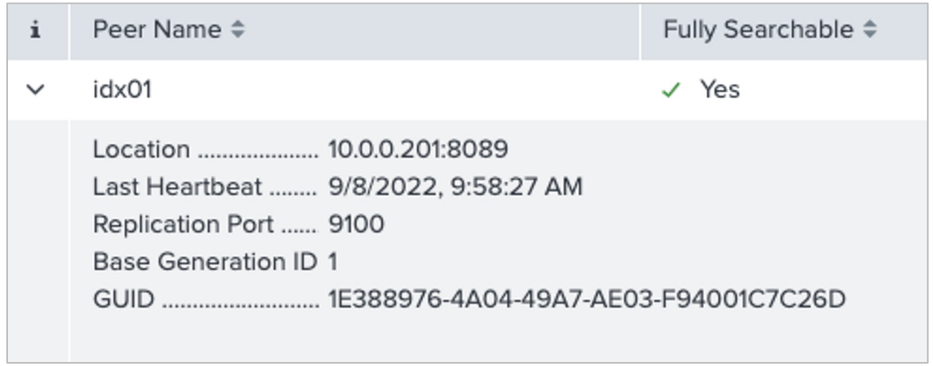
- If the peers come back online, Splunk will automatically perform primary rebalancing and reassign primary buckets
- There will only be one primary at a time which will result in excess copies



Excess Buckets

- Manager Node tracks excess buckets by generation ID
- Generation ID of cluster increases when Manager Node initiates primary bucket fix-up
- List of excess buckets (and cleaning utilities) found under **Bucket Status**





Notable Indexer Cluster Log Channels

- Cluster peers communicate via the **`/services/cluster`** endpoints
  - **`/services/cluster/manager:`** peer to manager communication
  - **`/services/cluster/peer:`** manager to peer communication
- splunkd_access.log: Indexer cluster communication logs

```
index=_internal sourcetype=splunkd_access
(uri="/services/cluster/peer/buckets*" OR uri="/services/cluster/manager/buckets*")
| convert ctime(_time) | table _time uri_path | sort _time
```

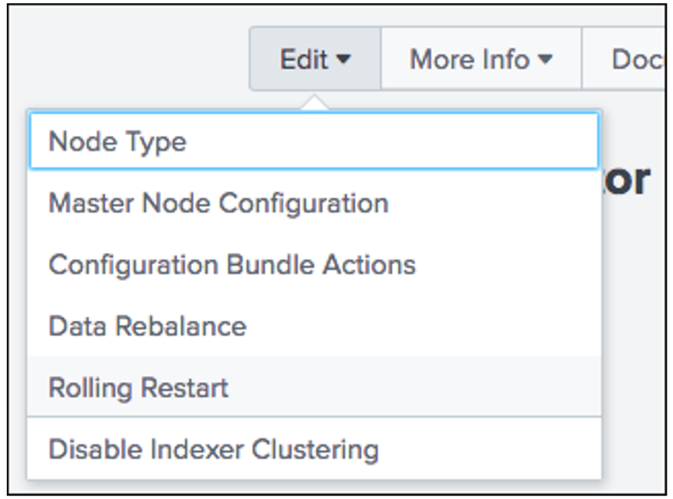
  - Higher response time indicates service overloading
  - Response status 200 is good, anything else is not good
- splunkd.log: indexer clustering activity logs
  - **`component=CM* OR component=Cluster*`**
  - Look for **`WARN/ERROR`** on the **`host=<manager>`**
- metrics.log
  - **`component=Metrics group=clusterout_connections`**

How to Restart an Indexer Cluster

If both manager and peers need to be restarted:

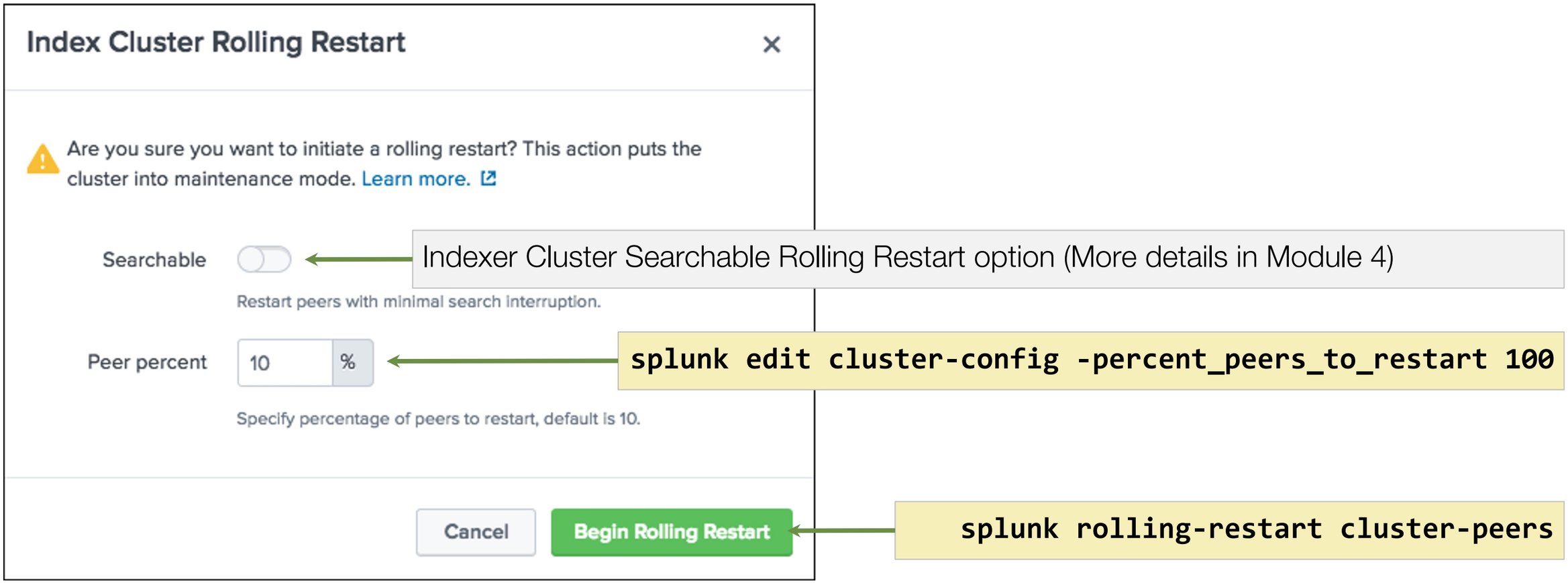
1. Restart the manager node with **`splunk restart`**
2. Run preliminary health checks
   - Check the manager dashboard for cluster status
   - Run: **`splunk show cluster-status --verbose`**
3. Restart peer nodes:
   - From Manager Node SplunkWeb UI:
     - Select **Edit > Rolling Restart**
   - CLIF
     - Run: **`splunk rolling-restart cluster-peers`**

> **Note**
>
> Restarting an entire cluster is not ordinarily required. Manager and peers can be updated in sets per configuration changes.



How a Rolling Restart Works

- Performs a phased restart of all the peer nodes
  - Restarts 10% of the peers at a time in random order (configurable)

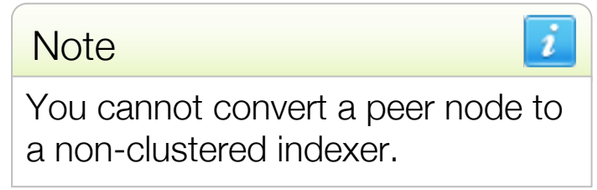


Migrating Non-clustered Indexers to a Cluster

- You can add a non-clustered indexer to a cluster as a peer node at any time

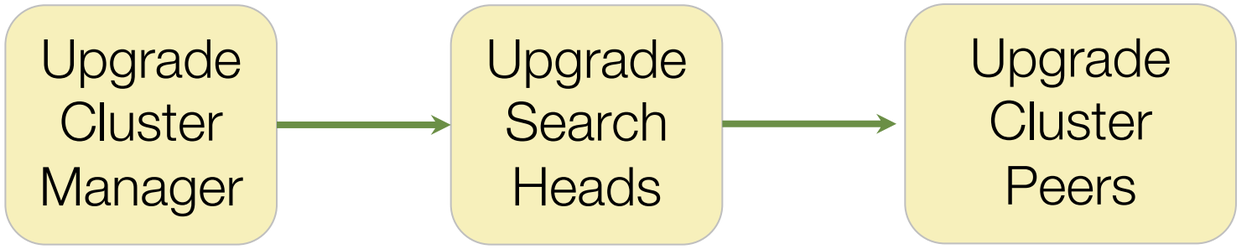
```
splunk edit cluster-config -mode peer ...
```

- Apps must be re-distributed via **manager-apps**
  - You learn about distributing apps in indexer clusters in Module 4
- Only new data coming into this peer is replicated
  - New data follows the cluster's replication factor
- Existing buckets are not replicated
  - Contact Splunk Professional Services if you must replicate legacy buckets



Upgrading and Applying Maintenance Releases

- The upgrade process differs considerably depending on the nature of the upgrade
- Always read the latest documentation first!



Note

- Manager node must run the highest version
- Search head must run higher version than the peer nodes
- Complete the entire process quickly

Further Reading: Single-site Indexer Cluster

- System requirements and other deployment considerations for indexer clusters
- Upgrade an indexer cluster
- server.conf (Spec)

Lab Exercise 2 – Enable Single-site Cluster

Time: 30 minutes

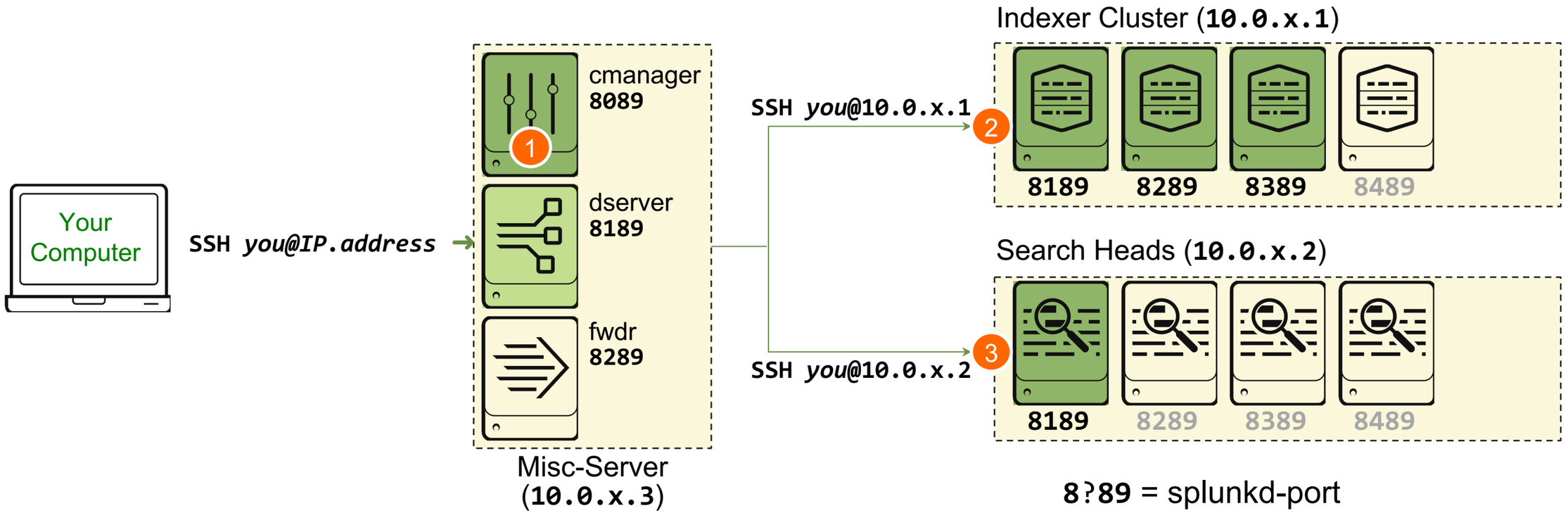
Tasks:

- Start all cluster members and configure licensing
- Configure the manager node for a single-site indexer cluster
- Configure three indexers to form the replication peers
- Configure a search head to join the cluster
- Monitor the cluster status with Splunk Web

Optional Tasks: (time permitting)

- Test a peer node failover scenario
- Investigate the peer outage with Splunk internal logs

Lab Exercise 2 – Enable Single-site Cluster (cont.)



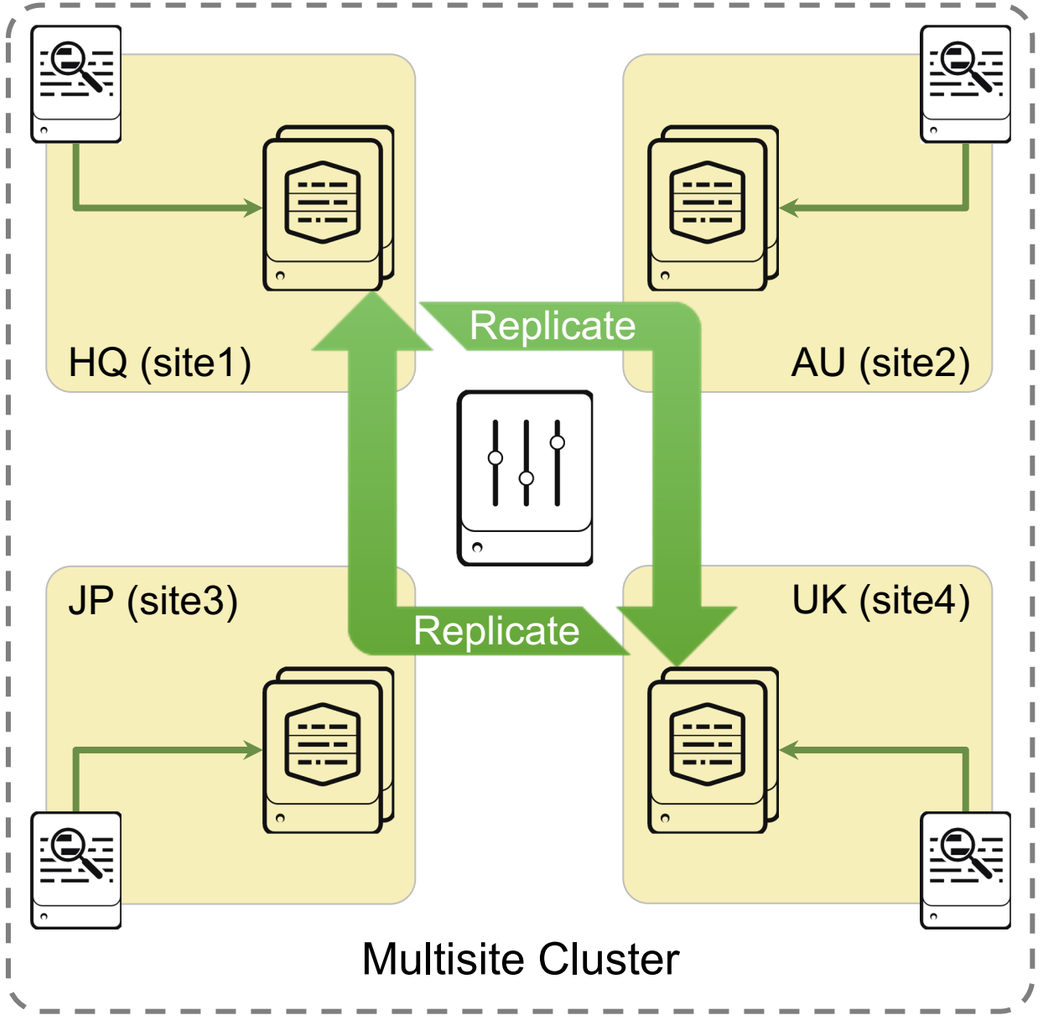
Module 3:
Multisite Indexer Cluster

Module Objectives

- Describe how Splunk multisite indexer clusters work
- Identify multisite indexer cluster terms
- Implement a multisite indexer cluster
- Describe optional configuration settings

Key Benefits of Multisite Indexer Cluster

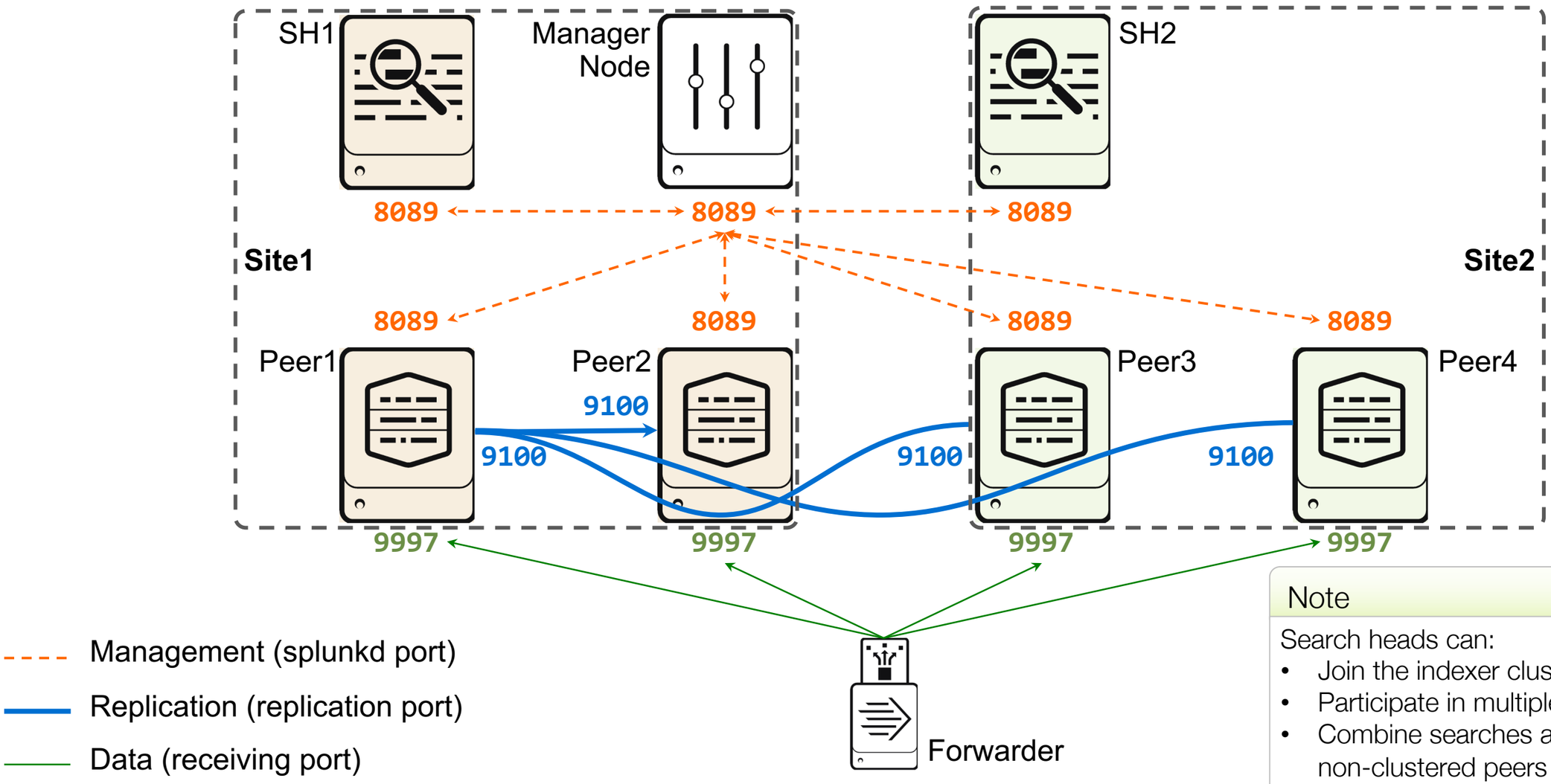
1. Improved disaster recovery
   - Stores index copies at multiple sites (i.e. geo-location or rack)
   - Provides automatic site-failover capability
   - In case of a disaster, indexing and search continue on surviving sites
2. Search affinity
   - Preferentially searches assigned site
   - Greatly reduces WAN network traffic



Multisite Indexer Cluster Deployment

1. Determine multisite cluster use cases and requirements
2. Install Splunk Enterprise and configure cluster instances
   - One manager node
   - At least two peer nodes per site
   - One or more search heads per site (recommended)
3. Enable clustering on the instances using Splunk CLI or edit **`server.conf`** in the following order:
   - Manager Node > Peer Nodes > Search Heads
4. Create and distribute the configuration bundle to the peer nodes
5. Configure forwarders to send data to the peer nodes

Multisite Cluster Topology



Note

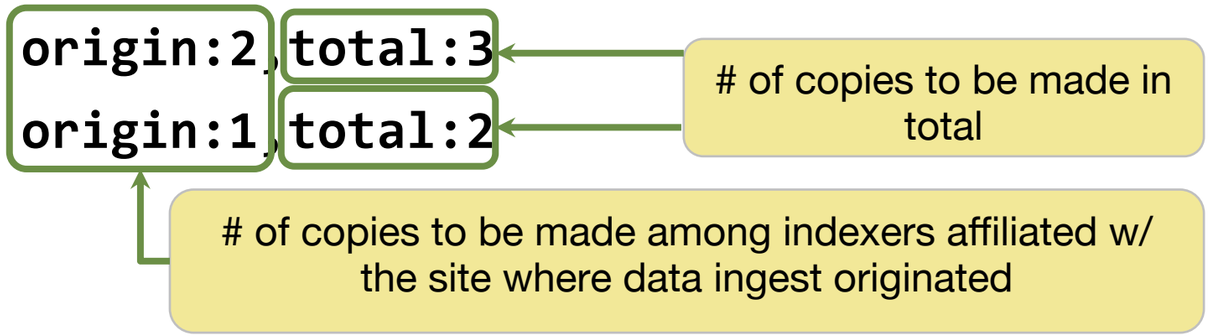
Search heads can:

- Join the indexer cluster any time
- Participate in multiple clusters
- Combine searches across clustered and non-clustered peers

Multisite Indexer Cluster server.conf Attributes

Property Name	Description
multisite	Enables multisite clustering
site	- The site *affiliation* (indexers) or site *affinity* (SH, Fwdrs) for host - All cluster members are assigned a site, including the manager node
available_sites	- **site1, site2, …site<*n*>** - Supports 1-63 sites
site_replication_factor	# of raw copies of data among the sites
site_search_factor	# of raw copies also including tsidx (searchable copies)

- **site_replication_factor** **origin:2,total:3**
- **site_search_factor** **origin:1,total:2**



Examples of Site Replication Factor

Configuration	Description
`origin:2, total:3`	Default. Put the extra copy on a site that doesn't have a copy
`origin:1, total:4` (where there are 4 sites)	Try to put a copy on any site that doesn't have one
`origin:2, site1:2, total:5`	Both site1 and origin have a minimum of 2 copies
`origin:2, site1:1, total:4`	If origin happens to be site1, then the higher value takes precedence
`origin:2, site1:2, site2:2, total:3`	Invalid

Note

- You must specify the origin
- A factor count must be greater than 0

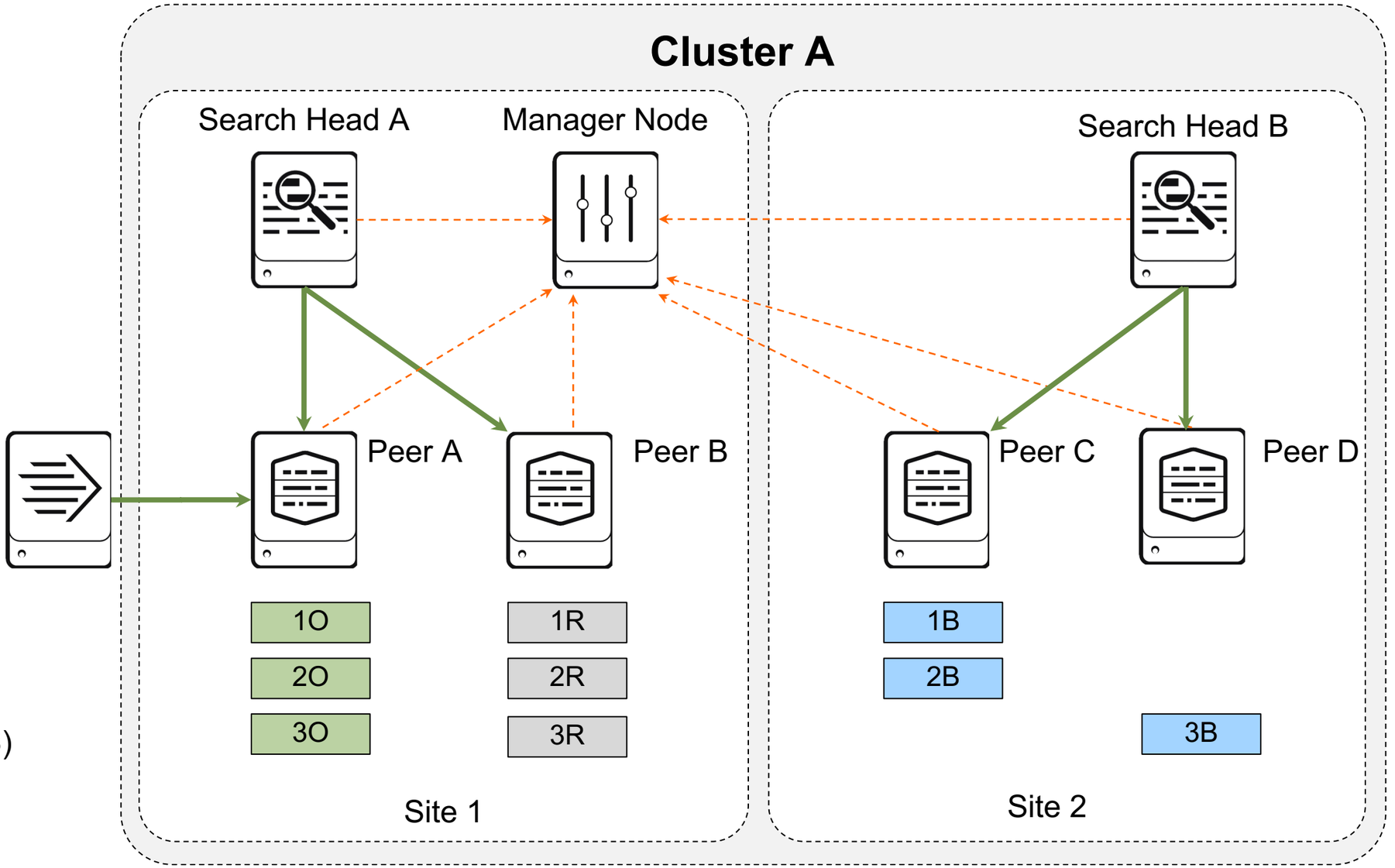
Multisite Factors in Action

Sites: 2
Peer Nodes: 4
Search Heads: 2
Manager Node: 1
Peer Nodes: 4

`site_replication_factor`
`origin: 2`
`total: 3`

`site_search_factor`
`origin: 1`
`total: 2`

Complete & Valid



rawdata / TSIDX — Origin (O)

rawdata / TSIDX — Searchable Backup (B)

rawdata — Rawdata Only (R)

Multisite Factors in Action – Primary Loss

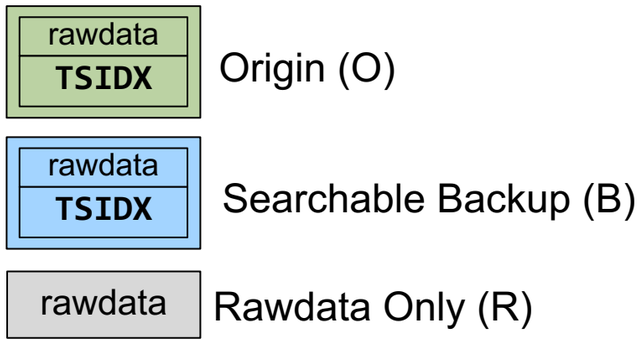
Sites: 2
Peer Nodes: 4
Search Heads: 2
Manager Node: 1
Peer Nodes: 4

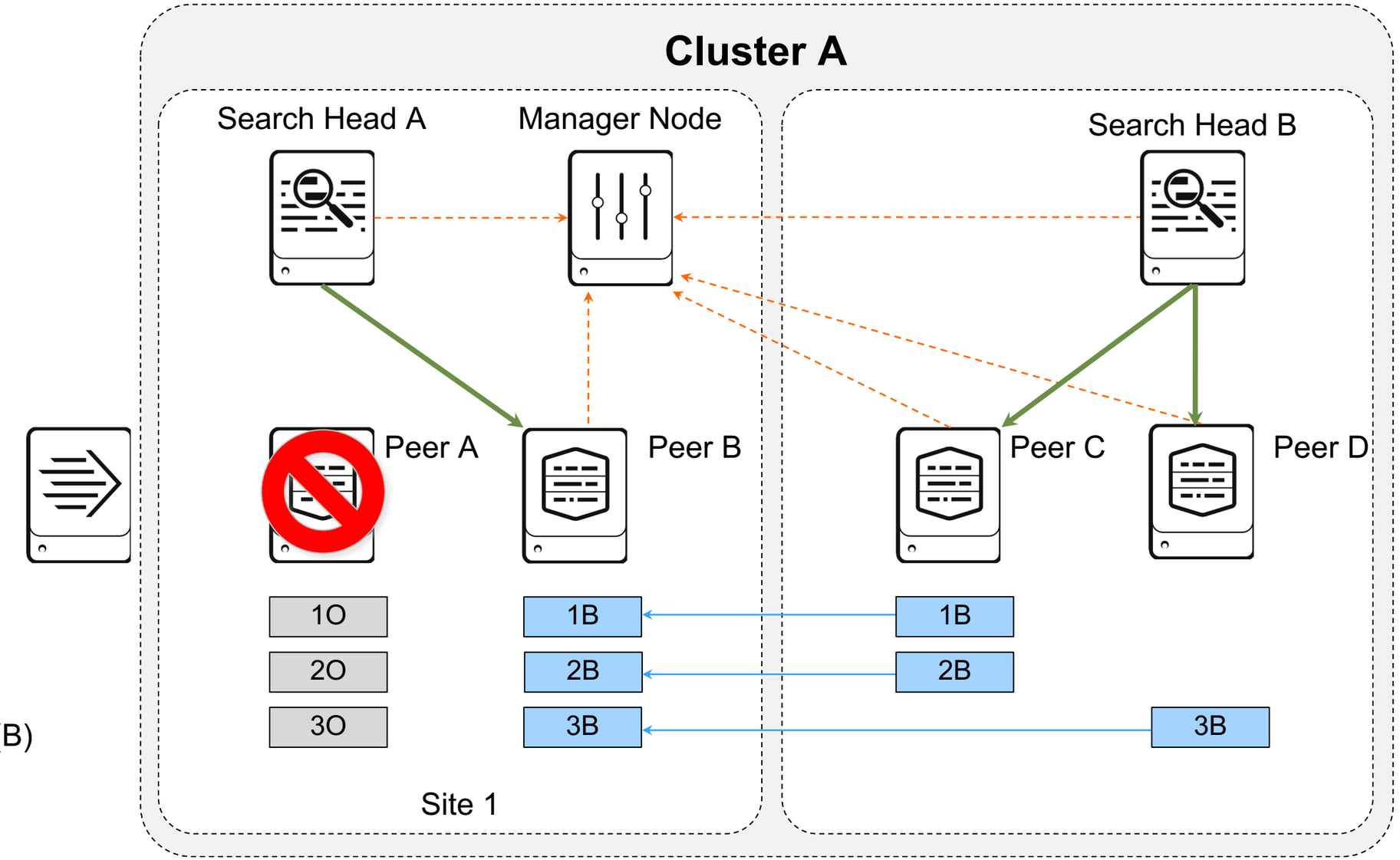
`site_replication_factor`
`origin: 2`
`total: 3`

`site_search_factor`
`origin: 1`
`total: 2`

Valid But Not Complete

origin:2 is not possible





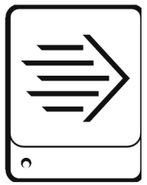
Multisite Factors in Action – Site Loss

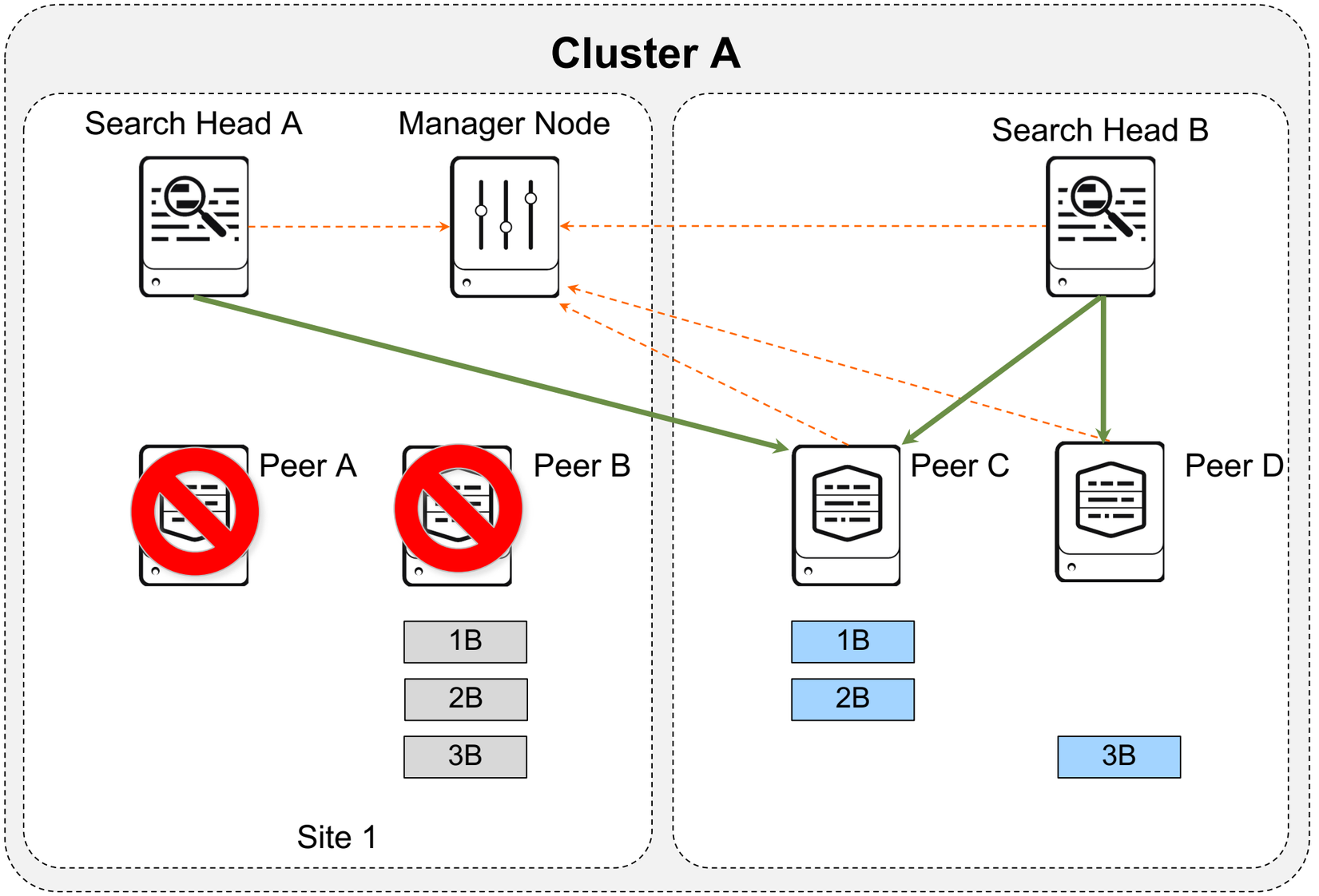
Sites: 2
Peer Nodes: 4
Search Heads: 2
Manager Node: 1
Peer Nodes: 4

`site_replication_factor`
`origin: 2`
`total: 3`

`site_search_factor`
`origin: 1`
`total: 2`

Valid But Not Complete





rawdata / TSIDX — Origin (O)

rawdata / TSIDX — Searchable Backup (B)

rawdata — Rawdata Only (R)

Configuring the Multisite Manager Node

```
splunk edit cluster-config -mode manager -multisite true -site site1
-available_sites site1,site2 -site_replication_factor origin:1,total:2
-site_search_factor origin:1,total:2 -secret <password>
```

SPLUNK_HOME/etc/system/local/server.conf

```
[general]
site = site1

[clustering]
multisite = true
mode = manager
available_sites = site1,site2
site_replication_factor = origin:1,total:2
site_search_factor = origin:1,total:2
pass4SymmKey = <hashed secret>
```


 |

Configuring Multisite Cluster Peer Nodes

```
splunk edit cluster-config -manager_uri https://10.0.1.3:8089
-mode peer -site site1 -replication_port 9100 -secret <password>
```

Peer1&2

```
splunk edit cluster-config -manager_uri https://10.0.1.3:8089
-mode peer -site site2 -replication_port 9100 -secret <password>
```

Peer3&4

Peer1&2 server.conf

```
[general]
site = site1

[clustering]
mode = peer
manager_uri https://10.0.1.3:8089
pass4SymmKey = <hashed secret>

[replication_port://9100]
```

Peer3&4 server.conf

```
[general]
site = site2

[clustering]
mode = peer
manager_uri https://10.0.1.3:8089
pass4SymmKey = <hashed secret>

[replication_port://9100]
```

Configuring Multisite Cluster Search Heads

If SH is not already a cluster search head, enable it first:

- `splunk edit cluster-config -mode searchhead ...`

To add the search head to another indexer cluster, run:

- `splunk add cluster-manager <manager_uri:port> ...`

If you need to change the clustering configuration or attributes:

- `splunk edit cluster-manager <manager_uri:port> ...`

Configuring a New Multisite Search Head

Enable a new or existing instance as a multisite cluster search head

```
splunk edit cluster-config \
-mode searchhead  \
-manager_uri https://10.0.1.3:8089  \
-site site2 \
-secret <password>
```

```
[general]
...
site = site2

...

[clustering]
manager_uri = https://10.0.1.3:8089
mode = searchhead
multisite = true
pass4SymmKey = <hashed secret>
```

server.conf

Configuring an Existing SH as Multisite

Add a single-site cluster to SH

```
splunk add cluster-manager https://20.0.2.6:8089 \
-multisite false -secret <password#1>
```

Convert existing single-site SH to multisite

```
splunk edit cluster-manager https://10.0.1.3:8089 \
-multisite true -site site1 -secret <password#2>
```

```
[clustering]
manager_uri = clustermanager:10.0.1.3:8089,
  clustermanager:20.0.2.6:8089
mode = searchhead

[clustermanager:20.0.2.6:8089]
manager_uri = https://20.0.2.6:8089
multisite = false
pass4SymmKey = <hashed secret #1>

[clustermanager:10.0.1.3:8089]
manager_uri = https://10.0.1.3:8089
multisite = true
pass4SymmKey = <hashed secret #2>
site = site1
```

server.conf

Search Affinity

- Search heads only get results from peer nodes on their (valid) local site
  - If a searchable bucket exists, it will be the primary bucket for that site
  - Searches extend across sites only when needed
- Reduces network traffic while providing access to full data set
  - Limits access of users to their local search heads
- Enabled by default

Three modes of search affinity:

- Single-site mode: only one set of “primary” searchable buckets
- Multisite mode: supports searchable replicas for each site
- Single peer: debugging purposes

Disabling Search Affinity on SH: `site=site0`

Edit the search head configuration:

```
splunk edit cluster-manager https://10.0.55.3:8089 -site site0
```

```
[clustermanager:10.0.55.3:8089]
manager_uri = https://10.0.55.3:8089
multisite = true
pass4SymmKey = Hashed_Secret
site = site0
```

SH2 server.conf (Search affinity disabled)

> **Note**
>
> All sites must be in close proximity with very low network latency

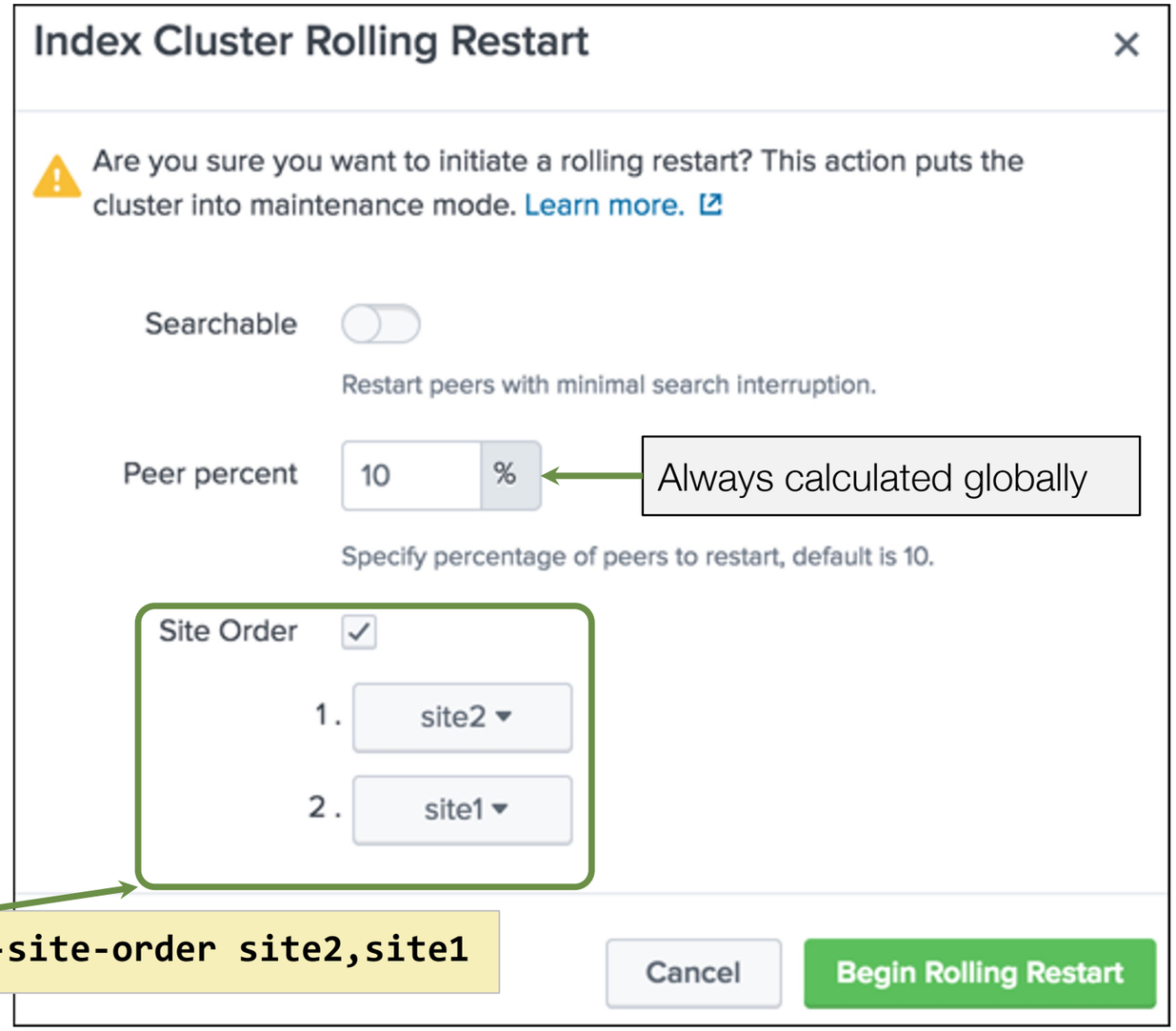
Spreads the search request across indexers on all sites

- Increases WAN traffic

Restarting a Multisite Indexer Cluster

If a cluster restart is required:

1. Restart the manager node
2. Run preliminary health checks
    1. Check cluster status on manager dashboard
3. Begin rolling restart
    1. Multisite rolling restart proceeds across all indexers
    2. Specify site order (optional):




```
splunk rolling-restart cluster-peers -site-by-site true -site-order site2,site1
```

Migrating Single-site to Multisite Cluster

1. Ensure all cluster nodes are running the same Splunk Enterprise version
2. Change the manager node to multisite mode and restart
   - Do not remove the existing single-site replication factor and search factor
   - New multisite factors must be at least as large as the single-site factors
3. Enable maintenance mode on the manager:

   ```
   splunk enable maintenance-mode
   ```

4. Change peer nodes to multisite mode with a site association and restart

   ```
   splunk edit cluster-config site=siteN
   ```

   - Do not restart a peer that hasn't been converted
5. Change search heads to multisite mode and restart:

   ```
   splunk edit cluster-manager https://CManager:8089 -multisite true -site site1 –secret <password>
   ```

6. Disable maintenance mode on the manager:

   ```
   splunk disable maintenance-mode
   ```

Indexer Cluster Maintenance Mode

Invoke when work on a peer node may cause excessive bucket status changes:

`splunk [enable|disable|show] maintenance-mode`

Considerations:

- Bucket fixup discontinues copying buckets between peers to maintain complete state
- Only reassigns primaries to maintain a valid state
- Peer outage while in maintenance mode may return incomplete results
- Maintenance mode persists across manager restarts
- No notion of sites; works for both indexer cluster configurations
- Bucket rolling that would occur due to peer outage is discontinued
- Manager node catches up on replication policies after maintenance mode disabled

The following commands automatically invoke maintenance mode:

- **`splunk apply cluster-bundle`**
- **`splunk rolling-restart`**

Bucket Behavior After Migration

By default, cluster holds buckets separately using the following rules:

- Single-site legacy buckets continue to follow the single-site **`replication_factor`** and **`search_factor`** settings
- New multisite buckets follow the multisite **`site_replication_factor`** and **`site_search_factor`** policies

If you want single-site legacy buckets to convert and use multisite replication, set **`constrain_singlesite_buckets=false`** in manager node **`server.conf:`**

- Restart required

```
[clustering]
manager_uri = https://10.0.1.3:8089
mode = searchhead
multisite = true
pass4SymmKey = <hashed secret>
constrain_singlesite_buckets=true
```

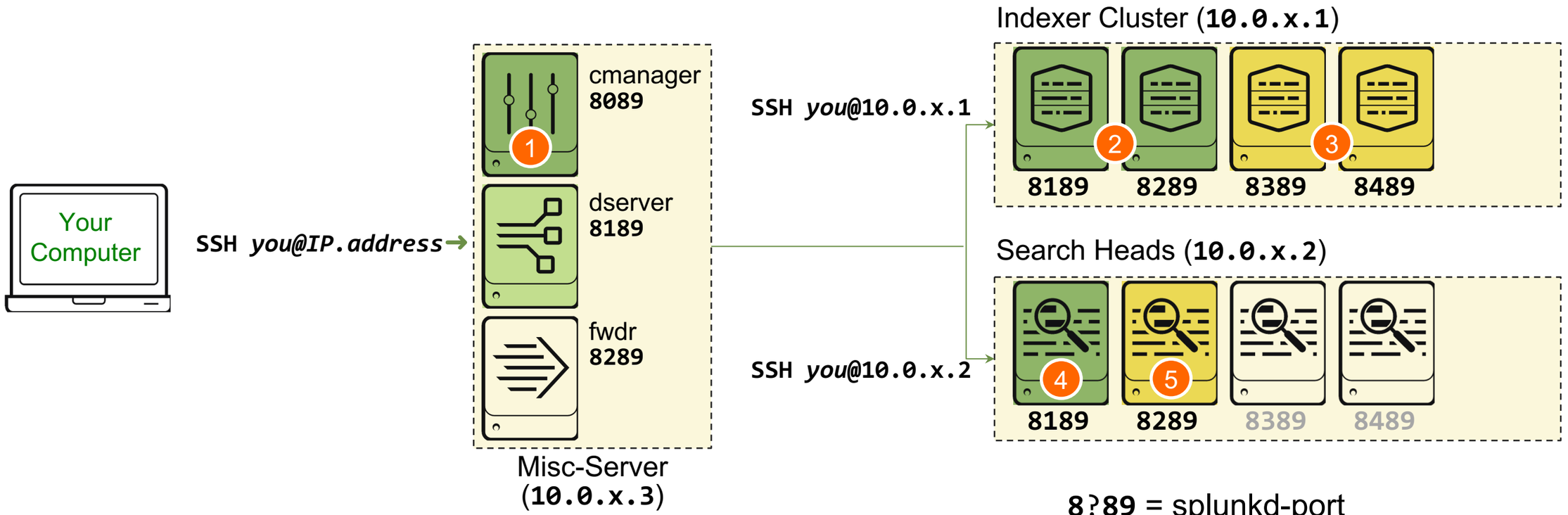

Further Reading: Multisite Indexer Cluster

- Basic clustering concepts for advanced users
- Replace the manager node on the indexer cluster
- Search head configuration overview

Lab Exercise 3 – Migrate to Multisite Cluster

- **Time:** 30 minutes
- **Tasks:**
  - Migrate the single-site manager node to the multisite mode
  - Migrate the existing peer nodes and add a new peer
  - To configure **`site2`**, convert **`IDX3`** and add a new peer **`IDX4`**
  - Convert **`SH1`** to **`site1`** search head and add **`SH2`** to **`site2`**
- **Optional Tasks:** (time permitting)
  - Test the indexer site failover scenario
    - Stop **`site1`** processes
    - Check the cluster status and verify the search affinity

Lab Exercise 3 – Migrate to Multisite Cluster (cont.)



Module 4:
Indexer Cluster Management and Administration

Module Objectives

- Enable replication for custom indexes
- Use manager node to deploy apps to peer nodes
- Take a peer offline temporarily
- Decommission a peer permanently
- Clean up excess cluster buckets
- Configure Monitoring Console for indexer cluster environment

Configuring Clustered Indexes: **repFactor**

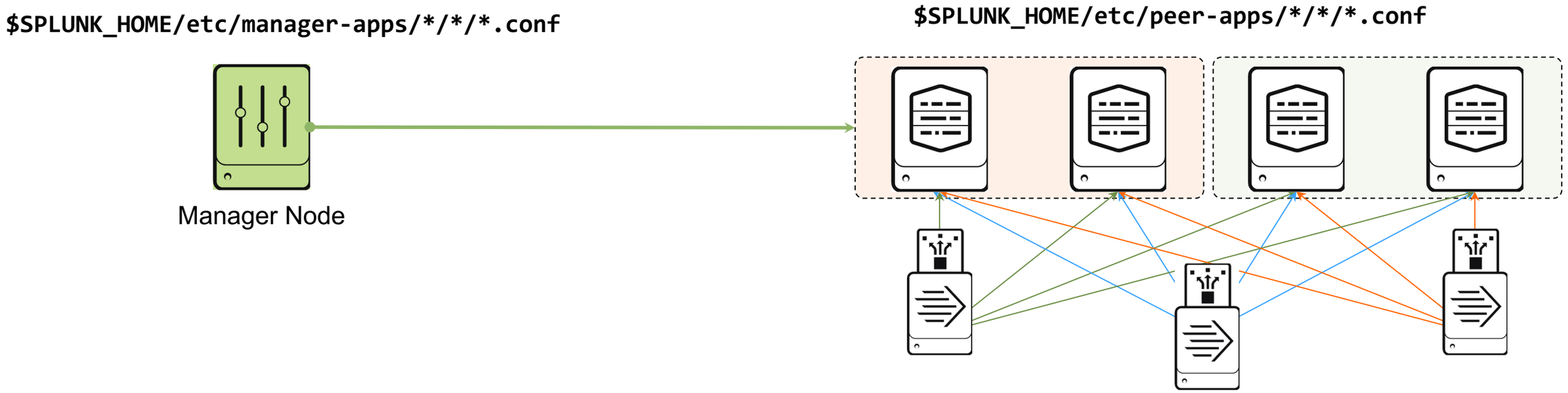
- Configure **repFactor** attribute in **indexes.conf:**
  - **repFactor = auto** (replicate)
  - **repFactor = 0** (default; do not replicate)
    - Note: internal indexes are set to replicate automatically when an indexer is configured as cluster indexer
- All peer nodes must use the same set of indexes.conf files
  - Deploy **indexes.conf** from the manager node

$SPLUNK_HOME/etc/peer-apps/_cluster/default/indexes.conf

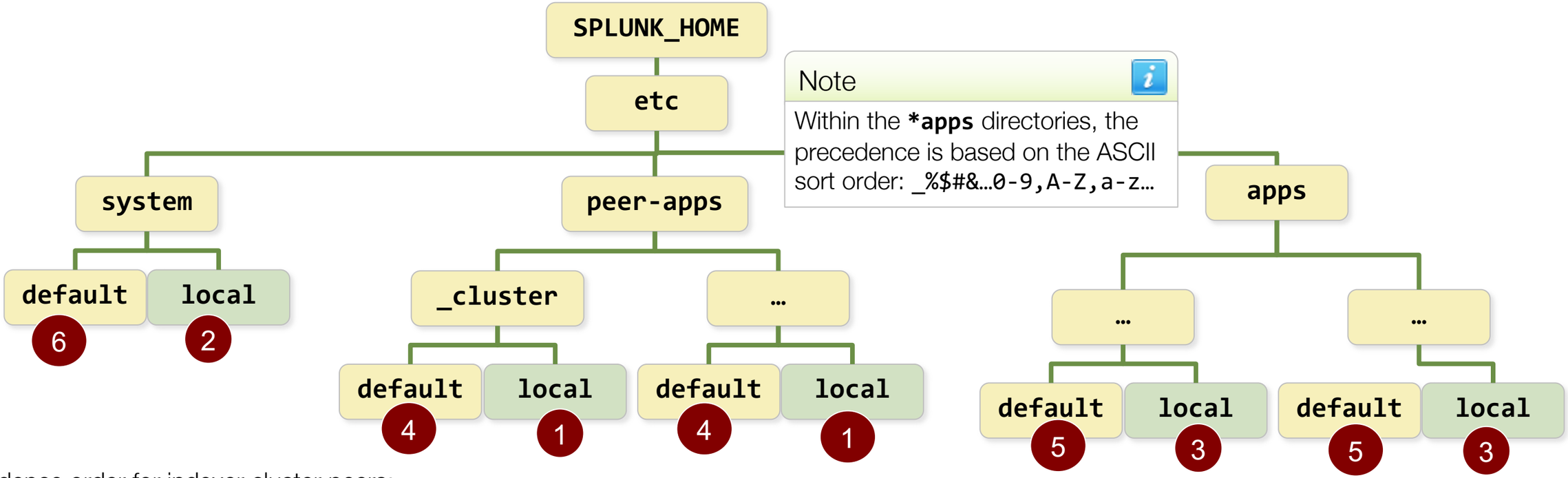
```
[main]
repFactor = auto

[history]
repFactor = auto

[summary]
repFactor = auto

[_internal]
repFactor = auto

[_audit]
repFactor = auto

[_thefishbucket]
repFactor = auto

[_telemetry]
homePath   = $SPLUNK_DB/_telemetry/db
coldPath   = $SPLUNK_DB/_telemetry/colddb
thawedPath = $SPLUNK_DB/_telemetry/thaweddb
repFactor = auto
...
```


Manager-apps Deployment Tool

- The manager node distributes configuration bundles (apps) to the peer nodes
  - Supports common configurations for inputs, parsing, and indexing (**inputs.conf**, **props.conf**, **transforms.conf**, **indexes.conf**)
- Stage deployment bundles in the manager's **SPLUNK_HOME/etc/manager-apps** directory
  - For common apps, copy them to the **<app-name>** subdirectory
  - For standalone files, copy them to the **_cluster/local** subdirectory
- If necessary, manager node initiates an automatic rolling restart of all peers



Clustered Indexer Configuration Precedence

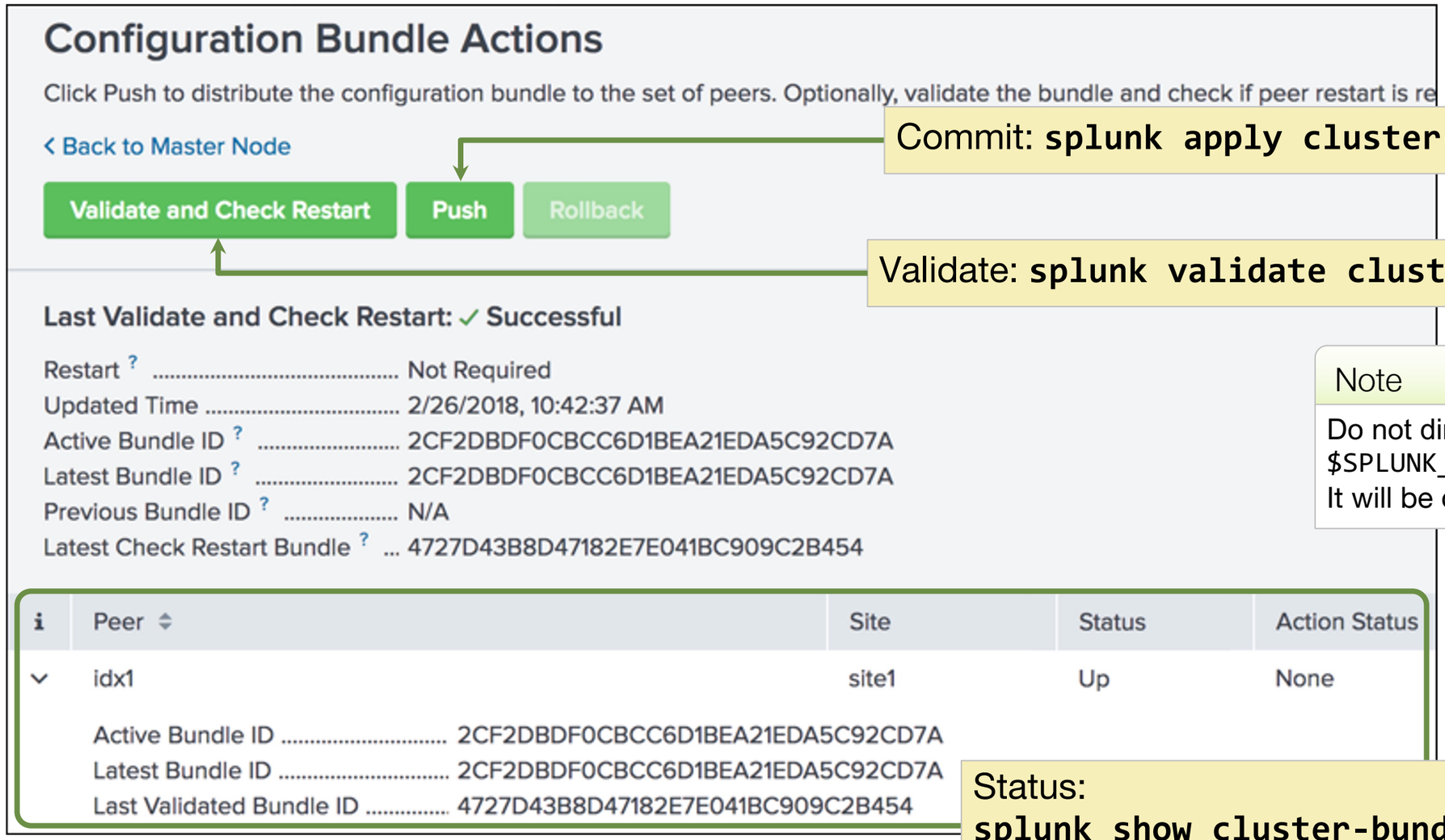


Precedence order for indexer cluster peers:

1. peer-apps local directories **SPLUNK_HOME/etc/peer-apps/*/local/*.conf**
2. System local directory: **SPLUNK_HOME/etc/system/local/*.conf**
3. App local directories: **SPLUNK_HOME/etc/apps/*/local/*.conf**
4. peer-apps default directories: **SPLUNK_HOME/etc/peer-apps/default/*.conf**
5. App default directories: **SPLUNK_HOME/etc/apps/*/default/*.conf**
6. System default directory: **SPLUNK_HOME/etc/system/default/*.conf**

Configuration Bundle Status and Actions

Manager Node: Settings > Indexer clustering



Commit: **`splunk apply cluster-bundle [--skip-validation]`**

Validate: **`splunk validate cluster-bundle [--check-restart]`**

Note

Do not directly edit
`$SPLUNK_HOME/etc/peer-apps/*/*/*.conf`
It will be overwritten on next push

Status:
`splunk show cluster-bundle-status [--verbose]`

 | 7 August 2023

Rolling Back a Configuration Bundle

If the **Push** action fails:

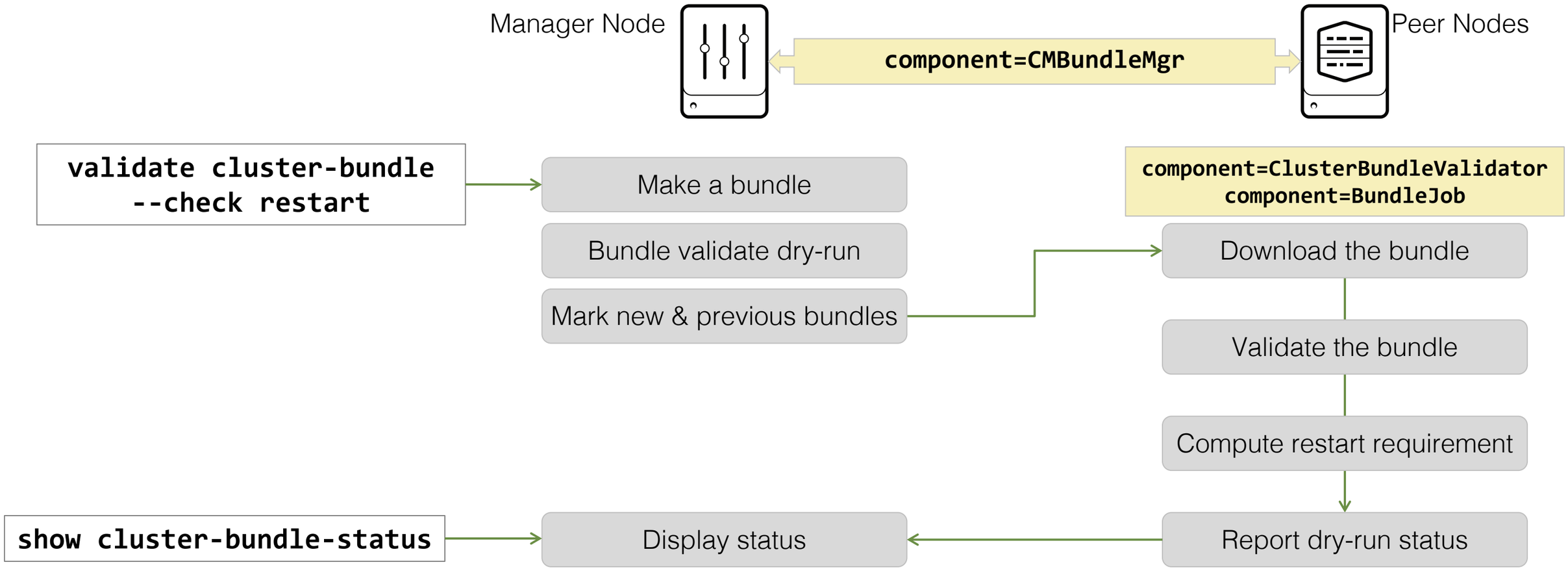
- Restore peers to the previously running state using the **Rollback** command on the manager node
  - **Rollback** only toggles between most recent & previous configuration bundles (!= undo)

Then

- Fix the problem in the **`manager-apps`** directory & re-apply
  - Peers only join the cluster if bundle validation succeeds during restart

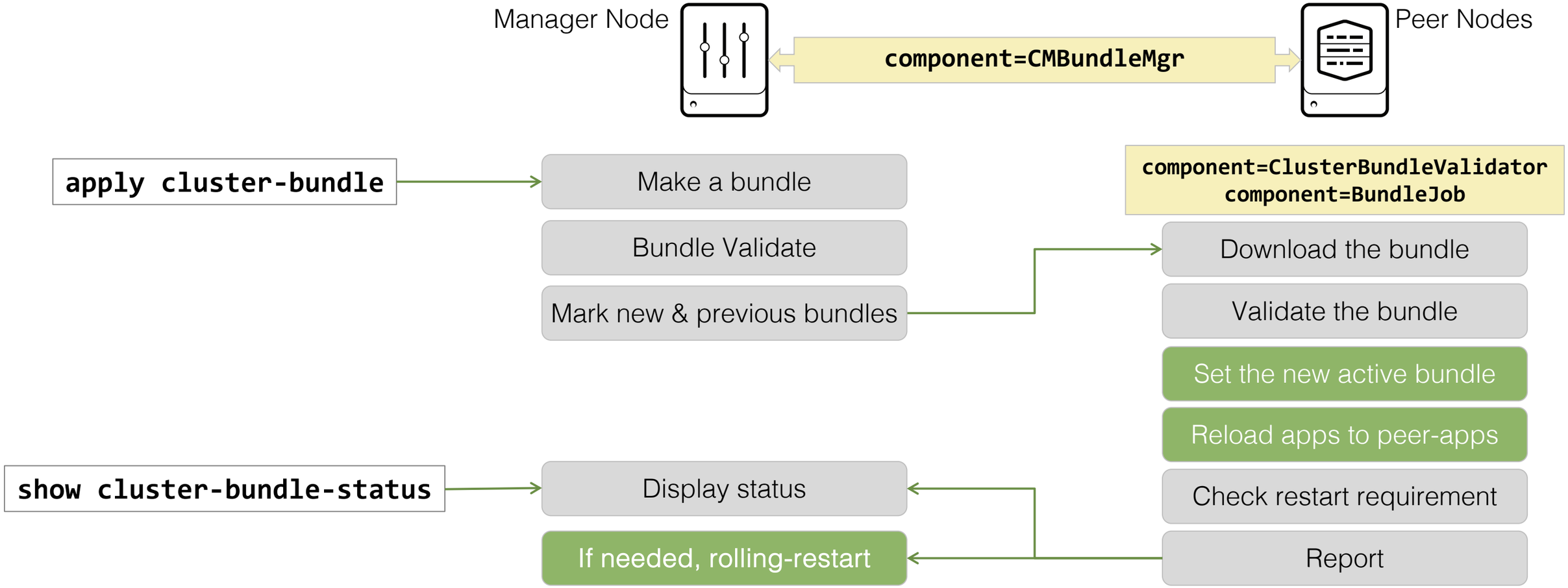
Alternatively, run: **`splunk rollback cluster-bundle`**

Log Channels for `manager-apps` Activities




```
index=_internal sourcetype=splunkd component IN (CMBundleMgr, BundleJob, ClusterBundleValidator)
| stats count by _time, host, component, event_message
```

Log Channels for `manager-apps` Activities (cont.)




```
index=_internal sourcetype=splunkd component IN (CMBundleMgr, BundleJob, ClusterBundleValidator)
| stats count by _time, host, component, event_message
```

Indexer Cluster Searchable Rolling Restart

Restarts peer nodes while maintaining the search availability

- Best effort is made to maintain a “valid” state throughout the restart
- Trades search availability with restart completion time
- May take longer to complete
- Completes in-progress searches before peer nodes restart
- Terminates searches that do not complete within the timeout (default: 180 sec.)

```
splunk rolling-restart -searchable true
```

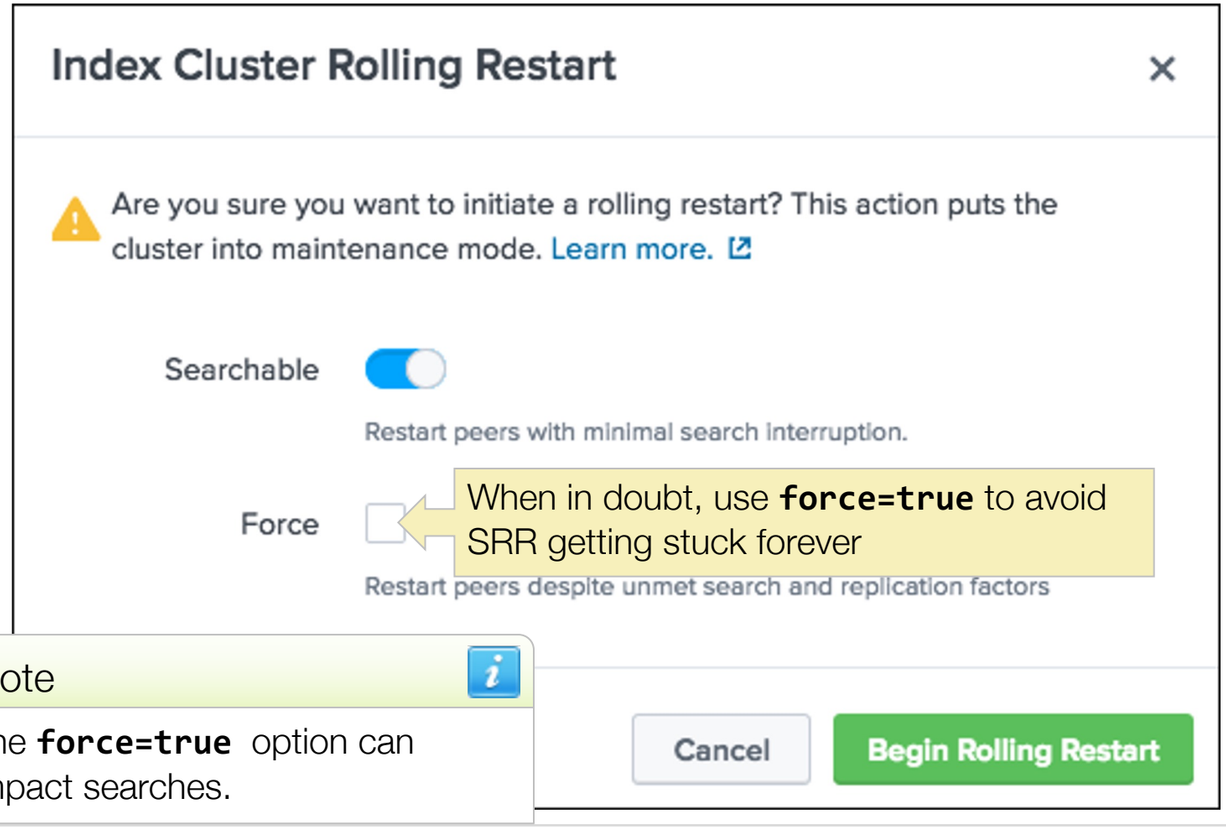

Searchable Rolling Restart (SRR) Options

- **`searchable <true|false>`**
  - **`false`** = ignore all options
- **`force <true|false>`**
  - **`true`** = proceed with the searchable rolling restart despite health check failures
  - Specify these additional parameters:
    - **`restart_inactivity_timeout <sec>`**

      Timeout before manager node gives up on current peer and moves on to next peer (default: 600 sec = 10 min)
    - **`decommission_force_timeout <sec>`**

      Timeout for a peer to finish and voluntarily restart before it is forced to restart (default: 180 sec = 3 min)

```
splunk rolling-restart cluster-peers
-searchable <true/false>
[-force <true/false>
-restart_inactivity_timeout <sec>
-decommission_force_timeout <sec>]
```



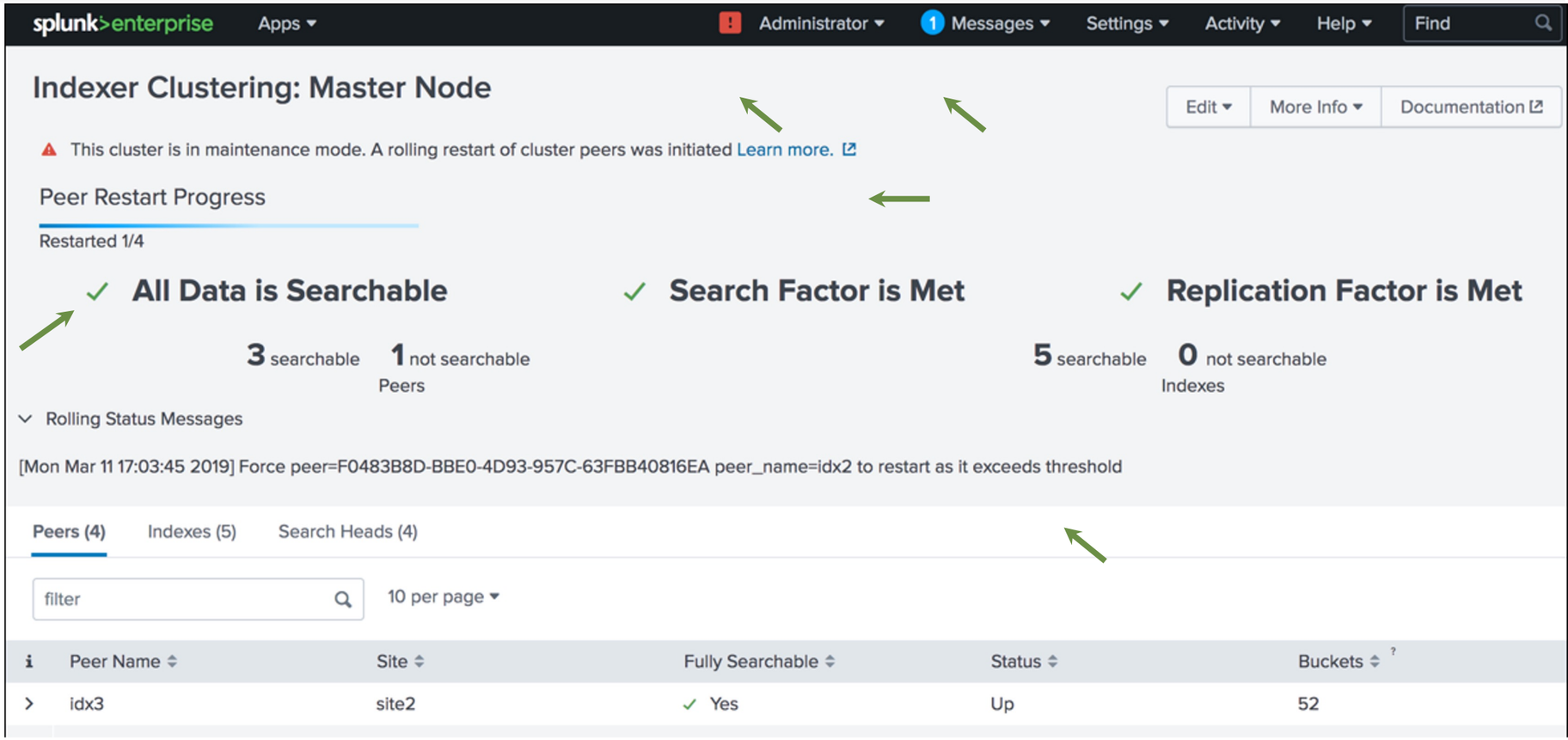
Note

The **`force=true`** option can impact searches.

Tracking Rolling Restart Progress

Web UI

- Health status
- Maintenance message
- Progress bar
- Additional messages



CLI

```
splunk show cluster-status --verbose
```

Taking a Peer Offline Temporarily

- Before taking a peer node offline ensure cluster is in complete state
- To take down a peer temporarily, run: **`splunk offline`**
  - The manager node delays bucket-fixing and just maintains a valid state
  - The peer node must be back online within 60 seconds (by default)
    - If the peer node does not return, the manager node initiates complete bucket fixup activities
  - If you need more time, extend the wait on the manager node using CLI or `server.conf`:

```
splunk edit cluster-config \
-restart_timeout <wait_seconds>
```

```
[clustering]
…
restart_timeout=wait_in_seconds
…
```

Decommissioning a Peer Node Permanently

- Run: **`splunk offline --enforce-counts`**
  - Delays shutdown until buckets are copied to other surviving nodes to achieve complete state
  - Does not shut down until all search and remedial activities complete
  - Does not remove the peer from the cluster
    - Accomplished by removing from `[clustering]` stanza in `server.conf`
- Manager retains the peer node information even when it is decommissioned
  - If you want to remove it from the manager node permanently, run:

    `splunk remove cluster-peers -peers <guid>,<guid>,...`

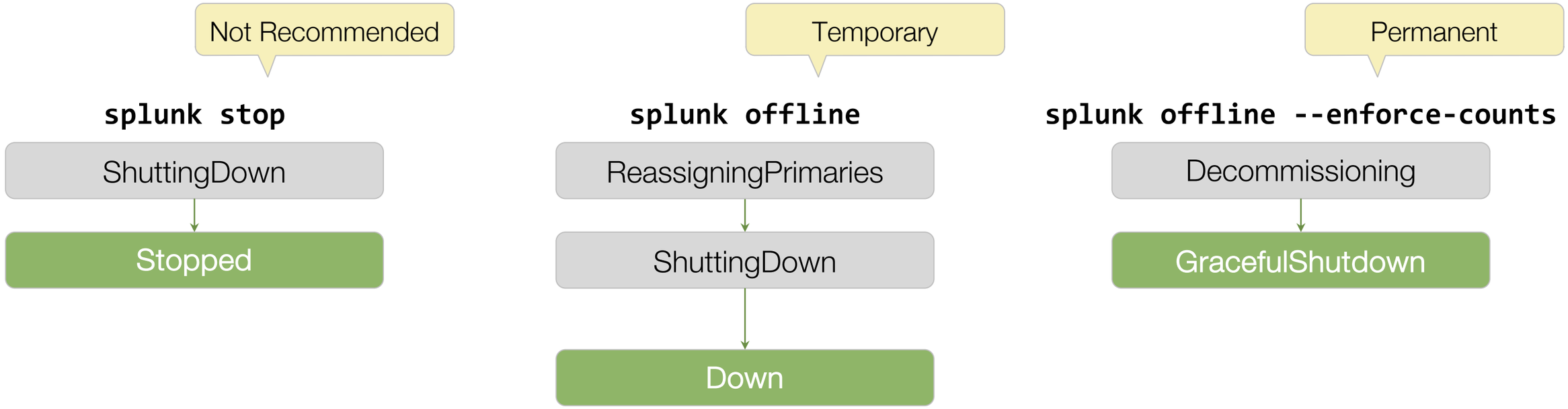
Adding New Peer Node Indexers to Cluster

Use the standard configuration procedures to add a peer node to an existing cluster

- Update the following attributes in **`server.conf`**:
  - **`[clustering]`**: Add new peer node
  - **`[replication_port://x]`**: Define listening port for replicated data
  - **`[general]`**: If multisite (**`site=site<n>`**)
- Restart Splunk on the new indexer peer

Indexer Cluster Peer Status

Manager node dashboard reports several possible status conditions for peer nodes



Decommissioning Sites

To decommission a disused site after an upgrade or migration:

- Ensure the cluster is in a complete state
- Ensure manager node is not a part of the decommissioning site
- Ensure at least one searchable copy of each bucket on remaining sites

Note:

- Prior standalone and single-site buckets will be lost
- Bucket fixup activities will start & may take awhile to complete

Steps for Decommissioning a Site

1. Reassign site search head(s) to a remaining site
2. If forwarders are associated with the decommissioned site using indexer discovery, re-assign them to a remaining site
3. On the manager node, run: **`splunk enable maintenance-mode`**
4. Update the following attributes in the manager's **server.conf**:
   - **`available_sites`**
   - **`site_replication_factor`**
   - **`site_search_factor`**
   - **`site_mappings`**
5. Restart the manager node
6. On the manager node, run: **`splunk disable maintenance-mode`**
7. Decommission each peer with **`splunk offline --enforce-counts`**

> Note
>
> **`site_mappings`** is a way to handle buckets that are stuck during fixups due to decommissioned origin site.

Replace Old Peers with New Site (Example)

Edit `server.conf` on the Manager Node:

```
[clustering]
…
available_sites = site1,site2
site_replication_factor = origin:1, total:2
site_search_factor = origin:1,total:2
…
```

Manager Node server.conf

Before

```
[clustering]
…
available_sites = site1,site3
site_replication_factor = origin:1,site1:1,site3:1,total:2
site_search_factor = origin:1,site1:1,site3:1,total:2
site_mappings = site2:site3
…
```

After

Replace All Peers (Example)

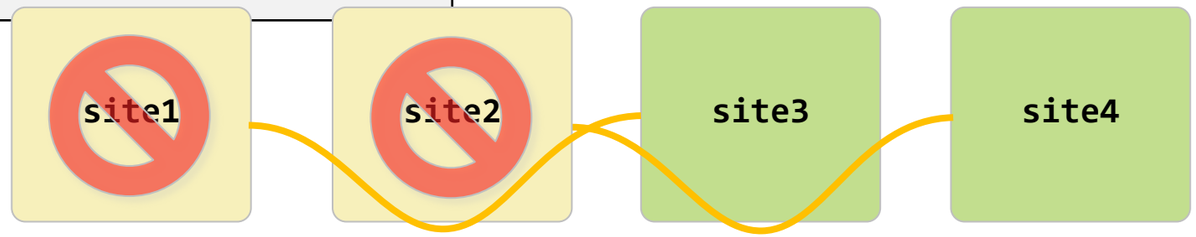
Edit server.conf on the Manager Node:

```
[clustering]
…
available_sites = site1,site2
site_replication_factor = origin:1, total:2
site_search_factor = origin:1,total:2
…
```

Before

```
[clustering]
…
available_sites = site3,site4
site_replication_factor = origin:1,site3:1,site4:1,total:2
site_search_factor = origin:1,site3:1,site4:1,total:2
site_mappings = site1:site3,site2:site4
…
```

After



Migrate From Multisite to Single (Example)

Edit server.conf on the Manager Node:

```
[clustering]
…
available_sites = site1,site2,site3
site_replication_factor = origin:1,site1:1,site3:1,total:2
site_search_factor = origin:1,site1:1,site3:1,total:2
site_mappings = site2:site3
…
```

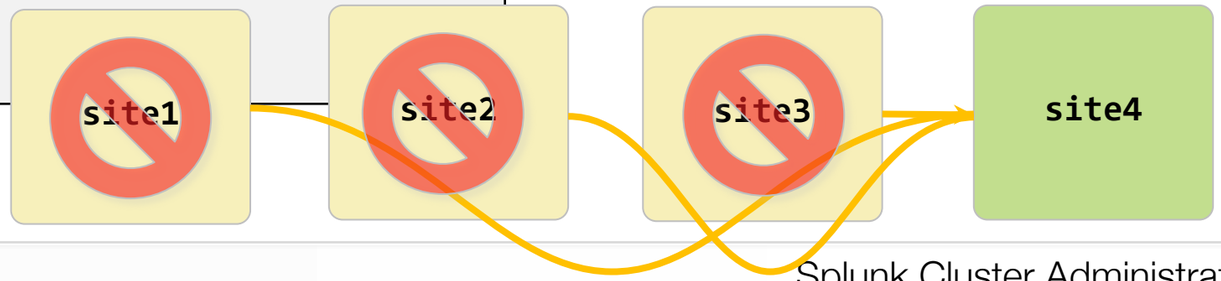
Before

```
[clustering]
…
available_sites = site4
site_replication_factor = origin:2,site4:2,total:2
site_search_factor = origin:2,site4:2,total:2
site_mappings = default_mapping:site4
…
```

After

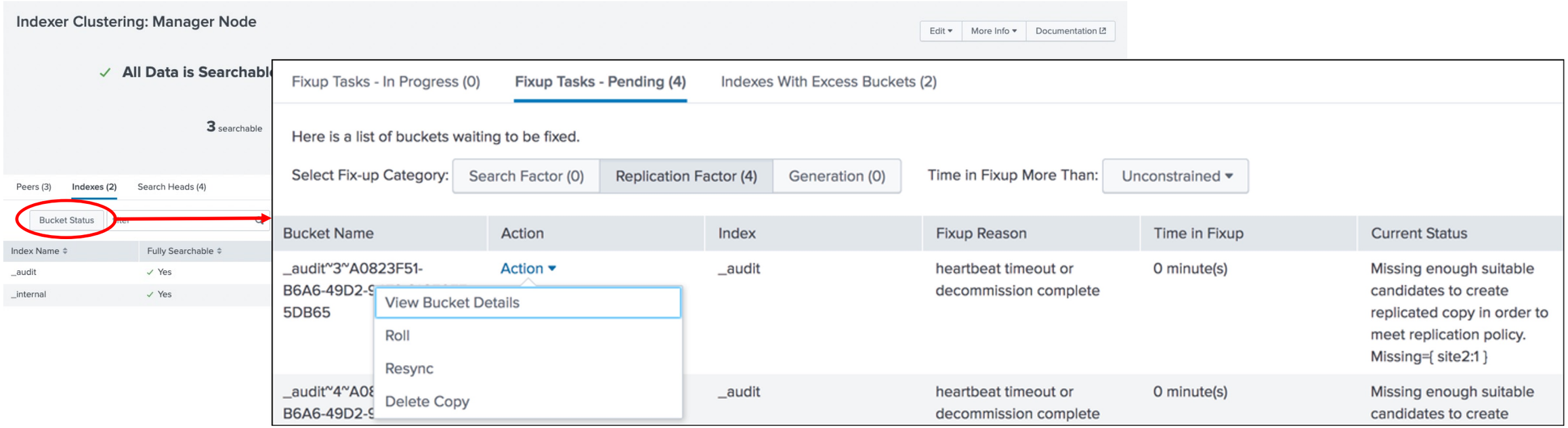
Note

If a site used in a mapping is later decommissioned, its previous mappings must be remapped to an available site.



Bucket Fixing (AKA Bucket Fixup)

- Process of manager node returning cluster to valid and complete state (if possible) by reassigning primaries and directing surviving indexers to copy both index and rawdata to meet search and rep factors
- Bucket Status page displays activity
  - Navigate to **Settings > Indexer Clustering > Indexes (Tab) > Bucket Status**



Primary Rebalancing

- Balances search load across all peer nodes
  - Manager node identifies duplicate searchable buckets
  - Attempts to reassign same number of primaries on each peer
  - Does not move searchable copies to different peer nodes
- Occurs independently for each site in a multisite cluster
  - Does not shift primaries between sites
- Triggers automatically when:
  - A peer node joins or rejoins the cluster
  - Manager node rejoins the cluster
  - Rolling restart completes
- Manually trigger rebalancing with REST endpoint on manager node:
  `services/cluster/manager/control/control/rebalance_primaries`

Data Rebalancing

Redistributes the number of bucket copies per index

- Balances storage distribution across peer nodes
- Operates on only warm and cold buckets, NOT hot buckets
- Rebalances all non-searchable, searchable and primary buckets
- Rebalances within sites as well as across sites in multisite cluster

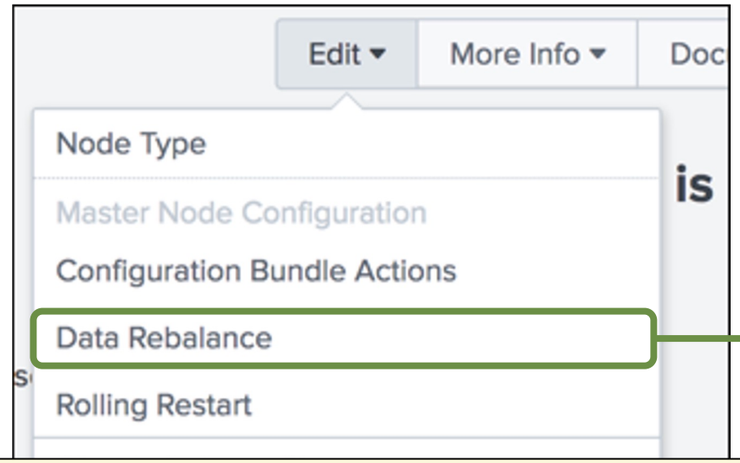
Uneven bucket distribution may cause higher load on peer nodes & may occur:

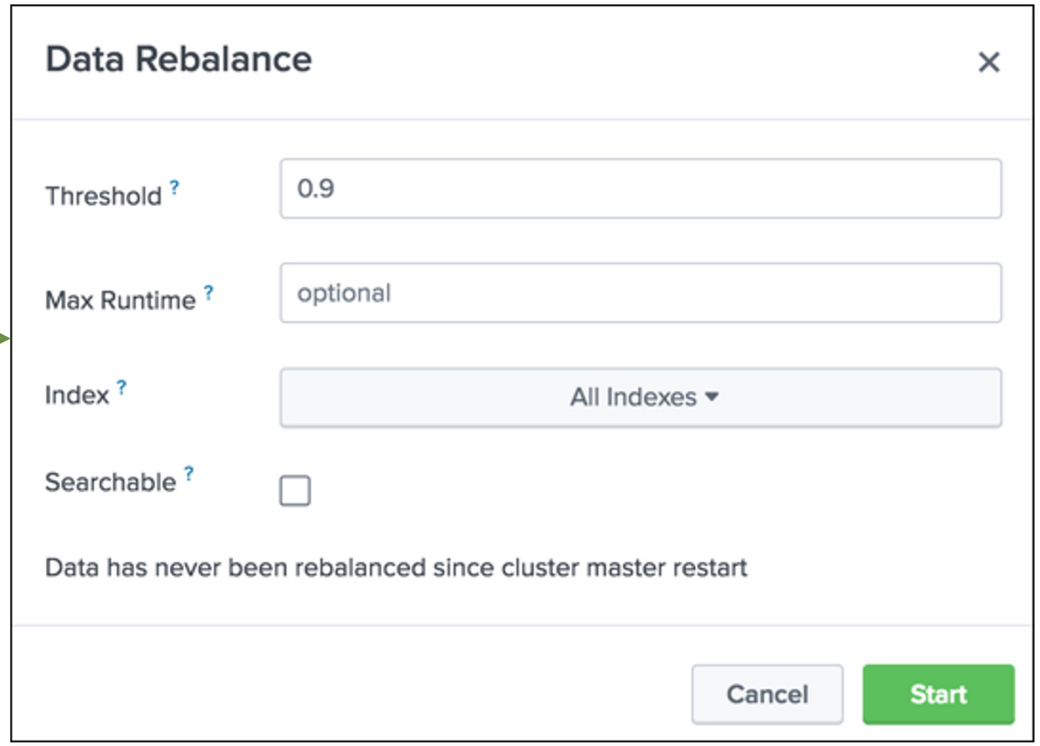
- After adding new peer nodes
- When forwarding data is skewed
- After frequent bucket fixups due to outages

i	Peer Name ^	Site	Fully Searchable	Status	Buckets
>	idx1	site1	✓ Yes	Up	68
>	idx2	site1	✓ Yes	Up	14
>	idx3	site2	✓ Yes	Up	40
>	idx4	site2	✓ Yes	Up	40

Duration (seconds)	Component	Invocations	Input count	Output count
8.33	dispatch.stream.remote	51	-	20,767,549
5.72	dispatch.stream.remote.idx1	27	→	12,316,674
2.61	dispatch.stream.remote.idx2	22	-	8,431,689
0.00	dispatch.stream.remote.idx3	1	-	9,590
0.00	dispatch.stream.remote.idx4	1	-	9,596

Data Rebalancing (cont.)





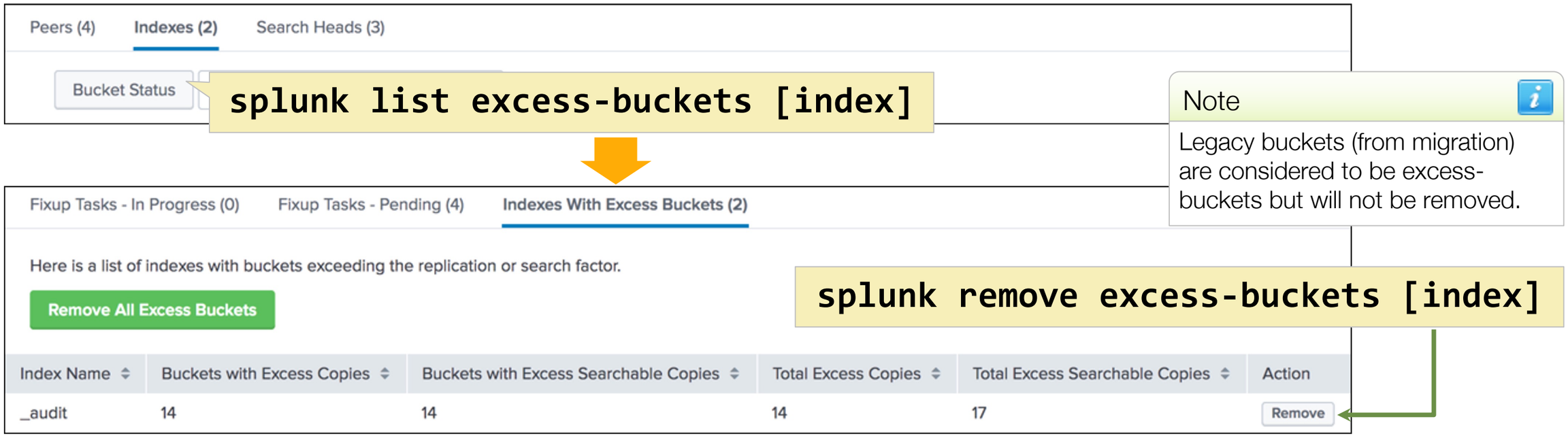

```
> splunk edit cluster-config -rebalance_threshold 0.90
> splunk rebalance cluster-data -action start
 [-searchable true] [-index <idx>] [-max_runtime <min>]
> splunk rebalance cluster-data -action status
> splunk rebalance cluster-data -action stop
```

Goal: achieve a practical balance, not a perfect balance

- Process impacts search performance
- Running in searchable mode takes longer to complete and requires more storage space

Cleaning Up Excess Bucket Replicas

- Excess copies may result from peers leaving and returning to cluster
  - No effect on operation of cluster, but consumes storage space
- Use Splunk Web or CLI on manager node to determine & remove



Peer Detention – Automatic

A peer enters **automatic detention** when the **`minFreeSpace`** threshold in **`server.conf`** is crossed (default value is 5GB)

When a peer is in automatic detention:

- Indexing is stopped (internal and external)
- Replication is stopped

Peer Detention – Manual

When a peer is in manual detention, it will:

- Stop replicating data from other peer nodes
- Disable external data ports and stop indexing external data (optional)
- Continue to index internal data
- Continue to participate in searches

Use cases for **manual detention**:

- Before decommissioning a peer to make it available only for searches
- Preempt automatic detention
- Diagnose a suspect peer by blocking indexing and replication
- Force new data to other peers
- Slow disk writes by only allowing indexing, but not replication

Enabling Manual Detention

On the manager node, run:

```
splunk edit cluster-config -manual_detention [on|on_ports_enabled|off]
-peers <guid1>, <guid2>,...
```

On the peer, run:

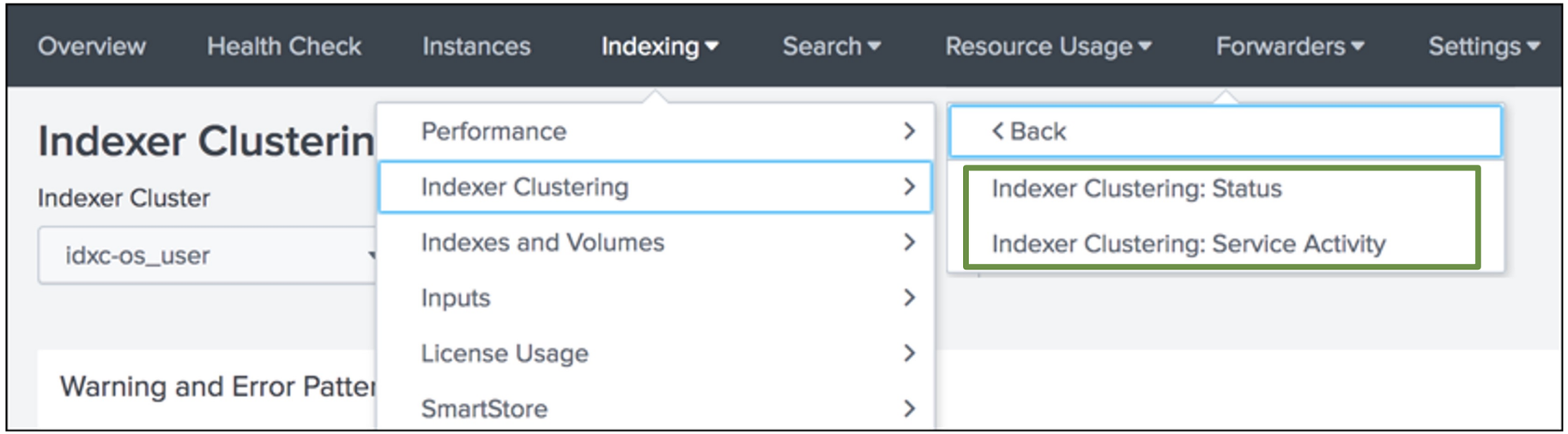
```
splunk edit cluster-config -manual_detention [on|on_ports_enabled|off]
```

- **on:** closes the TCP, UDP, and HEC ports; disables indexing and replication for all data inputs; does not disable indexing local monitor and scripted inputs
- **on_ports_enabled:** blocks incoming replication; continues to index
- **off:** disables the manual detention

Checking the detention status

- On the manager node:
  - CLI: **splunk show cluster-status**
  - UI: Settings > Indexer Clustering > Peers
- On the peer node: **splunk list cluster-config**
- On monitoring console: Indexing > Indexer Clustering > Indexer Clustering: Status

Indexer Cluster Dashboards in MC



- Indexer Clustering: Status
  - Provides the same information as the manager's indexer clustering page
- Indexer Clustering: Service Activity
  - For an ideal healthy cluster, most of the panels should be blank
  - The trending down of fixup tasks and service jobs count is normal
  - Pay attention to an increasing trend on pending tasks and jobs

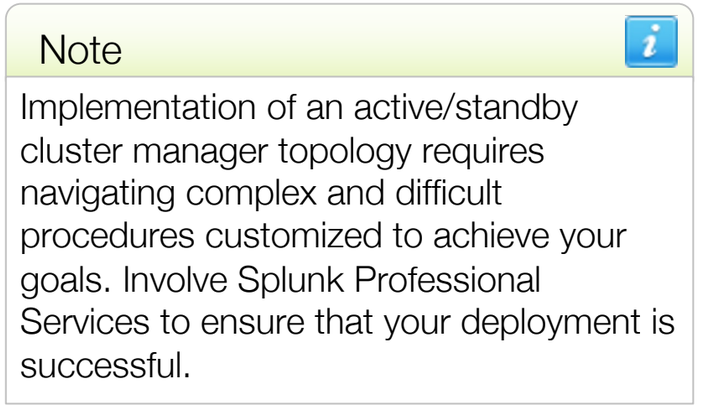
Cluster Manager Redundancy

New feature & recommended best practice of Splunk Enterprise 9.0+

- Addresses problems associated with manual node failover from previous versions
- Supports automatic or manual failover
  - Configured in **`server.conf`**

Requires configuration in three areas:

- Multiple cluster managers
- Peer nodes, search heads, and forwarders when indexer discovery is enabled
- Third party load balancing or DNS mapping



Load Balancer Option – Normal State

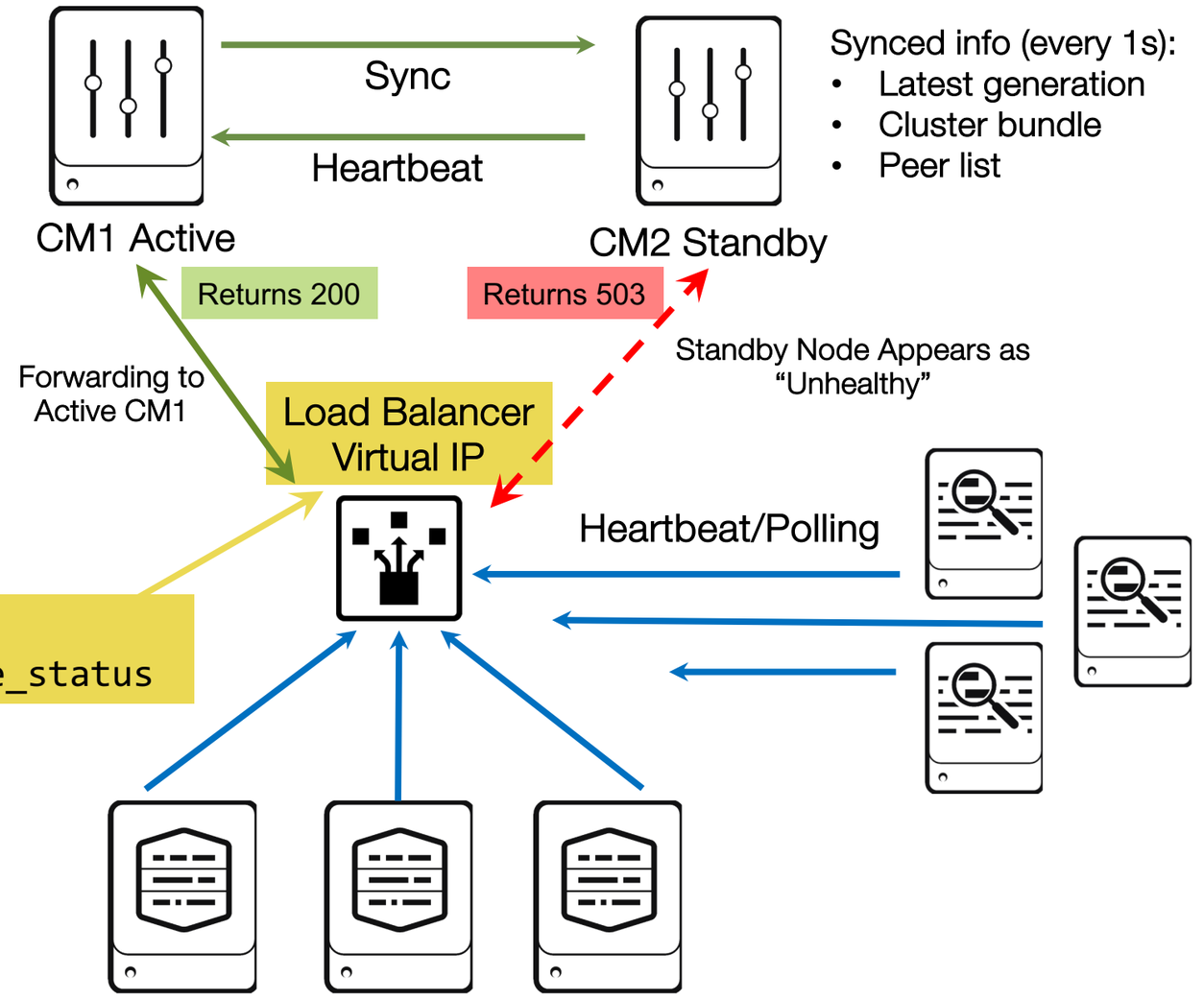
server.conf, Manager Nodes

```
[clustering]
mode = manager
manager_switchover_mode = auto
manager_uri = clustermanager1:cm1,clustermanager2:cm2
pass4SymmKey = Hashed_Secret

[clustermanager1:cm1]
manager_uri = https://10.16.88.3:8089

[cmanager2:cm2]
manager_uri = https://10.16.88.4:8089
...
```

LB configured to poll MN using
`https://CMX:8089/cluster/manager/ha_active_status`

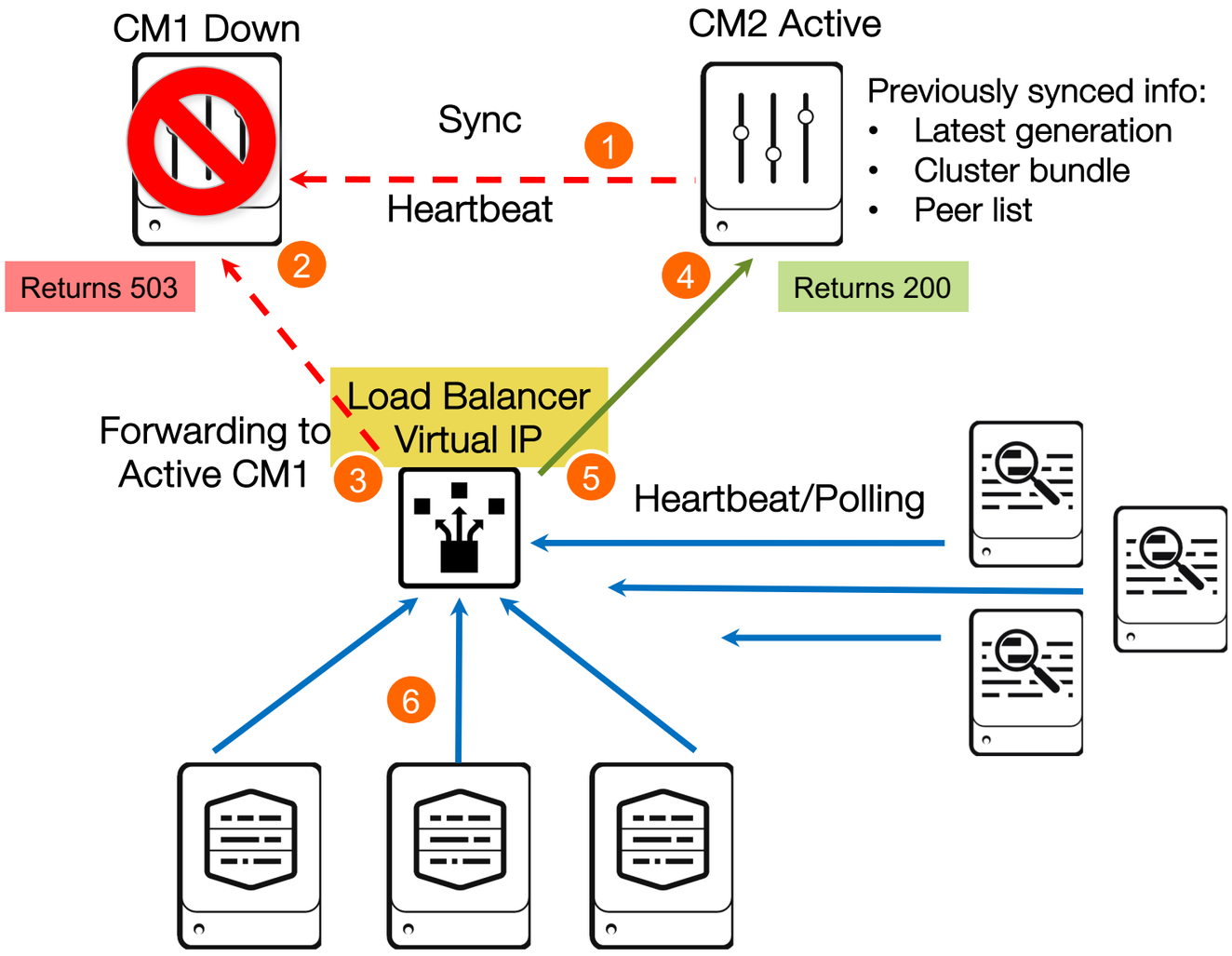
server.conf, Indexer peers and Search Heads

```
[clustering]
manager_uri = https://<LB-IP-OR-DNS-HOSTNAME>:8089
...
```



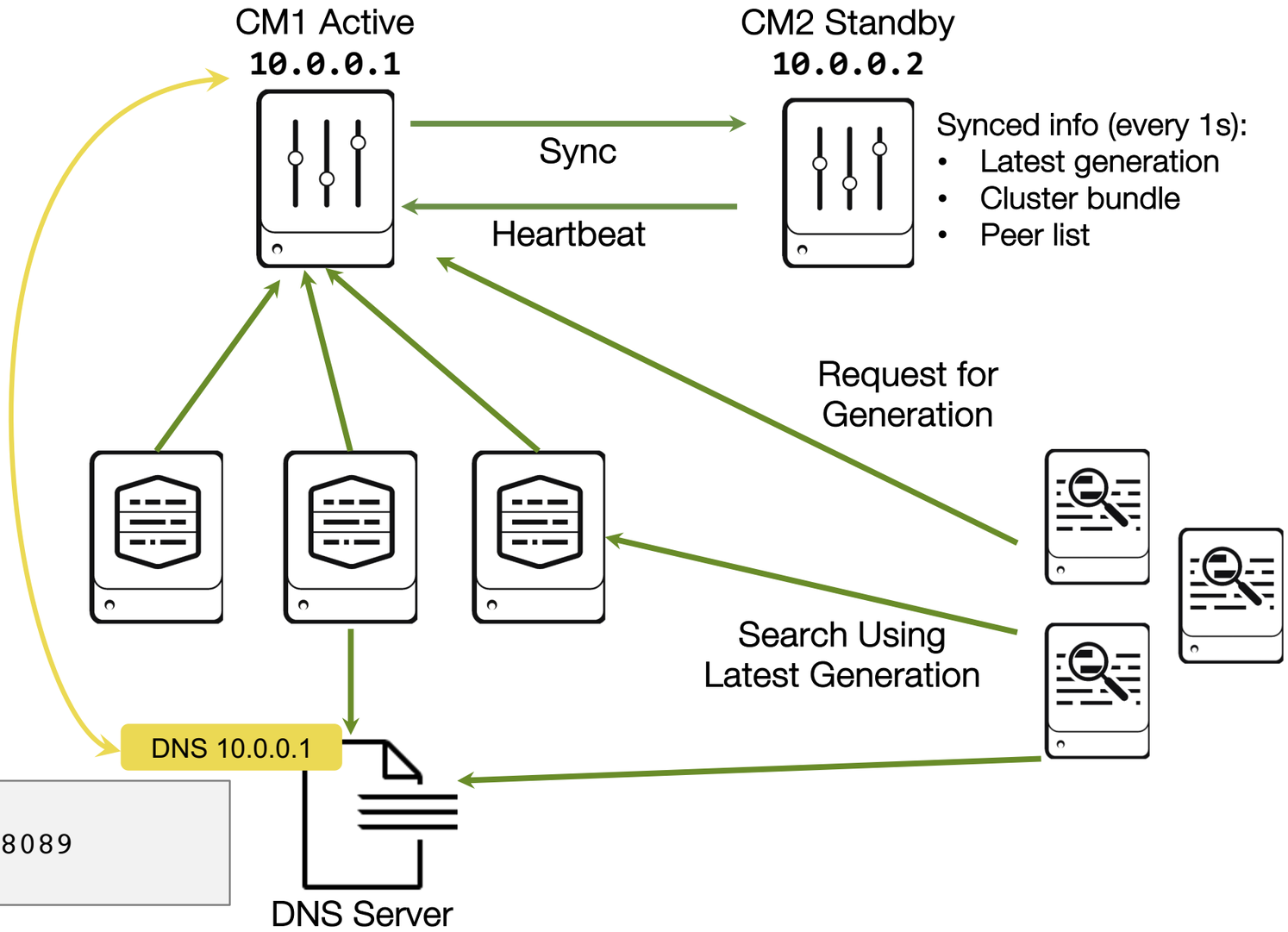
Load Balancer Option – Failover

1. CM1 down and CM2 detects the heartbeat loss
2. CM1 is either gone or advertises itself as “Unhealthy” to the load balancer
3. Load balancer stops forwarding traffic to CM1
4. CM2 promotes itself is active and advertises itself as “Healthy” to load balancer
5. Load balancer starts forwarding traffic to CM2
6. Indexers re-add themselves to CM2



DNS Option – Normal State

- DNS-based solution is typically used when CMs are configured in manual mode
- Automatic switchover can be used, but the DNS record must be manually updated
- Requires external monitoring to detect CM loss



server.conf

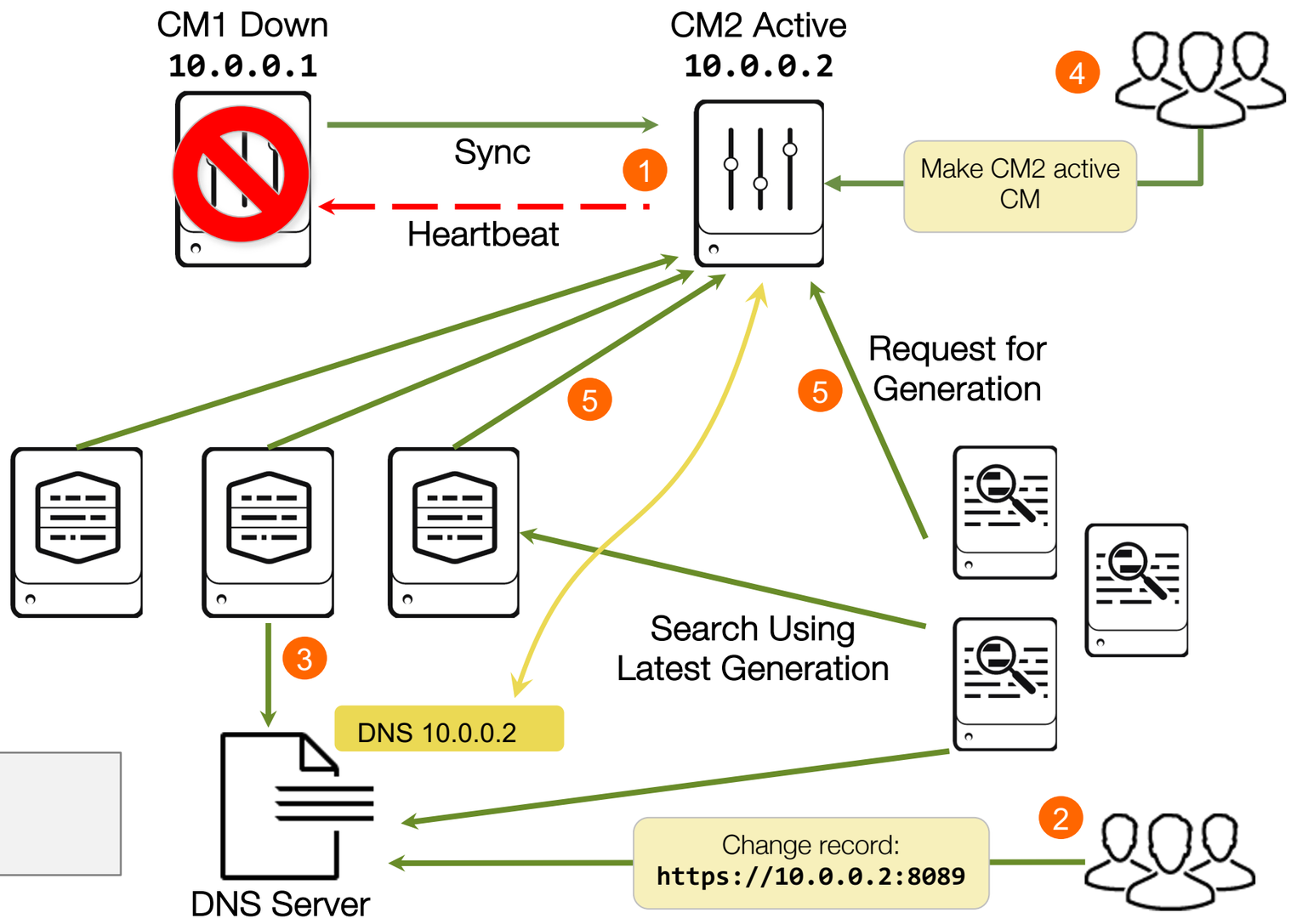
```
[clustering]
manager_uri = https://DNS:8089
...
```

DNS Option – Normal State

1. CM1 down and CM2 detects the heartbeat loss
2. Admins detect through monitoring mechanism that the CM has gone down and decides to change the DNS record for the CM to point to Standby CM2
3. Nodes will refresh the DNS entry to pickup the active CM IPAddress
4. Admins execute CLI to make standby CM2 active
5. Nodes connect to the new CM

server.conf

```
[clustering]
manager_uri = https://DNS:8089
...
```



Indexer Clustering Health Report

Reports every 20 seconds by default

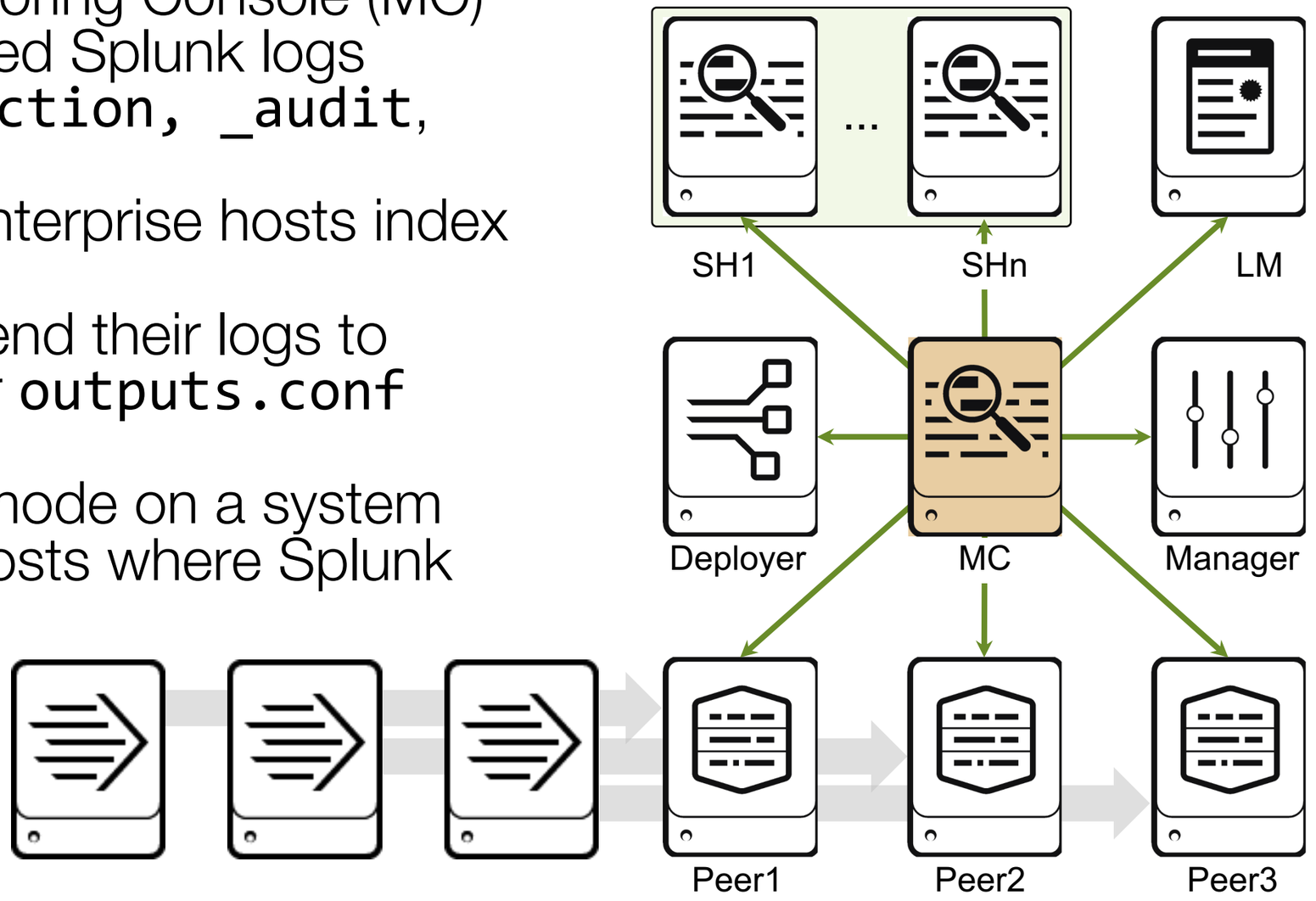
- **`feature:cluster_bundles`** reflects whether there are validation errors in the last bundle that were pushed to cluster peers (yellow only)
- **`feature:data_durability`** reflects whether or not:
  - The configured replication factor is met (red only)
  - The configured search factor is met (red only)
- **`feature:data_searchable`** turns red if one or more buckets lack a primary
- **`feature:indexers`** tracks whether any peer nodes are in detention mode
  - Yellow if in manual detention
  - Red if in automatic detention
- **`feature:missing_peers`** tracks any peer nodes that are in transition
  - Yellow if nodes are stopping, stopped, decommissioning, pending or restarting
  - Red if nodes are down
- **`feature:indexing_ready`** stays green when the cluster is functional

Monitoring Clusters with Monitoring Console

In distributed mode, Monitoring Console (MC) needs to search the indexed Splunk logs (`_internal, _introspection, _audit`, etc.)

- By default, all Splunk Enterprise hosts index these inputs locally
- Universal Forwarders send their logs to indexers defined in their `outputs.conf`

Enable MC in distributed mode on a system that searches all Splunk hosts where Splunk logs are indexed



> **Note**
>
> Best practice: configure non-indexer Splunk Enterprise hosts to forward inputs to indexing tier

Monitoring Console Config Prerequisites

- Users require **admin_all_objects** capability to configure the MC
- Each instance must use a unique **servername** and **default-hostname**
- Platform instrumentation is enabled for every instance (UF optional)
- Optionally on the cmanager: **splunk edit cluster-config -cluster_label idxc1**
- Forward all inputs (including internals and summaries) from all non-indexer Splunk Enterprise hosts, including search heads and the cmanager node to the indexing tier

outputs.conf

```
[indexAndForward]
index = false

[tcpout]
defaultGroup = default-autolb-group
forwardedindex.filter.disable = true
indexAndForward = false

[tcpout:default-autolb-group]
server=idx1:9997,idx2:9997,idx3:9997,idx4:9997
useACK = true
```


MC Post-Configuration Checklist

- Add all instances, except peer nodes, as search peers to the MC
  - Optional if not forwarding Splunk log inputs
- Enable distributed mode on MC
- Verify all instances are discovered and the server roles are correct
  - Ensure only peer nodes (indexers) are checked as indexers
  - Ensure dual roles are checked (e.g: if search head is also a license manager)
  - Click **Edit** to update roles accordingly
- Use custom groups to organize related components
  - Custom groups are used for view selection in the MC dashboards
- If the environment changes:
  - Return to **Setup**
  - Check server roles
  - Update if necessary

Note

To propagate the changes, always click **Apply Changes** on the MC Setup page.

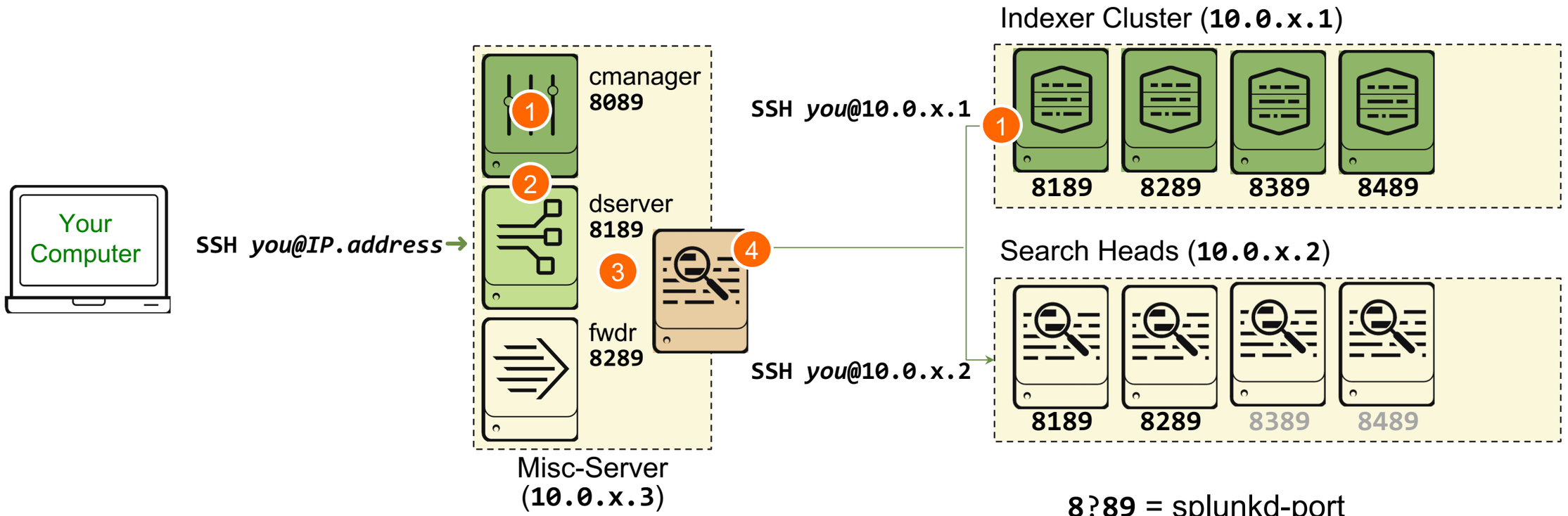
Further Reading: Indexer Cluster Management

- Take a peer offline
- View the manager node dashboard
- Rebalance the indexer cluster
- Reduce tsidx disk storage
- How indexer clusters handle report and data model acceleration summaries
- Update common peer configurations and apps

Lab Exercise 4 – Monitor CM Service Activities

- **Time**: 30 minutes
- **Tasks**:
  - Stage an app and deploy it to the peer nodes
  - Disable indexing on **`cmanager`** and **`dserver`**
  - Enable the Monitoring Console to run in distributed mode on **`dserver`**
  - Monitor the indexer clustering service activities from Monitoring Console

Lab Exercise 4 – Monitor CM Service Activities (cont.)



Module 5: Forwarder Configuration

Module Objectives

- Use indexer discovery to configure forwarders in a clustered environment
- Describe optional indexer discovery settings
  - Polling rate
  - Weighted load balancing
- Optimize indexing loads with volume-based load balancing

Configuring Indexer Acknowledgment

Enable acknowledgement in **`outputs.conf`**:

```
[tcpout:<target_group>]
indexerDiscovery = <name>
useACK = true
```

- Optional capability that helps prevent data loss by ensuring cluster receives and indexes all incoming data
- `true` = forwarder retains a copy of each sent event until the receiving system sends an acknowledgment
  - If the forwarder does not receive an acknowledgment, it resends the data to an alternative receiver
- `false` = forwarder considers data fully processed when it finishes writing it to the network socket
  - Default value = false

Set `ack_factor` in **`server.conf`**:

```
[general]
ack_factor = 1
```

- Sets the number of copies of ingest data that must be made across the indexer cluster before an acknowledgement (ACK) is returned from the source peer to the forwarder.
- Only valid if **`useACK = true`**
- Default value = 0 (uses replication factor)
- All peer nodes must use same value

Note

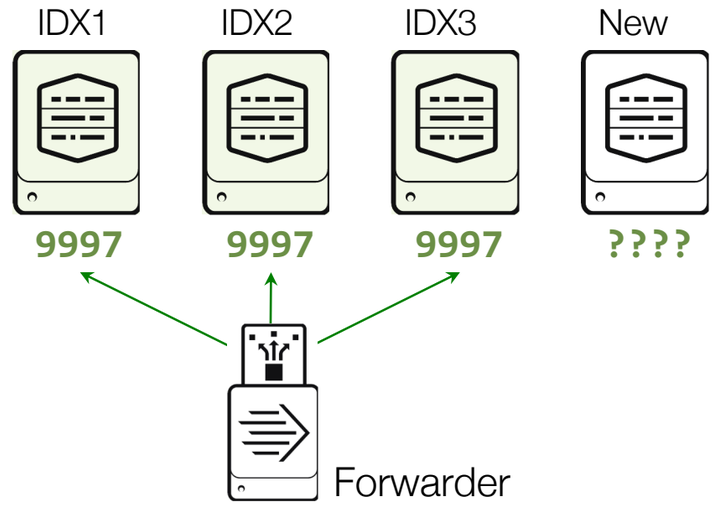
Forwarders are not required to be configured for clustering. `useACK` required to ensure end-to-end data fidelity.

Forwarder Management Challenges

In non-clustered environments, Splunk administrators:

- Must track target server changes
  - Deploying an updated outputs.conf to forwarders require restarting splunk on the forwarders
- Update the forwarder **`outputs.conf`** settings manually
  - A static list of indexers must be deployed to each forwarder

```
[tcpout:indexers]
server = IDX1:9997,IDX2:9997,IDX3:9997
useACK = true
```



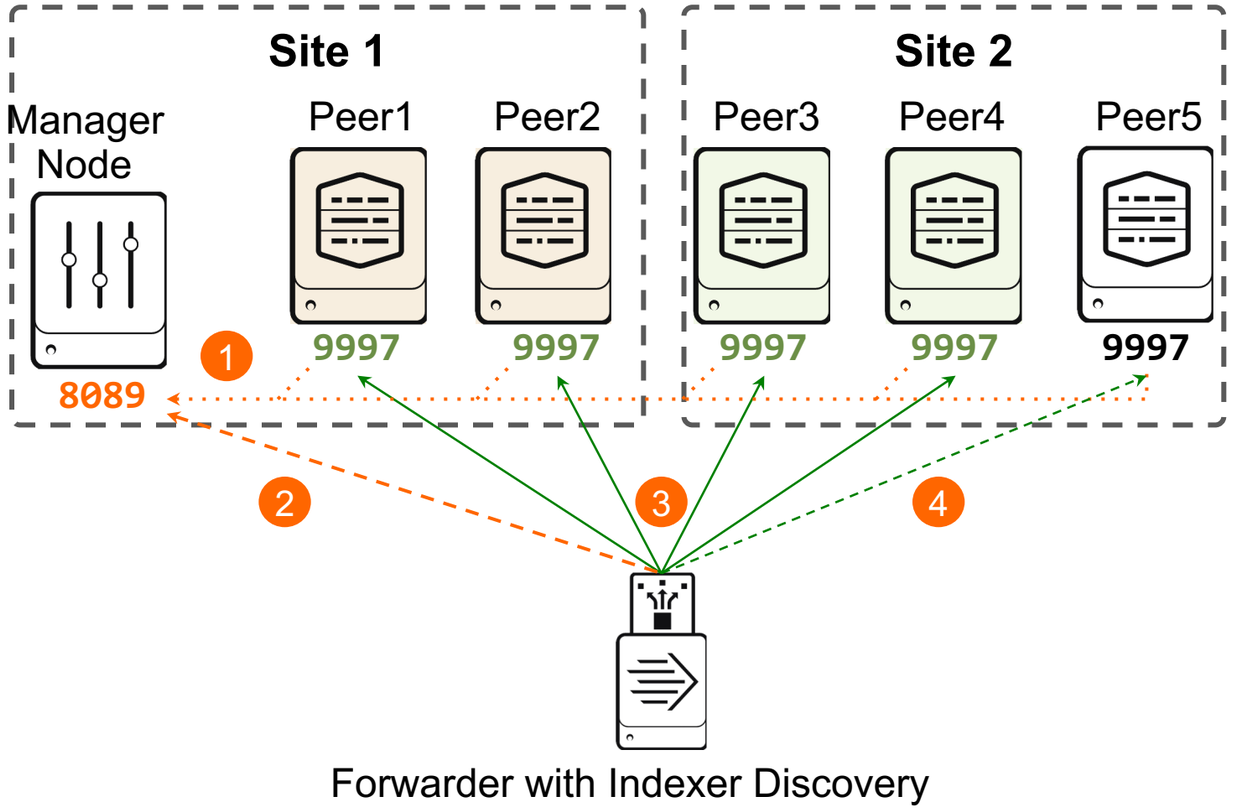
Using Indexer Discovery (Optional)

Dynamically & automatically connect forwarders to peer nodes in indexer clusters

- Minimizes the forwarder restarts
- Scales well and easier to manage
- Reduces the load on DNS servers
- Configurable site-awareness
- Optional weighted load balancing

1. Peers report their receiving ports to manager node
2. Forwarders poll manager node to get the latest list of peer nodes
3. Forwarders send data to the peers in the list
4. A peer can be added or removed without affecting the forwarder configurations

Configured in `server.conf` on Manager Node & `outputs.conf` on forwarders



Configuring Indexer Discovery: Manager Node

Edit `server.conf` on manager node:

```
[indexer_discovery]
pass4SymmKey = <Hashed Secret>
```

Note

Direct edit required. No CLI or SplunkWeb UI support for configuring indexer discovery.

- `pass4SymmKey:`
  - Unique value used on all forwarders and the manager node
  - For better security, use a different secret than manager & peer nodes
- No changes to peer nodes to receive data from forwarders
  - Just need listening enabled

```
splunk enable listen 9997
```


Configuring Indexer Discovery: Forwarders

outputs.conf

```
[tcpout:<target group>]
indexerDiscovery = <name>

[indexer_discovery:<name>]
manager_uri = https://<managerDNSorIP>:8089
pass4SymmKey = <Same Hashed Secret>
```

Use the indexers in this Manager Node's cluster

Configure which Manager Node the Forwarder polls for list of available indexers in cluster

Must match Indexer Discovery pass4SymmKey on Manager

inputs.conf

```
[monitor:///path/to/]
…
_TCP_ROUTING = <target group>
```

Determine indexers to be used

server.conf (if multisite)

```
[general]
site = site<n>
```

Configure site-awareness

- Indexer discovery is not site-aware by default; forwards to all indexers at all sites
- Multisite config requires site-id; use `site0` to forward to peers across all sites

Indexer Discovery Option: Polling Rate

Manager node determines polling interval dynamically using number of connected forwarders and **`polling_rate`**

- poll_interval (seconds) = #_of_forwarders / **`polling_rate + 30`**
- **`polling_rate:`** fixed factor manager uses to calculate polling interval
  - Set a factor between 1 and 10 (the default is 10)

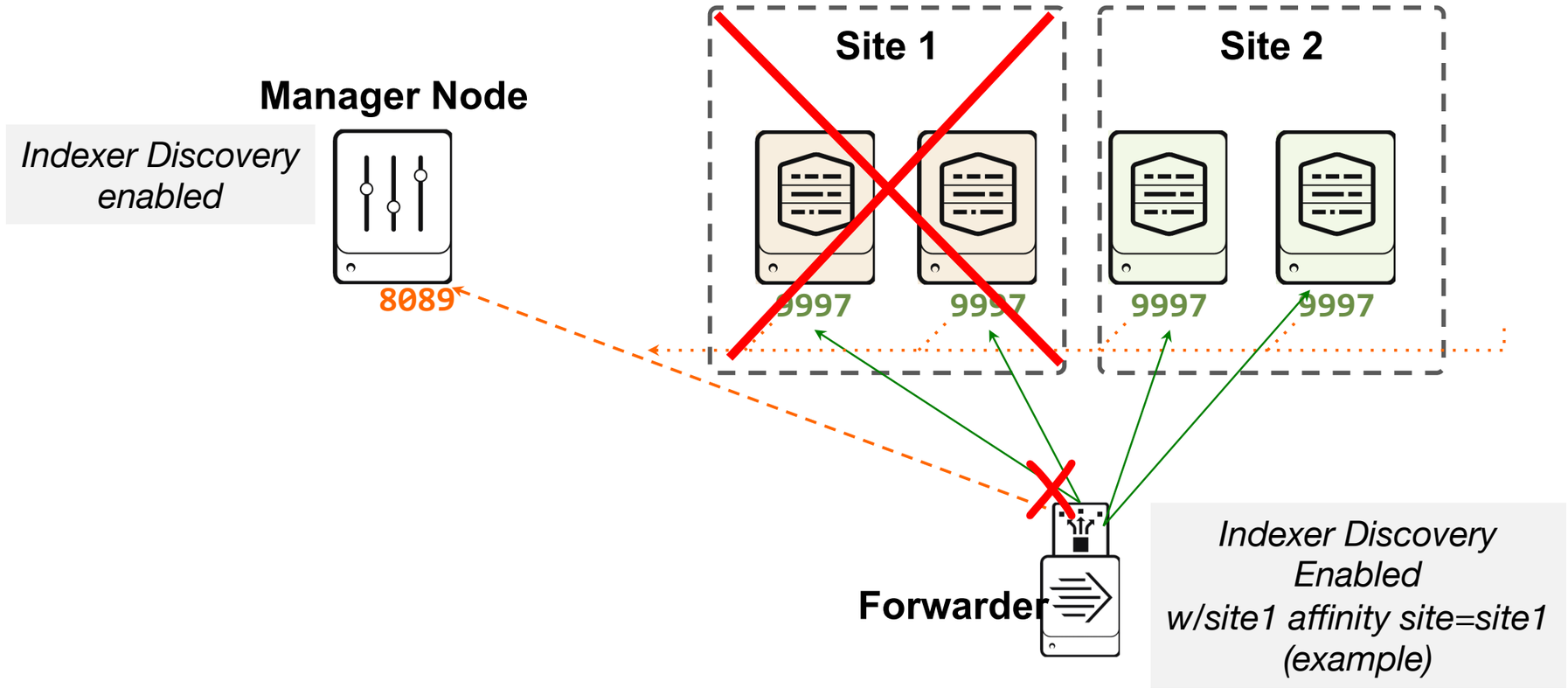
# of forwarders	polling_rate	polling_int_sec	polling_interval
100	1	100/1+30=**130**	2 min. 10 sec.
100	10	100/10+30= **40**	40 sec.
1,000	10	1000/10+30=**130**	2 min. 10 sec.

Manager Node server.conf

```
[indexer_discovery]
pass4SymmKey = <string>
polling_rate = <1-10>
```


Multisite Cluster Forwarder Site Failover

Manager Nodes w/ multi-site indexer clusters can optionally configure forwarder site failover in the Mgr Node's server.conf (Next Slide)



Forwarding to site affinity (the `site=siteN` set on fwdr) resumes when preferred indexer(s) back online

Configuring Forwarder Site Failover

- Multisite forwarders must use indexer discovery to enable failover
- Site-aware forwarders will not failover to another site by default
- Forwarding to original site resumes when node is back online

Configure on the Manager Node by editing **server.conf** or by running the following command:

```
splunk edit cluster-config -forwarder_site_failover <site-id>:<failover-id>
```

Manager Node `server.conf`

```
[clustering]
mode = manager
multisite = true
available_sites = site1,site2, site3
forwarder_site_failover = site1:site3,site2:site3
...
```

Note

This example uses three sites. but three sites are not required for failover: site1 could failover to site2 and vice versa.

Indexer Discovery Log Channels

Search manager node indexer discovery log channel for errors:

```
index=_internal component=CMIndexerDiscovery
```

Search forwarder **splunkd.log** for indexer discovery events:

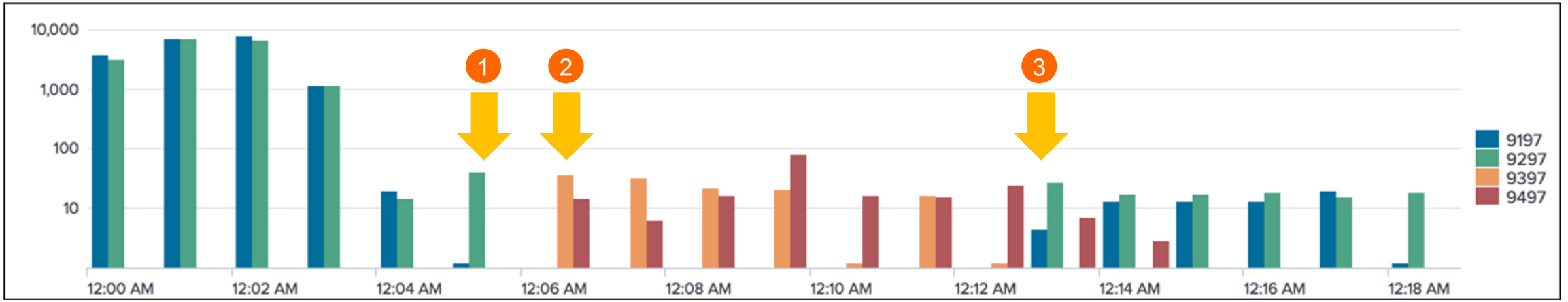
```
index=_internal component
IN(IndexerDiscoveryHeartbeatThread, HttpPubSubConnection,
TcpOutputProc)
```

Search forwarder **`metrics.log`** for data distribution and failover:

```
index=_internal host=uf component=Metrics
group=tcpout_connections
| timechart span=1h sum(kb) by destIp
```

Example: Indexer Discovery Site Failover

```
index=_internal host=uf Metrics group=tcpout_connections | timechart span=1h sum(kb) by destPort
```



1. Forwarder sending to site1 (9197 & 9297) until this point
2. Detected targets in site1 are no longer available and switch to failover site (9397 & 9497)
3. Original site recovered and forwarding targets gradually transition back to site1 peers

Note

In this example (student lab environment), the search uses **`destPort`**, instead of **`destIp`**.

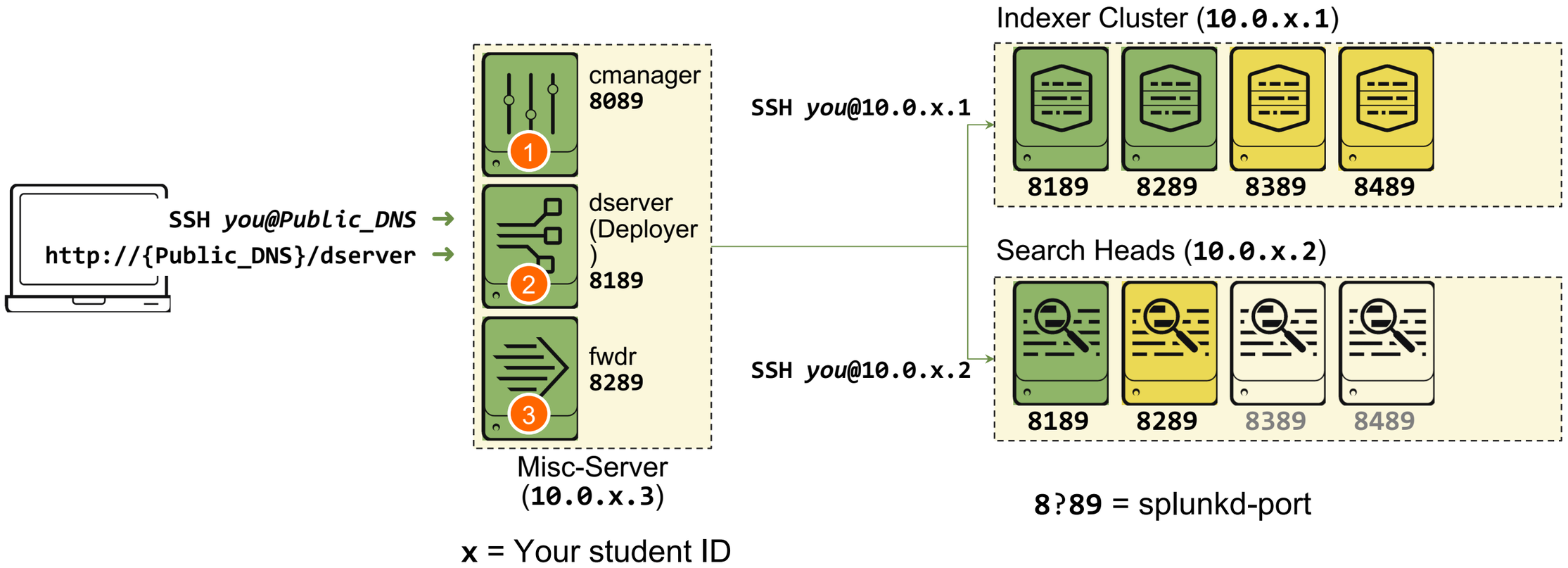
Further Reading: Forwarder Configuration

- Use indexer discovery to connect forwarders to peer nodes
- outputs.conf (Spec)
- Protect against loss of in-flight data
- server.conf (Spec)
- Set up load balancing
- props.conf (Spec)

Lab Exercise 5 – Configure a Forwarder

- **Time:** 30 minutes
- **Tasks:**
  - Enable the indexer discovery option with forwarder site failover on the manager node
  - Configure the deployment server
  - Enable the deployment client setting on the forwarder
  - Update the instance server role in Monitoring Console
  - Verify the forwarder app deployment
  - Test the forwarder site failover scenario

Lab Exercise 5 – Configure a Forwarder (cont.)



Module 6:
Search Head Cluster

Module Objectives

- Describe search head cluster architecture
- Configure a search head cluster
- Identify the captain and monitor cluster status
- Describe optional configuration settings

Search Head Cluster (SHC) Architecture

Composed of search heads (AKA cluster members) sharing:

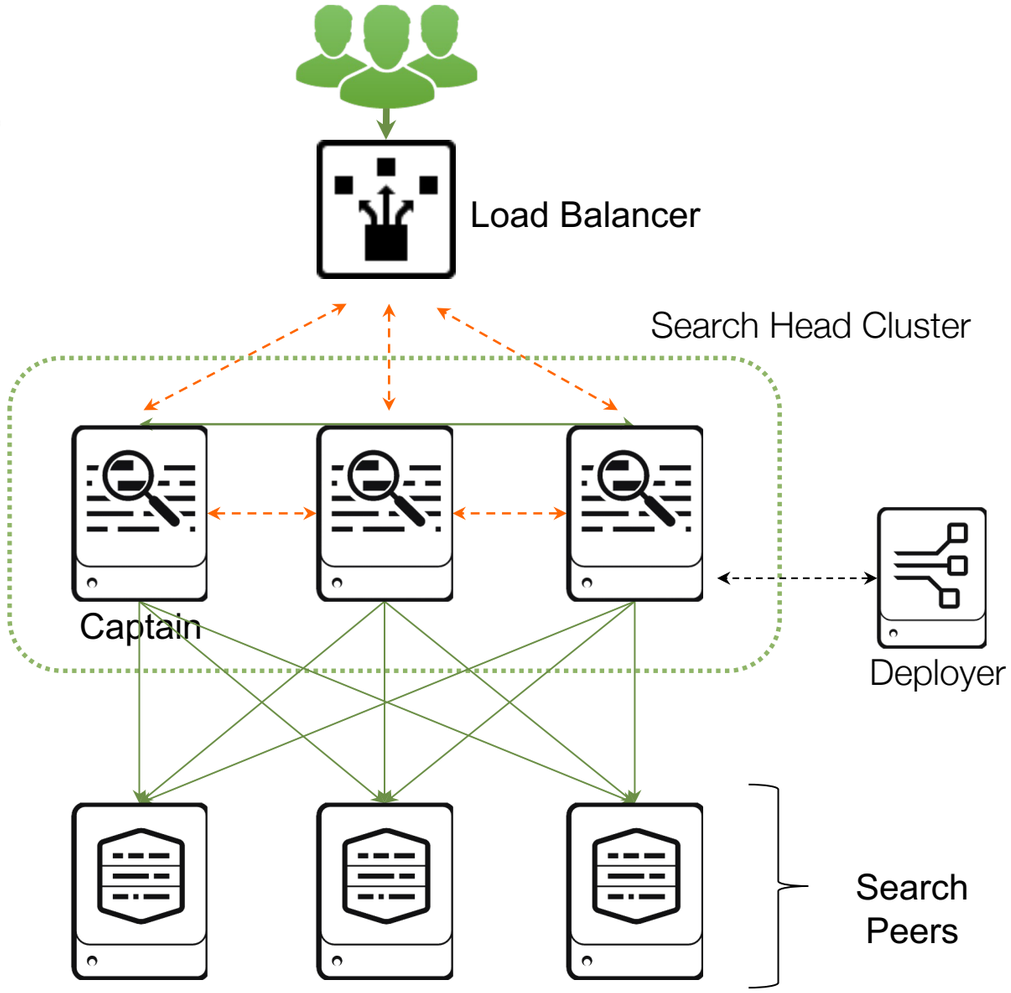
- Configurations: Cluster requires all members share same configs
- Job scheduling: Allocated to optimal member
- Search artifacts: Available to all members via replication

Uses raft distributed consensus

- One member serves as captain, coordinating cluster activity
- Members elect captain dynamically

Functioning cluster also requires:

- Deployer: Distributes apps and configs to cluster members
  - Not to be confused with Deployment Server
- Search peers: Either independent indexers or nodes in cluster
- Load balancer



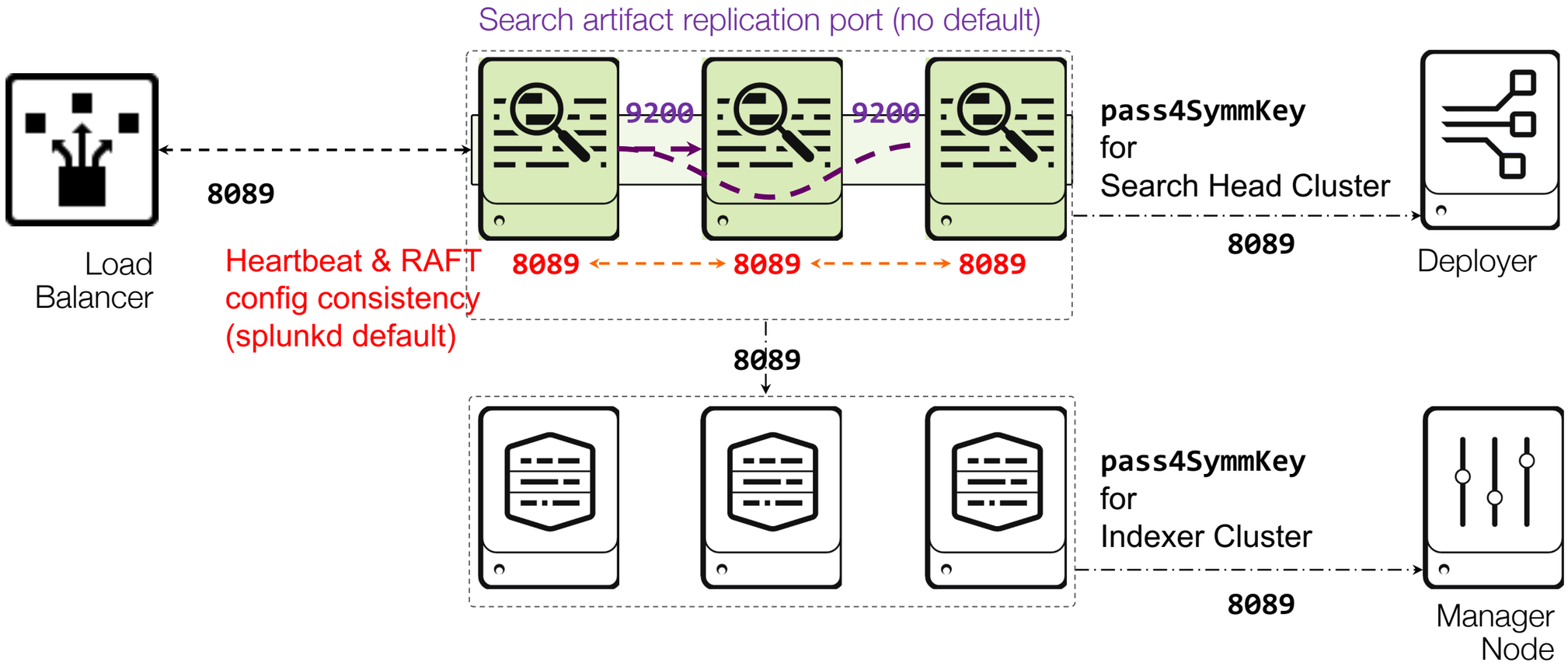
Benefits of Search Head Clusters

- Horizontal scaling
  - Add search heads as number of users and search load increases
- High availability
  - Because cluster members share jobs, search artifacts, and configurations, it doesn't matter which search head a user accesses. The user has access to the same set of dashboards, searches, etc.
- No single point of failure
  - Dynamic captaincy
  - Replication of configuration changes

Search Head Cluster Key Considerations

- Always use new Splunk instances
  - Minimum three members (able to survive 1 outage w/ majority)
  - Same install baseline default config as starting point
- Same hardware requirements as the dedicated search head
  - Use identical specifications for all members (bare metal or VM)
    - Works on all operating systems supported for Splunk Enterprise
    - Same version of Splunk Enterprise

Search Head Cluster Ports



Configure Search Head Cluster w/ CLI

1. Install Splunk Enterprise on each SH:
2. Configure `[shclustering]` stanza in `server.conf` on each SH followed by restart:

```
splunk init shcluster-config
-mgmt_uri https://SH2:8089
-replication_port 9200 -secret shcluster
```

3. Assign captaincy to one of the members and set a member list (include captain in the list):

```
splunk bootstrap shcluster-captain -servers_list
https://SH2:8089,https://SH3:8089,https://SH4:8089
```

4. Check search head cluster status:

   SplunkWeb: *Settings > Searchhead Clustering*

   CLI:
   ```
   splunk show shcluster-status
   splunk list shcluster-members
   ```

Note

CLI writes to **`SPLUNK_HOME/etc/system/local/server.conf.`**

`mgmt_uri` is HTTPS endpoint other members use for heartbeat and RAFT with this localhost

 |

Connecting a SHC to Non-Clustered Indexers

Two approaches to search non-clustered indexers:

1. Individually add the indexers (peers) from each SHC member w/ CLI:

```
splunk add search-server https://<peer>:8089 -remoteUsername <user> -remotePassword <pw>
```

2. Enable search peer replication by directly editing server.conf on all members :

```
[raft_statemachine]
disabled = false
replicate_search_peers = true
```

Once enabled, you can add new peers with CLI, Web, or REST API

Connecting a SHC to Indexer Clusters

Manually configure each SHC member as a search head on an indexer cluster

- The search head members get their list of search peers from the manager node of the indexer cluster

Connecting to a single-site indexer cluster (or use SplunkWeb UI:

```
splunk edit cluster-config -mode searchhead -manager_uri https://10.0.1.3:8089
-secret idxcluster
```

Connecting to a multisite indexer cluster (no SplunkWeb UI):

```
splunk edit cluster-config -mode searchhead -manager_uri https://10.0.1.3:8089 -site site2
-secret idxcluster
```


Search Head Cluster Member server.conf

Checking SH Cluster Status

```
> splunk show shcluster-status
```

```
Captain:
  dynamic_captain : 1
  elected_captain : Mon Mar 11 21:33:16 2019
               id : 75DBEA2A-0204-4C15-983C-116F1
 initialized_flag : 1
            label : sh2
        mgmt_uri : https://10.0.1.2:8289

min_peers_joined_flag : 1
 rolling_restart_flag : 0
   service_ready_flag : 1

Members:
         sh4    label : sh4
last_conf_replication : Mon Mar 11 21:54:39 2019
             mgmt_uri : https://10.0.1.2:8489
       mgmt_uri_alias : https://10.0.1.2:8489
               status : Up
```

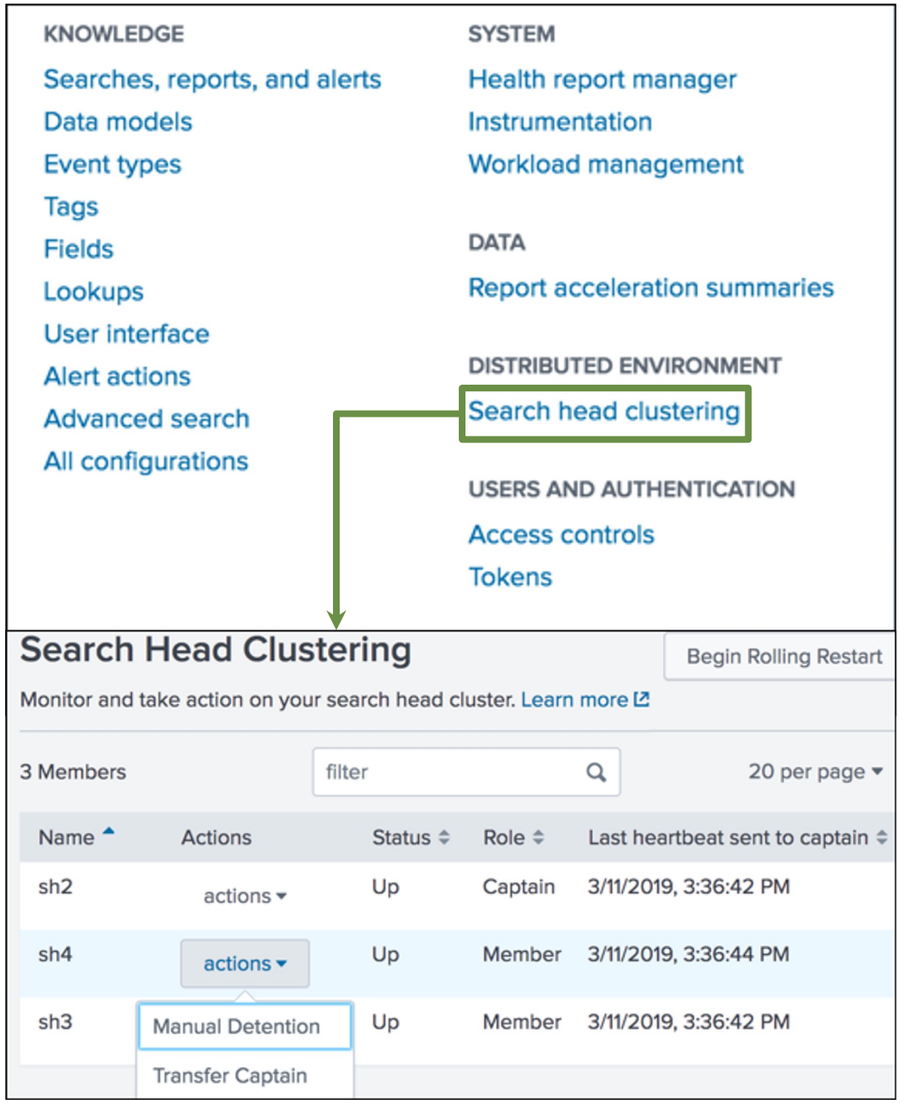
```
     sh2         label : sh2
last_conf_replication : Mon Mar 11 21:54:40 2019
             mgmt_uri : https://10.0.1.2:8289
       mgmt_uri_alias : https://10.0.1.2:8289
               status : Up

     sh3         label : sh3
last_conf_replication : Pending
             mgmt_uri : https://10.0.1.2:8389
       mgmt_uri_alias : https://10.0.1.2:8389
               status : Up
```

Note

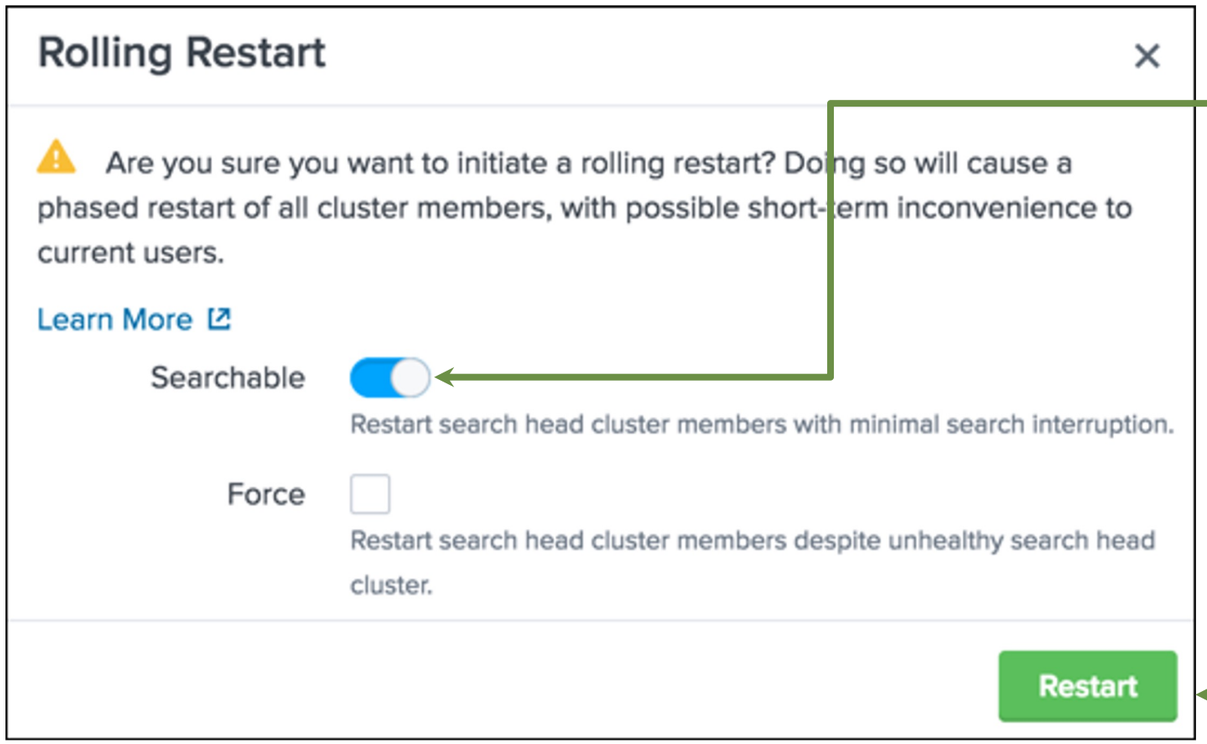
A raft metadata corruption can cause the captain election to fail. To confirm, look for **ERROR SHCRaftConsensus** in **splunkd.log**.

Splunk Web Settings Menu Changes

- When search heads become members of a search head cluster, the **Settings** menu in Splunk Web changes
  - Hides all non-replicable options
    - You can unhide them, if necessary
  - Enables the search head clustering UI on all SHC members
    - Able to perform rolling restart, manual detention and captaincy transfer
  - If you need to make changes to the settings that are hidden, use the **deployer** to push the underlying changes



Restarting a Search Head Cluster



Puts each SHC member in manual detention to complete in-progress searches before restarting SHC members

```
> splunk rolling-restart shcluster-members

> splunk rolling-restart shcluster-members
  -searchable true
  -decommission_wait_time 180
  -force false

> splunk rolling-restart shcluster-members -status 1
```

- Members restart in phases so the cluster can continue to operate
- The captain is the final member to restart and automatically invokes captaincy transfer, thus preventing captaincy from changing during the restart process
- Deployer automatically initiates a rolling restart, when necessary

Configuration and Artifact Replication

- Captain orchestrates both configuration and artifact replication
- Knowledge object configurations (replicated to all members):
  - Changes made via Splunk Web, CLI, or API are replicated
  - Direct **`.conf`** file edits must be implemented using the deployer
  - More details are discussed in the next module
- Artifacts (based on the search head cluster replication factor):
  - Only the artifacts resulting from scheduled reports are replicated
  - Real-time and ad-hoc search artifacts are not replicated, instead they are proxied (discussed later)
  - Captain enforces artifact fixups according to its replication policy

Ad-hoc Search Management

Ad-hoc searches:

- Are any non-scheduled search
- Generate artifacts
- Are not replicated
  - If the search artifact is needed (accessed) from another member, artifact proxying calls the owner member to get the results

Configure ad-hoc search preference in the `[shclustering]` stanza in `server.conf`:

captain_is_adhoc_searchhead	**true | false (default)**	If set to true, disable running scheduled searches on the captain; only run ad-hoc
adhoc_searchhead	**true | false (default)**	If set to true, disable schedule searches on SH; only run ad-hoc

Alerts

- When results of search meet alert condition to trigger action(s):
  - Configured actions are fired locally on the member that ran the search job
  - Alert information is reported to the captain
- Captain merges and maintains global view of alerts by:
  - Centralizing suppression (“throttling”) settings for Alerts
  - Sending merge alerts and suppression information are to all members
  - Not dispatching triggering search for suppression period

Handling Summary Indexing in SHC

- Accelerated Report and Data Model search summaries are stored on indexers
- Scheduled reports for indexing the summary results are *only* indexed on the SH that generates them.
  - They are not replicated by SHC
- Best practice for SH & SHC:
  - Forward any inputs (including summary and internal indexes) to the indexing layer

Search Head, `outputs.conf`

```
[indexAndForward]
index = false

[tcpout]
defaultGroup = default-autolb-group
indexAndForward = false

[tcpout:default-autolb-group]
server=idx1:9997,idx2:9997,…
```

Artifact Reaping

- Reaping of search job artifacts happens when artifact TTL expires
- Original member deletes its search artifacts and notifies captain
- Captain directs other indexers with replicas to delete their copy
- If the original member (owner of artifact) is unavailable, captain waits beyond TTL and directs indexers with replicas to delete copies

SHC Manual Detention

- Manual detention disables searching on the Search Head but allows continued participation in the cluster for maintenance and diagnostics
- Removing a cluster member from services (detention state) results in minimal impact on end-user search experience
  - Completes in-progress searches
  - Directs scheduled searches to remaining members
  - Continues to participate in election and conf/bundle replication
  - Does not participate in artifact replication
- Implement with the following commands:
  - **`splunk edit shcluster-config -manual_detention [on|off]`**
  - **`splunk show shcluster-status`**
  - **`splunk list shcluster-member-info`**

Useful SHC Debugging Searches

- Election history:
  - `index=_internal host IN (sh2, sh3, sh4) sourcetype=splunkd component=SHCRaftConsensus "All hail leader" | stats values(event_message) by _time host`
- Job scheduling status:
  - `index=_internal host IN (sh2, sh3, sh4) sourcetype=scheduler status=* | eval status_host=status."-".host | timechart span=5m count by status_host limit=0 usenull=f`
- Skipping jobs:
  - `index=_internal host IN (sh2, sh3, sh4) sourcetype=scheduler status=continued OR status=skipped | eval status_host=status."-".host | timechart span=5m count by status_host limit=0 usenull=f`
- Artifact proxy:
  - `index=_internal host IN (sh2, sh3, sh4) sourcetype=splunkd_access uri_path="/services/search/jobs*" isProxyRequest=true | stats count by method host file`

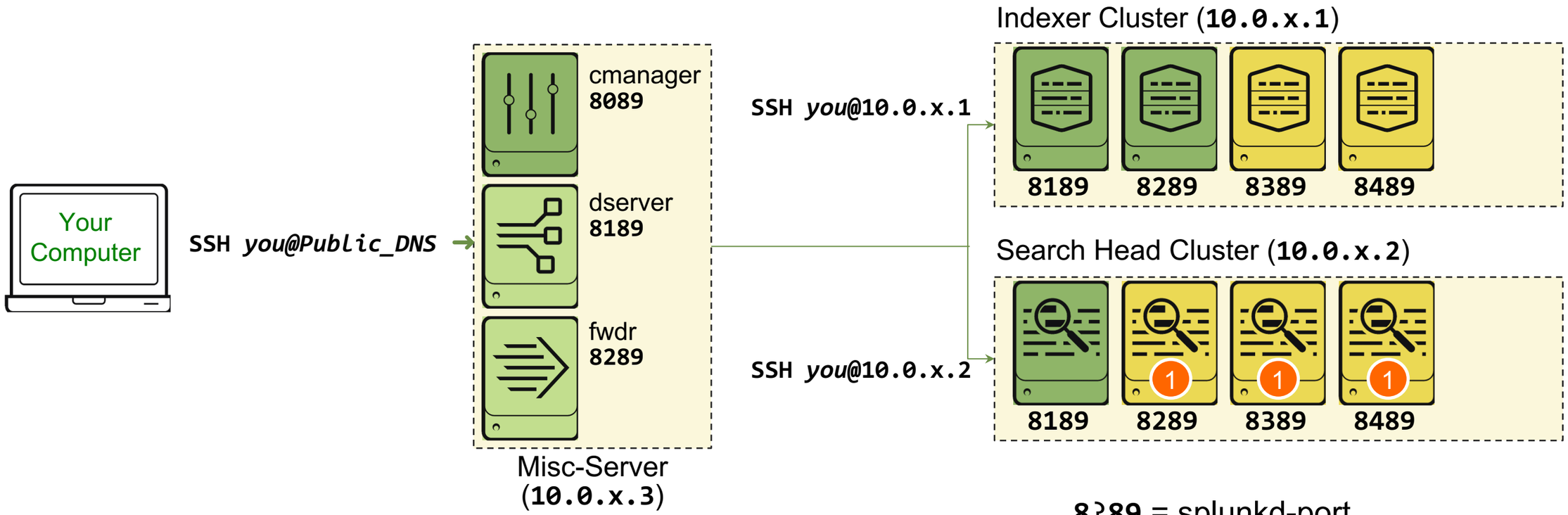
Further Reading: Search Head Clusters

- Search head clustering architecture
- About search head clustering
- Migrate from a search head pool to a search head cluster
- Control search concurrency on search head clusters
- Configure the priority of scheduled reports
- Configuration updates that the cluster replicates
- In Search of an Understandable Consensus Algorithm

Lab Exercise 6 – Deploy a Search Head Cluster

- **Time:** 30 minutes
- **Tasks:**
  - Add two more search heads to **`site2`**
  - Enable a search head cluster with **`site2`** search heads
  - Verify that your search head cluster is functioning

Lab Exercise 6 – Deploy a Search Head Cluster (cont.)



8?89 = splunkd-port

x = Your student ID

Module 7:
SHC Management and Administration

Module Objectives

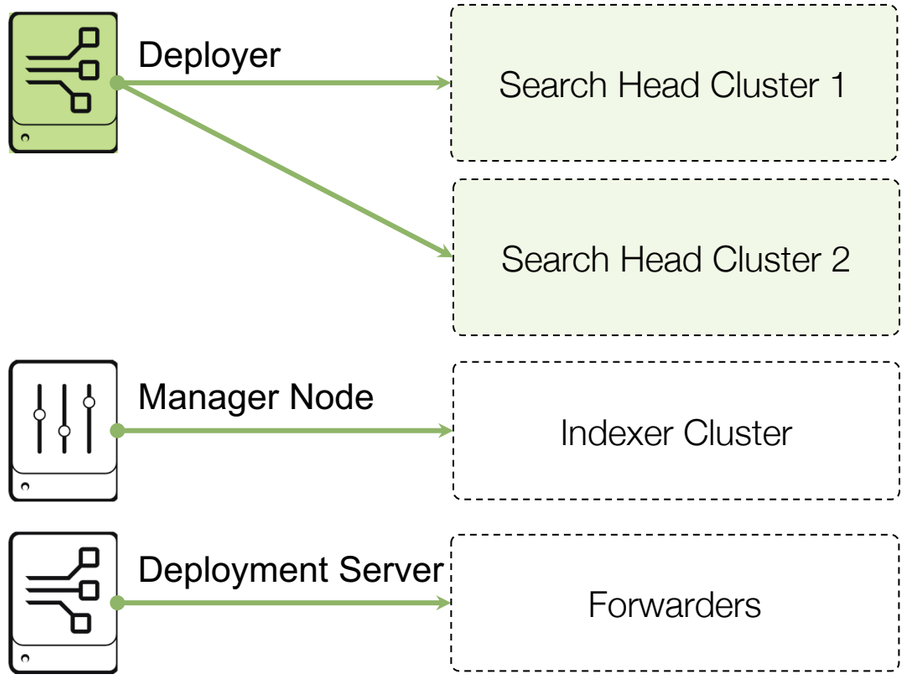
- Deploy apps to a search head cluster
- Describe when and how to transfer captaincy
- Upgrade and manage search head cluster members
- Monitor SHC health with Monitoring Console

Replication of Configuration Bundle

- All SHC members share the same configuration baseline
- Captain functions as a source control server
  - Members replicate changes from/to captain every 5 seconds (replication cycle)
  - Members generate a snapshot every minute and purge old sets every hour
    - **`var/run/splunk/snapshot/*.bundle`**
- All members SHOULD achieve *eventual consistency* at some point
  - In each replication cycle, members first pull (sync) outstanding changes from the captain, then push (submit) local changes to the captain
  - Members resolve conflicts on pull (sync)
  - All members keep a journal of changes in **`etc/system/replication/ops.json`**
    - Members check if the diverging point is at the end of captain's log
    - Pull all changes after the diverging point and then insert the changes

Splunk Deployer

- Distributes apps and non-replicable files into a SHC
  - Does not represent a "single source of truth"
  - Cannot use it alone to restore members to the latest state
- Must run on a non-search head member
  - Can be enabled on an existing Splunk instance with other responsibilities
- No CLI configuration support
  - Associate the deployer with a SHC by setting the **`pass4SymmKey`** in **`server.conf`**
- Can share a deployer with multiple SHCs if:
  - The clusters have exactly the same apps and configurations and use the same ***`secret`***



Deployer server.conf

```
[shclustering]
pass4SymmKey = <secret>
```

Deploying Bundles With Deployer

- Stage apps (configuration bundles) in deployer's **`etc/shcluster/apps`** dir
  - Optionally, configure push modes of each app
- Implement one of two approaches:
  1. Push bundles out from deployer to all search head members
     - **`splunk apply shcluster-bundle –target <member:port>`**
       - Can specify any member, but the deployer pushes the bundles to all members
  2. Configure search head members to fetch and sync after a restart

     `splunk edit shcluster-config -conf_deploy_fetch_url https://<deployer>:8089`
     - Adds the deployer address to the existing search head members

Setting the Deployer Push Modes

Configure push mode for app configuration deployment under **`[shclustering]`** stanza in **`app.conf`**

- **`deployer_push_mode`**
- **`deployer_lookups_push_mode`**

> WARNING
>
> DO NOT change the push mode when a member is down. Doing so can cause members to go out-of-sync.

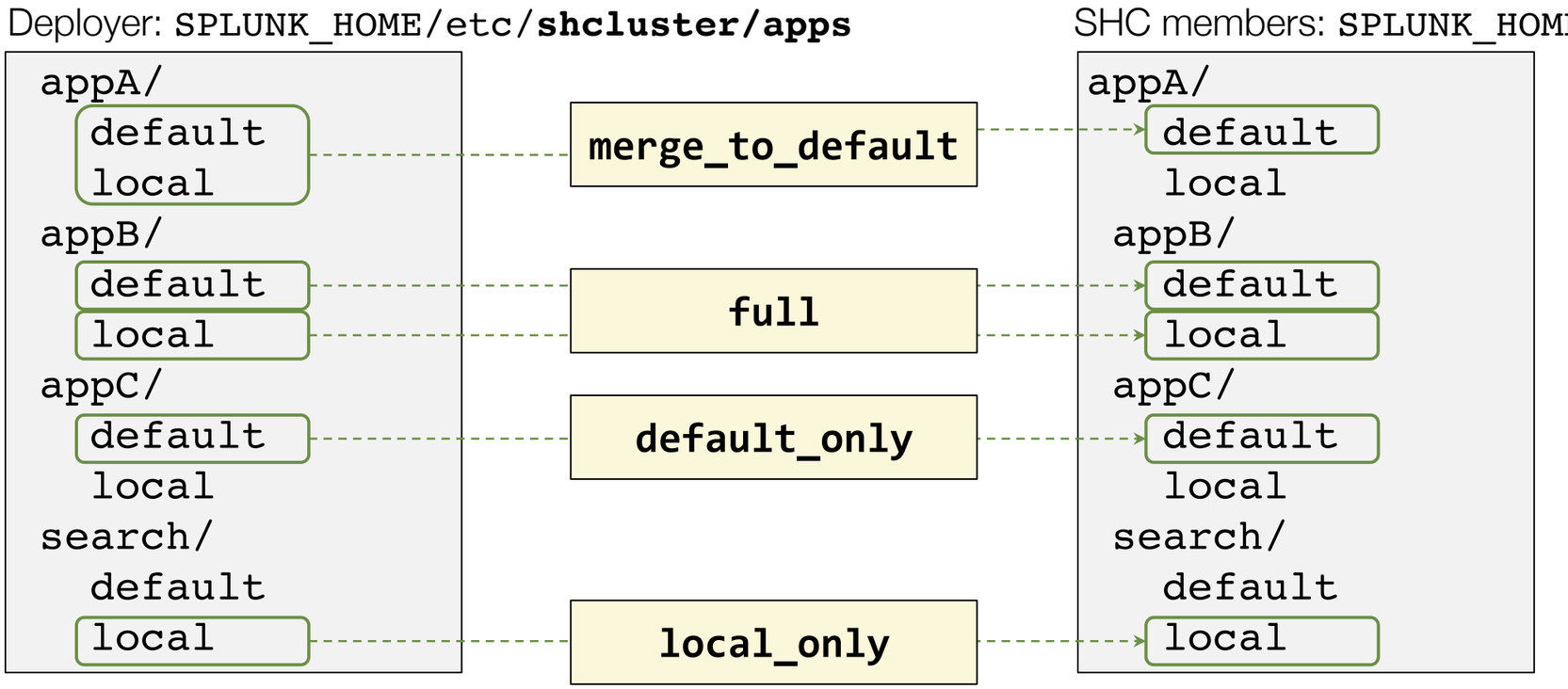
- Push mode can apply at the system level or on a per app basis
- Push mode options do not apply to user bundles
  1. Each user bundle is first sent to the captain
  2. The captain commits it to the user's **`local`** directory
     - Treats the commits as member configuration changes
     - Only adopts the new stanzas and ignores the existing stanzas

Deployer Push Mode Options

- Provides flexible ways to deploy apps from the deployer

  `deployer_push_mode = merge_to_default | full | default_only | local_only`

  - **`merge_to_default`** is the default if a mode is not specified
    - Is the only behavior for the versions prior to 7.3



Note

Bundles targeting **`local`** is sent to the captain. All others are sent to all members.

WARNING

DO NOT use the deployer to push built-in apps without first setting the push mode to **`local_only`**.

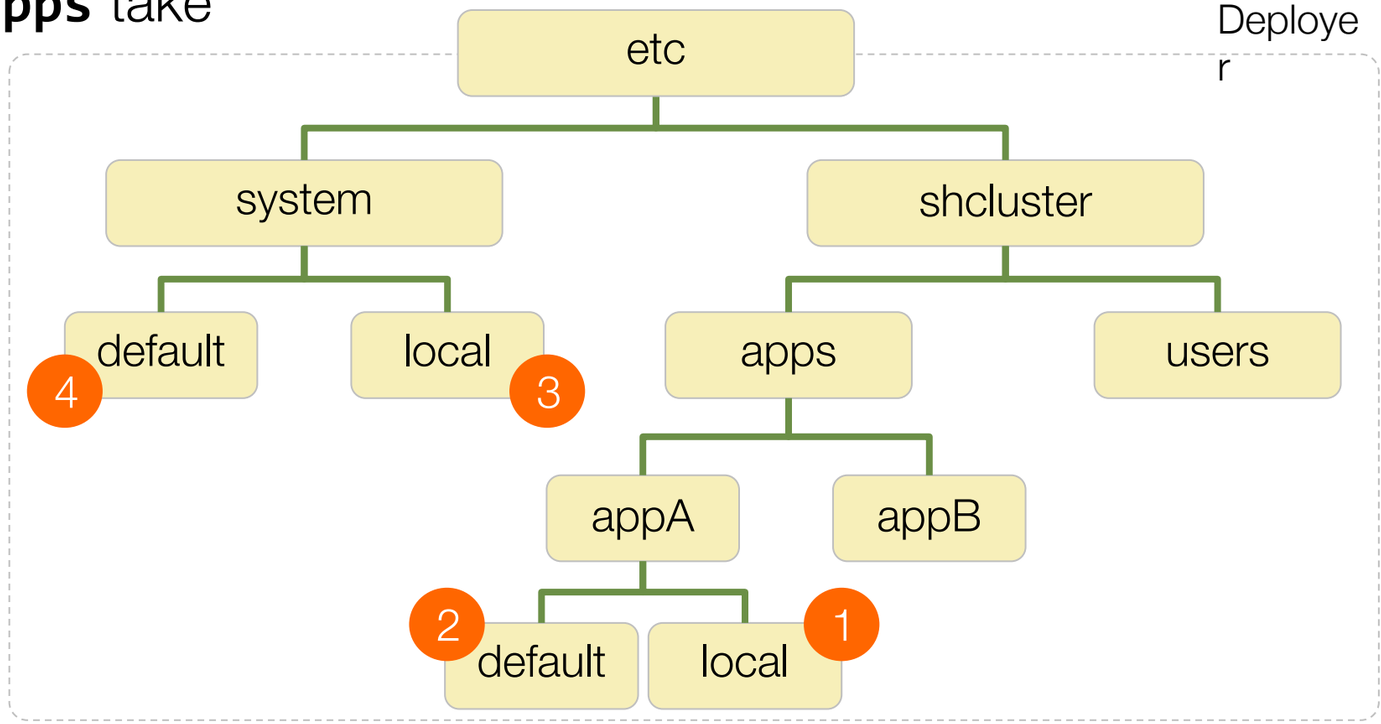
Deployer Lookup Push Mode Options

- Prior to 7.3, the populated lookup tables in SHC get overwritten from the latest bundle update from the deployer by default
  - To preserve the populated lookup tables, you execute:

    splunk apply shcluster-bundle -target *member* **-preserve-lookups true**
    - Can only preserve or overwrite all lookup tables
    - Even experienced admins can make a mistake
- The **deployer_lookups_push_mode** setting is introduced in 7.3
  - Provides more granular ways to preserve or overwrite lookups
  - Allows a way to persist the setting per app
  - **deployer_lookups_push_mode = preserve_lookups** (default)
    - Follows the **-preserve-lookups** flag of the command
  - **deployer_lookups_push_mode = always_preserve** | **always_overwrite**
    - Ignores the **-preserve-lookups** flag
    - Allows the **deployer_lookups_push_mode** of the app to decide

Deployer Push Mode Precedence

- The push mode can be set globally or locally for specific apps
  - If the push mode settings are not explicitly set under the specific app, the app follows the global deployer policy
  - The **`local`** takes precedence over any corresponding **`default`** settings
  - The settings under **`etc/shcluster/apps`** take precedence over **`etc/system`**



Note

The SHC members' existing configurations in **`local`** always take precedence over pushed configurations from a deployer.

Controlling the Deployment Process

- By default, **`splunk apply shcluster-bundle`** automatically triggers a rolling-restart as necessary
- To control the rolling-restart, you can apply in phases

```
splunk apply shcluster-bundle -target <member:port> -action stage
splunk apply shcluster-bundle -target <member:port> -action send
splunk rolling-restart shcluster-members
```

CAUTION: Use the following process with extreme care

- To delete all the apps on SHC members previously distributed by the deployer, you can push an empty bundle
  - The app must be in **`state=enabled`** (check its **app.conf**)
  - The deployer cannot remove the app if it is in **`state=disabled`**

```
splunk apply shcluster-bundle -target <member:port> -force true
```

Multithreaded SHC Deployer Bundle Push

- The **`deployerPushThreads`** allows you to concurrently send a large number of apps to a large number of SHC members
- Splunk checks the setting in the **`shclustering`** stanza of **`server.conf`** on the Deployer and launches the specified number of threads

  `auto` = deployer auto-tunes threads (one per member) depending on the number of members returned by the captain (preferred setting)

  `1` = default setting

  `<positive integer>` = sets thread number

```
[shclustering]
deployerPushThreads = auto
```

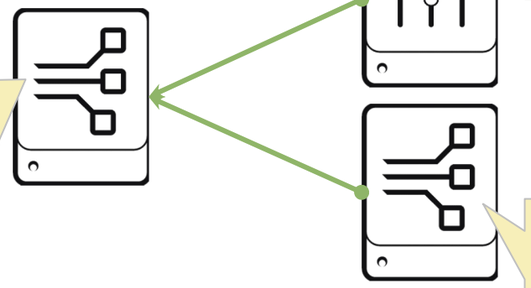
server.conf

Working with Deployment Server

- You CANNOT use deployment server to directly distribute apps to the peer nodes or SHC members
- You can use it to distribute apps to the **manager-apps** and **shcluster** directories

`serverclass.conf`
on **Deployment Server**

```
[serverClass:idxc_x]
stateOnClient = noop
restartSplunkd = false

[serverClass:shc_y]
stateOnClient = noop
restartSplunkd = false
```



`deploymentclient.conf` on **Manager Node**

```
[deployment-client]
serverRepositoryLocationPolicy = rejectAlways
repositoryLocation = SPLUNK_HOME/etc/manager-apps
```

`deploymentclient.conf` on **Deployer**

```
[deployment-client]
serverRepositoryLocationPolicy = rejectAlways
repositoryLocation = SPLUNK_HOME/etc/shcluster/apps
```

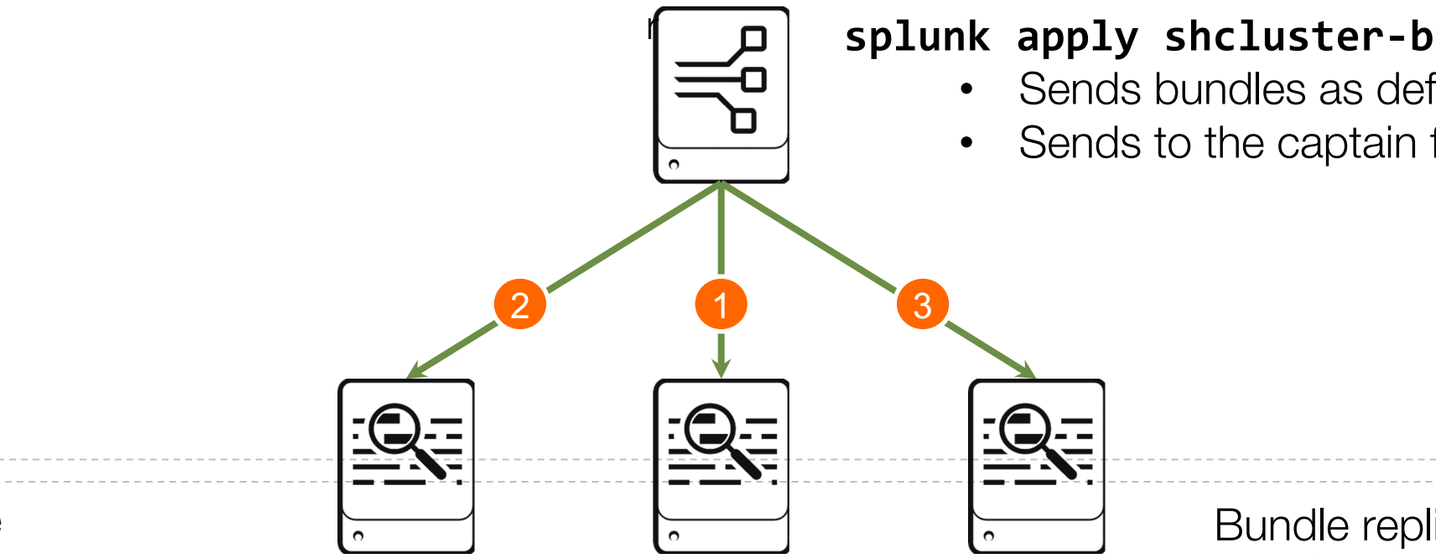
Bundle Deployment Summary

App bundle deployment

Deploye

`splunk apply shcluster-bundle`

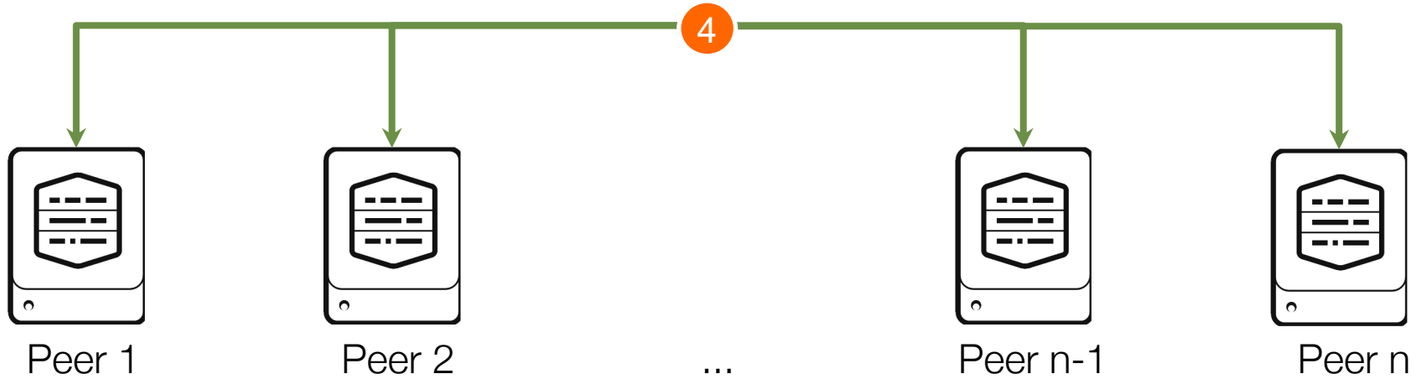
- Sends bundles as defined in **`server.conf`**
- Sends to the captain first



Search bundle deployment

Bundle replication to search peers happens only from the captain

- Sends bundles in parallel per bundle



SH Cluster Conf Replication Log Channels

- **conf.log** records recent configuration replication activity

  index=_internal sourcetype=splunkd_conf |timechart list(data.task) by host

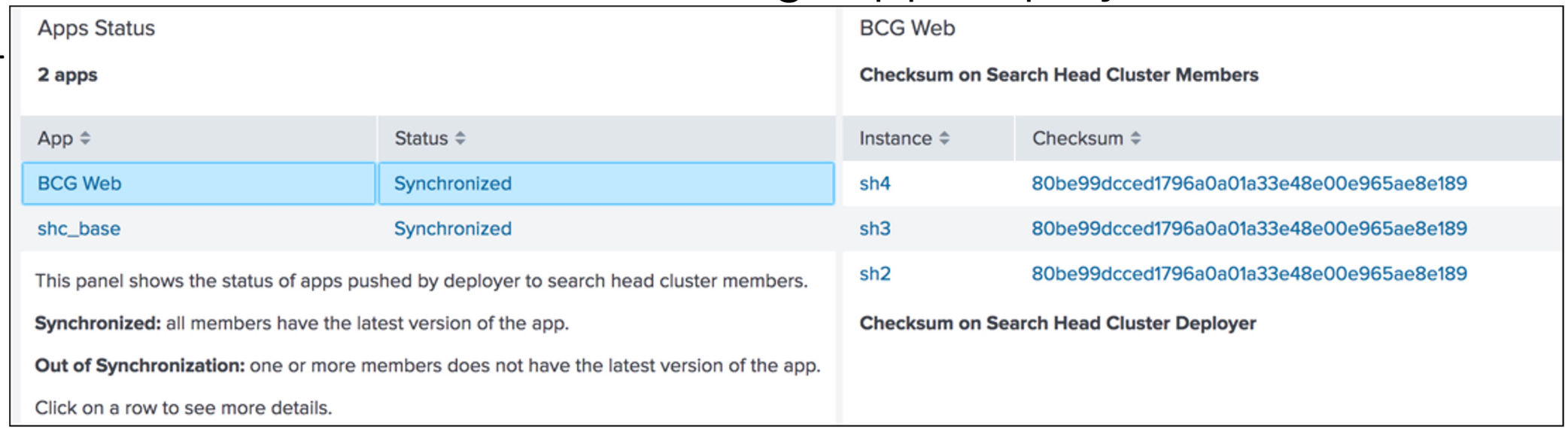
  – Deployer-specific actions:
    - **createDeployableApps**: build bundles in **var/run/splunk/deploy**
    - **populateDeployableInfo**: read bundles from **var/run/splunk/deploy**
    - **sendDeployableApps**: push baseline apps to a search head cluster member

  – SHC member-specific actions:
    - **addCommit**: changes made locally on this member
    - **pullFrom**: changes pulled from the captain to this member
    - **acceptPush**: changes pushed from a member to this instance
    - **computeCommon**: initial negotiation with the captain
    - **purgeEligible**: purge in-memory and on-disk data to reduce resource usage
    - **installSnapshot**: update to a snapshot of the latest bundle from the captain
    - **downloadDeployableApps**: install baseline apps from the deployer on startup

Checking SHC Bundle Deployment Status

- Search from a search head member:

```
index=_internal component=ConfDeployment data.task=*Apps
| table host data.source data.target_label data.task data.status
```

- Check the Search Head Clustering: App Deployment dashboard
  - 

Apps Status

2 apps

App	Status
BCG Web	Synchronized
shc_base	Synchronized

This panel shows the status of apps pushed by deployer to search head cluster members.

Synchronized: all members have the latest version of the app.

Out of Synchronization: one or more members does not have the latest version of the app.

Click on a row to see more details.

BCG Web

Checksum on Search Head Cluster Members

Instance	Checksum
sh4	80be99dcced1796a0a01a33e48e00e965ae8e189
sh3	80be99dcced1796a0a01a33e48e00e965ae8e189
sh2	80be99dcced1796a0a01a33e48e00e965ae8e189

Checksum on Search Head Cluster Deployer

Some Useful Debugging Searches

- Find missing baseline:

```
index=_internal sourcetype=splunkd_conf
STOP_ON_MISSING_LOCAL_BASELINE | timechart count by host
```

- Overall configuration replication behavior

```
index=_internal sourcetype=splunkd_conf pullFrom
data.to_repo!=*skipping* | timechart count by data.to_repo
```

- Evidence of captain switching

```
index=_internal sourcetype=splunkd_conf pullFrom
data.from_repo!=*skipping* | timechart count by
data.from_repo
```

- Find the destructive resync events:

```
index=_internal sourcetype=splunkd_conf installSnapshot |
timechart count by host
```

More Useful Conf Replication Log Channels

- **splunkd.log** components
  - **ConfReplication**
  - **ConfReplicationThread**
  - **ConfReplicationHandler**
  - **loader** – check events during startup
- Replication performance: **metrics.log group=conf**
  - **wallclock_ms_total** is for an action and **wallclock_ms_max** is for a single invocation
- Network activity between the members:
  - **sourcetype=splunkd_access uri_path="/services/replication/configuration*"**
- Network activity between the members and the deployer:
  - **sourcetype=splunkd_access uri_path="/services/apps/local*" OR uri_path="/services/apps/deploy*"**

Types of SHC Captaincy

- SHC has two types of captaincy designed to handle specific cases
  - Dynamic captain (default)
    - If you want to reassign captaincy, you can manually transfer captaincy to a preferred member
  - Static captain
    - When members are unable to elect a captain

Controlling Captaincy

- Why?
  - SHC Captain consumes more CPU and memory resources
  - If a system with less number of cores becomes the captain, the overall scheduling capacity gets lowered
    - SHC search capacity **= *captains_scheduler_count x #_of_members*** with scheduled search enabled (**adhoc_searchhead = false**)
  - Assign captaincy to members based upon geographic preference
- How?
  - Set **preferred_captain=false** in **server.conf** to exclude a member
  - SHC tries to elect a member with **preferred_captain=true** (default), but not always possible
    - For example, if no preferred members are reachable by the majority
  - When a non-preferred captain is chosen, the first available preferred captain member requests a captaincy transfer

Transferring Captaincy

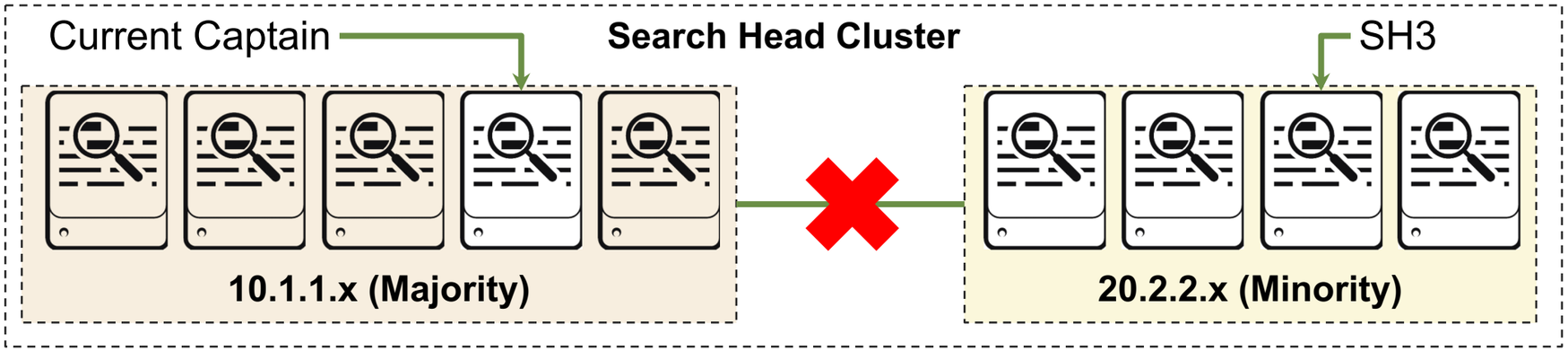
- To transfer the captaincy to a preferred member, run this command from any member:

  `splunk transfer shcluster-captain -mgmt_uri <new_captain>`

- NOTE:
  - **`splunk rolling-restart shcluster-members`** automatically invokes the captaincy transfer
- To check the transfer status, run this from any member:

  `splunk show shcluster-status`

Designating a Static Captain



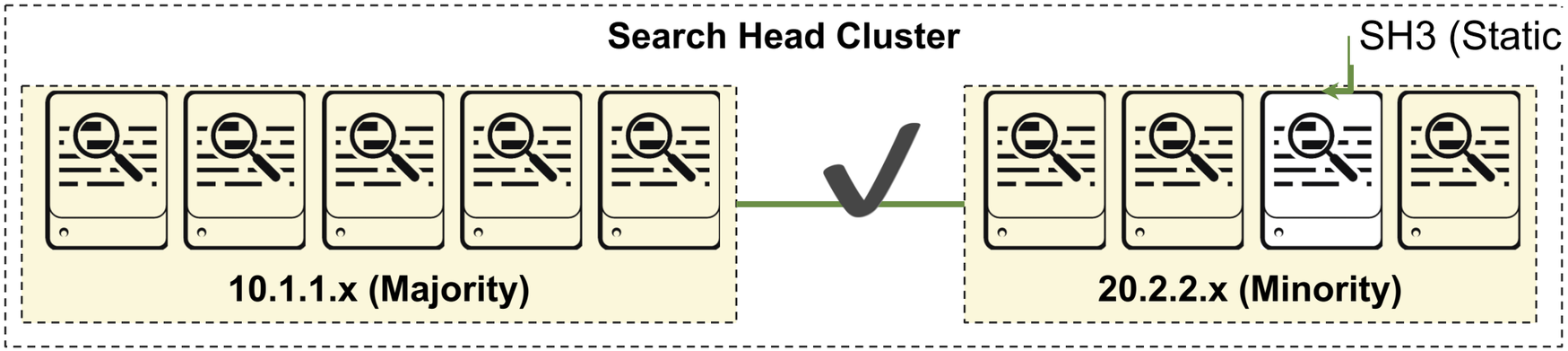
- During a network partition failure, minority group members are unable to elect a captain
- Splunk admins can designate a captain for the minority group as a temporary workaround
  - On the new captain node (**SH3**):

```
splunk edit shcluster-config -election false -mode captain -captain_uri https://SH3:8089
```

  - On the rest of the minority group members:

```
splunk edit shcluster-config -election false -mode member -captain_uri https://SH3:8089
```

Restoring the Cluster to Dynamic Captaincy



Upon resolution of the issue, revert to the dynamic captain mode

1. On all majority members: Disable election and report to the static captain

```
splunk edit shcluster-config -election false -mode member -captain_uri https://SH3:8089
```

2. On all members: Re-enable election; change the static captain last

```
splunk edit shcluster-config -election true -mgmt_uri <THIS_MEMBER>:8089
```

3. On **SH3**: Bootstrap as the new dynamic captain for the entire SHC

```
splunk bootstrap shcluster-captain -servers_list <SEARCH_HEAD_MEMBER_LIST>
```

Adding Search Head Members

- Why always use new Splunk instances?
  - SH cluster configuration consistency is derived from the journals
  - The **`ops.json`** file is fixed in size (configurable) and gets rolled
  - Non-members do not have this file
  - Members downed for a long time may not be able to find the diverging point
- Add an offline member to the search head cluster:
  - Restart the offline member
    - Receives the set of intervening changes from the captain and should resync
  - If unable to fully recover, you can force the resync on the re-joining member:

    `splunk resync shcluster-replicated-config`
- Add a new member or a previously decommissioned member:
  - Details on the next slide

Adding a New or Decommissioned Member

1. Delete Splunk Enterprise, if exists
2. Install and initialize the instance
3. Join the search head cluster
   - To announce itself to an existing SH cluster, run on the new instance:

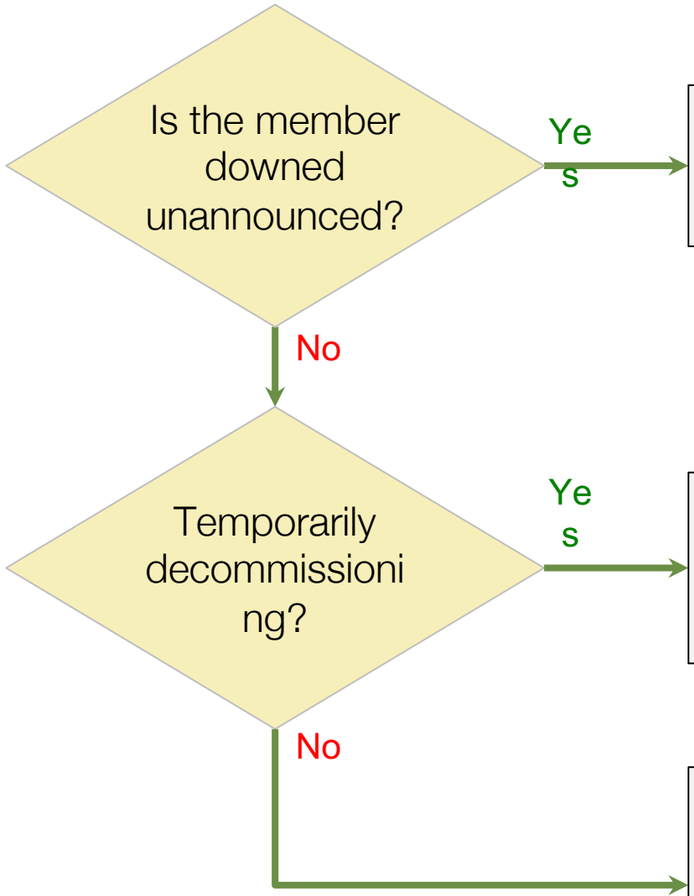
     ```
     splunk add shcluster-member -current_member_uri
     https://<any_existing_member>:8089
     ```

   - To introduce a new member to the cluster, run from any existing member:

     ```
     splunk add shcluster-member -new_member_uri
     https://<new_member>:8089
     ```

- When a new member joins the cluster, it gets:
  - The deployed bundles from the deployer
  - The replicated configurations and artifacts from the captain

Decommissioning a SH Cluster Member



Run from any SHC member:

```
splunk remove shcluster-member -mgmt_uri <downedSH:m_port>
```

Run on the decommissioning member:

```
splunk remove shcluster-member
splunk stop
```

Run on the decommissioning member:

```
splunk remove shcluster-member
splunk disable shcluster-config
```

Important

DO NOT just run: **`splunk stop`**.

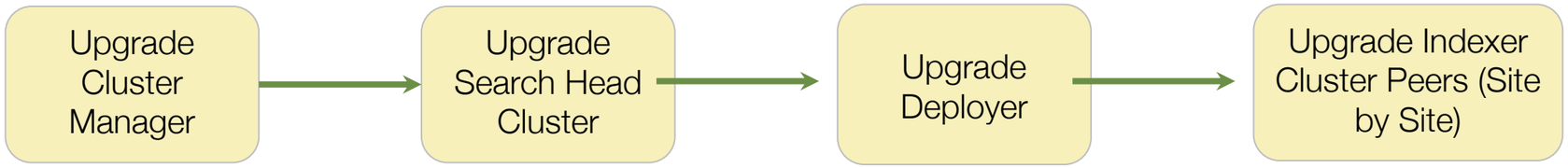
Performing a Rolling Upgrade of SHC

Beginning with Splunk Enterprise 9.1.0:

- Rolling upgrade process is **automated**
- Enabled by default
- Straightforward configuration
- Provides 2 interfaces: REST API & CLI
- Only supported on Linux
- Only upgrades from tarballs

Prior to Splunk Enterprise 9.1.0:

- Rolling upgrade process is **manual**
- All SHC members and indexer nodes must be running version 7.1 or later



Performing a Rolling Upgrade of SHC >=9.1

Specify path to newer Splunk version in **rolling_upgrade.conf:**

```
[downloader]
package_path=https://<URL>/splunk-9.1.0-1c86ca0bacc3-Linux-x86_64.tgz

# package_path=file://<ABSOLUTE PATH>/splunk-9.1.0-1c86ca0bacc3-Linux-x86_64.tgz
```

$SPLUNK_HOME/etc/apps/splunk-rolling-upgrade/local/rolling_upgrade.conf

Specify user with permission to retrieve new session key:

```
[script://$SPLUNK_HOME/etc/apps/splunk-rolling-upgrade/bin/complete.py]
passAuth=<SPLUNK USER>
```

Trigger on any SHC peer:

- `$SPLUNK_HOME/bin/splunk rolling-upgrade shc-upgrade`

Monitor process:

- `$SPLUNK_HOME/bin/splunk rolling-upgrade shc-upgrade`

Repeat process on the deployer.

Note

The specified user requires a newly-introduced capability: **upgrade_splunk_shc,** only enabled on admin user by default.

 |

Performing a Rolling Upgrade of SHC <9.1

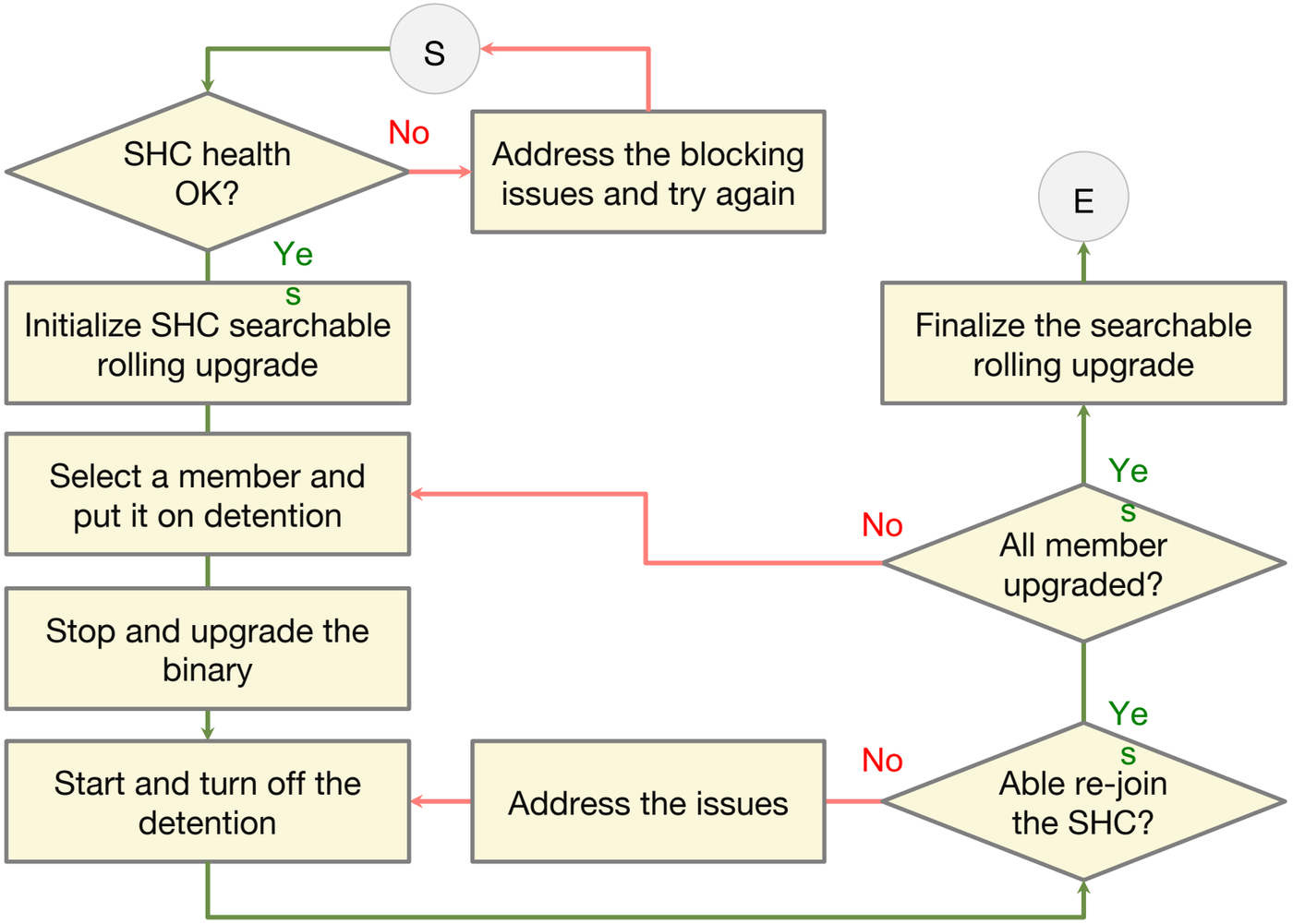
```
splunk show shcluster-status --verbose
```

```
splunk [upgrade-init | upgrade-finalize]
 shcluster-members
```

```
splunk list shcluster-member-info
```

Confirm `active_historical_search_count = 0`

```
splunk edit shcluster-config
 -manual_detention [on | off]
```



Note

Splunk provides a template script, **`SPLUNK_HOME/bin/shc_upgrade.py`**, that you can modify and use.

Key SH Cluster Maintenance Commands

```
> splunk help shclustering
```

- Helpful CLI commands (can be run on Captain or members)

```
splunk rolling-restart shcluster-members
splunk show shcluster-status
```

- Helpful CLI commands to run on the SHC members

```
splunk [edit|list] shcluster-config
splunk add shcluster-member
splunk resync shcluster-replicated-config
splunk clean raft
```

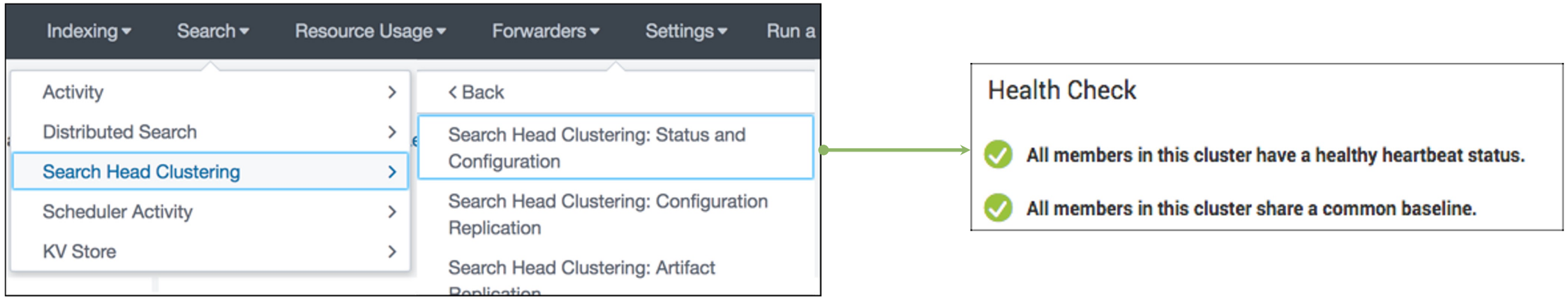
- Helpful CLI commands to run on the SHC deployer

```
splunk apply shcluster-bundle
```

Search Head Cluster Health Report

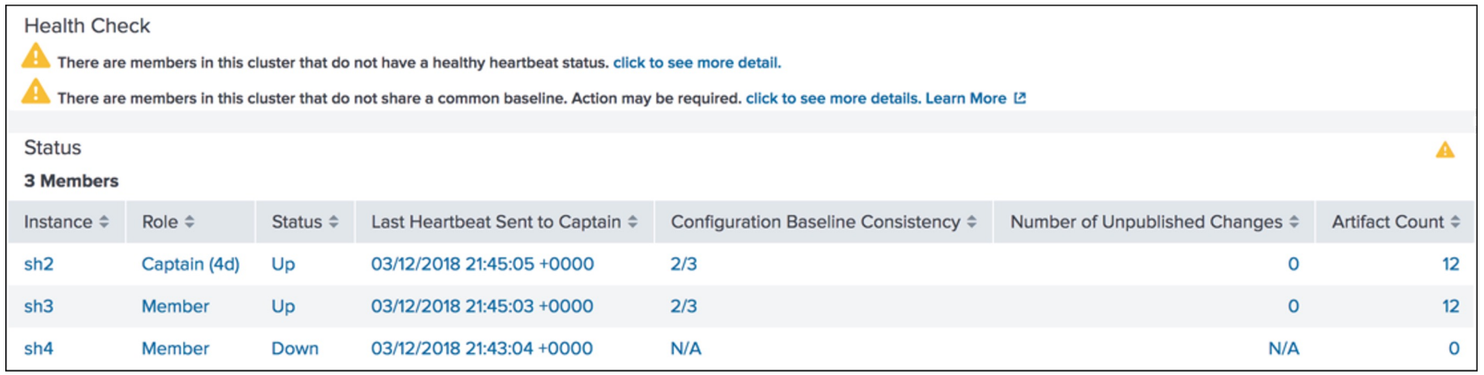
- **`feature:shc_members_overview`** tracks the status of SHC
  - **`status`** turns yellow if members skip a heartbeat and red if they skip more times
  - **`replication_factor`** checks if enough search head cluster members exist to honor the artifact replications (yellow only)
  - **`detention`** turns yellow if a member is in manual detention, red if a member is in automatic detention
- **`feature:shc_captain_election_overview`** tracks the quorum majority required to re-elect a dynamic captain (yellow only)
- **`feature:shc_captain_common_baseline`** reflects if the bundles are synced across members (red only)

Search Head Cluster Dashboards in MC

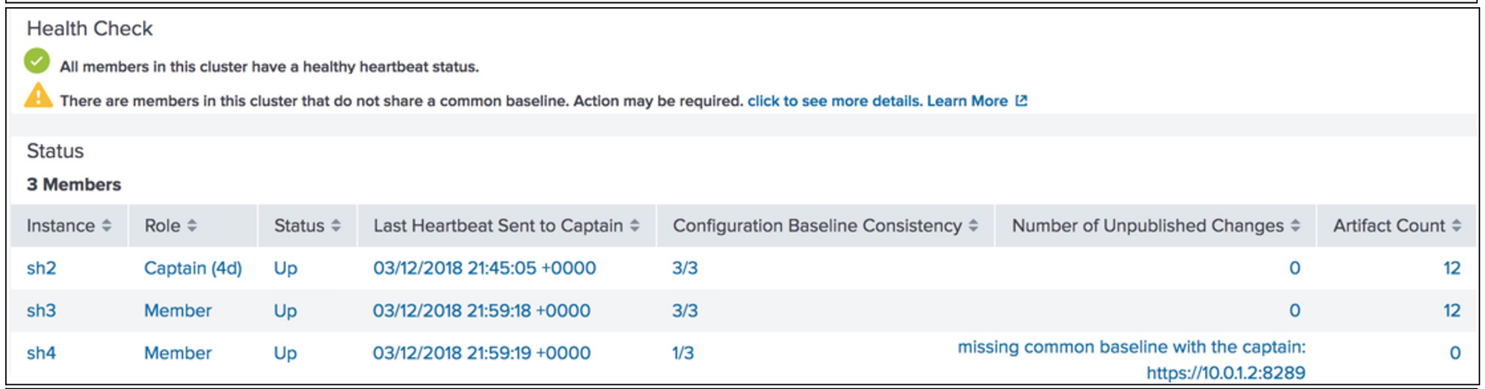


- **SHC Status and Configuration** provides high-level SHC health
  - The last heartbeats should be nearly the same
  - A member heartbeat drift and frequent captain election indicate possible problems
- **Configuration Replication** shows how user changes propagate among its members
  - Drill down to see members' common baseline and detect unpublished changes
- Growing backlogs in **Artifact Replication Job Activity** should be investigated
- In **Scheduler Activity**, monitor **Skip Ratio** and **Execution Latency**

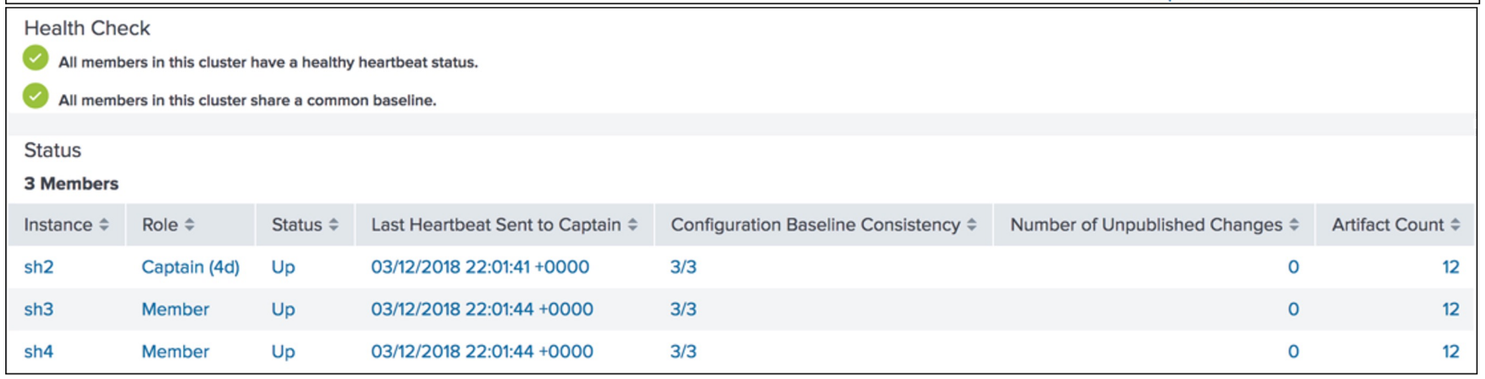
Troubleshooting the SHC with Monitoring Console



By default, members report a heartbeat every five seconds and the captain marks a member as **Down** after missing for 60 seconds



If a member was down for a long period of time, it may not be able to find a baseline and a manual resync on the member is required



To restore consistency:

```
splunk resync
shcluster-replicated-config
```

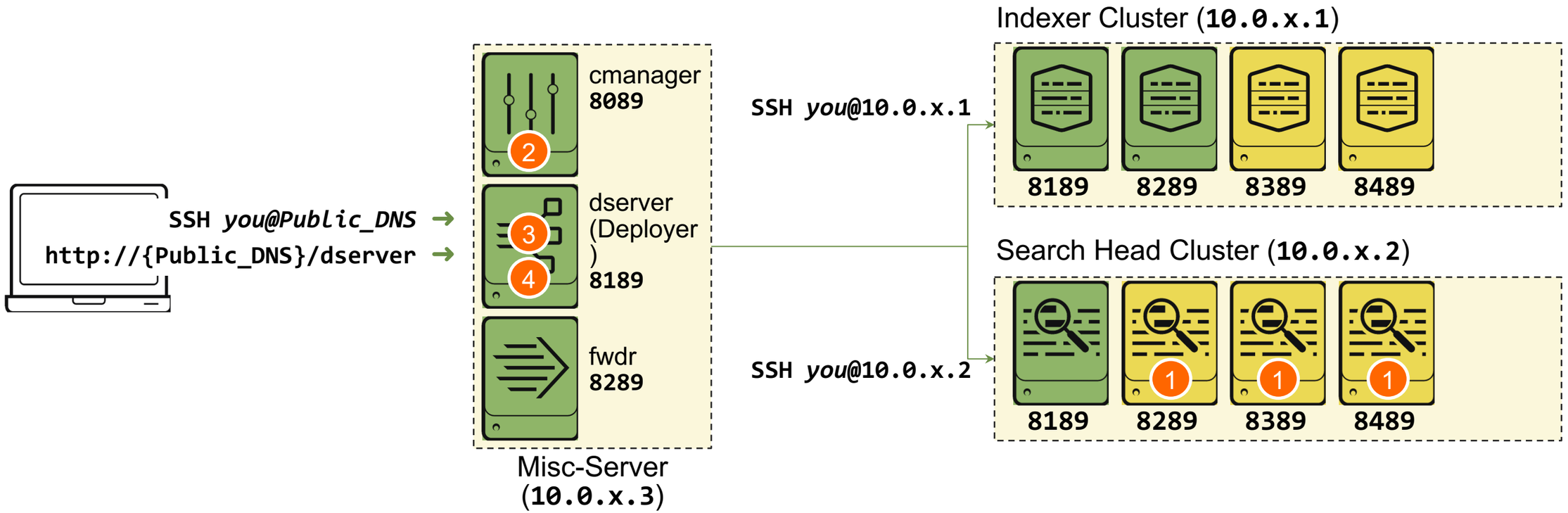
Further Reading: SHC Management

- Best practice: Forward manager node data to the indexer layer
- Use the deployer to distribute apps and configuration updates
- Use the monitoring console to view search head cluster status and troubleshoot issues
- Use static captain to recover from loss of majority
- Configuration updates that the cluster replicates
- Upgrade a search head cluster
- Handle failure of a search head cluster member

Lab Exercise 7 –Deploy a SHC App

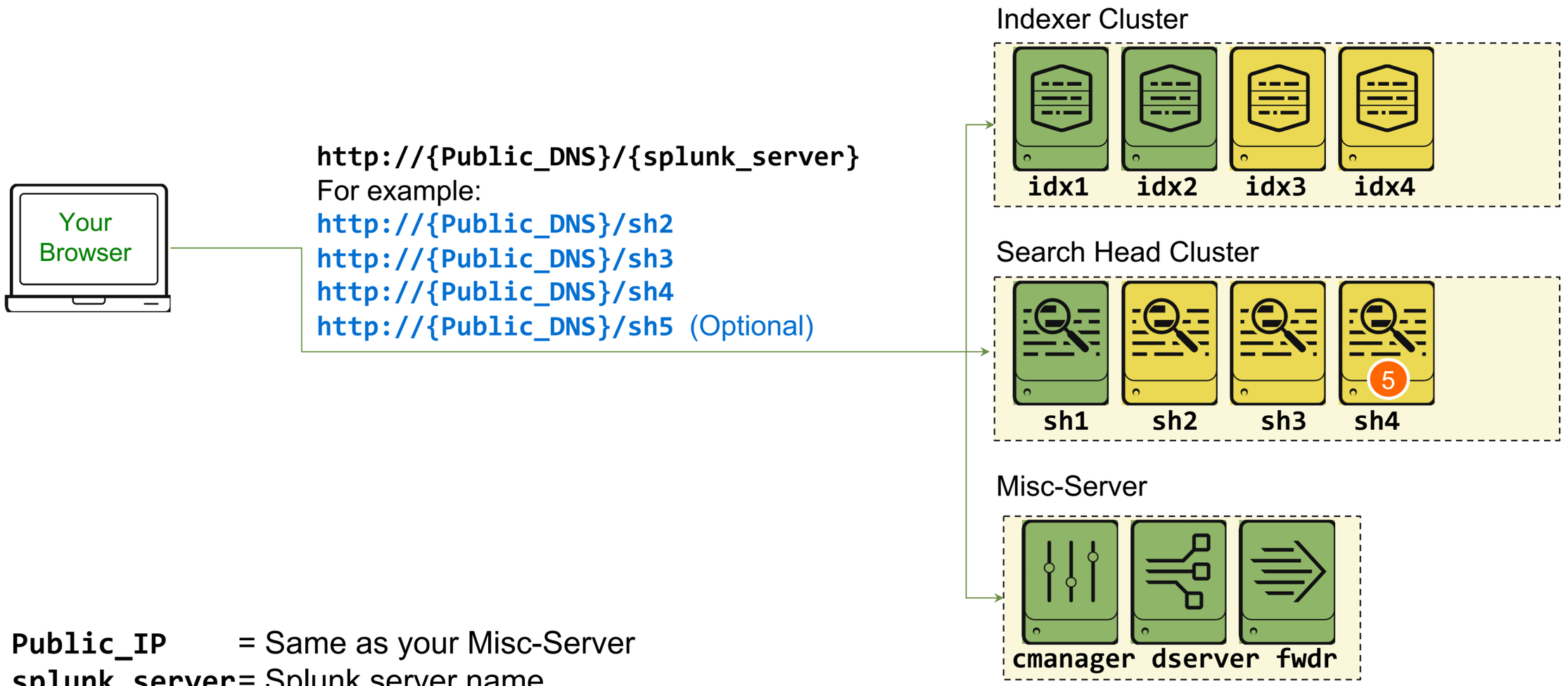
- **Time**: 30 minutes
- **Tasks**:
  - Configure the SHC members and the deployer
  - Stage and distribute apps to search head cluster members
  - Verify the app deployment
  - Complete the Monitoring Console setup on **`dserver`**
  - Review the search head cluster dashboards

Lab Exercise 7 – Deploy a SHC App (cont.)



x = Your student ID
8?89 = splunkd-port

Lab Exercise 7 – Deploy a SHC App (cont.)



Public_IP = Same as your Misc-Server
splunk_server = Splunk server name

Module 8:
KV Store Management in Splunk Clusters

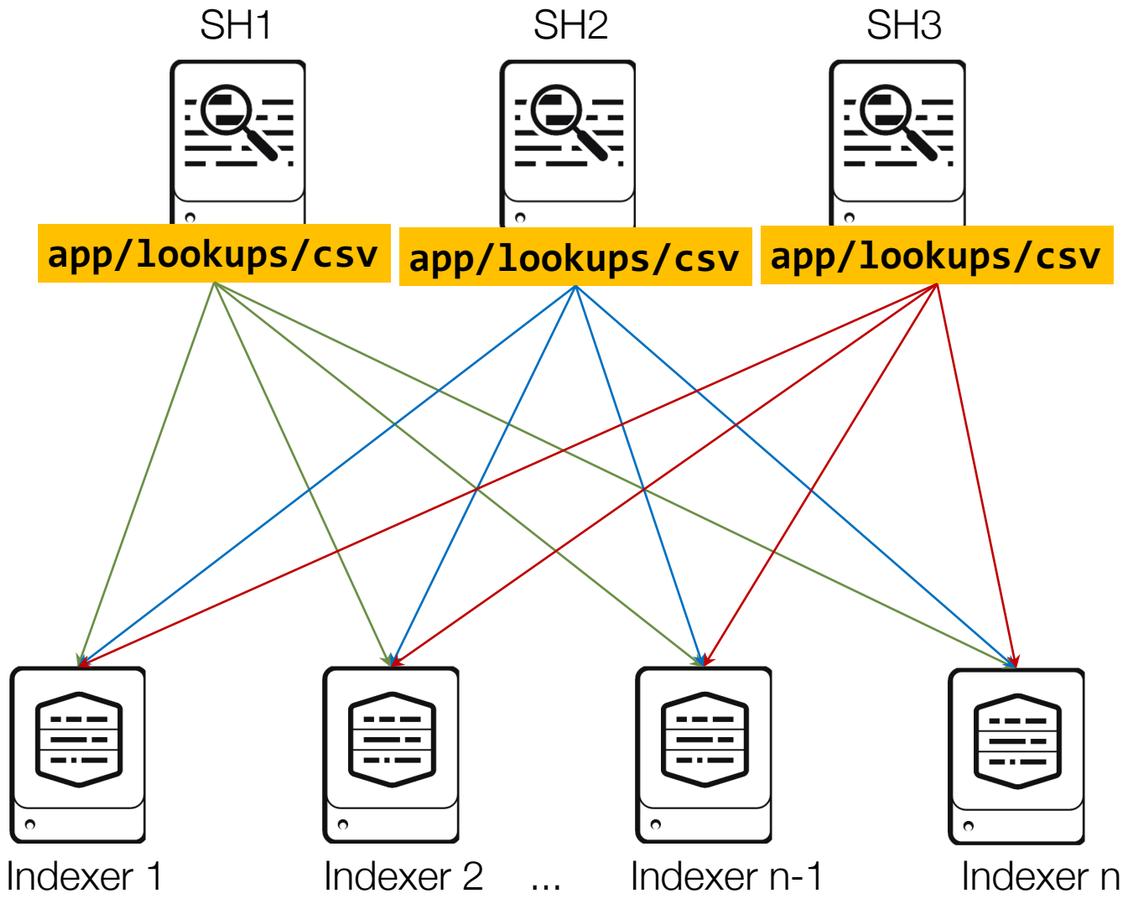
Module Objectives

- Enable a KV store collection under a search head cluster
- Manage KV store collections in a search head cluster
- Monitor and troubleshoot KV store issues in Splunk clusters
- Implement search history replication using KV store collections

Overview of Splunk Lookups

- A lookup is a Splunk data enrichment knowledge object
  - Splunk lookups are supported during search time and the parsing phase
  - Defined in **`props.conf`** and **`transforms.conf`**
- **File-based lookup** is used for datasets that are small and/or change infrequently
  - Uses CSV files stored in the **`lookups`** directory
  - Gets replicated to search peers
- **KV store lookup** is designed for large key-value collections that frequently change
  - Need to define **`collections.conf`** additionally
    - Can optionally configure data type enforcement and field accelerations
  - Collections live only on the search heads by default

CSV Lookup Challenges in Distributed Environment



- Each search head manages and replicates its own CSV lookup files
  - Updates to lookup files propagate to search peers independently
- Searches can fail if a bundle is not replicated to search peers in time
  - By default, a bundle larger than 2GB is not replicated to the search peers
- In Splunk clusters, the timely replication of lookup files is more critical
  - By default, SHC conf replication rejects a bundle size larger than 2GB

Benefits of KV Store Lookup

- Quick per-record updates
  - Performs Create-Read-Update-Delete (CRUD) operations on individual records using the Splunk REST API and SPL commands
- Standardized interface for app developers
  - REST API and data type validation
- Faster lookup replication from SHC to search peers
  - Enable KV store replication first per collection
    - Distributes a collection in CSV lookup files
    - Can perform automatic lookups on the index tier
- Can facilitate backup and restore of KV store data

Enabling KV Store Collections

- Before users can use a KV store, an admin must create a collection
- A collection is defined in **`collections.conf`**
  - Must be placed in an app's **`default`** or **`local`** directory
    - Other attempts are ignored
  - Specify the name of the collection (stanza) and optional attributes
    - Matching lookup field, output fields, etc.
  - Enforcing data types is optional
    - If enforced, any input that does not match the type is silently dropped
  - More options discussed later

`collections.conf`

```
[collection_name]
enforceTypes = [true|false]
field.<name1> = [number|string|bool|time]
field.<name2> = [number|string|bool|time]
accelerated_fields.<xl-name> = <json>
```

Example:

```
[mykv]
enforceTypes = true
field.x = number
field.y = string
accelerated_fields.xl2 = {"x": 1, "y":
```


Populating KV Store Lookups

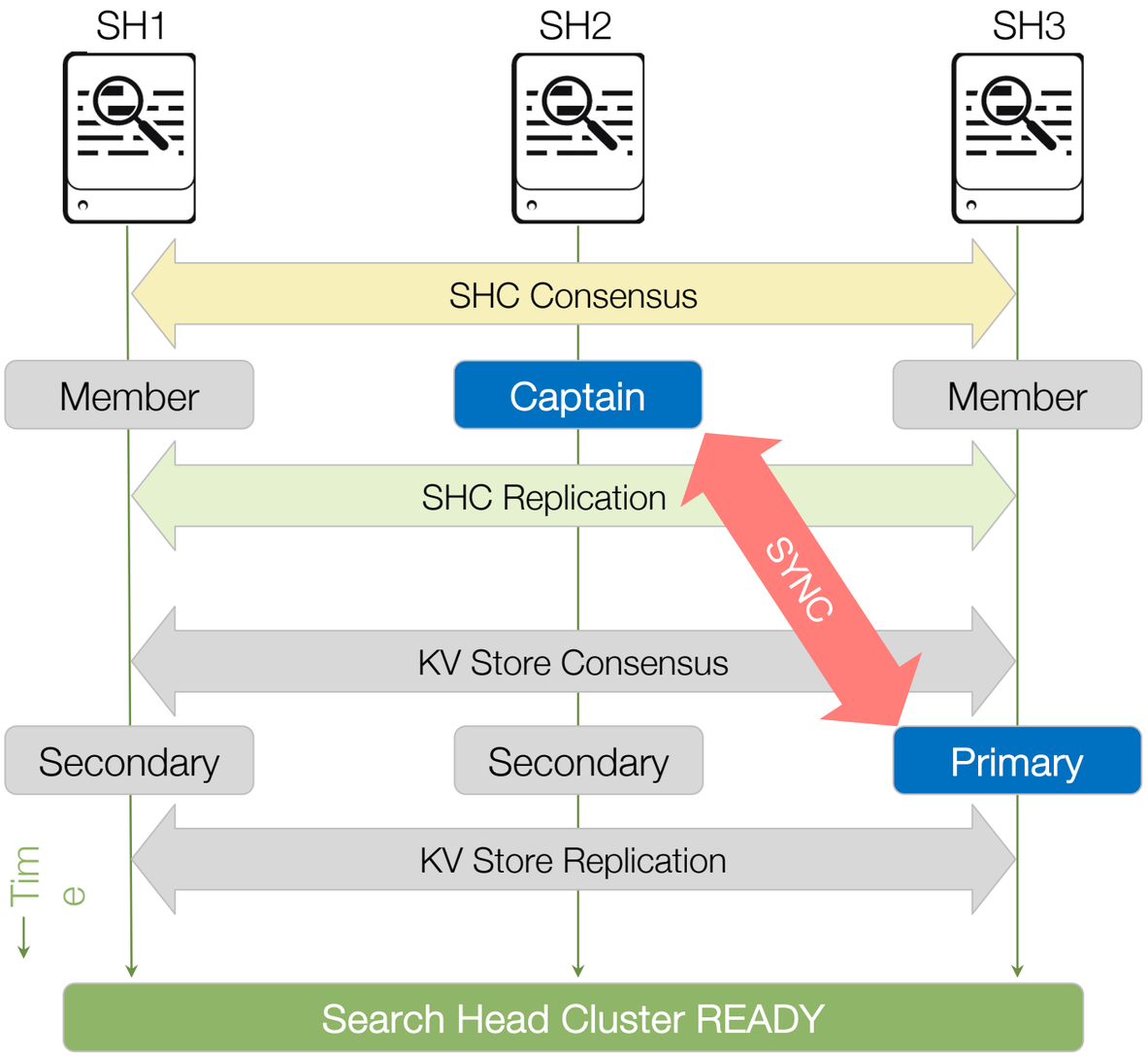
- Create a KV store collection stanza in **`collections.conf`**
  - Enable optional attributes, if needed
- Add lookup definition for KV store
  - Click **Settings** > **Lookups** > **Lookup definitions**
    - The resulting configuration is saved in **`transforms.conf`**
- Write data to the KV store
  - Search: **`... | fields id, location, type | outputlookup <lookup_name>`**
  - Or, use REST APIs

```
curl -k -u <user>:<pw>
https://<url:mport>/servicesNS/<user>/<app>/storage/collections/data/<lookup_name>
-H 'Content-Type: application/json'
-d '{"id": 001, "location": "CA", "type": "basic"}'
```

Note

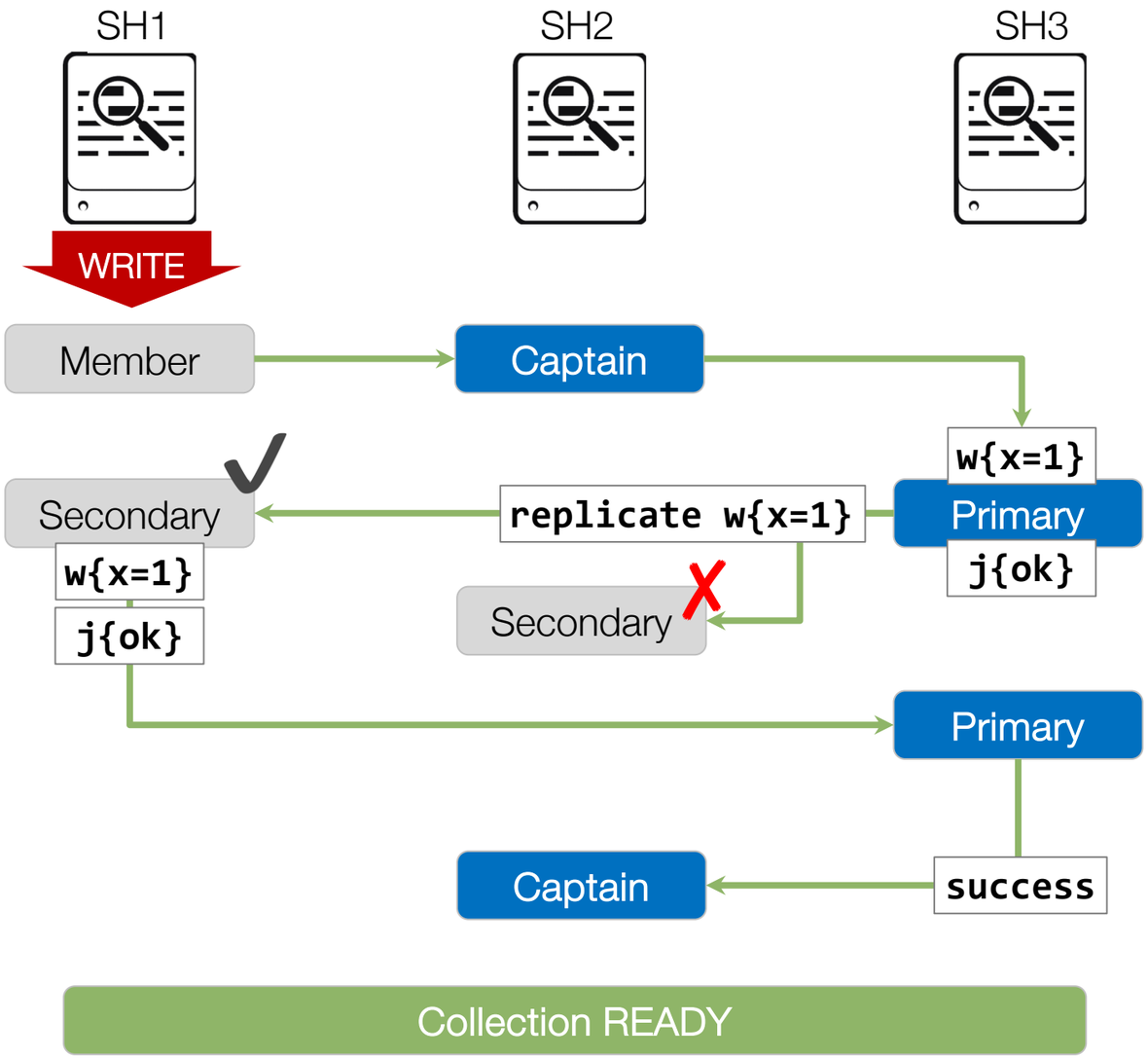
Working with KV store is discussed in detail in Building Splunk Apps course.

How KV Store Works in SH Cluster



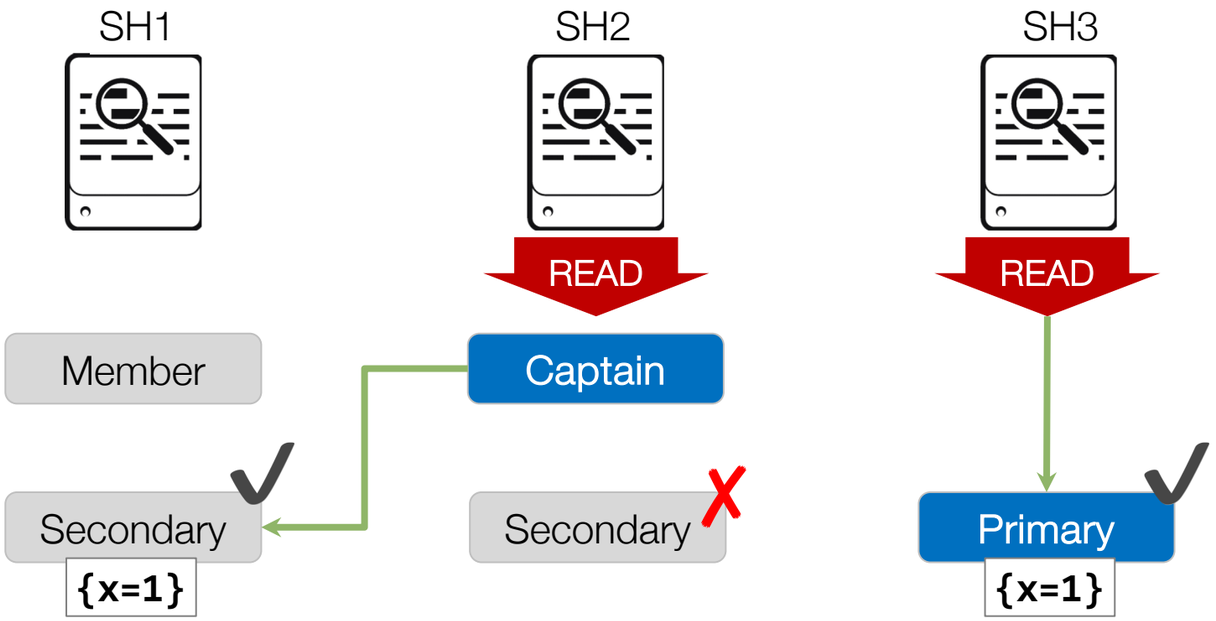
- Within a SHC, the KV store forms its own cluster
  - Can use up to 50 SHC members
- KV store port must be accessible from all SHC members
  - Uses 8191 by default
- SHC captain and KV store primary synchronize their member list every time there is a status change
  - Add, remove, or restart
- By default, the value of **`mgmt_uri`** in **`[shclustering]`** is used for KV store connection and replication

KV Store Collection Replication – Write



- The primary receives all write operations and records them in its journal
- The secondaries copy and apply these operations asynchronously
- Writes are acknowledged when:
  - The majority of voting KV store nodes have applied the operations
  - The writes have successfully been logged to their respective journals
- Checkpoints in journals provide data consistency
  - Provide recovery information

KV Store Collection Replication – Read



- READ operations can be done on any member
- Routed to the *nearest* member
  - Read from the lowest-latency member
  - Most reads are done with local instance preference

KV Store CLI Commands

- The similar conditions that cause SHC members to end up in an inconsistent replication state can cause the KV store collections to be out of sync
  - Frequent status changes to the SHC member instances
  - Changing the GUID or hostname of a member
  - Depending on the condition of SHC, KV store cluster can also be in the state where it is impossible to sync
- One workaround is to bootstrap SHC from scratch
  - WARNING: Proceed ONLY when it is absolutely necessary
- Useful commands
  - **`splunk show kvstore-status`**
  - **`splunk [backup | restore]`**
  - **`splunk clean kvstore`**
  - **`splunk resync kvstore -source <GUID>`**

splunk show kvstore-status

Example ran on SH2

splunk show shcluster-status

```
Captain:
        dynamic_captain : 1
        elected_captain : Fri Mar  9 00:32:52 2018
                     id : CE1F9057-981B-4419-BBA1-F10A5F3FBC6A
       initialized_flag : 1
                  label : sh2
               mgmt_uri : https://10.0.1.2:8289
  min_peers_joined_flag : 1
   rolling_restart_flag : 0
     service_ready_flag : 1

Members:
 sh2
                  label : sh2
               mgmt_uri : https://10.0.1.2:8289
          mgmt_uri_alias : https://10.0.1.2:8289
                 status : Up

 sh3
                  label : sh3
    last_conf_replication : Wed Mar 14 22:58:43 2018
               mgmt_uri : https://10.0.1.2:8389
          mgmt_uri_alias : https://10.0.1.2:8389
                 status : Up

 sh4
                  label : sh4
    last_conf_replication : Wed Mar 14 22:58:43 2018
               mgmt_uri : https://10.0.1.2:8489
          mgmt_uri_alias : https://10.0.1.2:8489
                 status : Up
```

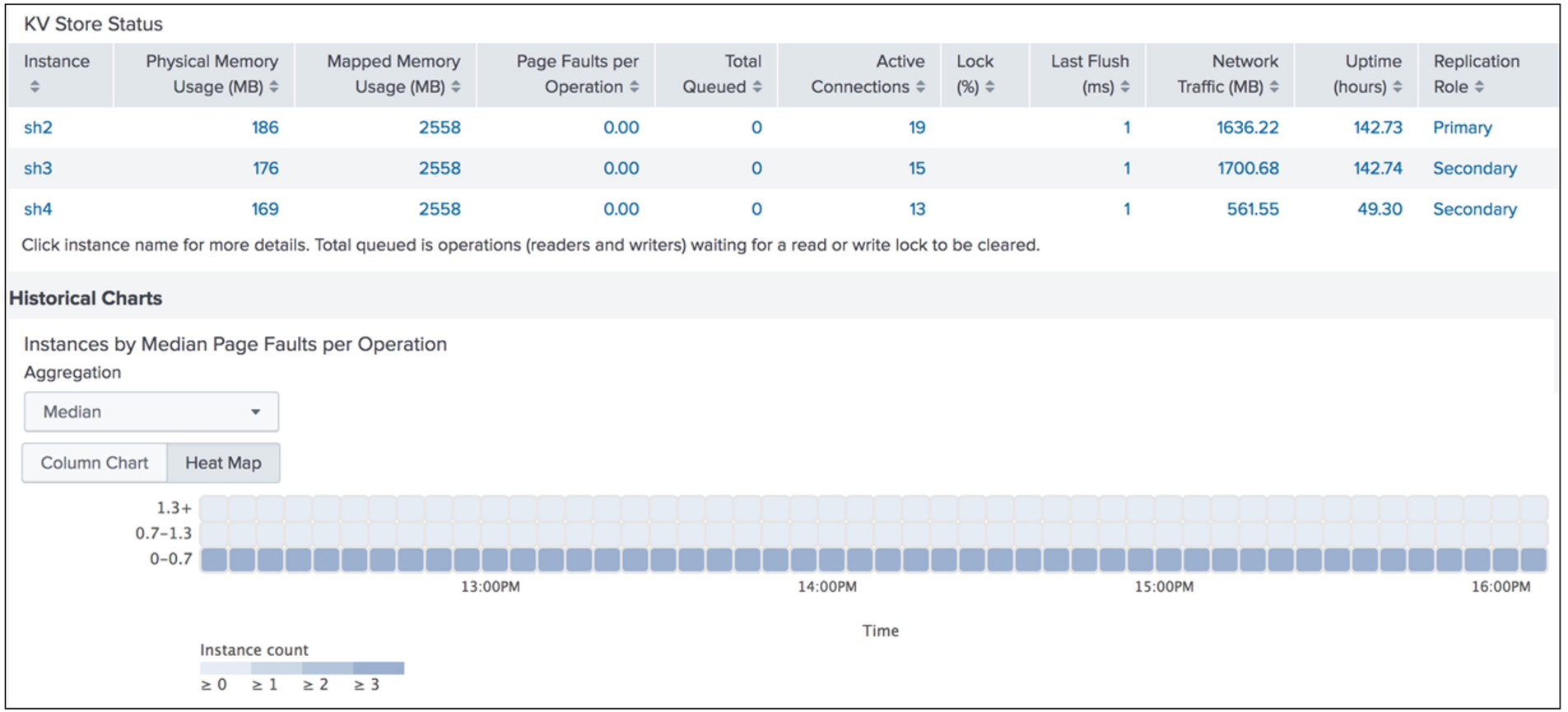
splunk show kvstore-status

```
This member:                    ...
            replicationStatus : Non-captain KV store member
                   standalone : 0
                       status : ready

Enabled KV store members:
10.0.1.2:8291                       guid : 13E6C3B3-293D-468B-A27D-66EC0D1358D8
                    hostAndPort : 10.0.1.2:8291
                                    ...
KV store members:
10.0.1.2:8391     configVersion : 3
                   electionDate : Mon Mar 12 21:43:19 2018
                 electionDateSec : 1520890999
                     hostAndPort : 10.0.1.2:8391
                   lastHeartbeat : Wed Mar 14 23:01:54 2018
                lastHeartbeatRecv : Wed Mar 14 23:01:55 2018
                                  ...
                     optimeDate : Wed Mar 14 23:01:54 2018
                   optimeDateSec : 1521068514
                           pingMs : 0
                replicationStatus : KV store captain
                          uptime : 176616
                                  ...
10.0.1.2:8291     configVersion : 3
                    hostAndPort : 10.0.1.2:8291
                     optimeDate : Wed Mar 14 23:01:54 2018
                   optimeDateSec : 1521068514
                replicationStatus : Non-captain KV store member
```


 | 7 August 2023

Monitoring KV Store Status with MC

- MC provides more details on KV store status
  - In the MC General Setup page, add the KV store server role to the instance
  - Go to **MC** > **Search** > **KV Store: Deployment**



Backing Up and Restoring KV Store

- To back up KV store data in a SHC:
  1. Confirm the KV store status with CLI or MC

     `splunk show kvstore-status`
  2. From a member with **`backupRestoreStatus`** and **`status`** in **`ready`** state, run:

     `splunk backup kvstore [-archiveName <archive>] [-collectionName <collection>]  [-appName <app>]`

     - Dumps an output to:
       `SPLUNK_HOME/var/lib/splunk/kvstorebackup/kvdump_<timestamp>.tar.gz`
- To restore when the majority SHC members are stale:
  1. Confirm the KV store status with CLI or MC

     `splunk show kvstore-status`
  2. Run the restore command:

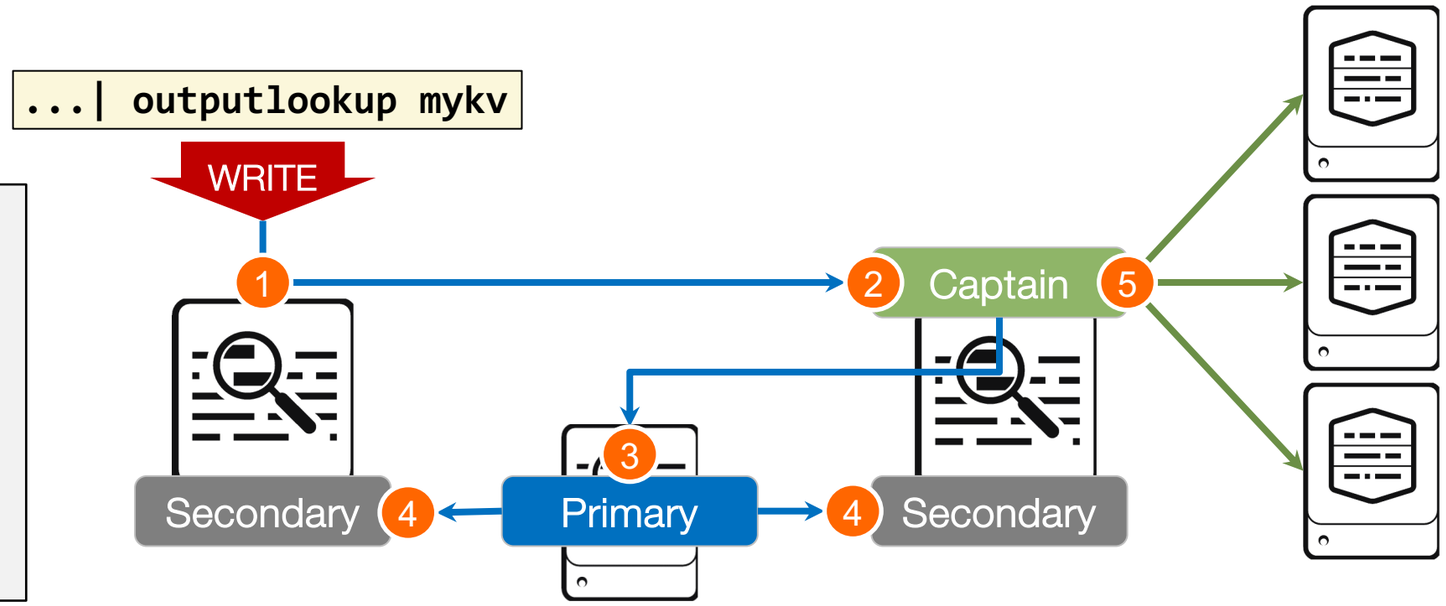
     `splunk restore kvstore -archiveName` `<archive>.tar.gz`

SHC KV Store Bundle Replication to Indexers

- To enable an automatic lookup with KV store data, you must enable replication in **`collections.conf`**
  - Each automatic lookup configuration is limited to a specific host, source, or source type

collections.conf on all members

```
[mykv]
enforceTypes = true
field.x = number
field.y = string
accelerated_fields.xl2 = {"x": 1, "y": 1}
replicate = true
replication_dump_strategy = one_file|auto
replication_dump_maximum_file_size = <KB>
```



Notable KV Store Log Channels

- **Metrics.log** has two subgroups under **group=kvstore** to track different activities
  - **name=dump OR name=sync**
  - Example: Show the sync operation duration over time

    ```
    index=_internal metrics group=kvstore | timechart max(msSyncTotal) by name
    ```
- **splunkd.log** contains various components for KV store status, start/stop, sync, and replication activities
  - Logs only **WARN/ERROR** events by default
  - Can increase the log verbosity in **server.conf**

    ```
    [kvstore]
    verbose = true
    ```
  - For list of KV store components, search:

    ```
    index=_internal sourcetype=splunkd component IN (kvstore*, collection*)
    ```
- **kvstore.log** contains the introspection data feeding the **_introspection** index

  ```
  index=_introspection sourcetype=kvstore
  ```

KV Store Operations Log Size

- The journal keeps record of all operations that modify the collections
- A larger journal can give a KV store cluster a greater tolerance for lag and even make the set more resilient
- The KV store allocates the full log size the first time Splunk is started, regardless of its utilization
  - The operations log is 1 GB by default
  - To adjust, edit **`[kvstore] oplogSize`** in **`server.conf`**
    - Must edit all nodes and run **`splunk clean kvstore -local`**
- *So, what is consuming all that journal space?*
  - A large **`| outputlookup`** operation can generate a lot of records
  - **`outputlookup append=true`** vs **`outputlookup append=false`**

Monitoring KV Store with Monitoring Console

- **`index=_introspection`** derives the historical KV store performance views in Monitoring Console
- Start with the deployment-wide view and drill down to an instance level
  - **Median Page Faults per Operation** shows the read activity involving disk I/O
  - Excessive hit (1.3+) indicates a need for more RAM
  - **Replication latency** shows the operation lags between the primary and secondary nodes during writes
    - Long lags (30+) can indicate replication (write) issues
    - You may need to increase the operation log size
  - Together with **High lock percentage** (50%+) and **Flushing rate** (50~100%), you can infer that there are heavy write operations or the system is sluggish
  - **Operations Log Window of KV Store Captain** shows the time between the first and last operations in the journal
- **Operations per Minute** in the **KV Store: Instance** dashboard shows the number of calls made to the KV store operations in detail

Replicating Search History

Search history may be replicated using KV store collections beginning w/ Splunk Enterprise 9.1

- Search history stored locally in CSV files by default; not synced

To implement, add the following to **`limits.conf`** on each SH:

```
[search]
enable_history = true
search_history_storage_mode = kvstore
```

Optionally specify the following:

```
max_history_length = <integer>                    #Default: 500
max_history_time_to_keep = <integer>[s|m|h|d] #Default: 90d
```


Further Reading: KV Store

- About the app key value store
- Configure KV Store lookups
- Back up and restore KV store

Lab Exercise 8 – Add a KV Store Collection

- **Time**: 30 minutes
- **Tasks**:
  - Identify the current SHC captain and KV store captain
  - Verify the state of the KV store service from Monitoring Console
  - Enable the KV store collection replication
  - Implement search history replication

Module 9:
Introduction to SmartStore

Module Objectives

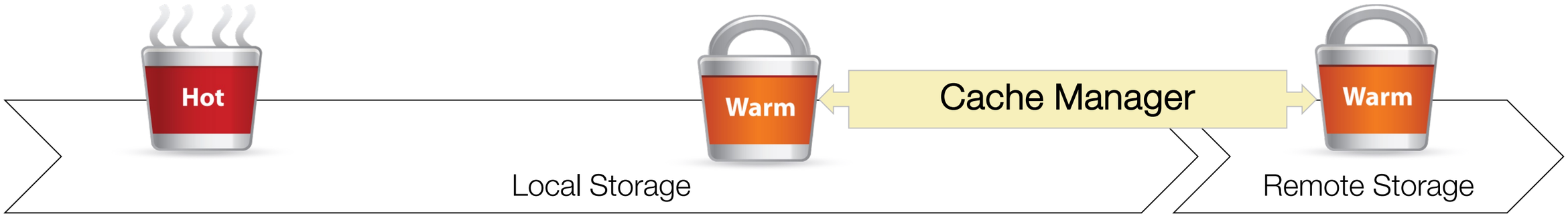
- List the use cases for deploying SmartStore
- Provide a SmartStore architecture overview
- Enable SmartStore in indexer cluster
- Monitor SmartStore status and identify potential issues

What is SmartStore and Why Use It?

- Scale-out distributed architecture is not a good fit for growing volumes
  - As Splunk deployment and adoption matures, demand for storage outpaces compute resources
  - Adding more peer nodes in response to data growth is expensive
- To scale beyond typical indexer clustering use cases, separating storage from computation resources has benefits
  - Lower cost of data retention
  - Higher reliability
  - Better scalability
- SmartStore is a Splunk indexer capability used to separate the compute resources from storage

SmartStore Architecture Overview

- S3-, Google Cloud- or Azure-compliant storage
- Cache manager
  - **Uploads** buckets to a remote storage when rolling from hot to warm
  - **Downloads** (fetch) buckets only if evicted and needed for search
  - **Evicts** buckets from the local cache based on the bucket's age, search frequency, and other tunable criteria



Considerations for Moving to SmartStore

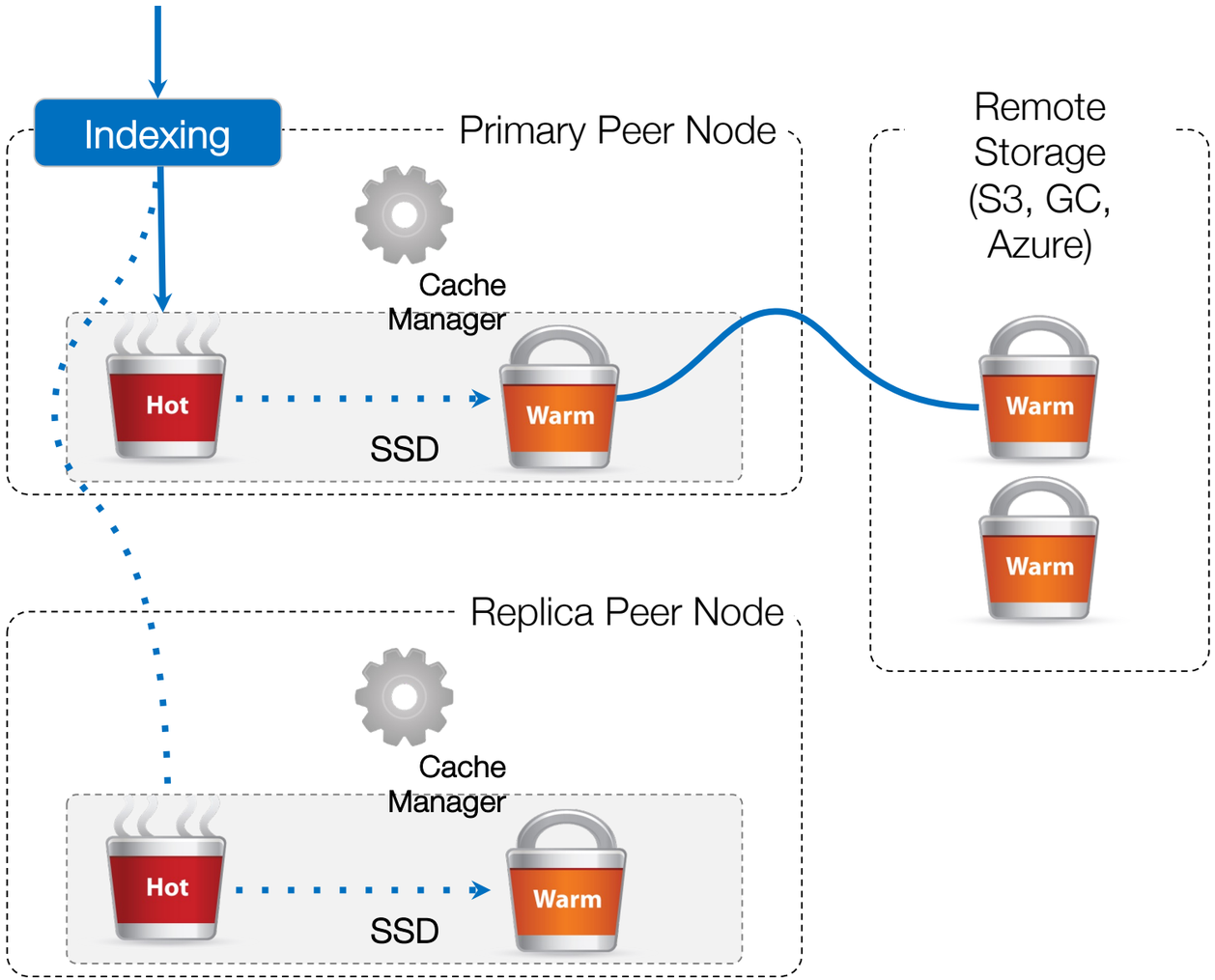
- Storage utilization is significantly outpacing other resources
- Considerable amount of admin time is devoted to managing the cluster because the manager node is always busy
  - Bucket fixups, data rebalancing, or software upgrade
- When most searches are over recent data
- NOT beneficial, if you run:
  - Frequent, rare searches with long lookbacks
  - If data is cached, no performance impact

Deploying SmartStore - Requirements

- Same as any Splunk indexer requirements
  - Except, the preferred local storage type is SSD
- Each on-prem machine must run the same Linux 3.x+ 64-bit OS
- For clustering, the peer node configurations must be identical
  - Can configure SmartStore globally or on a per-index basis
  - The local storage on each peer must be in proportion to the eviction policy
    - Recommend to reserve enough storage for a 30-day retention
  - Replication factor and search factor must be equal
    - Only hot buckets follow traditional replication policies
- **`homePath`** and **`coldPath`** must point to the same partition
- Resize **`maxDataSize=auto_high_volume`** to **`auto`**

SmartStore Index Time Workflow

1. Data is written to a hot bucket and gets replicated as normal
2. As the bucket rolls into warm (cached), it is uploaded to remote storage
   a. Replicated copies contain directory to support warm buckets, but no data
3. After the warm bucket is successfully uploaded to the remote storage, the local bucket is eligible for eviction
4. Buckets freeze directly from warm



Deploying SmartStore: S3 Example

1. Enable a manager for a new indexer cluster
2. Configure SmartStore settings in
   manager-apps/_cluster/local/indexes.conf
   - **remotePath:** Root path for remote
   - **storageType:** Set index to use SmartStore (i.e: remote volume)
   - **path**: Set remote storage location for index
   - **remote.*scheme*.*:** Settings specific to remote (auth, URL)
   - Specify an index (optional)
     - Index stanza requires **homePath**, **thawedPath**, & **coldPath**
     - **coldPath** is not used but required
3. Add peer nodes
   - Peers get the manager-apps bundles when they join
4. Check the deployment
   - When the cluster is complete, verify the connection from a peer node:
     splunk cmd splunkd rfs -- ls index:myidx

```
[default]
remotePath = volume:s3/$_index_name
repFactor = auto

[volume:s3]
storageType = remote
path = <scheme>://<remote-path>

# S3 example
path = s3://path/to/mysplunk/buckets/
remote.s3.access_key = <S3 access key>
remote.s3.secret_key = <S3 secret key>
remote.s3.endpoint = https://<S3 host>

[myidx]
homePath = $SPLUNK_DB/myidx/db
thawedPath = $SPLUNK_DB/myidx/thaweddb
coldPath = $SPLUNK_DB/myidx/colddb
maxGlobalDataSizeMB =
frozenTimePeriodInSecs =
```

indexes.conf

Deploying SmartStore - Migration

- Pre-existing single-site buckets must follow the multisite policy
  - Set **`constrain_singlesite_buckets=false`** in the manager's **server.conf**
- Review the current restrictions and reconfigure all nodes in the cluster to conform to the SmartStore requirements
- Test the SmartStore configurations and remote connectivity
- If the test is successful, apply the bundle from the manager node
  - The manager kicks off the migration but it happens entirely on the peer nodes
- SmartStore index conversion can take a long time
  - To monitor the migration process, search on the manager:

```
| rest /services/admin/cacheman/_metrics | fields splunk_server
migration.*
```

Deploying SmartStore - Bootstrap

- Bootstrapping fetches buckets present on a remote storage to new node(s)
  - Confirm all buckets are present before bringing down the old node(s)
- To bootstrap, use the same remote storage settings on the new node(s)
- The manager node coordinates a discovery process per index
  1. The manager requests a peer to retrieve a bucket list for a given index
  2. The peer reports back with the bucket list
  3. The manager distributes a set of primaries to fetch across all available peer nodes
  4. Each peer
     - Gets its assigned buckets from the remote storage
     - Recreates bucket's metadata locally and sets it as the primary
     - Replicates only primary's metadata to other peer nodes
  5. Other peers recreate buckets per metadata and report their buckets to the manager as replicated buckets

Index Retention Attributes with SmartStore

- SmartStore indexes do not use cold buckets; roll to frozen directly from warm
- Data retention is managed cluster-wide
  - **`maxGlobalDataSizeMB`**
    - Applies to all buckets on a per-index basis and across all peers in the cluster
    - Includes the total size on remote storage plus all warm buckets on all peer nodes to be uploaded to the remote storage
    - Counts only one copy of each bucket
  - **`frozenTimePeriodInSecs`** – behaves same as in non-SmartStore index rolling
  - The existing **`maxTotalDataSizeMB`** and **`maxWarmDBCount`** settings are ignored
- When a bucket ages out of index it is deleted from remote then local storage

SmartStore Index Freezing (Aging) Process

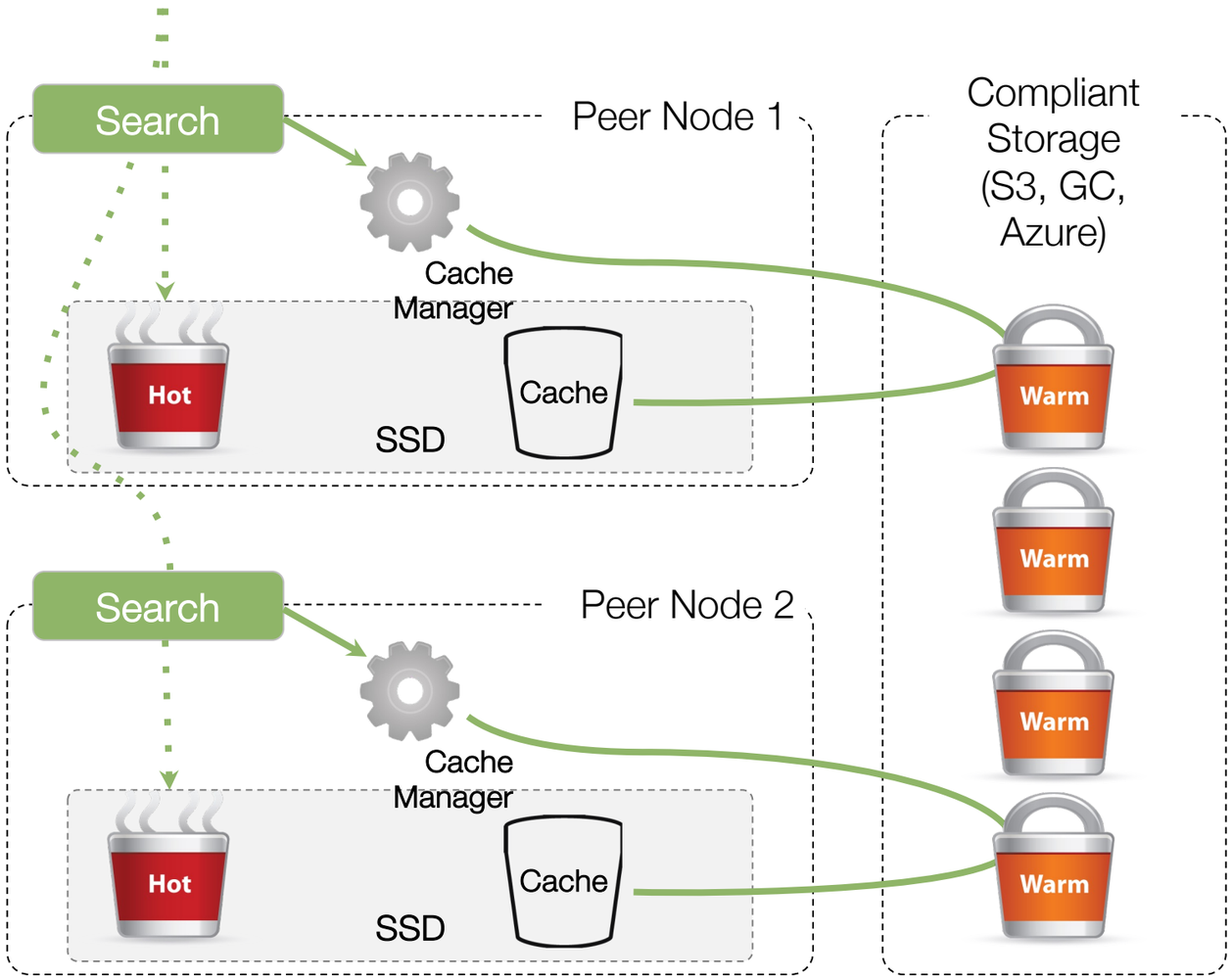
- The manager node assigns a freeze job to a peer node that has a local bucket copy
  - Identifies freeze candidates every 15 minutes by default
- If the index does not have **`coldToFrozenDir`** set, peer node removes remote bucket first and then local copy (no longer available to bootstrap)
  - Any buckets participating in a search are forestalled from deletion for up to 5 minutes, then force removed
  - Manager node is notified when bucket is deleted
- The manager node instructs other peer nodes to delete their local copies
- If **`coldToFrozenDir`** is set and bucket is evicted, only raw data is downloaded and copied, then deleted from remote first followed by local

SmartStore Search Considerations

- SmartStore is best-suited for the following:
  - Splunk searches that typically occur over near-term & recent data
  - Hot buckets which are always (and only) local; never evicted
- The searches always occur in local storage
  - Cannot search copies on remote store
  - The cache manager avoids evicting recently created and recently searched buckets
  - Cached buckets searched infrequently (often older data) will likely get evicted
- Only the primary hot buckets participate in a search
- Only the primary indexer (peer node) fetches the bucket if needed for a search (replicas are always empty)

Basic SmartStore Search-Time Workflow

1. SH dispatches a job to search peers
2. Peers spawn search processes and read a bucket list
3. Each search process searches only the "opened" buckets
   - Not all searches require the entire content
   - Fetches bucket files on as needed basis
   - A bucket is "opened" only when it is available locally
   - The cache manager blocks the process until a bucket is "opened"
   - Hot buckets are always "opened"
4. When the search is done, buckets are "closed" and they are eligible for eviction



More SmartStore Search-Time Workflow

- DMA & RA and scheduled search Summary Indexing
  - The Primary running the search uploads its summary buckets to remote store
  - When a search needs the summary, the cache manager opens the summarized bucket
- **`|delete`**
  - The primaries delete the events locally
  - Cache manager uploads/syncs the delete journals in the remote storage
- When a peer fails, bucket fixup to Complete state is notably faster
  - Reassign primary to surviving peer (no download from remote store unless/until needed for search)
  - Replicas are basically empty directories (not tsidx or raw)

Configuring SmartStore Cache Manager

Initiate eviction with the following settings in **server.conf**:

```
[diskUsage]
minFreeSpace = numInMB (default 5000)

[cachemanager]
eviction_padding = numInMB (default 5120 (5G))
max_cache_size = numInMB (default 0  = not set)
```

- Indexer will go into automatic detention if less than **minFreeSpace** on filesystem
- Add **eviction_padding** to **minFreeSpace** value to protect additional space
- Eviction will begin when indexed data uses/occupies **max_cache_size** threshold

When occupied space on cache partition exceeds **max_cache_size**, or the partition free space falls below (**minFreeSpace +eviction_padding**), cache manager begins to evict data.

Configuring SmartStore Eviction Policy

Apply globally in **server.conf** or per index in **indexes.conf** (priority):

```
[cachemanager]
…
hotlist_recency_secs = numInSeconds (default 8640 (24h))
hostlist_bloom_filter_recency_hours = numInHours (default 360 (15d))
```

Note

Do not change **eviction_policy** without consulting Splunk Support. Default value is **lru**, which prioritizes evicting “least recently used”.

- Avoid evicting buckets if most recent event within **hotlist_recency_secs**
- Eviction removes tsidx & rawdata, leaving behind metadata like bloomfilter, which will be evicted after aging beyond **hostlist_bloom_filter_recency_hours**

Notable SmartStore Log Channels

- splunkd.log
  - **CacheManager**, **CacheManagerHandler** – remote storage activities (INFO)
  - **S3Client**, **CacheManagerEviction**, **StorageInterface** (WARN)
- search.log on peer nodes
  - **CacheManagerHandler** – the cache manager bucket operations
  - **S2BucketCache** – search-time bucket management (bucket open/close)
- audit.log –bucket upload, download, eviction, and removal events
- metrics.log – metrics for remote storage operations

```
index=_internal metrics group IN(cachemgr*, spacemgr)
```

- splunkd_access.log – records the cache manager calls from searches

```
index=_internal sourcetype=splunkd_access
uri_path="/services/admin/cacheman*"
```

Monitoring Console – SmartStore

- Per instance and deployment-wide SmartStore activity information
  - Remote storage connectivity
  - Cache hits and misses
  - Bucket upload and download
  - Cache thrashing
  - Migration process
  - Bootstrapping progress

Further Reading: Splunk SmartStore

- About SmartStore
- How search works in SmartStore
- Troubleshoot SmartStore
- Configure the SmartStore cache manager
- Migrate existing data on an indexer cluster to SmartStore
- System requirements and other deployment considerations for indexer clusters
- Amazon_S3 (Wikipedia)
- https://github.com/splunk/s3-tests
- Configure SmartStore
- Deploy SmartStore on a new standalone indexer

Lab Exercise 9 – Migrate a Cluster to SmartStore

- **Time**: 30 minutes
- **Tasks**:
  - Verify the SmartStore connectivity from a test instance
  - Test the SmartStore configuration on a test instance
  - Upgrade the indexer cluster to meet the SmartStore prerequisites
  - Configure and deploy the SmartStore bundle to peer nodes
  - Validate the SmartStore-enabled cluster

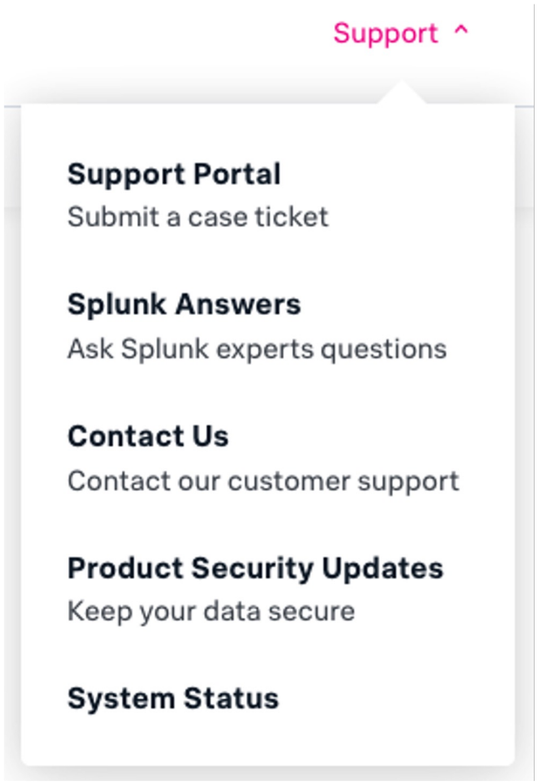
Wrap-up Slides

Community

- Splunk Community Portal
  community.splunk.com
  - Answers
  - Discussions
  - Splunk Trust
  - User Groups
  - Ideas
- Splunk Blogs
  splunk.com/blog/
- Splunk Apps
  splunkbase.com
- Splunk Dev Google Group
  groups.google.com/forum/#!forum/splunkdev
- Splunk Docs on Twitter
  twitter.com/splunkdocs
- Splunk Dev on Twitter
  twitter.com/splunkdev
- Splunk Live!
  splunklive.splunk.com
- .conf
  conf.splunk.com

Support Programs

- **Web**
  - Documentation: dev.splunk.com and docs.splunk.com
  - Wiki: wiki.splunk.com
- **Splunk Lantern**
  Guidance from Splunk experts
  - lantern.splunk.com
- **Global Support**
  Support for critical issues, a dedicated resource to manage your account – 24 x 7 x 365
  - Web: splunk.com/index.php/submit_issue
- **Enterprise, Cloud, ITSI, Security Support**
  - Web: splunk.com/en_us/about-splunk/contact-us.html#tabs/customersupport
  - Phone: (855) SPLUNK-S or (855) 775-8657



Learning Paths

Search Expert - Recommended Courses

Free eLearning courses are in blue and courses with an * are present in both learning paths.

- What is Splunk *
- Introduction to Splunk *
- Using Fields *
- Scheduling Reports and Alerts
- Visualizations
- Statistical Processing
- Working with Time
- Comparing Values
- Result Modification
- Leveraging Lookups and Subsearches
- Correlation Analysis
- Search Under the Hood
- Multivalue Fields
- Search Optimization *

Learning Paths (cont.)

Knowledge Manager - Recommended Courses

Free eLearning courses are in blue and courses with an * are present in both learning paths.

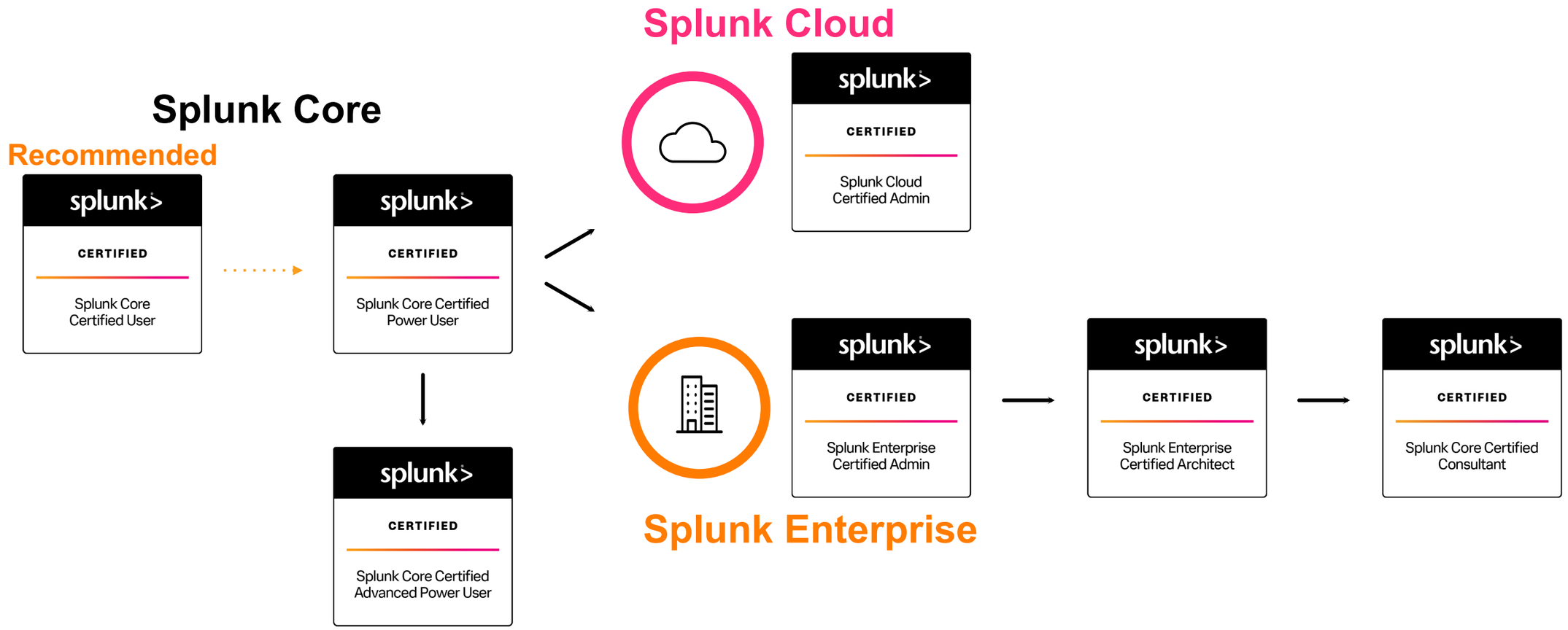
- What is Splunk *
- Introduction to Splunk *
- Using Fields *
- Introduction to Knowledge Objects
- Creating Knowledge Objects
- Creating Field Extractions
- Enriching Data with Lookups
- Data Models
- Introduction to Dashboards
- Dynamic Dashboards
- Using Choropleth
- Search Optimization *

Splunk Certification

Offerings & Requirements

Splunk Core and Beyond

Regardless of which Splunk product you use, it all starts with Splunk Core



App-Specific Offerings

For Splunk Add-Ons

App Developer

ES Administration

ITSI Administration

SOAR Automation Developer

Splunk Core Certified User

This entry-level certification demonstrates an individual's basic ability to navigate and use Splunk software

Prerequisite Certification(s):

- None

Prerequisite Course(s):

- None

Splunk Core Certified User Exam

Time to study! We suggest candidates looking to prepare for this exam complete Fundamentals 1 **or** the following courses:

- What is Splunk?
- Intro to Splunk
- Using Fields
- Scheduling Reports and Alerts
- Visualizations
- Statistical Processing
- Working with Time
- Leveraging Lookups and Subsearches
- Search Optimization
- Enriching Data with Lookups
- Data Models

See here for registration assistance.

Congratulations! You are a...

Recommended Next Step

- Splunk Core Certified Power User

Splunk Core Certified Power User

This entry-level certification demonstrates an individual's foundational competence of Splunk's core software

Prerequisite Certification(s):

- None

Prerequisite Course(s):

- None

Splunk Core Certified Power User Exam

Time to study! We suggest candidates looking to prepare for this exam complete Fundamentals 2 **or** the following courses:

- Visualizations
- Statistical Processing
- Working with Time
- Comparing Values
- Result Modification
- Correlation Analysis
- Search Under the Hood
- Introduction to Knowledge Objects
- Creating Knowledge Objects
- Creating Field Extractions
- Data Models
- Using Choropleth

See here for registration assistance.

Congratulations! You are a...

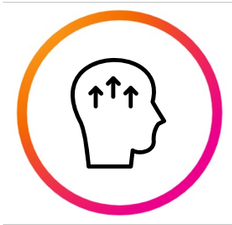
Recommended Next Steps

- Splunk Core Certified Advanced Power User
- Splunk Enterprise Certified Admin
- Splunk Cloud Certified Admin

 | 7 August 2023

Splunk Core Certified Advanced Power User

This certification demonstrates an individual's ability to generate complex searches, reports, and dashboards with Splunk's core software to get the most out of their data



Prerequisite Certification(s):

- Splunk Core Certified Power User

Prerequisite Course(s):

- None

Splunk Core Certified Advanced Power User Exam

Time to study! We suggest candidates looking to prepare for this exam complete Fundamentals 3, Creating Dashboards, and Advanced Searching & Reporting **or** the following courses:

- Using Fields
- Working with Time
- Comparing Values
- Result Modification
- Leveraging Lookups and Subsearches
- Correlation Analysis
- Search Under the Hood
- Multivalue Fields
- Search Optimization
- Creating Field Extractions
- Enriching Data with Lookups
- Data Models
- Using Choropleth
- Introduction to Dashboards
- Dynamic Dashboards

See here for registration assistance.

Congratulations! You are a...

Recommended Next Steps

- Splunk Enterprise Certified Admin
- Splunk Cloud Certified Admin

Splunk Cloud Certified Admin

This certification demonstrates an individual's ability to support the day-to-day administration and health of a Splunk Cloud environment

Prerequisite Certification(s):

- Splunk Core Certified Power User

Prerequisite Course(s):

- None

Splunk Cloud Certified Admin Exam

Time to study! We suggest candidates looking to prepare for this exam complete **either** the Splunk Cloud Administration **or** the Transitioning to Splunk Cloud course.

Both courses will equally prepare candidates for the exam, but are tailored to meet the needs of the individual based on prior Splunk experience.

Splunk Cloud Administration is designed for net-new administrators working in a Splunk Cloud environment. **Transitioning to Splunk Cloud** is for experienced Enterprise administrators looking to maximize their success in migrating to a Cloud environment.

See here for registration assistance.

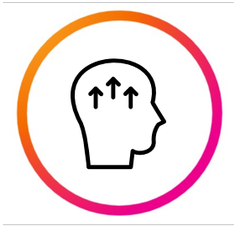
Congratulations! You are a...

Recommended Next Steps

- Splunk Certified Developer

Splunk Enterprise Certified Admin

This certification demonstrates an individual's ability to support the day-to-day administration and health of a Splunk Enterprise environment



Prerequisite Certification(s):

- Splunk Core Certified Power User

Prerequisite Course(s):

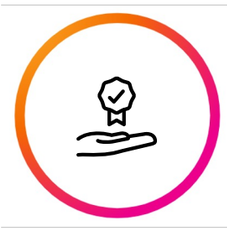
- None

Splunk Enterprise Certified Admin Exam

Time to study! We suggest candidates looking to prepare for this exam complete the following courses:

- Splunk System Administration
- Splunk Data Administration

See here for registration assistance.



Congratulations! You are a...

Recommended Next Steps

- Splunk Enterprise Certified Architect
- Splunk Certified Developer

Splunk Certified Architect

This certification demonstrates an individual's ability to deploy, manage, and troubleshoot complex Splunk Enterprise environments

Prerequisite Certification(s):

- Splunk Core Certified Power User
- Splunk Enterprise Certified Admin

Prerequisite Course(s):

- Architecting Splunk Enterprise Deployments
- Troubleshooting Splunk Enterprise
- Splunk Cluster Administration
- Splunk Deployment Practical Lab

Splunk Enterprise Certified Architect Exam

Time to study! We **require** candidates looking to register for this exam to complete the following prerequisite courses:

- Architecting Splunk Enterprise Deployments
- Troubleshooting Splunk Enterprise
- Splunk Cluster Administration
- Splunk Deployment Practical Lab

Candidates who are **Splunk Enterprise Certified Admin** and have completed all of the above courses will automatically receive an exam authorization for the Splunk Enterprise Certified Architect exam within 5-7 business days of receiving their passing lab results.

See here for registration assistance.

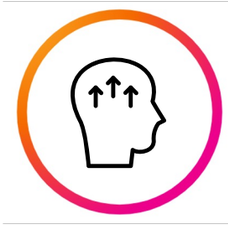
Congratulations! You are a...

Recommended Next Steps

- Splunk Core Certified Consultant

Splunk Core Certified Consultant

This certification demonstrates an individual's ability to properly size, install, and implement Splunk environments and to advise others on how to utilize the product and maximize its value for their needs



Prerequisite Certification(s):

- Splunk Core Certified Power User
- Splunk Enterprise Certified Admin
- Splunk Enterprise Certified Architect

Prerequisite Course(s):

- *Advanced Power User courses **or** digital badge**
- Core Consultant Labs
  - Indexer Cluster Implementation
  - Distributed Search Migration
  - Implementation Fundamentals
  - Architect Implementation 1-3
- Services Core Implementation

Splunk Core Certified Consultant Exam

Time to study! We **require** candidates looking to register for this exam to complete the following prerequisite courses:

- *Fundamentals 3*, *Creating Dashboards*, *Advanced Searching & Reporting**
- Core Consultant Labs
- Services Core Implementation

Candidates who are **Splunk Enterprise Certified Architects** and have completed all of the above courses must contact certification@splunk.com to request their Core Consultant exam authorization.

See here for registration assistance.

These Advanced Power User courses can be replaced with a Splunk Certified Advanced Power User badge **or completion of the following courses:*

- Using Fields
- Creating Field Extractions
- Enriching Data with Lookups
- Data Models
- Search Optimization
- Working with Time
- Leveraging Lookups and Subsearches
- Comparing Values
- Correlation Analysis
- Result Modification
- Multivalue Fields
- Search Under the Hood
- Introduction to Dashboards
- Dynamic Dashboards
- Using Choropleth

Congratulations! You are a...

Recommended Next Steps

- None

Splunk Enterprise Security Certified Admin

This certification demonstrates an individual's ability to install, configure, and manage a Splunk Enterprise Security deployment

Prerequisite Certification(s):

- None

Prerequisite Course(s):

- None

Splunk Enterprise Security Certified Admin Exam

Time to study! We suggest candidates looking to prepare for this exam complete the following course:

- Administering Splunk Enterprise Security

Please note: all candidates are expected to have working knowledge and experience as either Splunk Cloud or Splunk Enterprise Administrators.

See here for registration assistance.

Congratulations! You are a...

Recommended Next Steps

- Splunk Phantom Certified Admin

Splunk IT Service Intelligence Certified Admin

This certification demonstrates an individual's ability to deploy, manage, and utilize Splunk ITSI to monitor mission-critical services

Prerequisite Certification(s):

- None

Prerequisite Course(s):

- None

Splunk IT Service Intelligence Certified Admin Exam

Time to study! We suggest candidates looking to prepare for this exam complete the following course:

- Implementing Splunk IT Service Intelligence

Please note: all candidates are expected to have working knowledge and experience as either Splunk Cloud or Splunk Enterprise Administrators.

See here for registration assistance.

Congratulations! You are a...

Recommended Next Steps

- Courses on Observability

Splunk SOAR Certified Automation Developer

This certification demonstrates an individual's ability to install and configure a SOAR server, integrate it with Splunk, and plan, design, create, and debug playbooks

Prerequisite Certification(s):

- None

Prerequisite Course(s):

- None

Splunk SOAR Certified Automation Developer Exam

Time to study! We suggest candidates looking to prepare for this exam complete the following courses:

- Administering SOAR (Phantom)
- Developing SOAR (Phantom) Playbooks
- Advanced SOAR (Phantom) Implementation

Please note: all candidates are expected to have working knowledge and experience as either Splunk Cloud or Splunk Enterprise Administrators.

See here for registration assistance.

Congratulations! You are a...

Recommended Next Steps

- None

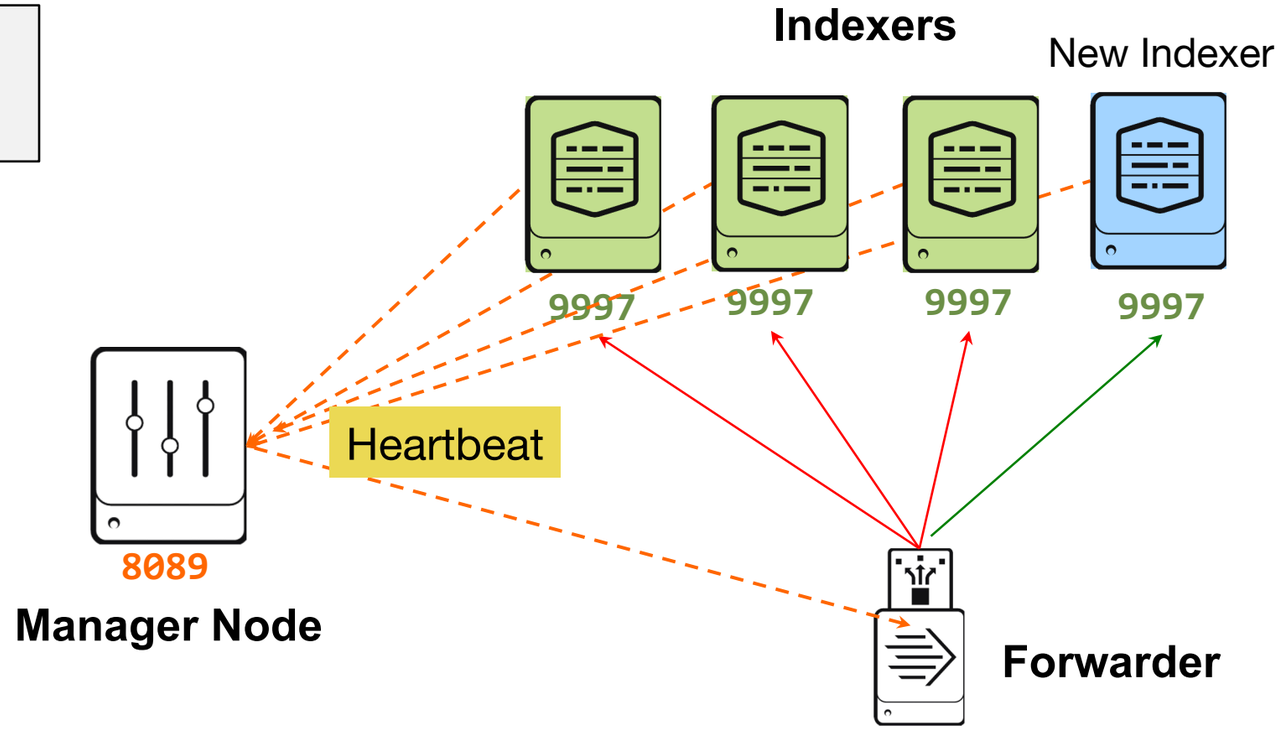
Thank You

Appendix

Using Weighted Load Balancing (Optional)

- Forwarders select next indexer based on relative disk capacities:
  - **`indexer_disk_capacity / total_disk_capacity_of_indexers_combined`**
- Selection is random but weighted towards indexers with larger disk capacity
- Enable in **`server.conf`** on Manager Node:

```
[indexer_discovery]
indexerWeightByDiskCapacity = true
…
```



Splunk Security 9.0 Improvements

New Security Features

Splunk Assist
Upgrade Readiness App
Native Smart Card Auth

No-action Needed Fixes

Secure by default.
Automatically implemented:
enforcement mode

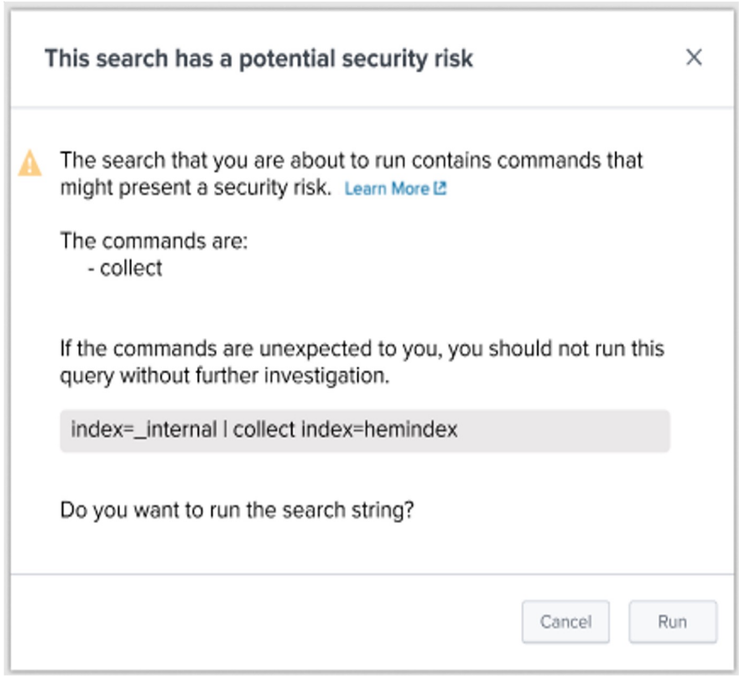
Action-required Advisories*

Require proactive steps to work. Mostly in **warning mode**

** For June 2022*

Splunk 9.0 Security – No Action Needed

- Improved dashboard security with sanitization on input fields
- Increased admin control with greater Splunk roles and capabilities with restriction options
- Updates to third party packages
  - Node.js
  - OpenSSL
- Easier risk management with user-friendly SPL safeguards
- Role-based filtering



Splunk 9.0 Security – Action Required

- Configure any action-required procedures:

  docs.splunk.com/Documentation/Splunk/latest/Security/Updates

Summary of changes

The following table lists a summary of the changes, the Splunk platforms on which the changes ship, the enforcement mode in which they currently operate, and links to procedures on how to configure Splunk software to enforce the changes.

Change	Description	Introduced in	Addresses	Current mode	Learn more
TLS certificate host name validation	Splunk platform instances verify the hostname in the TLS certificate they receive when they connect to other Splunk platform instances.	Splunk Cloud Platform (SCP) 8.2.2202, Splunk Enterprise (SE) 9.0	SVD-2022-0602, SVD-2022-0603	Warning	Learn more
Python module TLS connection hardening	Python modules on Splunk platform instances always validate TLS connections.	SCP 8.2.2202, SE 9.0	SVD-2022-0601	Warning	Learn more
TLS certificate host name validation using Splunk CLI	The Splunk Command Line Interface (CLI) validates TLS certificates for any connections it makes to other Splunk platform instances.	SCP 8.2.2203, SE 9.0	SVD-2022-0606	Warning	Learn more
Universal forwarder security	Universal forwarders always validate connections from	SE 9.0	SVD-2022-0605	Enforcement	Learn more

 |

Splunk Security Upgrade Resources

- Splunk Docs: Product Security page

  www.splunk.com/en_us/product-security.html

- Splunk Docs: Security Update Documentation

  docs.splunk.com/Documentation/Splunk/latest/Security/Updates

- Splunk Lantern: Upgrading Splunk Enterprise

  lantern.splunk.com/Splunk_Platform/Product_Tips/Enterprise/Upgrading_Splunk_Enterprise

Key Indexer Cluster Maintenance Commands

- Helpful CLI commands to run on the Manager Node

```
splunk help clustering
splunk rolling-restart cluster-peers
splunk [enable|disable|show] maintenance-mode
splunk set indexing-ready
splunk validate cluster-bundle
splunk show cluster-bundle-status
splunk apply cluster-bundle
```

Scaling Search Capacity on SH Cluster

- Load-based scheduling heuristic
  - Captain is the only job scheduler
    - Captain establishes authority and is also a member
    - The normal scheduler on all members is suppressed
    - Captain schedules and delegates jobs to its members
  - Captain maintains global knowledge of all search jobs
    - Members regularly report their job loads to the captain
    - Ad-hoc and real-time search results (artifacts) are not replicated
    - Saved and scheduled search artifacts are replicated per search head cluster replication factor
  - Captain directs which member to contact to access search results

Report Scheduler

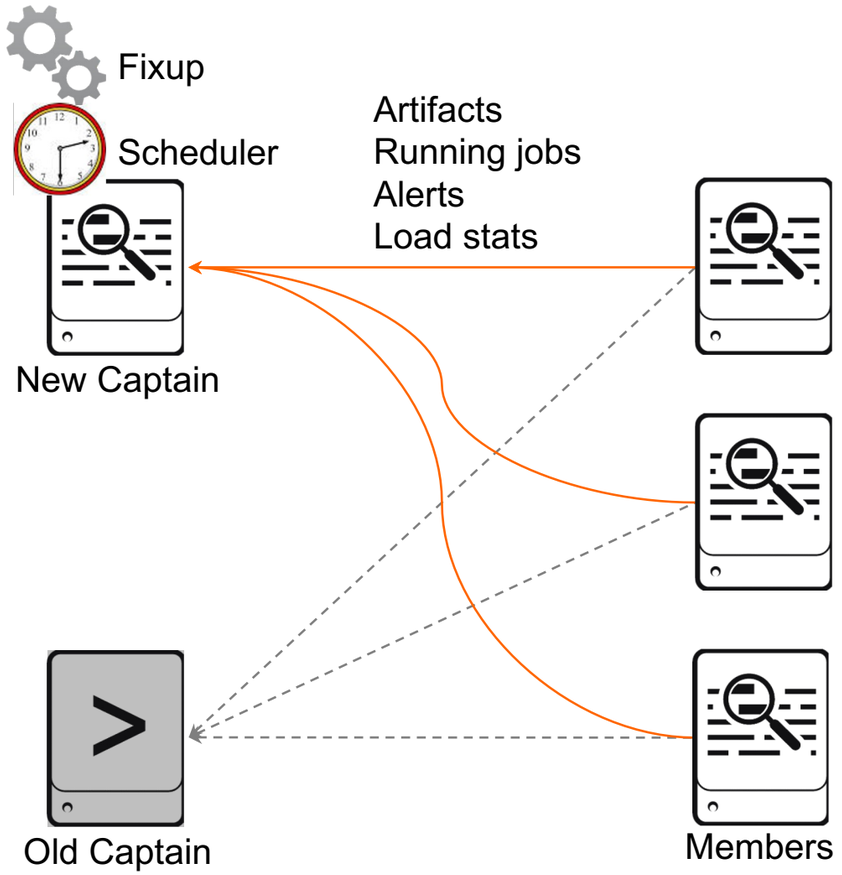
- Without SHC, a search head can defer and skip search jobs if:
  - Search head restarts (could cause a gap in summary index)
  - Search head encounters resource constraints (max. concurrent search limit)
    - A user performing prolific, ad-hoc searches can overwhelm other searches
    - Infrequent long-running searches can starve out frequent short-running searches
    - A deferred job is implicitly retried (repeated for the duration of its window)
- With SHC, the captain mitigates starvation and recovers missed jobs
  - Implements a heuristic approach based on job load, priority scoring, search history introspection, and schedule window

Search Head Cluster Log Channels

- Heartbeat and election
  - splunkd_access.log: **uri_path="/services/shcluster/member/consensus*"**
  - Corresponding events in **sourcetype=splunkd**
    - **component=SHCRaftConsensus OR component=SHClusterMgr**
    - **component=Metrics group=captainstability upgrades_to_captain=1**
- Job scheduling
  - scheduler.log: **sourcetype=scheduler component=SavedSplunker status=***
  - Corresponding events in **sourcetype=splunkd**
    - **component=SHCMaster delegate**
    - **component=Metrics group=search_concurrency**
- Artifact proxy and reaping
  - **sourcetype=splunkd_access uri_path="/services/search/jobs*"**
    - **status!=20*** indicates an issue
    - **method=GET isProxyRequest=true** indicates a proxied request
  - **sourcetype=splunkd_access uri_path="/services/shcluster/captain/artifacts*"**

How Does Cluster Provide Always-On Services?

- Auto SH captain failover
  - Elect new captain via Raft
  - Persists its records in **`var/run/splunk/_raft/<server>/log`**
  - Members register their list of artifacts, running jobs, alerts, and search load statistics to a new captain
  - New captain enables its scheduler
  - New captain executes fixups if needed



Note

Use DNS names when initializing members.