Archiving Data From Splunk Indexes

Keith Schon, Principal Engineer – Hunk

Theresa Vu, Director of Product Marketing – Big Data

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.

Meet Your Speakers

Keith Schon

- Principal Engineer for Hunk
- Joined Splunk in Oct 2014
- Most interesting item ordered online:
  - Indonesian octopus
- Favorite Splunk t-shirt saying:
  - Put That in Your | and Splunk IT

Theresa Vu

- Director of Product Marketing for Hunk and Big Data
- Joined Splunk in April 2015
- Secretly hoping future releases are named after care bears
- Favorite Splunk t-shirt saying:
  - I Like Big Data and I Cannot Lie

Agenda

- What's new in Hunk 6.3
- Background: How Splunk stores data
- Non-Hadoop data retention alternatives
  - Cold-to-frozen path
  - Cold-to-frozen script
- Hunk archive index functionality
  - How it works
  - Unified Search
  - Connecting to other Hadoop applications via Archive Bucket Reader

What's New in Hunk 6.3

splunk>

Introducing Hunk 6.3

Drive Down TCO

- Archive to Hadoop
- Single Splunk Interface to Search Real-Time & Historical Data

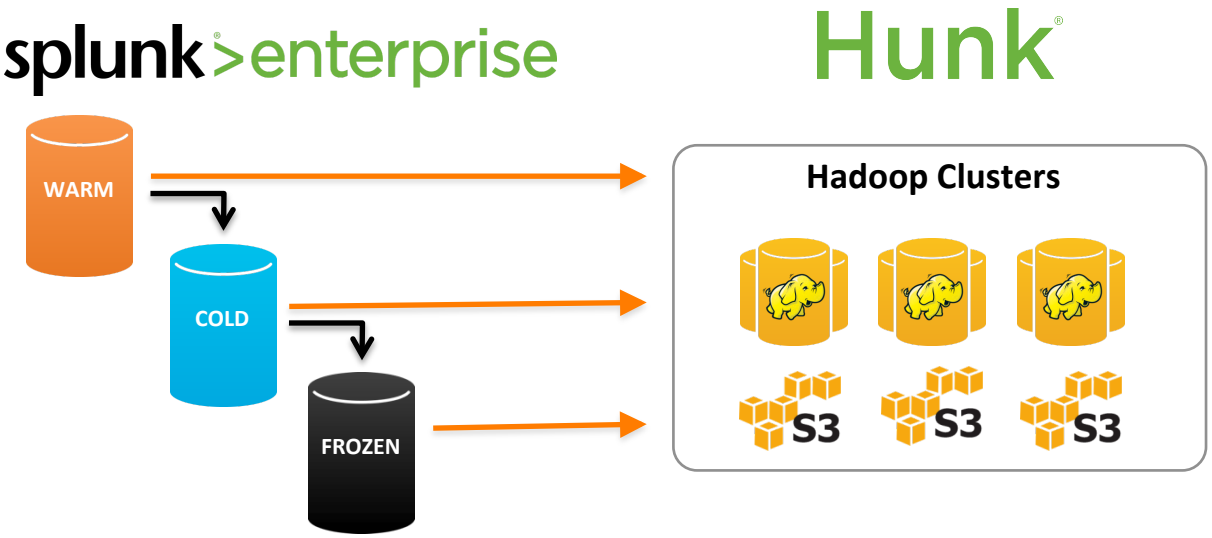
Open Access for 3rd-Party Hadoop Tools

- Access Data Using Hive or Pig
- Query Without Moving or Replicating Data

Advanced Analytics & Visualizations

- Anomaly Detection
- Geospatial Visualization
- Contextual Display

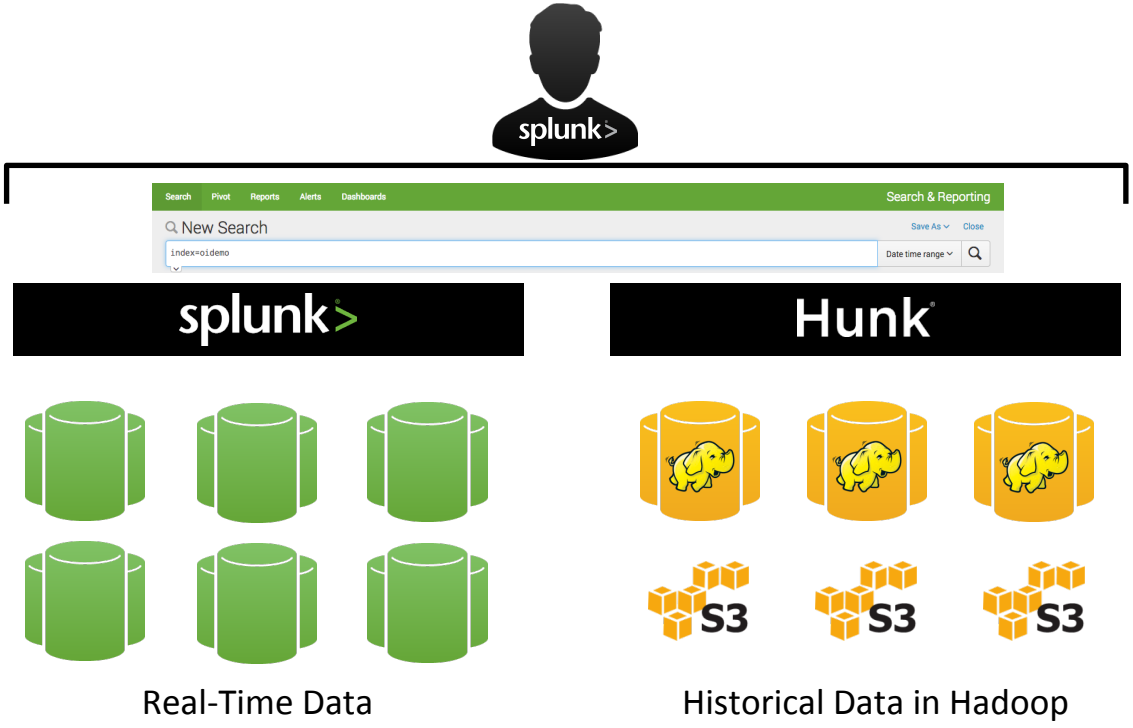
Archive Splunk Data to HDFS or AWS S3



Drive Down TCO by Archiving Historical Data to Commodity Hardware

Unified Search

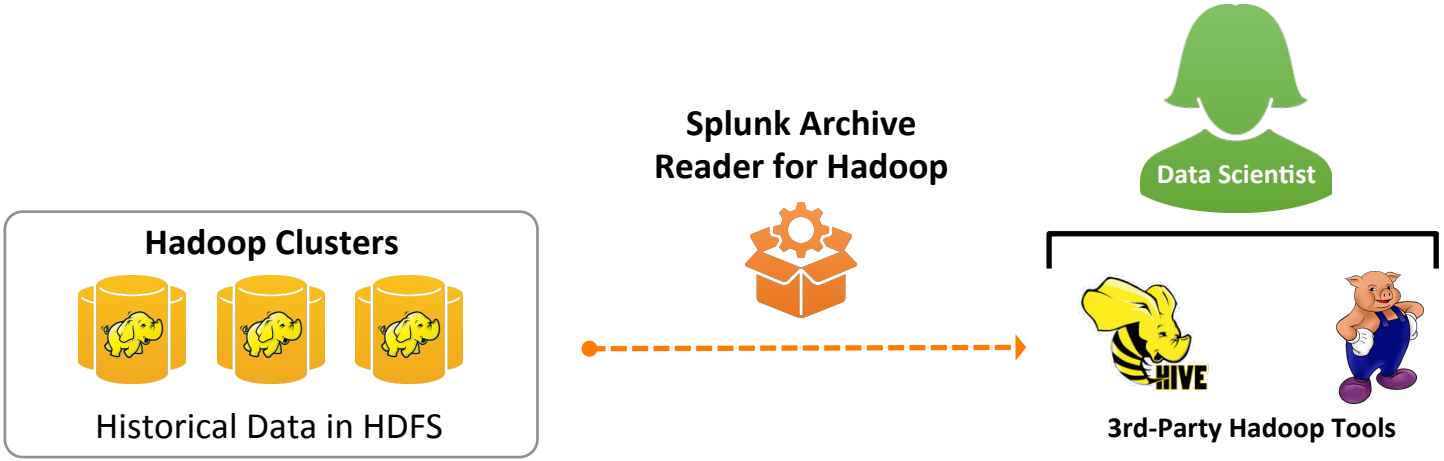
Intelligently Search Across Real-Time and Historical Data using the Same Splunk Interface



Real-Time Data

Historical Data in Hadoop

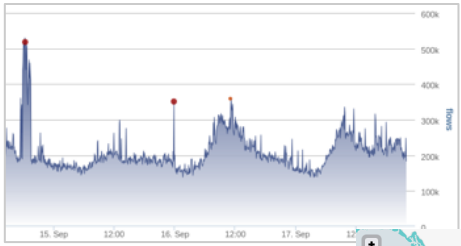
Open Access to Historical Data Using 3rd-party Hadoop tools

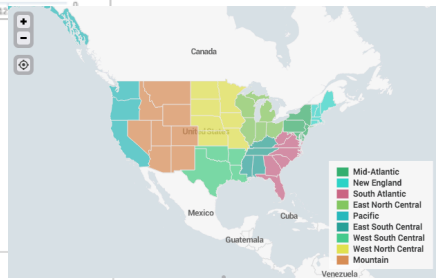


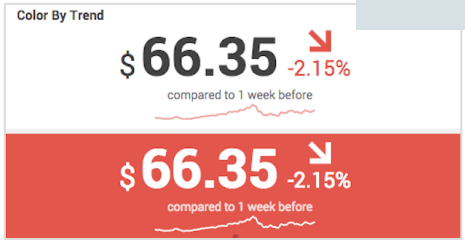
- Use 3rd-party Hadoop tools (e.g., Hive, Pig) to perform additional analysis
- Broaden data access to wider set of audiences, e.g. data scientists and analysts
- Run queries without moving or replicating data

Advanced Analytics and Visualization Capabilities

- Anomaly Detection
  - *Incorporates Z-Score, IQR & histogram methodologies in a single command*
- Geospatial Visualization
  - *Visualizes metric variance across a customizable geographic area*
- Single Value Display
  - *Derive more context by layering on visual cues and more flexible formatting*







Get a Free Hunk® License!

Index 100GB data per day or higher

Visit Hunk booth in the App Showcase

Attend “Archive Splunk Data and Access Using Hadoop Tools” session

Why Is Archiving Important?

Want to keep some data for a very long time

- Historical data useful for long-term modeling and machine learning
- Legal requirement and/or corporate data retention policies may mandate the data be accessible and searchable for set minimum periods

Also want to control Total Cost of Ownership (TCO) of data systems

- Keeping more and more data in production means more and more $$$ spent on hardware, utilities, IT effort, etc.

Splunk provides a myriad of options, giving you the flexibility to manage the trade-off of data availability vs. TCO in the way that is best for your business

Review: How Splunk Stores Data

splunk>

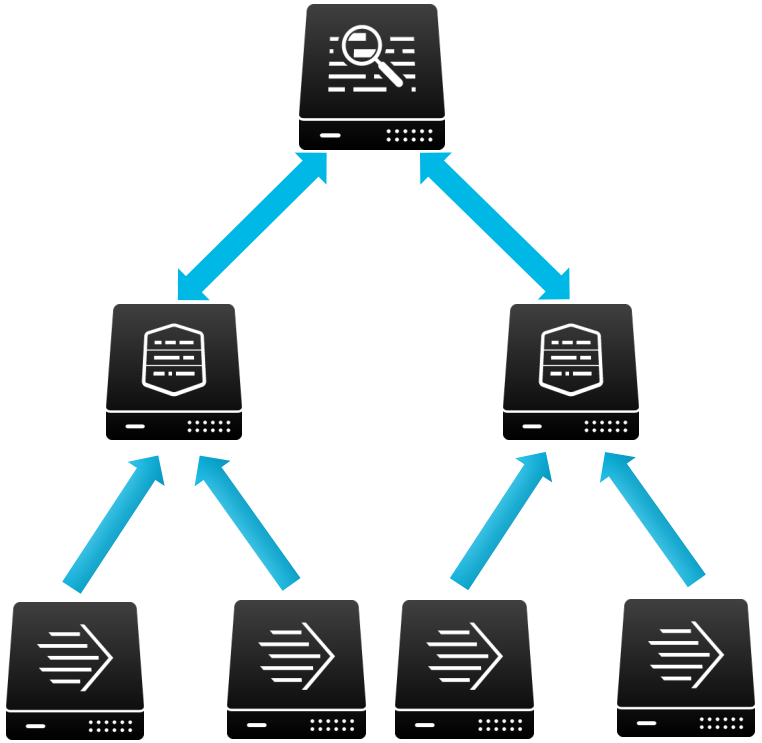
Splunk Enterprise Architecture

Search Head: Provides Splunk Web interface. Coordinates queries across indexers. Final processing of query results. May be clustered.

Indexers: Maintain Splunk indexes. Do much of the work of queries. Need good storage, and CPU.

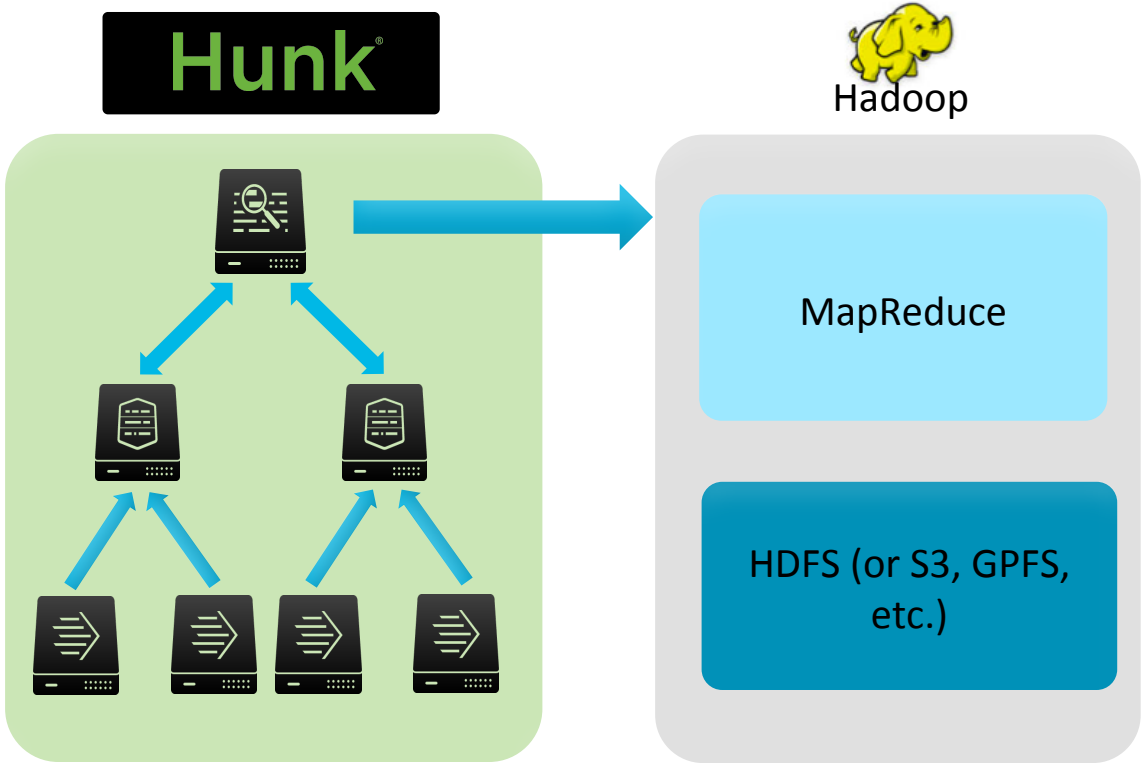
- *What do we do when these run out of space?*

Forwarders: Live where the data is created. Forward data to the indexers.



Hunk And Virtual Indexes

- **Virtual index:** Data on a distributed file system, typically HDFS or S3.
- MapReduce used to compute queries in parallel fashion.
- Virtual and non-virtual indexes may be queried together from the same Search Head.



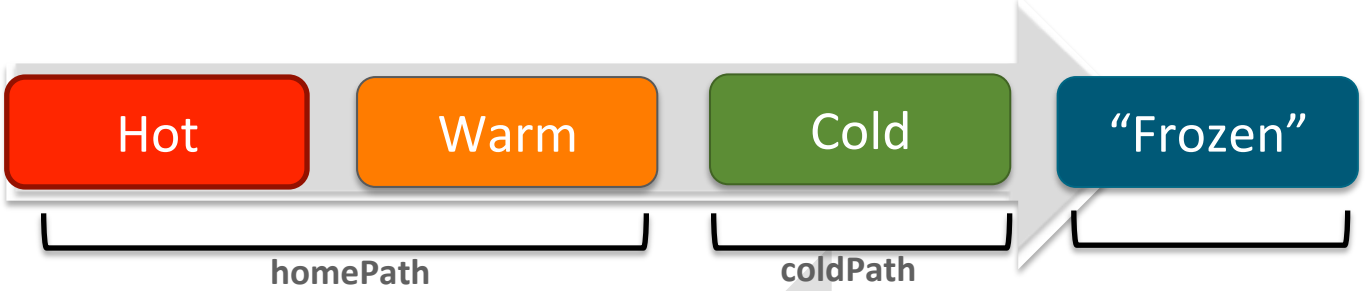
Understanding Splunk “Buckets”

- A Splunk index stores data in chunks called buckets
- A buckets contains events from only one index, from finite period of time
  - Contains raw events, may contain additional files for indexing and filtering
- Buckets age or (“roll”) through lifecycle phases

	Hot	Warm	Cold	Frozen	Thawed
Can be written to?	Yes	No	No	No	No
Can be searched?	Yes	Yes	Yes	No	Yes

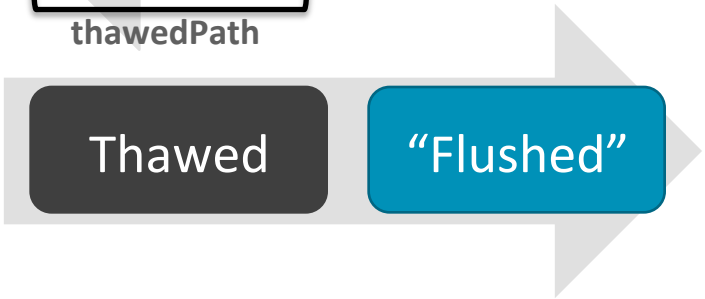
Bucket Lifecycle

Bucket aging pipeline



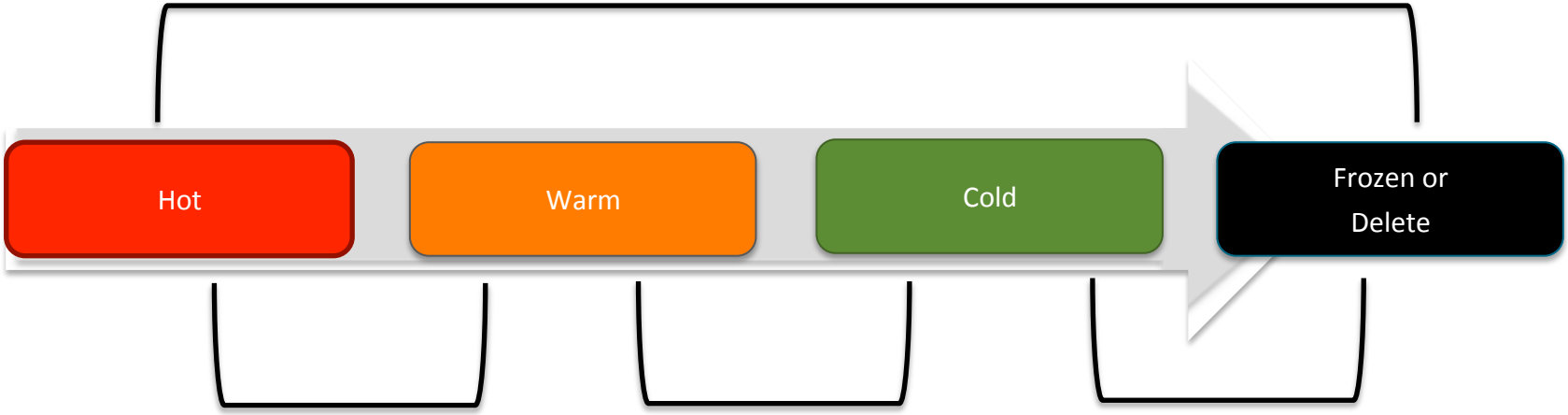
Make frozen data searchable again

Delete data from Thawed



Bucket Lifecycle Configurations

maxTotalDataSizeMB (default is 500,000 MB)
maxVolumeDataSizeMB



maxHotBuckets
maxDataSize

maxWarmDBCount

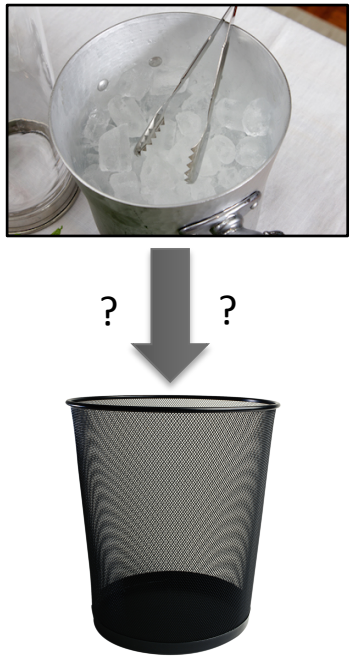
frozenTimePeriodInSecs
(default is 6 years)

“Frozen” Buckets: (Non-hunk) Archiving Alternatives

splunk>

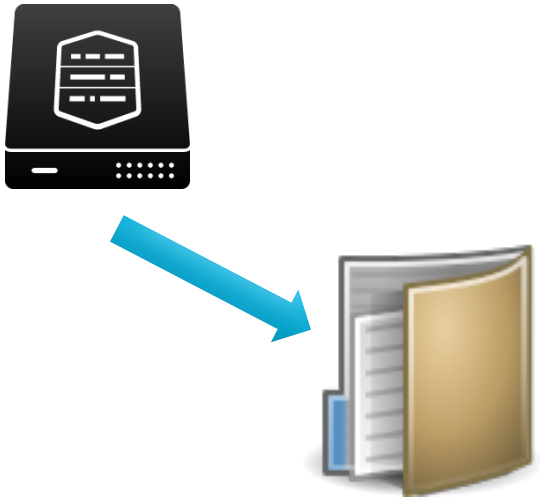
Freezing Buckets

- Buckets are rolled from “cold” to “frozen” when the index is taking up too much space
- **By default, this means the bucket gets deleted**
- In many cases this is fine, but what if:
  - You need fine-grained control to comply with data retention policies?
  - You want to minimize TCO by storing older data on cheaper hardware?



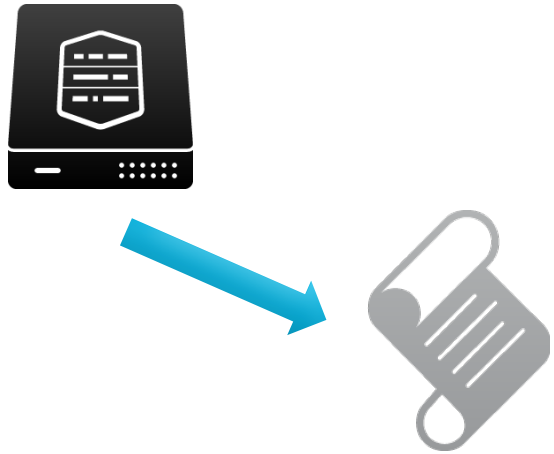
Alternative 1: Archive To File System

- Have Splunk copy frozen buckets to another directory
  - Directory specified in the index configuration via “coldToFrozenDir”
  - May be located on slow and/or inexpensive storage, possibly a NAS or SAN
- Advantages:
  - Easy to configure
- Disadvantages:
  - Archive location must be accessible via the file system
  - Frozen data is not searchable
    - Buckets can be made searchable again by copying (by hand) to a “thawed” directory for the index



Alternative 2: Archive Via Script

- Run an arbitrary script on buckets as they are rolled from cold to frozen
  - Script specified in the index configuration via “coldToFrozenScript”
  - Script can perform any desired action, e.g. encrypt data, scp data to remote system, send to tape backup system
  - Splunk will respect the return value of the script: will only delete the bucket following a successful script execution
- Advantages:
  - Extremely flexible
- Disadvantages:
  - User has responsibility for maintaining the script and guaranteeing its reliability
  - Frozen data is still not searchable

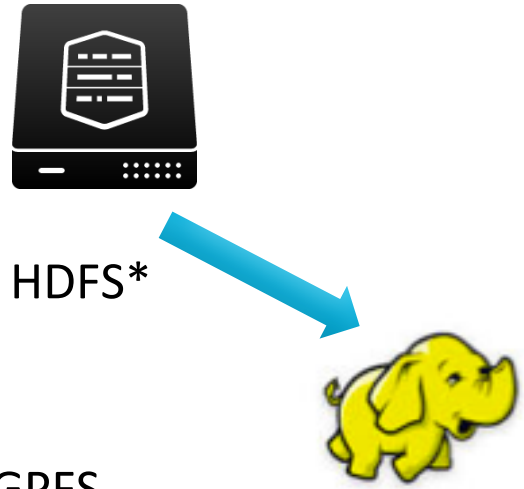


Hunk Archive Indexes

splunk>

Alternative 3: Hunk Archiving To Hadoop

- Archiving to Hadoop available with Hunk
- Extremely scalable
- Data remains searchable, using Hunk

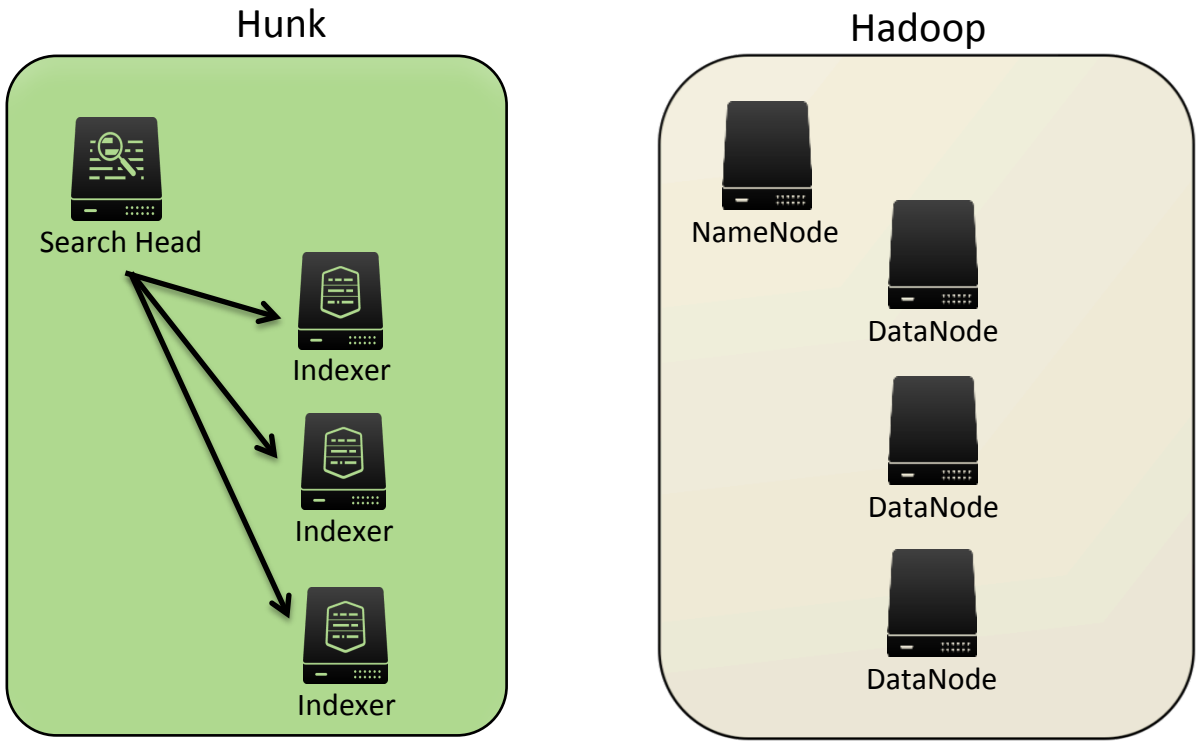


Keep the same interface and use the same queries you already use to search your data, but store your data on HDFS* instead of on your indexers

*Or any other Hadoop-compatible file system (e.g. S3, GPFS, CleverSafe)

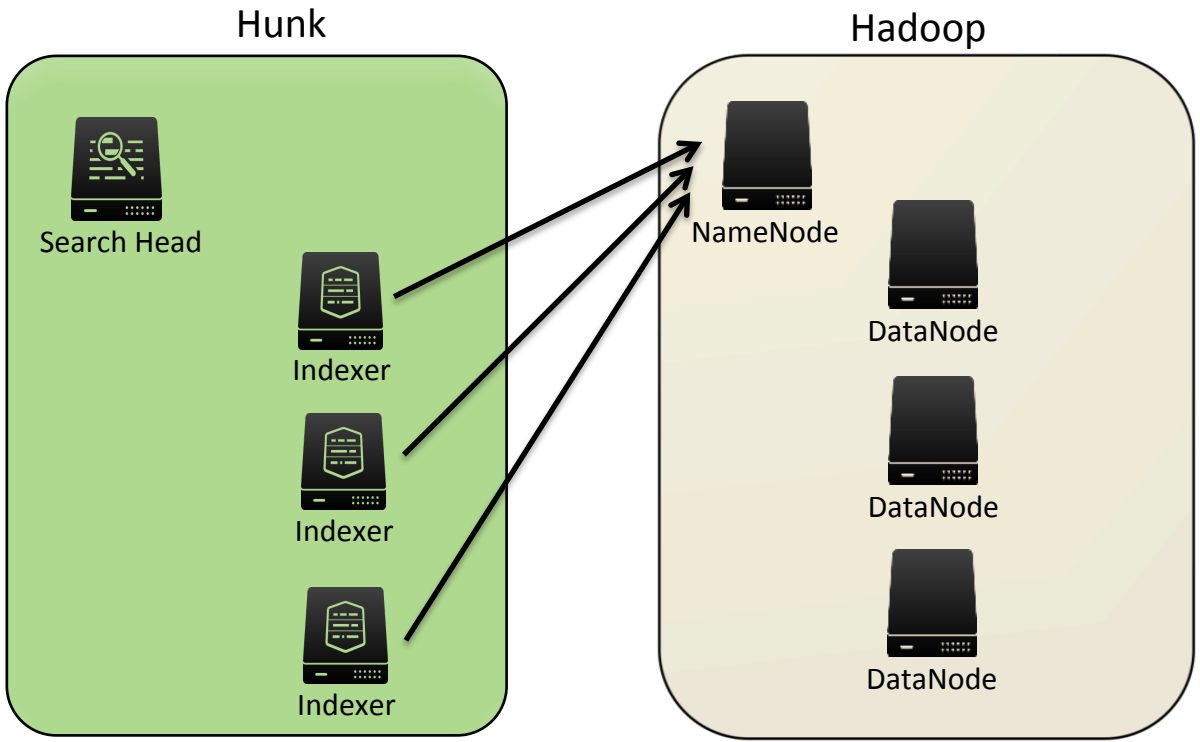
Splunk Archiving To Hadoop

Once per hour, the Search Head tells all the indexers to archive to Hadoop



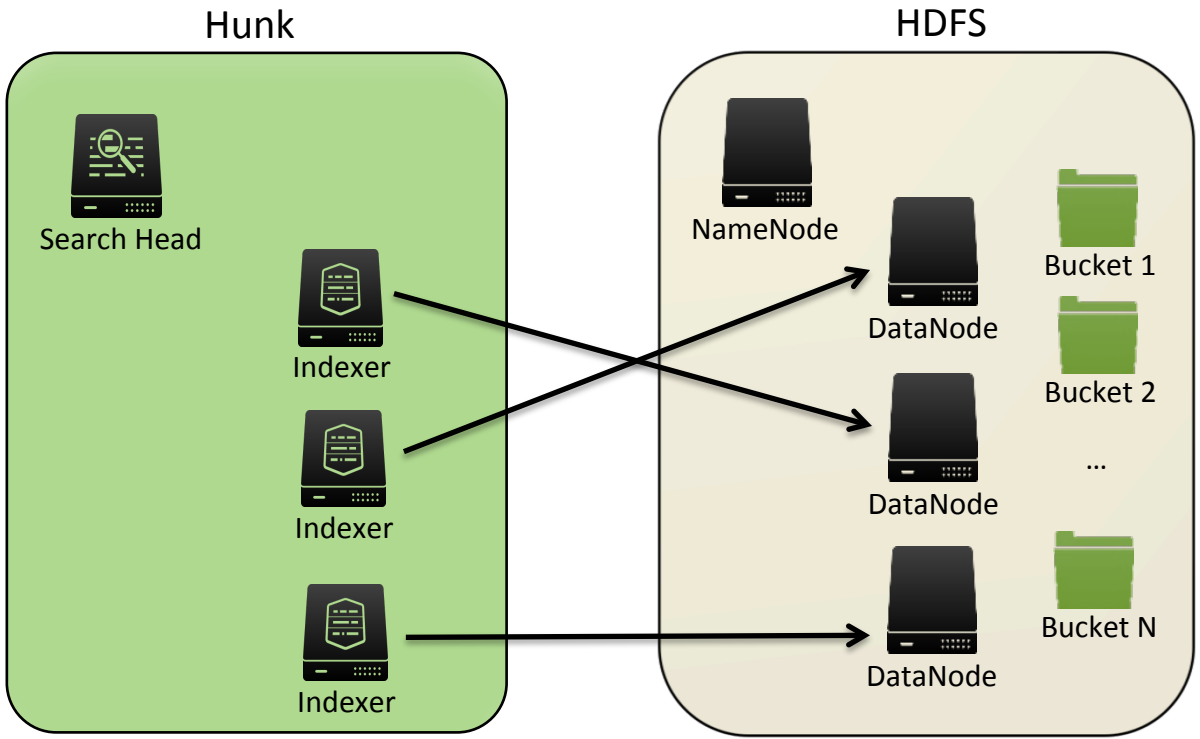
Splunk Archiving To Hadoop

The indexers look for buckets that qualify for archiving. They compute an appropriate HDFS path, and ask the name node where to write to.



Splunk archiving to Hadoop

Buckets are written to HDFS, with appropriate replication.



What Gets Copied?

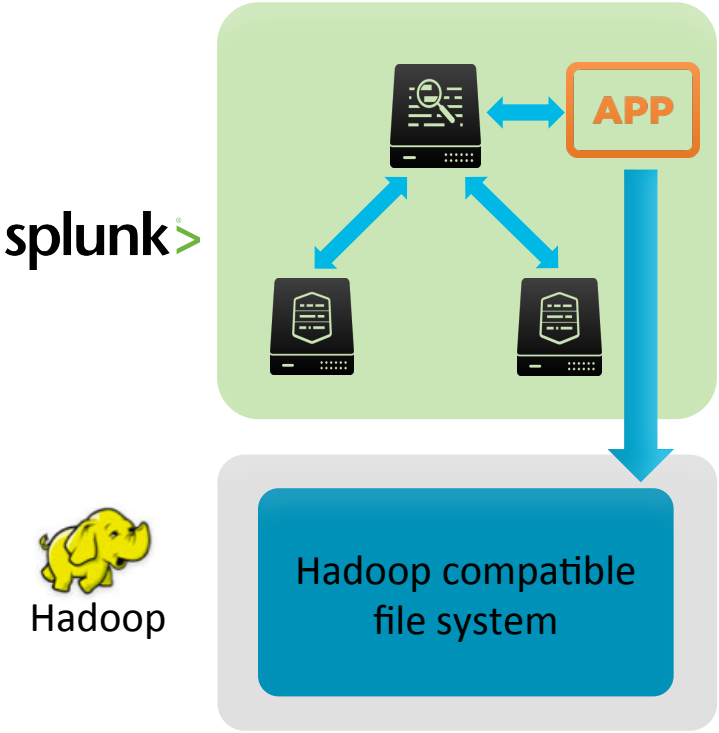
- The compacted raw data, in the format in which it's stored in the bucket (journal.gz files)
- The timestamps of the earliest and latest events in the bucket are encoded in the path, to speed up later search
- Auxiliary files (e.g. tsidx, bloom filters) are not copied
  - Less suitable to Hadoop searches, and take up a lot of space

What About Hadoop Connect?

HC is an older solution implemented as an app. Runs a query, and writes them to HDFS. But this can be very resource intensive:

- Network:
  - Data is first transferred from the indexers, then to HDFS
    - Data is transferred twice
    - Makes SH server a network bottleneck
- CPU:
  - All events must be parsed from the indexes, then transformed into final storage format

As data sizes increase, so does the load on the Search Head.



Archiving Scalability

- Each indexer transfers data directly to Hadoop: no single machine bottleneck, scale with number of indexers
  - Note: You can configure the maximum bandwidth used per indexer so as to not max out your network
- Raw data files (journal.gz) bulk copied as is
  - No need to parse events with
  - No need to transform final format
- Copy only data necessary to support later searches

Configuring An Archive Index

In indexes.conf, add a stanza like this:

```
[foo_archive]
vix.output.buckets.from.indexes = foo
vix.output.buckets.older.than = 86400 # age in seconds
vix.output.buckets.path = /hdfs/path/to/archive
vix.provider = hadoop_provider
```

Optional parameter (N is MB / sec, setting to 0 means no limit):

```
vix.output.buckets.max.network.bandwidth = N
```

(Optional) Add this to the to the stanza of the original index (in this example, foo):

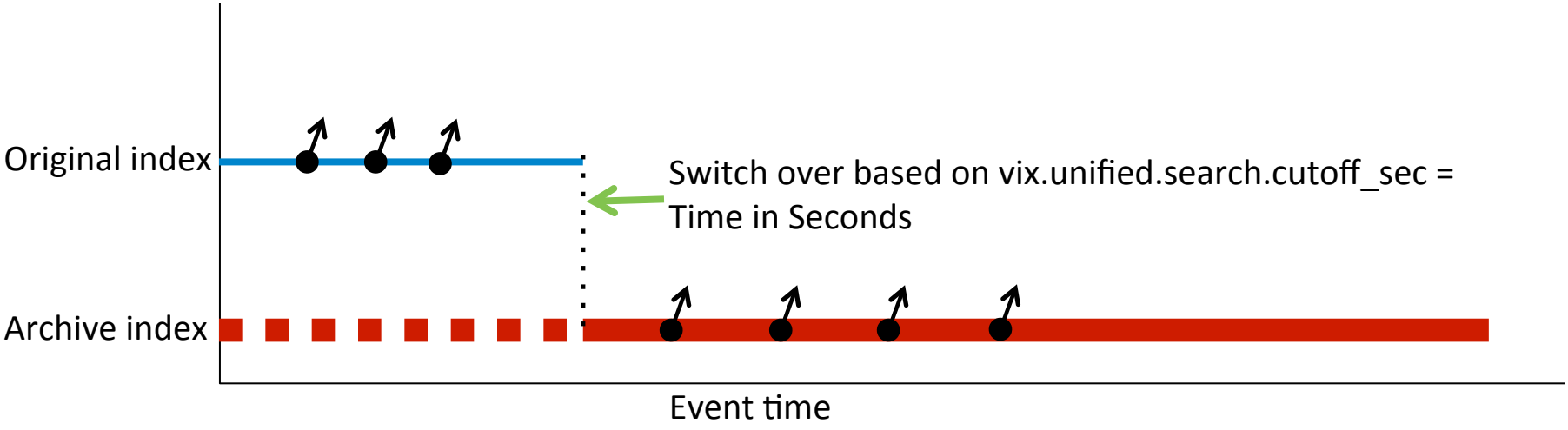
```
coldToFrozenScript = "$SPLUNK_HOME/etc/apps/splunk_archiver/bin/coldToFrozen.sh"
```

Is An Archive Index Searchable?

- Yes! If “foo_archive” is an archive index of “foo”, then:
  - Count the number of events in original index: “index=foo | stats count”
  - Count the number of events in the archive: “index=foo_archive | stats count”
- Remember that the original and the archive indexes will have overlapping events, so using both in the same query may produce duplicates
  - Which is why we’ve introduced…

Unified Search

- New for Hunk 6.3
- Transparently searches across both archive and original indexes, without duplicates
  - E.g. “index=foo | stats count” will give the total number of unique events currently in foo, or in foo_archive, with no double-counting
  - Requires the index be explicitly identified: will not work with index=* or index!=foo.



Configuring Unified Search

Enable unified search on the installation: In limits.conf, add this stanza:

```
[search]
unified_search = true
```

Then for each archive index for which you want to allow unified search, add this to its stanza:

```
[foo_archive]
…
vix.unified.search.cutoff_sec = N
```

Where N is the number of second before search time (i.e. before “now”) that you want the cutoff from the archive index to the original index to occur

Splunk Archive Bucket Reader

splunk>

- Problem:
  Archived buckets are in a Splunk proprietary format. What if you need to integrate with 3rd party software systems?

- Answer:
  Splunk Archive Bucket Reader

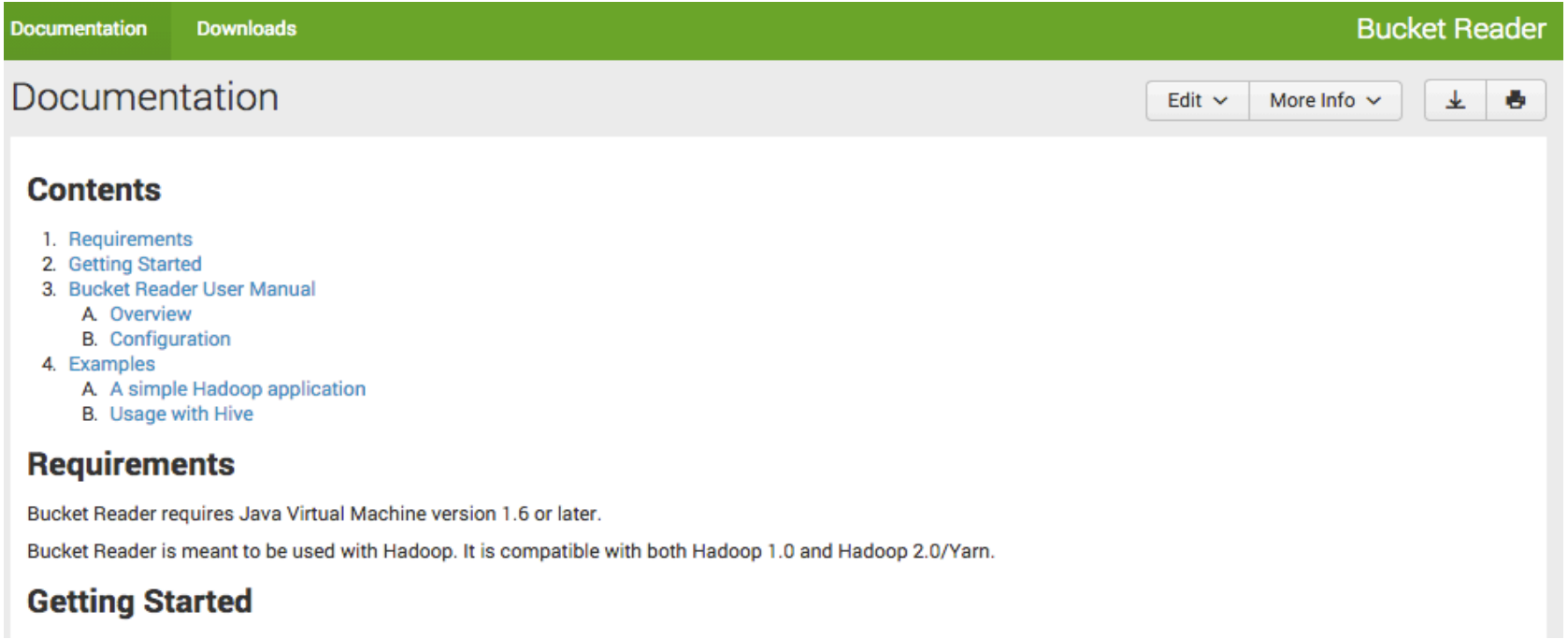
Packaged as a Splunk app. Available on https://splunkbase.splunk.com as “Bucket Reader”.

Capabilities

- Implements Hadoop InputFormat (and related classes) to allow any Hadoop-based application to read archived buckets.
- Provides the raw data of the original event, plus all indexed fields
  - NOTE: Does NOT provide access to search-time extracted fields.
- Use as a plug-in for custom applications, Hive, Pig, etc.
- Use for real-time querying, or transform archived buckets into new format.

Bucket Reader App Includes Full Documentation

Includes javadoc, sample application, usage examples, etc.



Hive Example

```
hive> CREATE TABLE splunk_event_table (
    >     Time              DATE,
    >     Host              STRING,
    >     Source            STRING,
    >     date_wday         STRING,
    >     date_mday         INT
    > )
    > ROW FORMAT SERDE 'com.splunk.journal.hive.JournalSerDe'
    > WITH SERDEPROPERTIES (
    >     "com.splunk.journal.hadoop.value_format" = "_time,host,source,date_wday,date_mday"
    > )
    > STORED AS INPUTFORMAT 'com.splunk.journal.hadoop.mapred.JournalInputFormat'
    > OUTPUTFORMAT 'org.apache.hadoop.hive.ql.io.IgnoreKeyTextOutputFormat';
OK
Time taken: 0.243 seconds
hive> LOAD DATA LOCAL INPATH './journal.gz' OVERWRITE INTO TABLE splunk_event_table;
Loading data to table default.splunk_event_table
Table default.splunk_event_table stats: [numFiles=1, numRows=0, totalSize=37779465, rawDataSize=0]
OK
Time taken: 2.465 seconds
hive> select * from splunk_event_table limit 10;
OK
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
2012-10-05      host::mchheda-hunktest-01.sv.splunk.com source::/home/mchheda/p4/splunk/current/new_test/data/access_combined_1m_events.log     friday  5
Time taken: 0.756 seconds, Fetched: 10 row(s)
hive>
```

Conclusions

splunk>

Multiple Options Available

Choose the option that suits your data retention needs and constraints.

- Cold-to-frozen path
  - Use cheap attached storage
  - Very easy to configure
- Cold-to-frozen script
  - Maximum flexibility
  - Leaves the work to the administrator
- Archive to Hadoop
  - Move data to Hadoop
  - Can search using Hunk
  - With Splunk Archive Bucket Reader, very flexible
  - Extremely scalable

THANK YOU

splunk>