Splunk Migrations

from here …. to THERE

mbarrie@splunk.com
mcronkrite@splunk.com

October 2018 | Version 2.0

Our Speakers

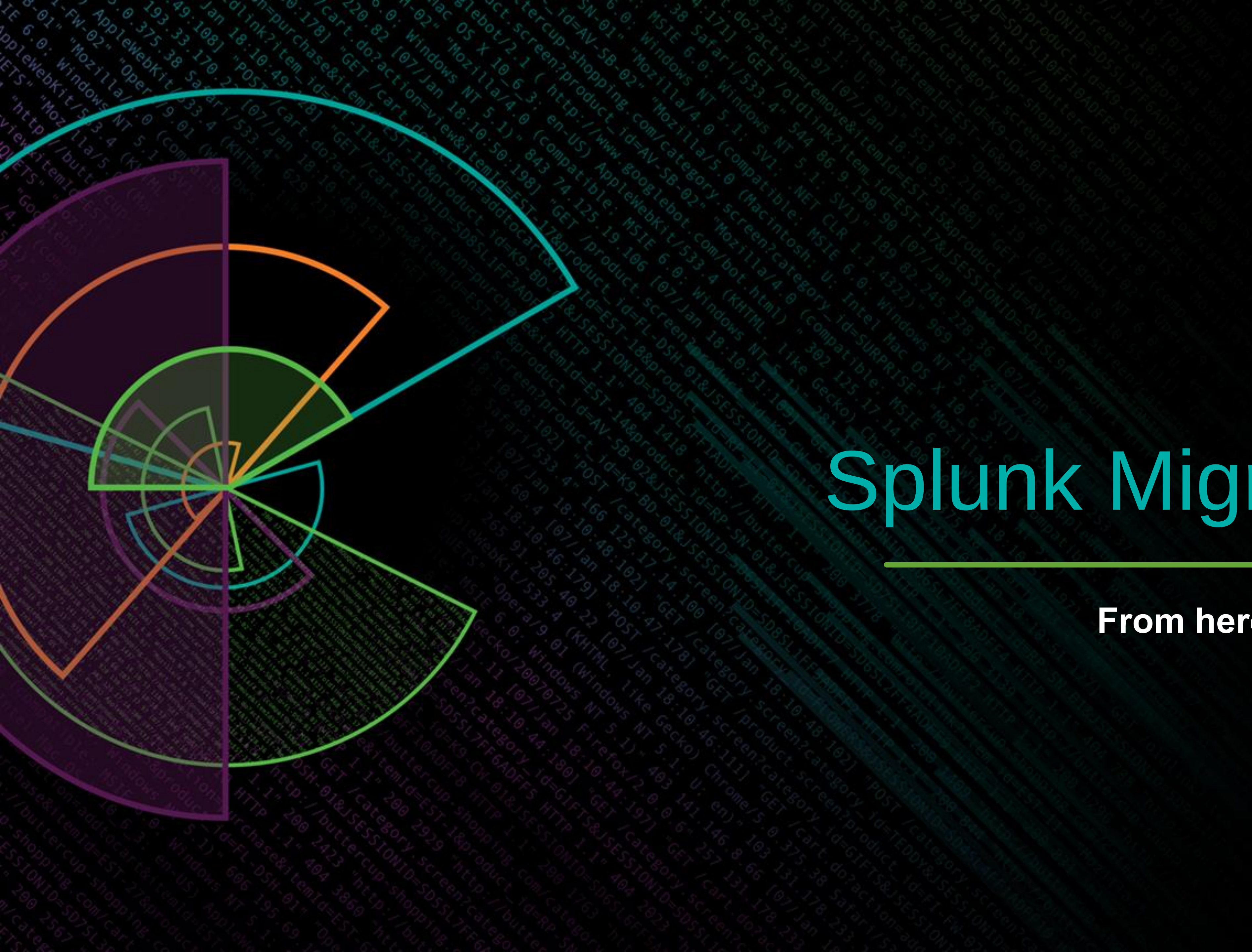
MIKE BARRIE

PS Architect West Region

MACY CRONKRITE

PS Architect Public Services

splunk> .conf18



Splunk Migrations

From here …. to THERE

splunk> .conf18

Anatomy of a Migration

The Five Factors that define any Migration

- Content
  - This is mostly just $SPLUNK_HOME/etc,
    - but you may have additional content in places like $SPLUNK_HOME/bin
- Run-time or Stateful Information
  - Usually the only interesting piece is the KVStore, except when it's not the only one
- Data
  - The are a LOT of ways to move data
- _time
  - Increasing lead time leads to more options.   Be the ant, not the grasshopper
- Bandwidth
  - The pipeline often defines the migration path.
- Services can Help!

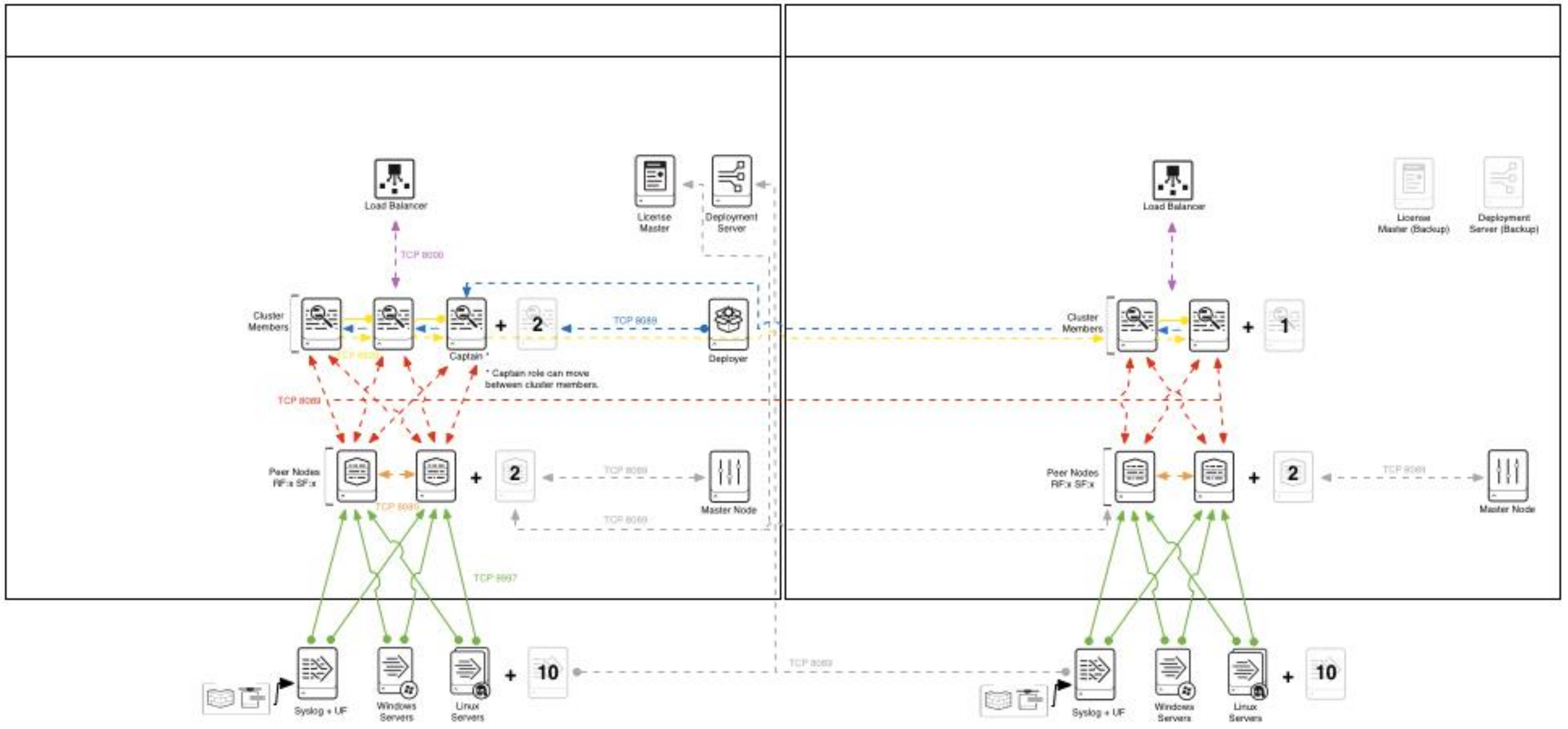
Splunk Architecture

The Splunk Elements that you need to consider

- Search Heads
  - Clustered?
  - Older? We card any Search Head that looks under 30
- Indexers
  - Clustered?
  - General Multi-Site layout?
    - Master or designated DR site?
    - Intra-site latency?
- Forwarders
  - Indexer naming
  - The decision to clone or split data should not be done lightly
  - Data routing and configurations are silent migration killers (firewalls, names. Etc.)

Splunk Architecture

The Splunk Elements that you need to consider

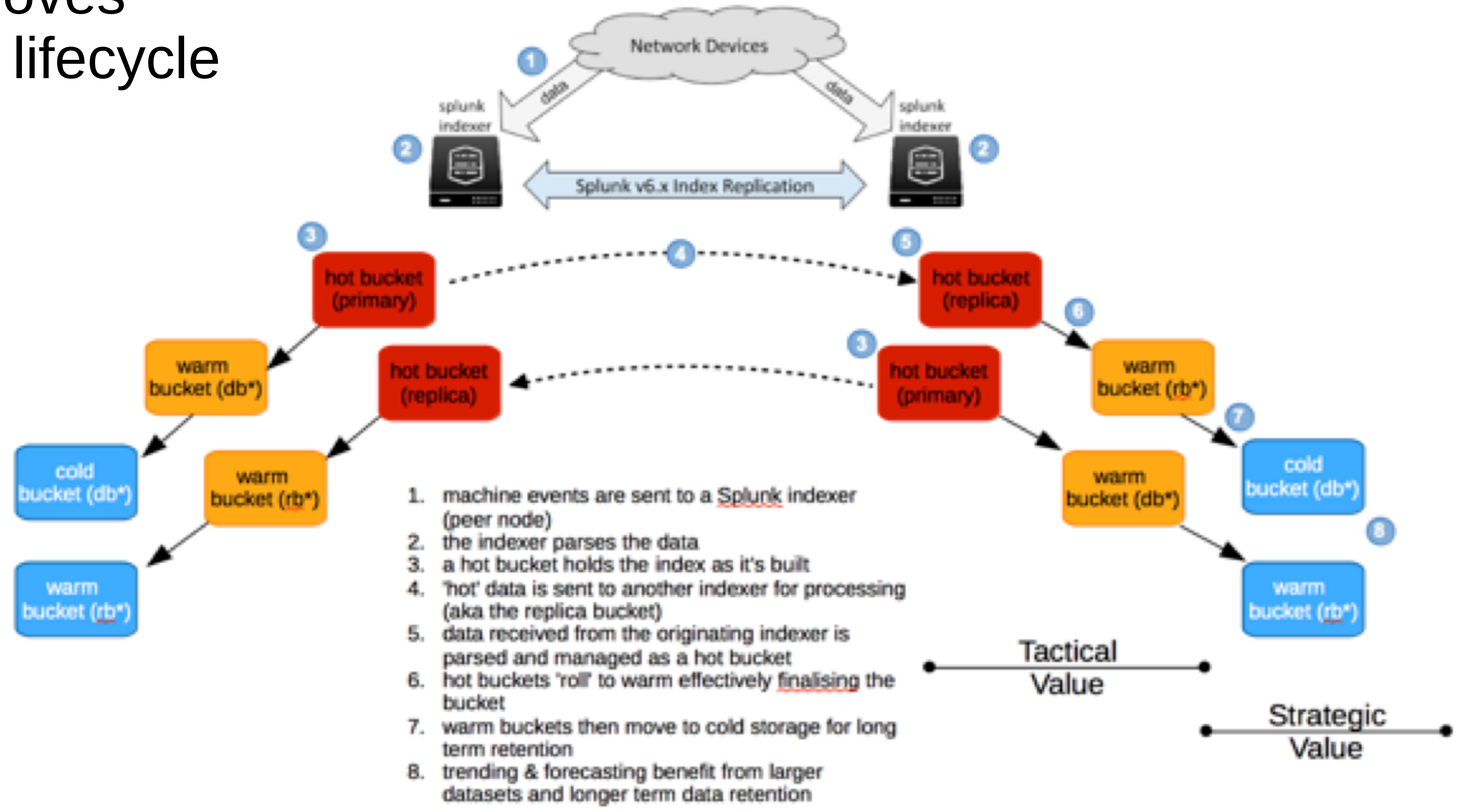


splunk> .conf18

Splunk Architecture

The Splunk Elements that you need to consider

- Understand the value of your data as it moves through the data lifecycle



1. machine events are sent to a Splunk indexer (peer node)
2. the indexer parses the data
3. a hot bucket holds the index as it's built
4. 'hot' data is sent to another indexer for processing (aka the replica bucket)
5. data received from the originating indexer is parsed and managed as a hot bucket
6. hot buckets 'roll' to warm effectively finalising the bucket
7. warm buckets then move to cold storage for long term retention
8. trending & forecasting benefit from larger datasets and longer term data retention

Splunk HA/DR

Let’s Have the Talk...

- High Availability and Disaster Recovery conversations in a migration concern two topics:
  - What are your true requirements or SLAs and who is driving them
  - How do the technologies in Splunk fit into those requirements

- Your ultimate choices will be a balance of:
  - Requirements from your organization (acceptable failover, acceptable losses, etc)
  - Cost
  - Seriously, cost
  - Adequate run-books and preparation by the Splunk team (as well as operations, networking, etc.)
  - The value and use cases for your data

First law of migrations, harm no data

primum data non nocere

The Timeline of Moving Data

A few words about moving data during a migration

- It takes time to transfer data up to the cloud
- It takes time to transfer data across a WAN to your new data center
- 1 TB takes less time than 10 TB, so...
  - Start planning NOW, every extra day you build into the migration is data you don't have to move
  - Pick and choose, not all data is created equal
  - Understand the limits of the pipe and use that to define your migration path

The Timeline of Moving Data

A few words about moving data during a migration

- The easy button is to let data age out and not move it at all
  - Hybrid search will allow you to span the old and the new while the data naturally ages
- Who wants easy?? Moving data falls into a few main methods:
  - For S2 based Splunk Cloud stacks, the S2 technology can automagically migrate data
  - The clustering logic can be used to gracefully migrate the data to a new location
  - If you have a problem... if no one else can help... and if you can find them... maybe you can hire... The Rsync-Team
  - Snowball and Snowball Edge (or your friendly local JBOD) can also be a viable approach
- Each of these methods has pros and cons
  (see the deep dive towards the end of the deck)

Indexer Migration Types

“With enough bandwidth, I could migrate the world” - Archimedes

- Going to clustering for the first time
- Single site Indexer to multi-site clustering
- Data center migration forced by lease, provider change, etc.
- Going to Splunk Cloud and the world of S2
- The forecast calls for Cloud (BYOL)
- Consolidating multiple smaller indexing environments
- Hardware refresh or restacking AMIs

Going to clustering for the first time

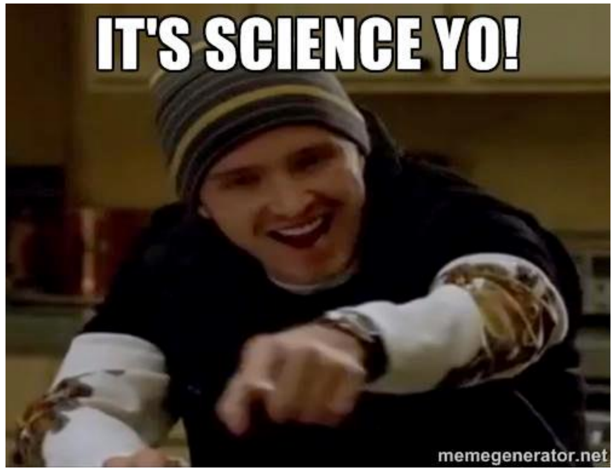
Standalone, or Distributed only indexers , SINGLE SITE

- Create new environment
  - Provision the new Indexing Cluster and config
  - Add the cluster to the existing Search Heads
    - But keep searching the existing Indexers
  - Point Universal and Heavy Forwarders at the new Indexers
    - Multi-site and HA concerns will be covered in a bit
  - Shut Off inputs on the old indexer(s)
- Let it age gracefully.. (easier) OR migrate the old data (harder)
  - Bucket names have changed
    - Single Site Buckets Need the GUID appended
    - Bucket collisions need to be accounted for

Going to clustering for the first time

Standalone, or Distributed only indexers , SINGLE SITE

- Non-clustered buckets look something like this:
  - db_<newest_time>_<oldest_time>_<localid>
    (there are also hot_v1_localid buckets, but let's ignore those)
- To convert them to single site clustered buckets, they need a new name:
  - db_<newest_time>_<oldest_time>_<localid>_<guid>
  - The guid can be found in $SPLUNK_HOME/etc/instance.cfg
- The localid's can't have collisions
  - Take the largest localid on the new Indexer's index (let's call that x) , add 100, and rename all your buckets to have a localid that is the current localid + x + 100
  - This renaming must be done in db and colddb directories in each index



Splunk Cloud Indexer Migration

Going Cloud and the wonderful world of S2

- Checklist for Splunk Cloud Migration Nirvana
  - Establish which components need on-prem presence.
    - Deployment Server?
    - Heavy Forwarders for api driven feeds, filtering 80% or more of the data, or dbConnect
    - Possibly Universal Forwarders to aggregate traffic
  - Determine a forwarding strategy
    - Firewall rules
    - Bandwidth considerations
  - Determine the need for a data migration
    - Cloud Services can help! They will upgrade your stack and use S2 to drain data into S3

Splunk Cloud Indexer Migration

Going Cloud and the wonderful world of S2

- No I can’t keep the data on site
  - Splunk Cloud search heads can’t search your indexers
  - Your search head can, however, search the Splunk Cloud indexers
    - Plan early and you can use hybrid search to age out your data

PLAN EARLY AND RETIRE YOUNG
NOW YOU KNOW, AND KNOWING IS HALF THE BATTLE
imgflip.com

Single site Indexer to multi-site clustering

Why are we doing this?

- Initially forgot to setup the new single site indexer as a multi-site
  - The old buckets will hang around, but won't replicate beyond their origin site
- A multi-site cluster with 1 site is totally a thing
  - When you convert from non multi-site to a multi-site cluster of 1 site, the clock starts
- Add new indexers that are site aware to the non-site aware cluster.
- Let the Cluster master heal the cluster by itself
- Decide on Search Affinity
  - Placing Search Heads in site 0 spread searchs across all Indexers
  - Pacing Search Heads in a site favors using only indexers in that site
  - In most cases, using site 0 is preferred

Single site Indexer to multi-site clustering

Why are we doing this?

- Side bar: Why do I even want Multisite???
- You have multiple sites that have minimal latency between them
  - This can allow for redundant copies of data that span data centers
  - This can allow for copies of data in a site earmarked as a repository
  - This can allow for the use of site aware search (but that’s a different talk, ask your Doctor is site affinity is right for you)
- You want to create logic sites within your single data center
  - This can be used to help with OS patching in large deployments

Data center migration

We’re movin’ on up (or East, or West, or to Akron)

- Data center migration forced by lease, provider change, etc.
  - Make sure your in your same compliance zone don’t high go from high to low.
- In a hard cut-over, timing is everything
- The size of the pipe really matters now
  - Moving data with clustering logic
  - Moving data with rsync
  - Moving data by admin / sneakernet
    - Snowball Edge
- Noisy Neighbor problems in a production environment can create problems
  - Impact on WAN traffic for other production applications
  - Impact of other traffic on your migration timelines

Your Cloud

AWS, Azure, Google, the list goes on...

- All cloud migrations have to reconsider
  - Networking
    - Data Flows / Paths / Private VPN / Firewalls
  - On-prem devices
    - (ldap, stream , adaptive response, scripts)
- On-Prem to BYOC
  - Use hybrid search / age out the dataBYOC to BYOC
  - Bandwidth between cloud costs $$$ (again, hybrid search)
- BYOC to On-Prem (not common)
  - backup
  - Store the data --- expand the racks ?

Consolidating Multiple Environments

- All of the bucket rules from previous slides apply
- Indexers clusters can be most easily joined when they are the same (multi-site, single site, etc.)
- Indexers that are not clustered can still be joined into a cluster
- All moving related rules from previous slides apply
- The actually joining is as simple as point the old Indexers to the new Cluster Master
  - Well not that simple, there needs to be a strategy in place for failing over search
  - This requires a window of change control to be done without duplicate or incomplete results

Hardware Refresh

- ‘Hardware’ refresh comes in a few flavors:
- Burn it all down and start with new Indexers
  - Clustering makes this easier as you can employ the clustering logic to retire older Indexers after new ones are added
  - Make sure you know where your Forwarders are forwarding and Search Heads are searching
- Add some new Indexers, remove some but not all of the old
  - This is reality sometimes, but be aware that the new Indexers are gated by performance of the old ones
- Restack AMIs in AWS (or the equivalent in your cloud)
  - If you have a requirement to restack AMIs, you can use the clustering logic (assuming you have > 1 searchable copy)
  - Without clustering you will need to rely heavily on non-ephemeral storage solutions

Search Heads

“Nobody in history has ever been fired for losing a macro” - Anonymous

Search Head Migration Paths

- Standalone Search Head to standalone Search Head
- Moving to Splunk Cloud
- Consolidating Multiple Search Heads into a single non clustered Search Head
- Consolidating Multiple Search Heads into an new Search Head Cluster
- Moving from one Search Head to a Search Head Cluster
- Premium App Considerations (Enterprise Security, UBA, and ITSI)
- Any windows SHC considerations

Standalone Search Head to standalone Search Head

- $SPLUNK_HOME/etc can be copied from the old Search Head to the new.
  - This brings over the splunk.secret which makes migrating configs easy.
  - Splunk.secret is the file used to hash passwords in config files
  - Certain pieces like the keys generated may not copy over well, which in turn means re-adding the Search Head to the peers or the Cluster Master(s)
    - These keys live under $SPLUNK_HOME/etc/auth, YMMV
    - $SPLUNK_HOME/etc/system/local often has host names baked in to fields like server.conf and inputs.conf that need to be updated to reflect new host information if you are moving data centers
  - Try to find any needed content out of band from Splunk (cron jobs, etc) that might rely on some host detail
  - Check $SPLUNK_HOME/bin

Standalone Search Head to standalone Search Head

- Run-time/Stateful Information
  - Usually the only interesting piece is the KVStore, except when it’s not the only one
    - Some apps store checkpoints in $SPLUNK_HOME/var/run
    - This is rare, but important to tease out
    - Normally affects api-based ingest methods
  - Splunk 7.1.x and later allows for simple kvstore backups.
  - For pre-7.1, there’s an app for that (Gemini tools)
- _time
  - For this migration the only consideration is how to cutover users (above and beyond Indexer migration strategy)

Moving to Splunk Cloud

- Step 1: Engage Splunk Cloud Services
  - Cloud Services specializes in the tools needed to thoughtfully pick up and move content to Splunk Cloud during the migration process.
    - App vetting
    - Moving and validating content
  - Your Job: Try to clean house before you make the move, there’s no better time like the present
- Run-time or Stateful Information
  - Again, Cloud Services has the tools to help with this process
  - This often includes a period of hybrid search to allow data to age out and test the migration effort
- _time
  - Starting early means having less headaches

Consolidating Multiple Search Heads Into a Single Non-clustered Search Head

- Traditionally this is a consolidation and streamlining decision
- The biggest concern in planning is to ensure there is no overlap or conflicting knowledge objects
  - Some of this overlap is easy to fix, some is hard
    - One solution is to script rest calls with curl and at least isolate various objects to app scope
    - Problem: What about objects created and used by other apps?
    - Solution: Hand review of the content
  - Diagnosing these conflicts can be hard
    - Imagine two field evals with the same field name but based on a different interpretation of the data.   This could silently break the logic of searches that use those fields.
      - There is no easy button, time to roll up your sleeves or engage with Professional Services

Consolidating Multiple Search Heads Into a Single Non-clustered Search Head

- Run-time or Stateful Information
  - This is where the Gemini Tools app shines, you can merge multiple KVStore backups
  - You aren't merging premium apps right? No Soup for you!
    - Merging multiple ES environments is graduate level work, talk about it with your team
- _time
  - Failover Windows here can be short if Data Model Accelerations are not heavily used.
  - In this scenario you can't avoid rebuilding most of the DMAs since there are multiple GUIDS

splunk> .conf18

Consolidating Multiple Search Heads Into a New Search Head Cluster

- Content
  - Search Head Clustering can change the workflow and a users experience.
    - If consolidated content is pushed from the Deployer, that content be not be moved or deleted
    - There is content available from support for injecting content into the cluster to avoid the Deployer's semi-read-only approach to content
    - Consider what should be managed from the deployer vs. the SHC itself
      - Do users routinely create or modify things like fields extractions?
      - Do users need control exercise control over correlation searches? Dashboards?
    - All of the issues in the "Consolidating Multiple Search Heads Into a Single Non-clustered Search Head" slides apply here
  - The biggest challenge is understanding your users and how they use the platform

Consolidating Multiple Search Heads Into a New Search Head Cluster

- Run-time or Stateful Information
  - All of the issues in “Consolidating Multiple Search Heads Into a Single Non-clustered Search Head” apply here as well
  - A common pitfall for standalone to SHC migrations is to drop content onto the Search Heads and hope for the best
    - Spoiler Alert: This ends badly.
- _time
  - Failover Windows here can be short if Data Model Accelerations are not heavily used.
  - In this scenario you can’t avoid rebuilding most of the DMAs
  - These migrations will almost always involve a large change control windov

Moving from one Search Head Cluster to a Search Head Cluster

- Content
  - Easy button: Use the clustering logic
    - Add the new nodes (assuming latency is sub 100ms)
    - Retire the old nodes
    - Profit
  - Hard(er) button:
    - Understand how the SHC treats local content vs. Deployer content
    - Contact support and/or Professional Services
    - Copying content from A->B won't work
      - The content in $SPLUNK_HOME/etc/apps/<app name>/local should stay local
      - You need code to properly seed the cluster if you want to preserve workflow
  - See the Deep Dives slide with this name for more details

And Migrating Everything Else…

Premium App Considerations

- Enterprise Security
  - Data model accelerations are tied to the GUID of the Search Head for Enterprise Security
  - On your new environment, disable the scheduler and copy the GUID over from the old environment
    - Remember the GUID on a Search Head Cluster lives in server.conf, but in etc/instance.cfg on a standalone Search Head
    - If you reuse the GUID, you can save on rebuilding all the DMAs
    - Don’t cross streams! Only one ES SH (or SHC) should be active at a time during the migration

Premium App Considerations

- Enterprise Security
  - Splunk PS cautions against migrations of Enterprise Security to a Search Head Cluster unless absolutely necessary
    - Modular inputs for things like Threat Intel need to be considered
    - Upgrading and maintaining ES creates challenges
    - It’s supported and it works, it’s just harder
    - Splunk PS can help, there is no easy button (or spoon) here

Premium App Considerations

- Run-time or Stateful Information for Enterprise Security
  - ES heavily leverages the KVStore, so migrating the content will make for a more seamless migration
  - Always confirm that your existing Search Head was forwarding to the Indexers
    - This ensures your notables and other summary indexes will be there waiting for you
- _time
  - You can't really have two ES Search Heads running in parallel, you need to plan a firm cutover
  - Repeat: There is no realistic side-by-side 'soft' migration for ES
  - Planning an ES or ITSI migration is mostly about understanding what in the environment is truly custom/unique
    - That clever cron job you wrote two years ago to populate a block list is a good example.

Forwarding Changes for Splunk Cloud

- Considerations for Universal or Heavy Forwarders as Aggregation Points
  - Universal Forwarders doa better job and are easier to manage
  - Reserve Heavy forwarders for what they are good at (filtering >80% of the data, data redaction, custom apps that use JDBC, API, or REST calls)
- How do I get data to Splunk Cloud? Load Balancers? Direct Connect?
  - Ideally have Forwarders send data diret to Splunk Cloud
  - Aggregate syslog traffic on prem and then send to Splunk Cloud
  - When you absolutely have to, an intermediate forwarder can solve problems
- Using the HEC in a Splunk Cloud Deployment
  - It's easy! Cloud Ops can create an Elastic Load Balancer for you to point HEC traffic at

To Aggregate or Not to Aggregate

That is the question

- Not aggregating is always a preferred path
  - One less point of failure
  - Better data distribution on the Indexers
  - One less box(es) to manage
- Universal Forwarders are the next best option
  - They are much better at balancing data evenly across Indexers
  - They are lighter weight and more efficient in sending data over the wire
    - Twice as efficient in most cases
  - They consolidate the index-time content/workload into Splunk Cloud, where you don't have to manage it
    - SAAS FTW!

To Aggregate or Not to Aggregate

That is the question

- Heavy Forwarders still have a place in this role
  - You have compliance needs to filter some data before it leaves your site
  - You are going to drop 80% or more of the data
  - You have data sources that require the Heavy Forwarder
  - That’s it, otherwise where possible use a Universal Forwarder
- Fun fact: Nothing is more likely to start a Splunk bar fight than this topic
  - The product evolves
  - Sometimes faster than habits

splunk> .conf18

Getting Data In

- Once you get your Splunk Cloud instance, you can download an app with the right SSL certs and outputs.conf to send data
- Ideal is every instance sending data to Splunk Cloud
- Reality is sometimes putting a tier of intermediate forwarders in place
- You will, at a minimum, in most cases need a syslog server and Deployment Server in your environment
  - You can get a stub license for any Heavy Forwarder or Deployment Server
  - Syslog, in most cases, is best paired with a Universal Forwarder
  - If you have a mature config management process in place (Salt, Chef, Puppet, etc.), you can leverage that and avoid managing via the Deployment Server
    - Using Deployment Server is the Easy Button, but integrating into your tool stack is the long term button

Getting Data In

- Data ingest that does not involve traditional syslog or UF ingest should be handled with care
  - Kafka, Routing through the previous SIEM, you know who you are...
  - Many message bus technologies could be useful and possibly in play in your environment, and that’s OK
  - While the transport mechanism is fine, the format of certain sources (Windows event logs for example) can break the OOTB app content when pulled in via more exotic clients (nifi, nxlog, the SenSage Windows Retriever, etc.)
    - Fixing this content can involve heavy manual effort (in some cases)
    - Pick your battles
      - Some data sources are easier (syslog device data)
      - Some are harder (Windows Event Logs)

The HEC is a Hammer, Your Data is a Nail

- The HTTP Event Collector is a great way to get data to Splunk Cloud
  - It can be used for application data (many many log adapters out in the wild)
  - Ephemeral instances (containers, etc) where a Universal Forwarder doesn't make sense
  - Syslog data as well (where appropriate)
  - Kafka, collectd, the possibilities are limitless

The HEC is a Hammer, Your Data is a Nail

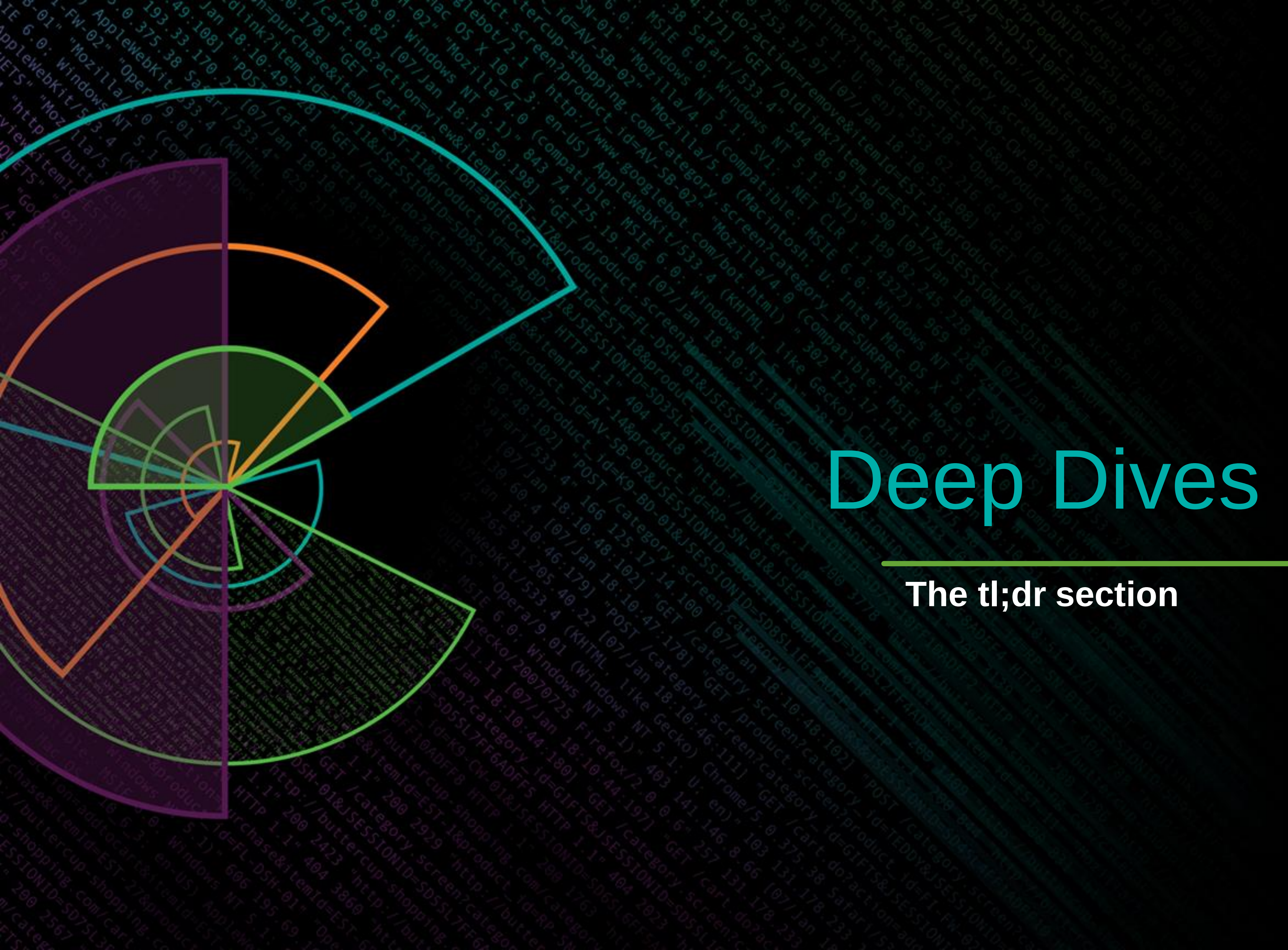
- By request, Splunk Cloud Ops will put an ELB in front of your Indexers for HEC traffic
  - You devices and apps then just send data to a single VIP and Round Robin across your Indexers
  - This is one place where a Load Balancer is an ideal and supported piece of a Splunk Architecture
    - I can totally see you thinking about additional uses for Load Balancing... Don’t be that person
      - That’s a different .conf talk, but normal forwarder traffic should never hit an LB (or ELB)
    - For now just remember ‘HEC and ELB are OK for me, UF to ELB to Indexer, and now I’m a Poindexter’
      - Fun Fact: Raise your hand if you are old enoug to remember Felix the Cat, Macy sure doesn’t

Thank You.

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

.conf18

splunk>



Deep Dives

The tl;dr section

The Timeline of Moving Data

The deep dive

- The clustering logic can be used to gracefully migrate the data to a new location
  - Join the new and old indexers to a single cluster
  - Make sure you are forwarding data to the new indexers
  - Place all the old indexers in manual detention
  - One indexer at a time (important!), run splunk offline --enforce_counts
  - The enforce_counts ensures that all buckets on the indexer have been replicated somewhere else before splunkd shuts down
  - This bucket cleanup runs at somewhat low priority, so the process can take a while
    - Which in turn minimizes the felt impacts of the offline activity

The Timeline of Moving Data

The deep dive

- If you have a problem... if no one else can help... and if you can find them... maybe you can hire... The Rsync-Team
  - Details: This was the tried and true method for years to move data off of an Indexer. You would create an rsync job to sync all warm and cold (but not hot) buckets to a staging directory on the same physical volume on the target Indexer.
  - When the process is more or less caught up (and assuming the Indexer is no longer receiving data), you would restart splunkd on the old Indexer to roll all remaining hot buckets to warm, and let the rsync complete.
  - Then you would need to rename all buckets so that the bucket id in the name does not collide with existing buckets on the Indexer.
  - Once collisions are dealt with, you stop the Indexer and move the buckets into place. This is normally done while the cluster is in maintenance mode to prevent the cluster from attempting to heal itself during the cutover.

Hardware Refresh – Deep Dive

Change your storage free your mind - moving a Mount Point

1. Place cluster in maintenance mode
2. Stage the volume configurations in a local "indexes.conf" on the indexers
3. Change the configuration in the "volume_indexes" app on the CM
4. Stop the indexers
5. Change the mount point
6. Bring the indexers up
7. Remove the local "indexes.conf" from the indexers
8. Deploy cluster bundle from CM
9. Take cluster out of maintenance mode

The Timeline of Moving Data

The deep dive

- More information on bucket naming can be found here: http://docs.splunk.com/Documentation/Splunk/7.1.2/Indexer/HowSplunkstoresindexes#Bucket_naming_conventions
- Shameless plug: This is a task that carries significant risk. Professional Services can help!
- Rsync allows you to rate limit the data drain with precision and can work on multiple Indexers in parallel. It can also run over long periods of time at slow transfer rates, if there is sufficient wall clock time.

Bucket Moving 101

Pseudo Code for Bucket Renaming Strategies

- Non-clustered to clustered bucket renaming
  - On each Indexer (where <guid> is the guid for that indexer in $SPLUNk-HOME/etc/instance.cfg:

```
for each index:
    for homePath and coldPath:
        rename db_<bucketID>to db_<bucketID>_<guid>
```

- Non-Multisite Clustered to Multisite Clustered
  - This cannot be done safely in earth's gravity.

Bucket Moving 101

Pseudo Code for Bucket Renaming Strategies

- Merging Buckets from One Indexer to Another

```
For each original Indexer I_old moving buckets to a new Indexer I_new
    For each index
        for homePath and coldPath
            create a <staging_directory> on I_new on the same physical volume as the current indexes
                named <index_name>/<pathname>
            copy/sync (db|rb)_* from I_old to I_new in <staging_directory>/<index_name>/<pathname>
When all_buckets have moved…
On each new Indexer with Splunk stopped
    For each index
        For homePath and coldPath
            Determine the largest bucketID in use in the .bucketManifest file in the homePath
            Add 10 to that number to get <new_max_bucket>
            Rename all (db|rb)_<bucketID>_<guid> to (db|rb)_<bucketID + new_max_bucket>_<guid>
            Move the renamed buckets from the staging directory into the the appropriate index directory
```

Bucket Moving 101

Pseudo Code for Bucket Renaming Strategies

Notes: The guid for the migrated buckets can carry over from the old Indexer. The cluster will still ensure it meets search and replication factors.

In clustered environments, these operations can generate a large amount of unnecessary bucket cleanup traffic unless done while all Indexers are down

...no seriously...

Bucket Moving 101

Which buckets are primary? Sometimes moving less is more.

- (must be run on the Cluster Master)

```
~~| rest splunk_server=local /services/cluster/master/buckets~~
~~        | rex field=title "^(?<repl_index>[^\~]+)  | search repl_index="*" standalone=0 frozen=*~~
~~        | rename title AS bucketID  | fields bucketID peers.*.search_state peers.*.bucket_flags frozen~~
~~        | untable bucketID peerGUID value~~
~~        | rex field=peerGUID "peers\.(?<peerGUID>[^\.]*?)\.(?:search_state|bucket_flags)"~~
~~        | rex field=bucketID "^(?<IDX>[^~]+)~(?<localBid>[^~]+)~(?<SrcGUID>[^~]+)"~~
~~        | eval bucket_primary=if(value=="0xffffffffffffffff",1,0)~~
~~        | eval peerGUID=if(peerGUID=="frozen", SrcGUID, peerGUID)~~
~~        | search bucket_primary=1 AND bucketID!="_*"~~
~~        | join type=outer peerGUID [ rest splunk_server=local /services/cluster/master/peers~~
~~                    | fields active_* host* label title status | eval PeerName= host_port_pair~~
~~                    | rename title AS peerGUID | table peerGUID PeerName ]~~
~~        | eval Peer=PeerName+":"+peerGUID | eval Peer=if(isnull(Peer), peerGUID, Peer)~~
~~        |table IDX,bucketID,PeerName~~
```

Check https://gist.github.com/mbarrie/b492aaec02b6a3cc87bfddc5dde5a8a9 as this changed in recent versions

Moving from one SHC to a SHC– Deep Dive

- Scenario 1: Use the clustering logic to your advantage
  - Description: In this approach you rely on the Search Head Clustering logic. You simply join new search Heads to the existing Search head cluster, wait for them to get a complete replication and then on the other end retire the old Search Heads from the cluster.
    - Pros: This is easy and safe. It's the recommended route if your latency is low enough.
    - Cons: High network latency (above ~200ms) will make this solution fragile or unworkable depending on how bad the latency is.
  - Details:
    - To start, you join the new Search Heads one at a time to the existing Search Head Cluster as adhoc-only non-preferred captain Search Heads.
    - This will prevent them for scheduling jobs but will force them to be replication targets for configs and KVStore data.
    - When you are ready to start retiring Search Heads, then you can start elevating the new Search Heads to be scheduler targets and possible captains.
    - Once this is happening you can fairly quickly retire the older Search Heads.
    - The final step will be to replicate the deployer, and then point all existing Search Heads to the new deployer using the Splunk cli.

Moving from one SHC to a SHC – Deep Dive

- Scenario 2: Seed the new cluster from the old
  - Description: If latency is too high, one approach would be to try to bootstrap the new cluster using an existing SHC member that you copied over.
  - Pros: Not many, although it is easier and much less work than recreating the cluster from scratch.
  - Cons: This is a under-the-hood operation and the risk of failure is much higher than using the clustering logic.
  - Details: There are scripts in the services organization that the Cloud Services team developed for this use case.   They will essentially create replication ID's so that the objects brought over to seed the new cluster look clustered.   It's a process that would have to be delivered by Splunk PS to be successful as it requires deep understanding of the SHC replication apparatus..