.conf18

splunk>

So, You’ve Inherited a Splunk Deployment

Reducing Technical Debt With a (mostly) Seamless User Experience

Ian Thiele | First Data
Jon LeBaugh | Splunk

October 2018

About Us

Ian Thiele

- Systems Engineer at First Data
- 5+ years of experience with Splunk. Three as a user, two as an admin.

Jon LeBaugh

- Sr. ITOA Architect at Splunk.
- Former technical debt contributor.
- 3 years at Splunk (today), using Splunk for 6+

"The concept of technical debt is central to understanding the forces that weigh upon systems, for it often explains where, how, and why a system is stressed. In cities, repairs on infrastructure are often delayed and incremental changes are made rather than bold ones. So it is again in software-intensive systems…"[1]
— *Grady Booch, 2014*

Suryanarayana, Girish (November 2014). Refactoring for Software Design Smells(1st ed.). Morgan Kaufmann. p. 258. ISBN 978-0128013977. Retrieved 19 November 2014.

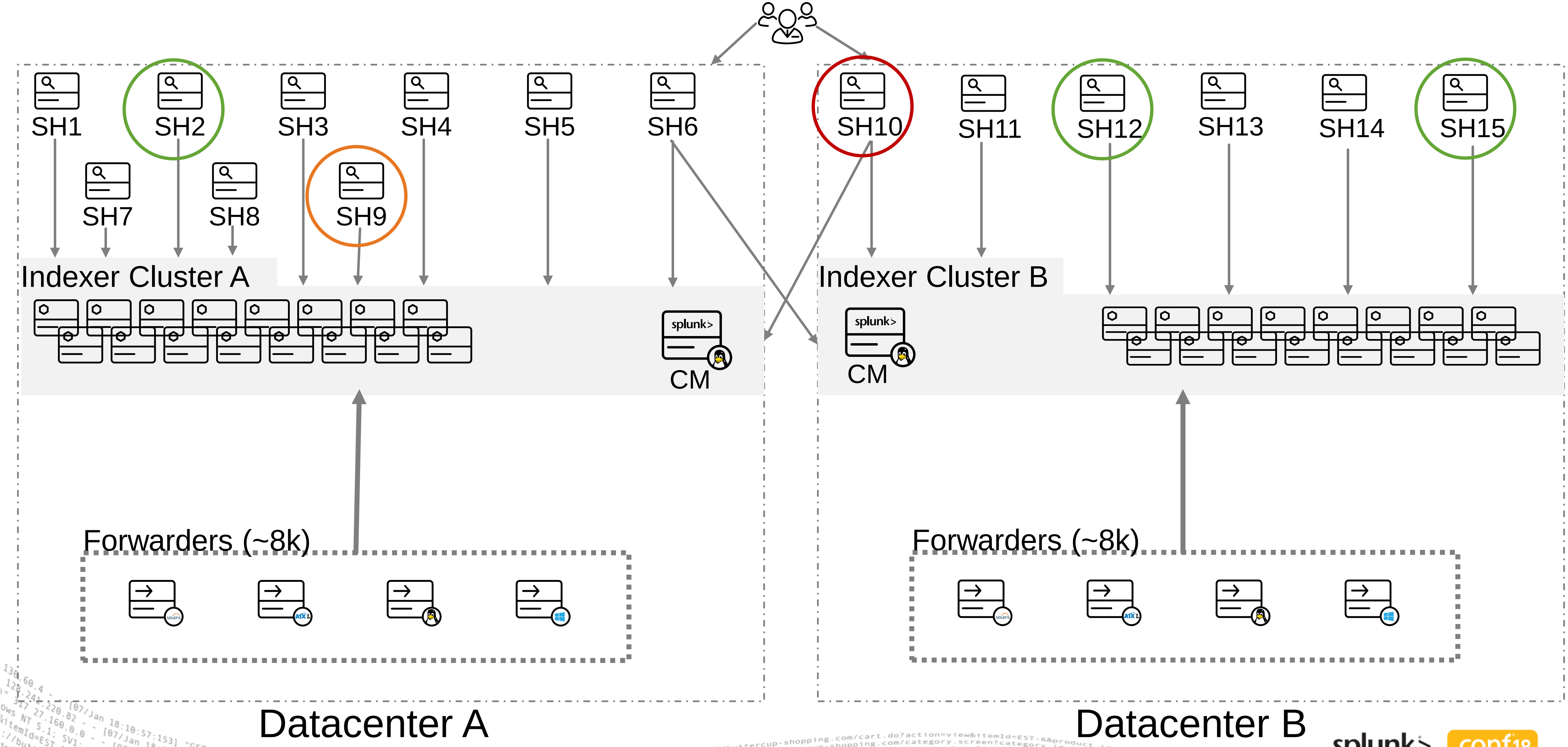
What Forms of Technical Debt Do You Suffer From?

- Hardware
- Version Drift
- Knowledge Object Divergence
- Haphazard Data Onboarding

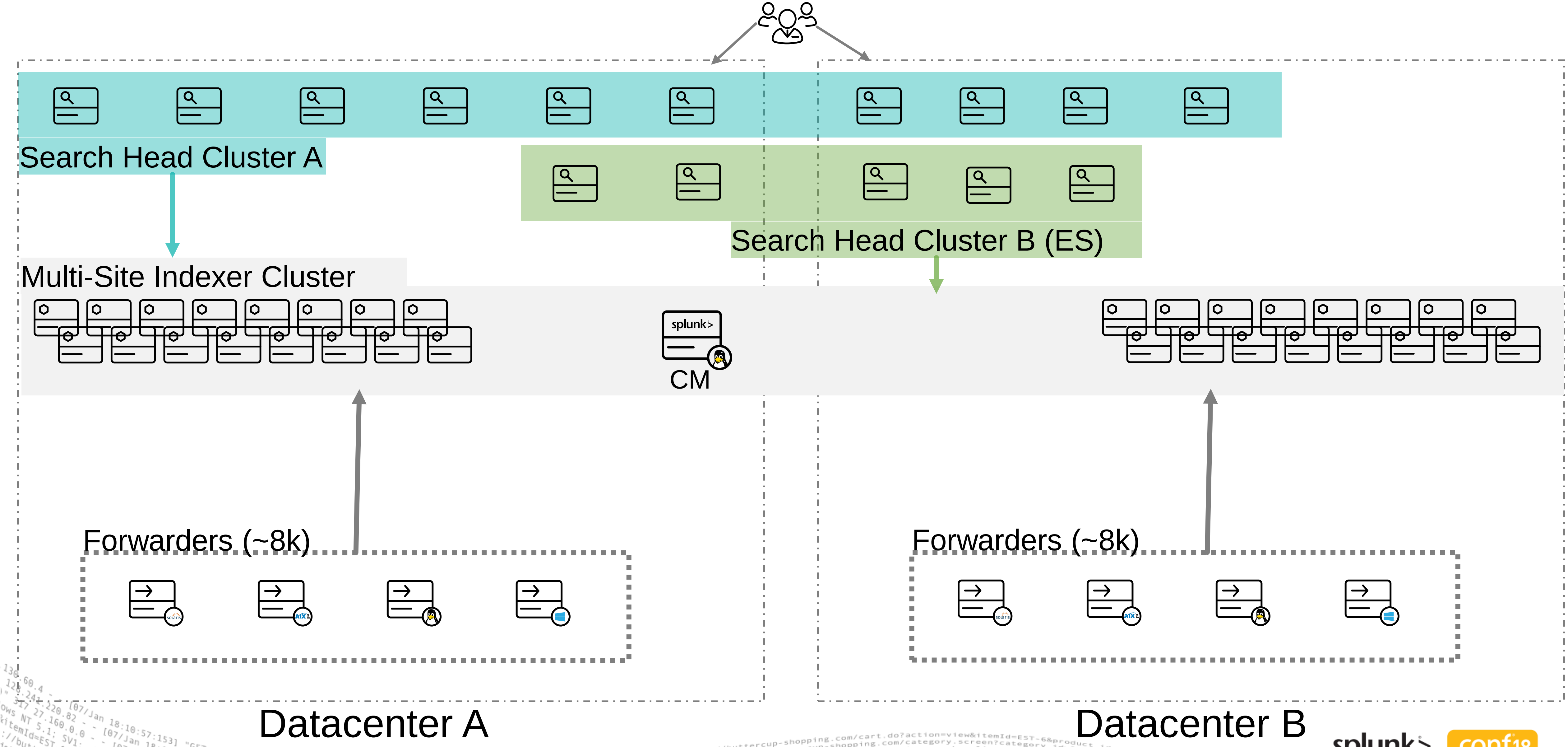
About Our Deployment

- On-prem datacenters
- Heterogeneous technology stack
- Licensed for 13.5 TB/day globally
  - ~10TB/day in North American deployment

The Inherited Environment



The Desired Environment



Technical Debt: Aging Indexing Infrastructure

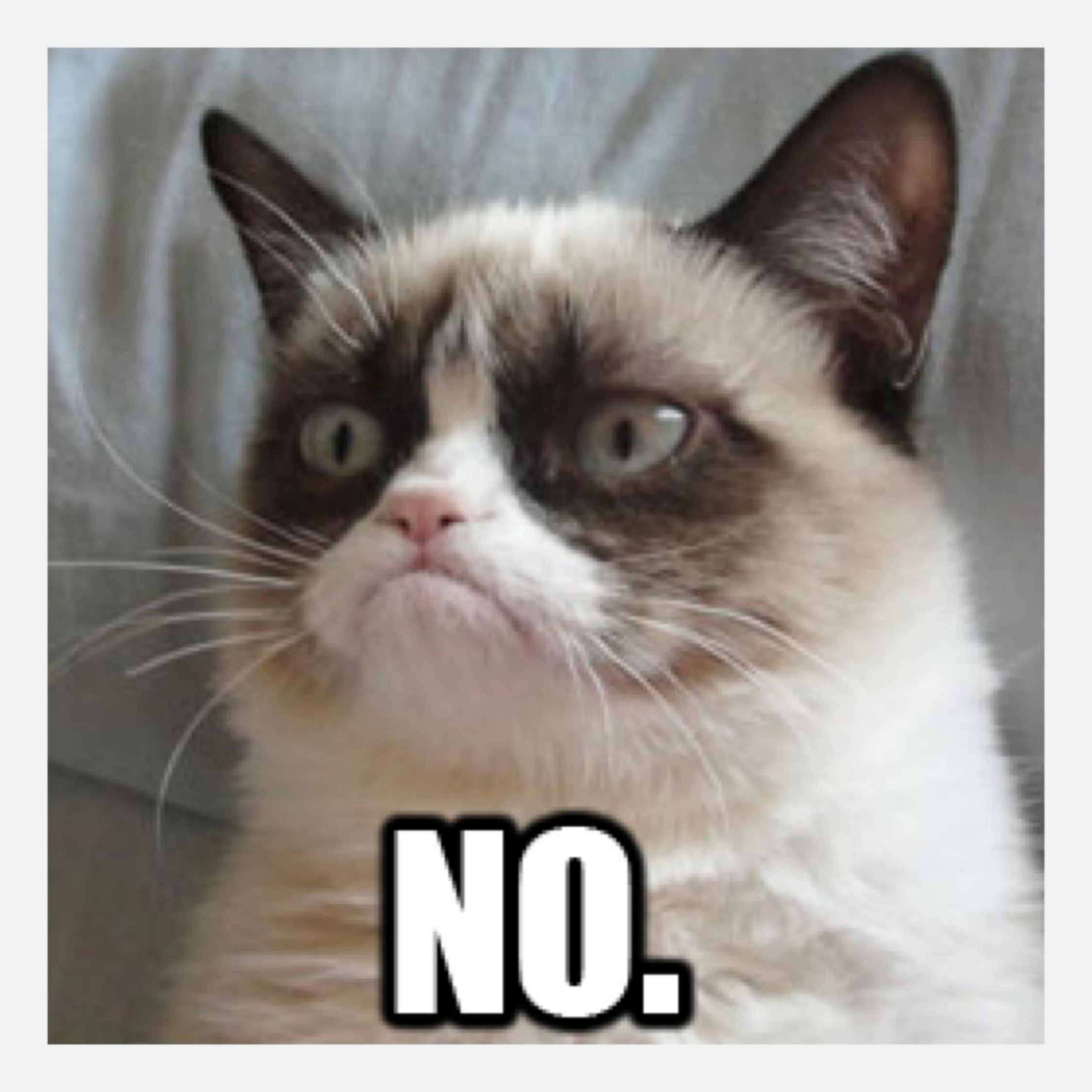
Migrating Indexing activity to new hardware.

The Indexing Hardware Migration Plan

- Stand-up new multi-site cluster
- Perform data cleanup on sourcetypes while migrating them into the new cluster.
- When data cleanup is complete, point inputs stanzas to new cluster using _TCP_ROUTING.
- Let the legacy system die on the vine after data retention period expires.

Best Laid Plans…

- Why are our HDD drives failing like dominoes?
- Wait, we bought all these drives at the same time right….uh oh.
- Fix it now!
- BTW, no downtime.



The New Plan

- Migrate buckets to new indexers
  - Splunk Admin Manual - https://docs.splunk.com/Documentation/Splunk/7.1.2/Installation/MigrateaSplunkinstance

- ~2PB of warm/cold buckets migrated using rsync.
  - Initial migration job took around a week.
  - Nightly incremental jobs were run to keep warm/cold buckets in sync as we swapped indexers in batches.

Example rsync script

```
declare -A hostmap=( [newsystema]=oldsystema [newsystemb]=oldsystemb [newsystemc]=oldsystemc )
dst=$(hostname | sed 's/\..*//' | tr '[A-Z]' '[a-z]')
src=${hostmap[$dst]}
.
.
rsync -av --partial --exclude "/*/db/hot_*" ${src}:/opt/splunk/var/lib/splunk/_* /opt/splunk/var/lib/splunk/ &
rsync -av --partial --exclude "/*/db/hot_*" ${src}:/opt/splunk/var/lib/splunk/[a-d]* /opt/splunk/var/lib/splunk/ &
.
.
rsync -av --partial --exclude "/frozen" ${src}:/opt/splunk/cold/_* /opt/splunk/cold/ &
rsync -av --partial --exclude "/frozen" ${src}:/opt/splunk/cold/[a-d]* /opt/splunk/cold/ &
.
.
wait
```

Swapping Indexers in the Cluster

- **Make sure indexes.conf settings are equivalent on both systems**
- Place cluster in maintenance mode
  - splunk enable maintenance-mode
- Shutdown splunk on source node:
  - splunk stop
- On destination node:
  - Run incremental rsync
  - Configure [clustering] stanza on destination node to point to cluster master
  - splunk start
- Remove source node from cluster master:
  - splunk remove cluster-peers -peers <guid>
- Deploy new outputs.conf

Technical Debt: Isolated Search Heads

Migrating content from multiple stand-alone search heads to a search head cluster.

Challenges

- Classifying apps across all instances
  - How many search heads is the app installed on?
  - Is the app visible to users and have they created local content?
- Knowledge Object Divergence
  - Identifying KO conflicts in apps that are on multiple search heads.
  - How to determine which conflicting setting is correct?
  - How many users will be affected by using one KO vs another?
- Comparing settings across multiple divergent instances of an app was very tedious and time consuming.

Merging Strategy

- Needed a repeatable mechanical merge process
- Tarballs of $SPLUNK_HOME/etc/apps and $SPLUNK_HOME/etc/users were collected from each search head.
- The app instance from the highest search volume search head was used as the baseline configuration.
  - Resulted in the least amount of user impact with regards to conflicts.
  - _audit and _internal data was used to identify access volume for each app.
- Detecting Conflicts
  - .conf/.meta files – Exact string matches for key/values.
  - Lookups/views – Hashed file contents.
- Implemented in Python

Example: .conf settings merge

Search Head A – 200 unique user access per day

```
[sourcetype:a]
EXTRACT-response_time =
duration=“(?P<duration>\d+\.\d+)”
EXTRACT-response_code =
response_code=“(?P<response_code>\d+)”
EXTRACT-dst_ip =
dst_ip=“(?P<dst_ip>\d{1,3}(?:\.\d{1,3}){3})”
```

Search Head B – 5 unique user accesses per day

```
[sourcetype:a]
EXTRACT-response_time =
duration=“(?P<response_time>\d+\.\d+)”
EXTRACT-response_code =
response_code=“(?P<response_code>\d+)”
EXTRACT-dest_ip =
dst_ip=“(?P<dest_ip>\d{1,3}(?:\.\d{1,3}){3})”
```

Merged Configuration

```
[sourcetype:a]
EXTRACT-response_time =
duration=“(?P<duration>\d+\.\d+)”
EXTRACT-response_code =
response_code=“(?P<response_code>\d+)”
EXTRACT-dst_ip =
dst_ip=“(?P<dst_ip>\d{1,3}(?:\.\d{1,3}){3})”
EXTRACT-dest_ip =
dst_ip=“(?P<dest_ip>\d{1,3}(?:\.\d{1,3}){3})”
```

Application Breakdown

- Over 200 apps across all stand-alone search heads.
- 150 apps were visible to users and allowed power users to create app level content.
- 40 Applications were present on multiple search heads and contained conflicting content.
- 5-10 users felt “actual” impact

Deploying Merged Content

- Deployed all merged content using search head cluster deployer
  - Quick, easy, and officially supported method of distributing content to a search head cluster.
- All content ends up under default/ & default.meta on the cluster.
  - Users lost the ability to remove content they owned.
- Developed python program to copy content the cluster locally
  - Removed app from deployer and applied shcluster-bundle
  - Returned app skeleton to deployer with default/ content and applied shcluster-bundle
  - Transferred user created content directly to the cluster via REST API.

Technical Debt: Data Onboarding Cleanup

The Problem

- Explosion of Splunk usage company-wide
- Hundreds of new sourcetypes
- Proprietary Log Formats
- ”Management-Driven” onboarding directives
- Limited Staff

Sourcetype Cleanup

- Correct sourcetype name, bad event breaking and/or field extractions.
  - Easy to correct.
  - Normally no need to inform users of fixes.

- Incorrect sourcetype name
  - Easy Fixes:
    - Change inputs.conf: sourcetype =
    - Change props.conf: rename =

  - Harder Fixes:
    - Identifying user content that is referencing the old sourcetypes.
    - Automatic remediation?
      - Use REST API to find props/eventtypes/savedsearches/views that use old sourcetypes
      - Automatically remediate using text replacement and POST’ing back to each affected object.

Thank You

Questions?

Don't forget to rate this session in the .conf18 mobile app

.conf18

splunk>