

# Search Head Clustering – Basics To Best Practices

Manu Jose, Yuan Xu

Sr. Software Engineer, Splunk

.conf2016

splunk >

# Disclaimer

During the course of this presentation, we may make forward looking statements regarding future events or the expected performance of the company. We caution you that such statements reflect our current expectations and estimates based on factors currently known to us and that actual events or results could differ materially. For important factors that may cause actual results to differ from those contained in our forward-looking statements, please review our filings with the SEC. The forward-looking statements made in the this presentation are being made as of the time and date of its live presentation. If reviewed after its live presentation, this presentation may not contain current or accurate information. We do not assume any obligation to update any forward looking statements we may make. In addition, any information about our roadmap outlines our general product direction and is subject to change at any time without notice. It is for informational purposes only and shall not, be incorporated into any contract or other commitment. Splunk undertakes no obligation either to develop the features or functionality described or to include any such feature or functionality in a future release.

# Agenda

- What Is Search Head Clustering?
- Cluster Operation
- Scalability And High Availability
- Configuration Management

# Search Head Clustering

Ability to group search heads into a cluster in order to provide Highly Available and Scalable search services



MISSION  
CRITICAL  
ENTERPRISE

# Business Benefits Of SHC

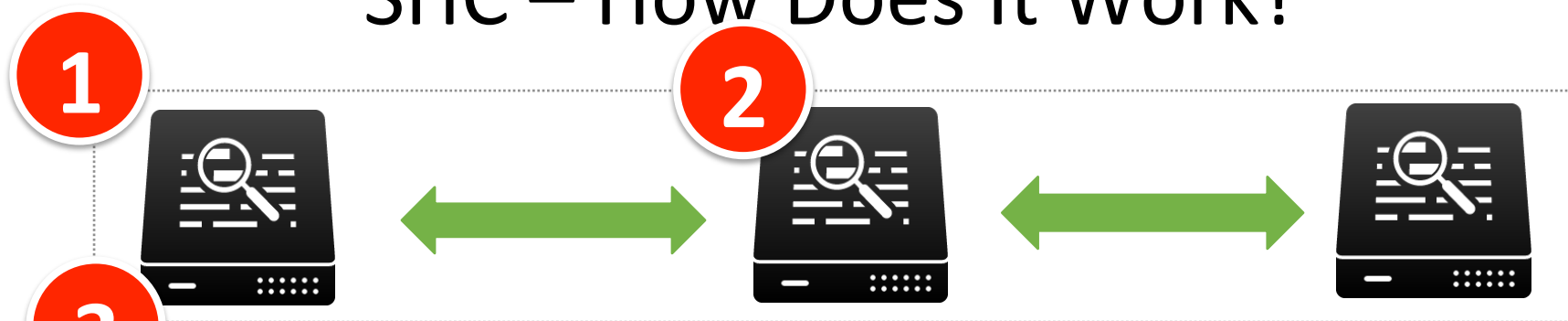
Horizontal Scaling

Consistent User Experience

Always-on Search Services

Easy to add / manage  
premium contents (apps)

# SHC – How Does It Work?



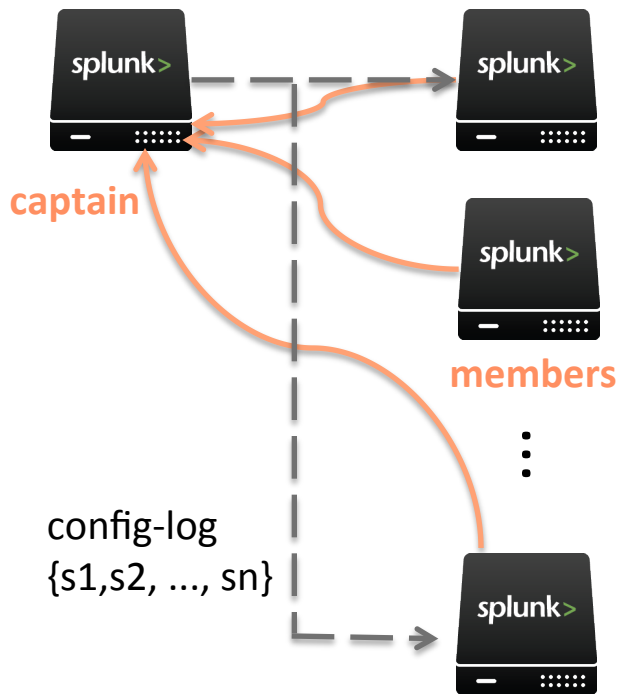
1. Group search heads into a cluster (Horizontal scaling)
2. Captain gets elected dynamically (No Single point failure)
3. User created reports/dashboards automatically replicated to other search heads (Consistent Configuration)

# Deploy A Cluster

.conf2016

splunk >

# Search Head Cluster Bring up



- Bootstrap captain
- Bring-up members
- Captain establishes authority
- Members join/register
- CLI based cluster scale/shrink



# Best Practices

- Add only fresh instances, if a node is re-purposed use “*Splunk clean all*”
- HA requires a minimum of 3 nodes
- All search heads on homogenous hardware and at same version
- Number of instances  $\geq$  replication\_factor
- Admin needs to manually do “*Splunk remove shcluster-member*” on captain to remove a dead node
- Multi-site clusters to have majority nodes at one site

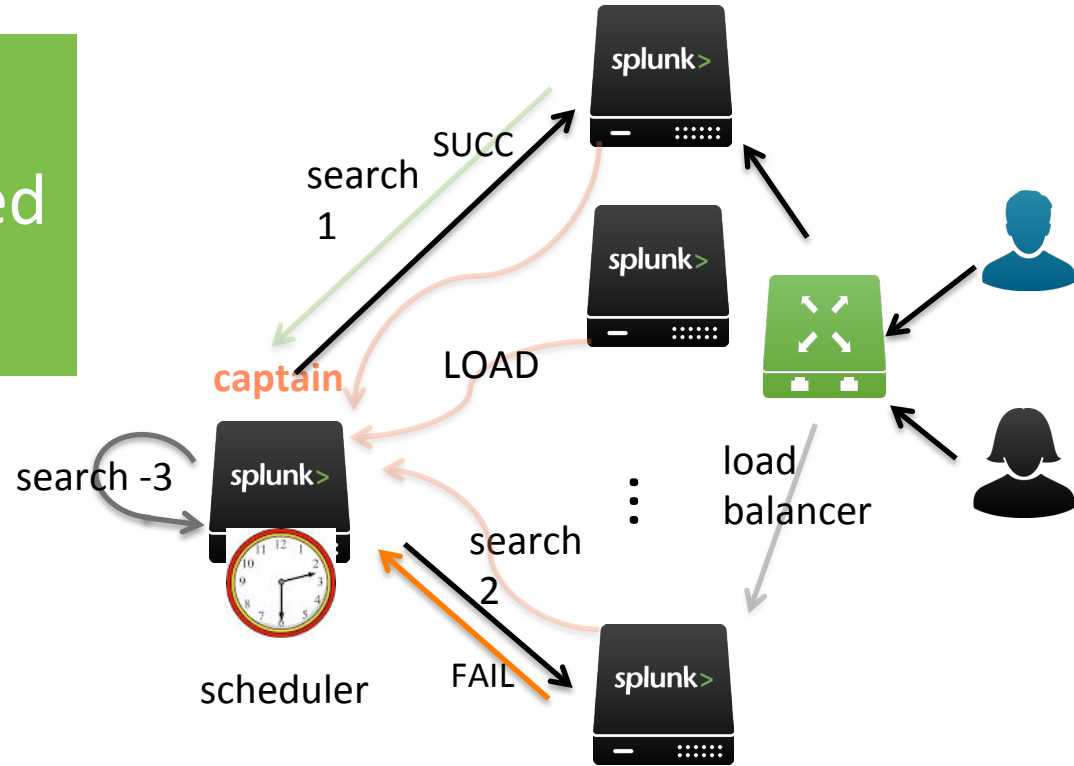
# Distributed Scheduling



.conf2016

# Job Scheduling Orchestration

- Captain is job scheduler
- Eliminates job-server need
- Load-based heuristic



# Details

- Auto-failover – New captain becomes scheduler
- `captain_is_adhoc_searchhead` knob to reduce captain load
- Captain updates RA/DM summaries on indexers.
- Scheduler limits honored across the cluster
- Real time scheduled searches run one instance across cluster
- Centralized user quota Management\*

# High Availability Of Search Results

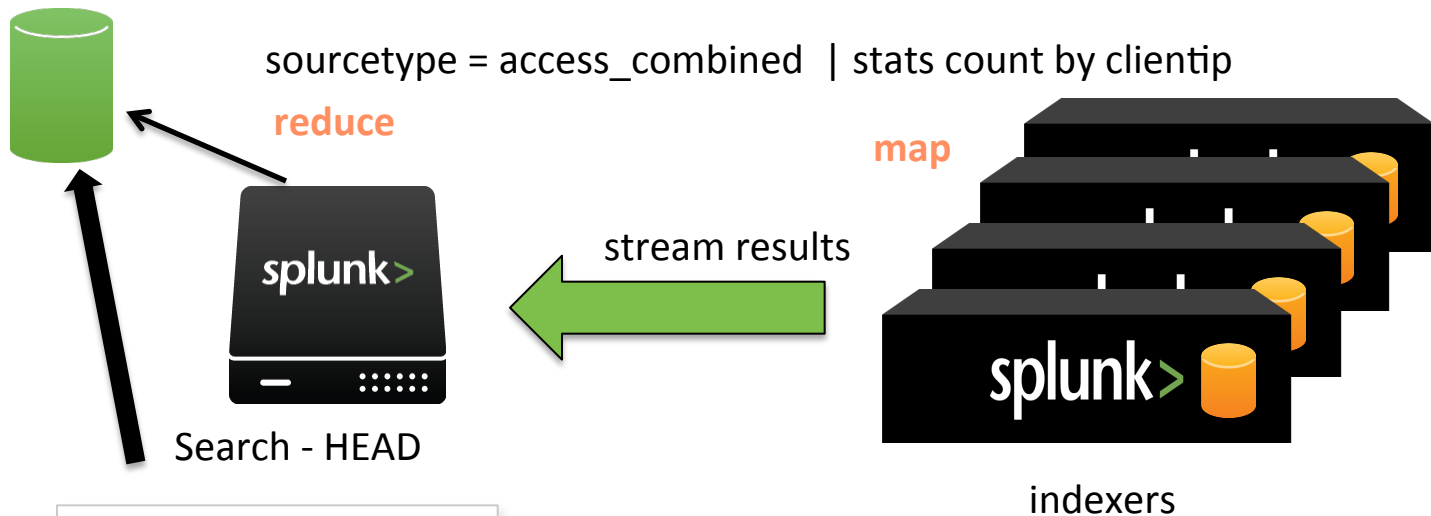


.conf2016

splunk >

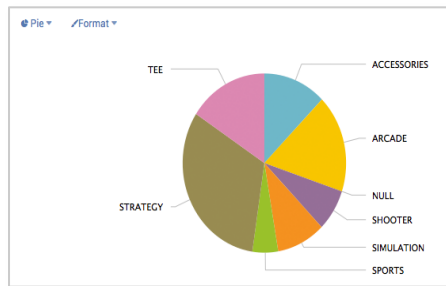
# Search Results Primer

```
$SPLUNK_HOME/var/run/  
splunk/dispatch/  
scheduler_admin_search_  
_mysearch_at_1410708600_  
345
```



other names

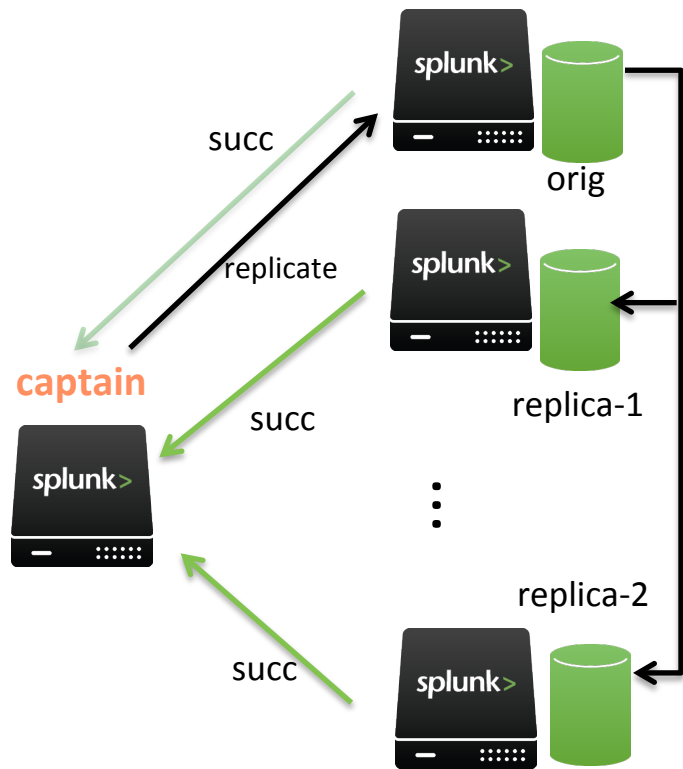
1. Search results
2. Search artifact
3. Dispatch directory
4. SID



**Dispatch dir needs to be replicated to multiple nodes to tolerate node failures**

# Artifact Replication

- Captain orchestrates replication
- Default replication\_factor=3
- Success failure ack'd to captain
- Asynch replicate on proxy
- Replication policy enforced by fixups



# Good To Know

- Adhoc searches are **not** replicated
- At least replication\_factor number of nodes should be in UP state for enforcing replication policy
- Replicated directory starts with “rsa\_<sid>” in the dispatch directory
- Captain orchestrates reaping of search artifacts from dispatch directory of all members
- An artifact is served based on availability from (1) itself, (2) search originating node, (3) captain



# Centralized Cluster State

- Captain maintains a global view of alerts and suppressions and updates the list to all members
- Captain registers all the adhoc searches run in the cluster
- Captain orchestrates reaping of search artifact replicas
- GET /services/search/jobs requests on any member will proxy to captain to get complete jobs

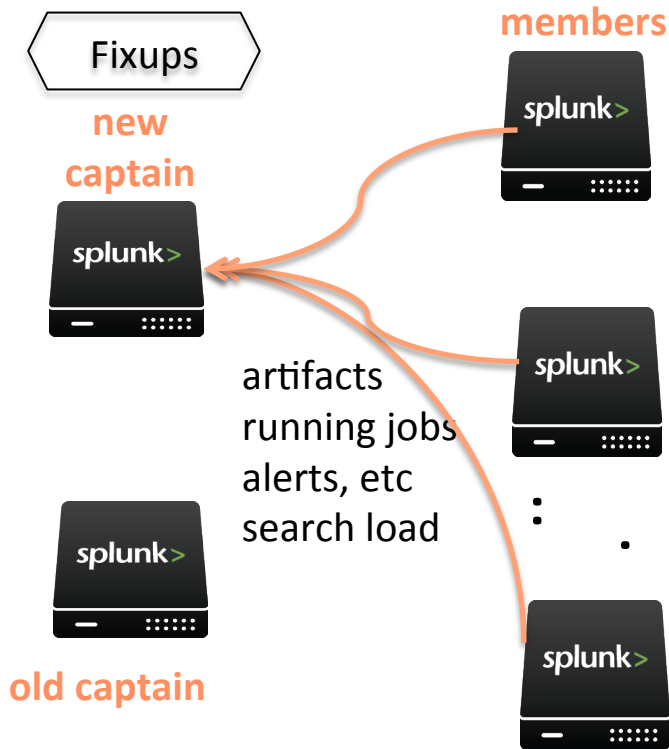
# High Availability Of Cluster

.conf2016

splunk >

# Dynamic Captain & Auto Failover

- Raft Consensus Protocol from Stanford
  - Diego Ongaro & John Ousterhout
- SHC uses RAFT for LE and Auto Failover



# Stable Captaincy

- Captain Switching should be extremely rare
- Repair a problem by transfer captain without restarts!!!
- Rolling-restart from the captain maintains the node as captain after restarts
- Captain preference added for members
- Disaster Recovery using static captaincy

# Configuration Management

.conf2016

splunk >

# Goal

- Consistent user experience across all search heads
- Changes made on one member are reflected on all members

# Configuration Changes

- Users customize **search and UI configurations** via Web/CLI/REST
  - save report
  - add panel to dashboards
  - create field extraction
- Administrators modify **system configurations**
  - configure forwarding
  - deploy centralized authentication (e.g. LDAP)
  - install entirely new apps

# Search And UI Configurations

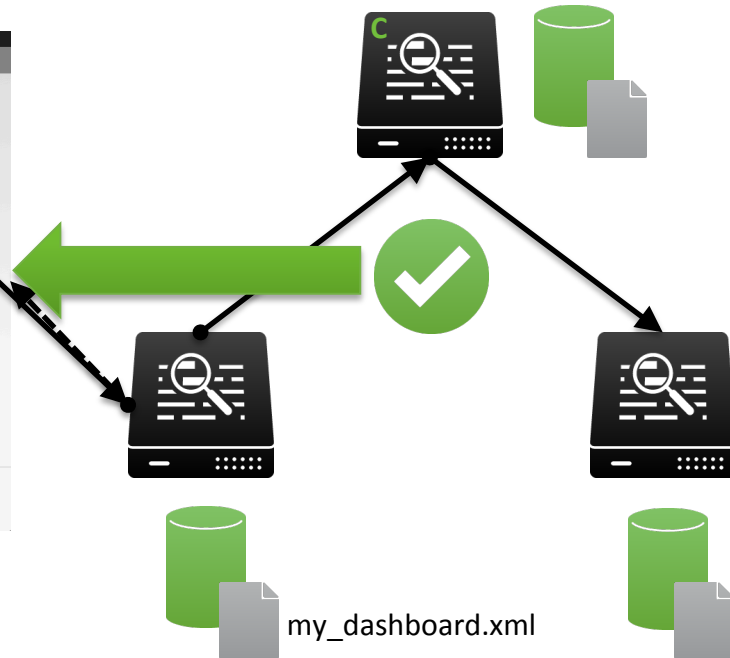
- Changes to search and UI configurations are replicated across the search head cluster automatically
- Goal: eventual consistency



# Conf Replication - Workflow

my dashboard

Time	Event
9/16/14 3:06:03.917 PM	09-16-2014 15:06:03.917 -0700 ERROR DistributedPeerManagerHeartbeat - Failed to get server info from peer http://localhost:3601, response code=404
9/16/14 3:06:03.907 PM	09-16-2014 15:06:03.907 -0700 WARN DistributedPeerManagerHeartbeat - Unable to get server info from peer http://localhost:3035 due to: Connection refused
9/16/14 3:05:03.863 PM	09-16-2014 15:05:03.863 -0700 ERROR DistributedPeerManagerHeartbeat - Failed to get server info from peer http://localhost:3601, response code=404
9/16/14 3:05:03.848 PM	09-16-2014 15:05:03.848 -0700 WARN DistributedPeerManagerHeartbeat - Unable to get server info from peer http://localhost:3035 due to: Connection refused
9/16/14 3:04:03.795 PM	09-16-2014 15:04:03.795 -0700 ERROR DistributedPeerManagerHeartbeat - Failed to get server info from peer http://localhost:3601, response code=404
9/16/14 3:04:03.787 PM	09-16-2014 15:04:03.787 -0700 WARN DistributedPeerManagerHeartbeat - Unable to get server info from peer http://localhost:3035 due to: Connection refused
9/16/14 3:03:03.735 PM	09-16-2014 15:03:03.735 -0700 ERROR DistributedPeerManagerHeartbeat - Failed to get server info from peer http://localhost:3601, response code=404
9/16/14 3:03:03.726 PM	09-16-2014 15:03:03.726 -0700 WARN DistributedPeerManagerHeartbeat - Unable to get server info from peer http://localhost:3035 due to: Connection refused
9/16/14 3:02:03.680 PM	09-16-2014 15:02:03.680 -0700 ERROR DistributedPeerManagerHeartbeat - Failed to get server info from peer http://localhost:3601, response code=404
9/16/14 3:02:03.671 PM	09-16-2014 15:02:03.671 -0700 WARN DistributedPeerManagerHeartbeat - Unable to get server info from peer http://localhost:3035 due to: Connection refused



# Conf Replication Progress

```
Captain:
    dynamic_captain : 1
    elected_captain : Tue Aug  9 10:27:23 2016
                  id : 61A55EA5-FDB6-496D-B8CC-E4205DDCE9DF
    initialized_flag : 1
                  label : yxu-mbp15-node3
                  mgmt_uri : https://localhost:9089
    min_peers_joined_flag : 1
    rolling_restart_flag : 0
    service_ready_flag : 1

Members:
yxu-mbp15-node1
    label : yxu-mbp15-node1
    last_conf_replication : Tue Aug  9 15:16:43 2016
    mgmt_uri : https://localhost:11089
    mgmt_uri_alias : https://yxu-mbp15:11089
    status : Up
yxu-mbp15-node3
    label : yxu-mbp15-node3
    mgmt_uri : https://localhost:9089
    mgmt_uri_alias : https://yxu-mbp15:9089
    status : Up
yxu-mbp15-node2
    label : yxu-mbp15-node2
    last_conf_replication : Tue Aug  9 15:16:43 2016
    mgmt_uri : https://localhost:8089
    mgmt_uri_alias : https://yxu-mbp15:8089
    status : Up
```

# Conf Replication - Health Check

## Search Head Clustering: Status and Configuration

Search Head Cluster:

SHC\_Yuan\_15

[Hide Filters](#)

### Health Check

**!** There are members in this cluster that do not share a common baseline. Action may be required. [click to see more details](#). [Learn More](#)

Select views: [All](#) [Snapshot](#) [Historical](#)

### Snapshots

Search Concurrency (Running/Limit)

Ad hoc + Scheduled (0 Running)

**0/207**

Historical

**0/207**

Real-time

Scheduled (0 Running)

**0/69**

Historical

**0/69**

Real-time

**0/33**

Summarization

[Click to see more details.](#)

Search concurrency limits can be set in limits.conf. [Learn More](#)

### Status

**3 Members**

Instance	Role	Status	Last Heartbeat Sent to Captain	Configuration Baseline Consistency	Number of Unpublished Changes	Artifact Count
<a href="#">fool02.sv.splunk.com</a>	Captain	Up	08/02/2016 20:01:57 -0700	3/3	0	2
<a href="#">fool01.sv.splunk.com</a>	Member	Up	08/02/2016 20:01:56 -0700	3/3	0	2
<a href="#">fool03.sv.splunk.com</a>	Member	Up	08/02/2016 20:01:57 -0700	1/3	missing common baseline with the captain: <a href="https://fool02.sv.splunk.com:1599">https://fool02.sv.splunk.com:1599</a>	1

[Click on instance name to see more details.](#)

[Click on configuration baseline ratio to see more details about configuration replication. Learn More](#)

### Configuration Baseline Consistency for: fool03.sv.splunk.com

Shares Common Baseline With	Does Not Share Common Baseline With	No Response From
<a href="#">fool03.sv.splunk.com</a>	<a href="#">fool01.sv.splunk.com</a> <a href="#">fool02.sv.splunk.com</a>	

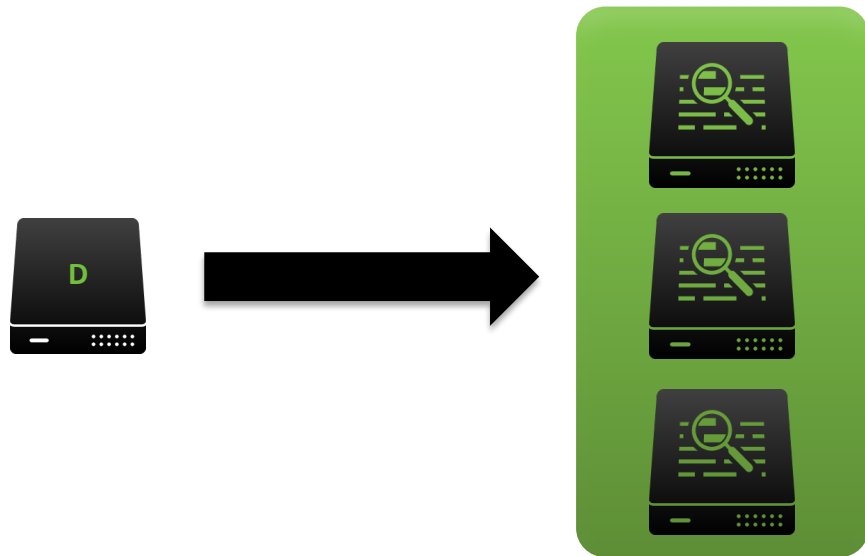
[Close this panel](#)

# System Configurations

- Recall: only changes to search and UI configurations are replicated across the search head cluster automatically
- Changes to **system configurations** are **not** replicated automatically because of their high potential impact
- How are system configurations kept consistent, then?

# Configuration Deployment

- Deployer: a single, well-controlled instance outside of the cluster
- Configurations should be tested on dev/QA instances prior to deploy



# THANK YOU

.conf2016

