Behind the Magnifying Glass

How Search Works

Jeff Champagne | Principal Architect, Splunk

Thursday, October 4th, 2018 | Walt Disney World

Who's This Dude?

Jeff Champagne

Principal Architect

jchampagne@splunk.com

- Started with Splunk in the fall of 2014
- Member of the Splunk Architecture Council
- Former Splunk customer in the Financial Services Industry
- Lived previous lives as a Systems Administrator, Engineer, and Architect
- Loves Skiing, traveling, photography, and a good Sazerac

10s, 10s, 10s Across the Board!

Rate My Session Please

Am I in the right place?

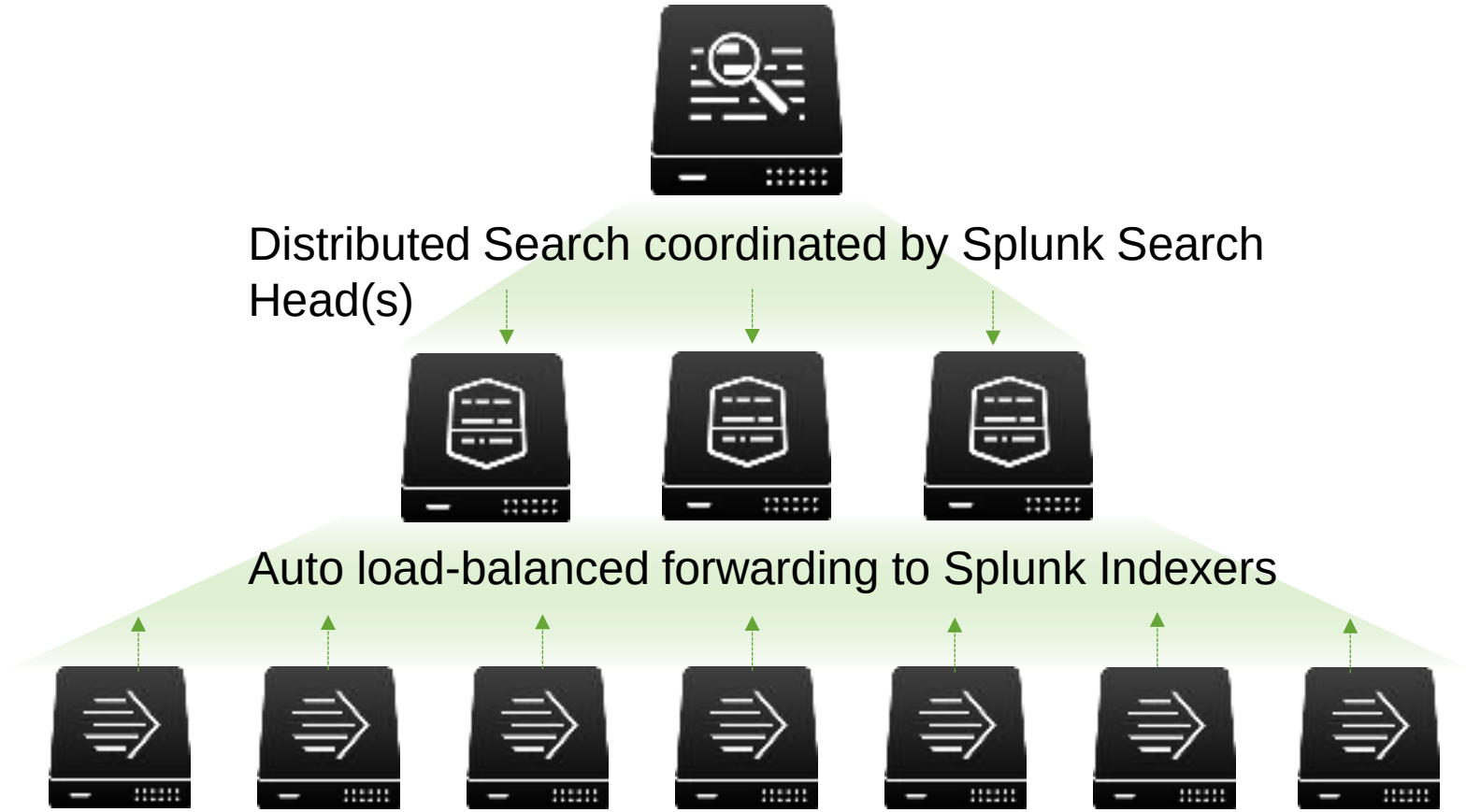
Some familiarity with…

- Splunk Components
  - Search Head, Indexer, Forwarder
- Splunk Search Interface
- Search Processing Language (SPL)

What Will I Learn?

- What is going on when you click search
- How to improve searches so they run faster
  - Splunk Architecture Overview
  - How Splunk stores events
  - Components of a search
  - Search tips and SPL command alternatives
  - Search command examples

Splunk Enterprise Architecture

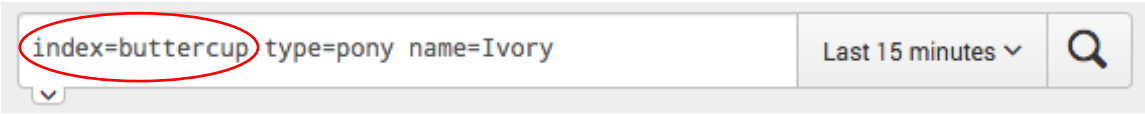


Send data from thousands of servers using any combination of Splunk forwarders

Index vs. Index

An Overloaded Term

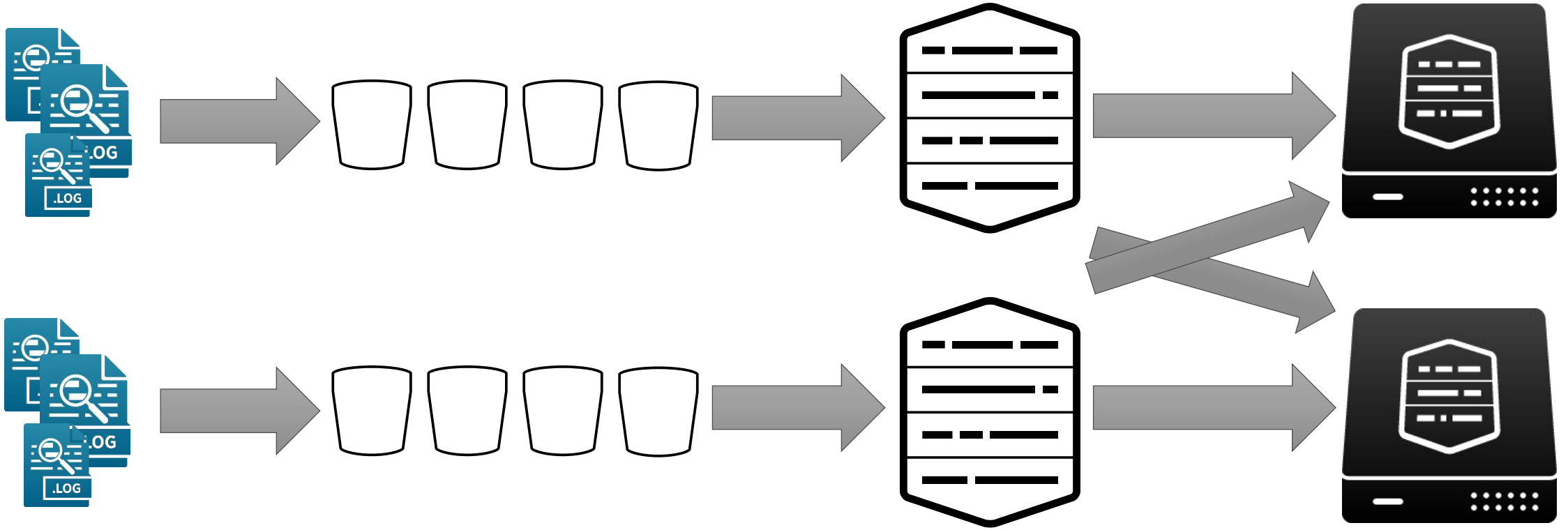
- Logical grouping for data
  - You or your Splunk admin create these
  - You reference these in your searches
    - Implicitly or explicitly



- TSIDX File
  - Time-series Index
  - Splunk's "secret sauce"
  - A logical Index is made up of many indexes/TSIDX files
  - This is how we search for your data
    - More on this later…

How Are Events Stored?

Buckets, Indexes, and Indexers



Events Buckets Indexes Indexers

(Logical Grouping)

How Are Events Stored?

Bucket Aging Process – Classic Mode

Hot/Warm Storage

- Fast Storage
- Recent data

Cold Storage

- Slower “bulk” storage
- Older data

Archive Storage

- Historical/Compliance data
- Online (searchable)/Offline

How Are Events Stored?

Bucket Aging Process – Smart Store Enabled

Hot/Cache Storage

- Fast Storage
- Recent (hot) data
- Cached data

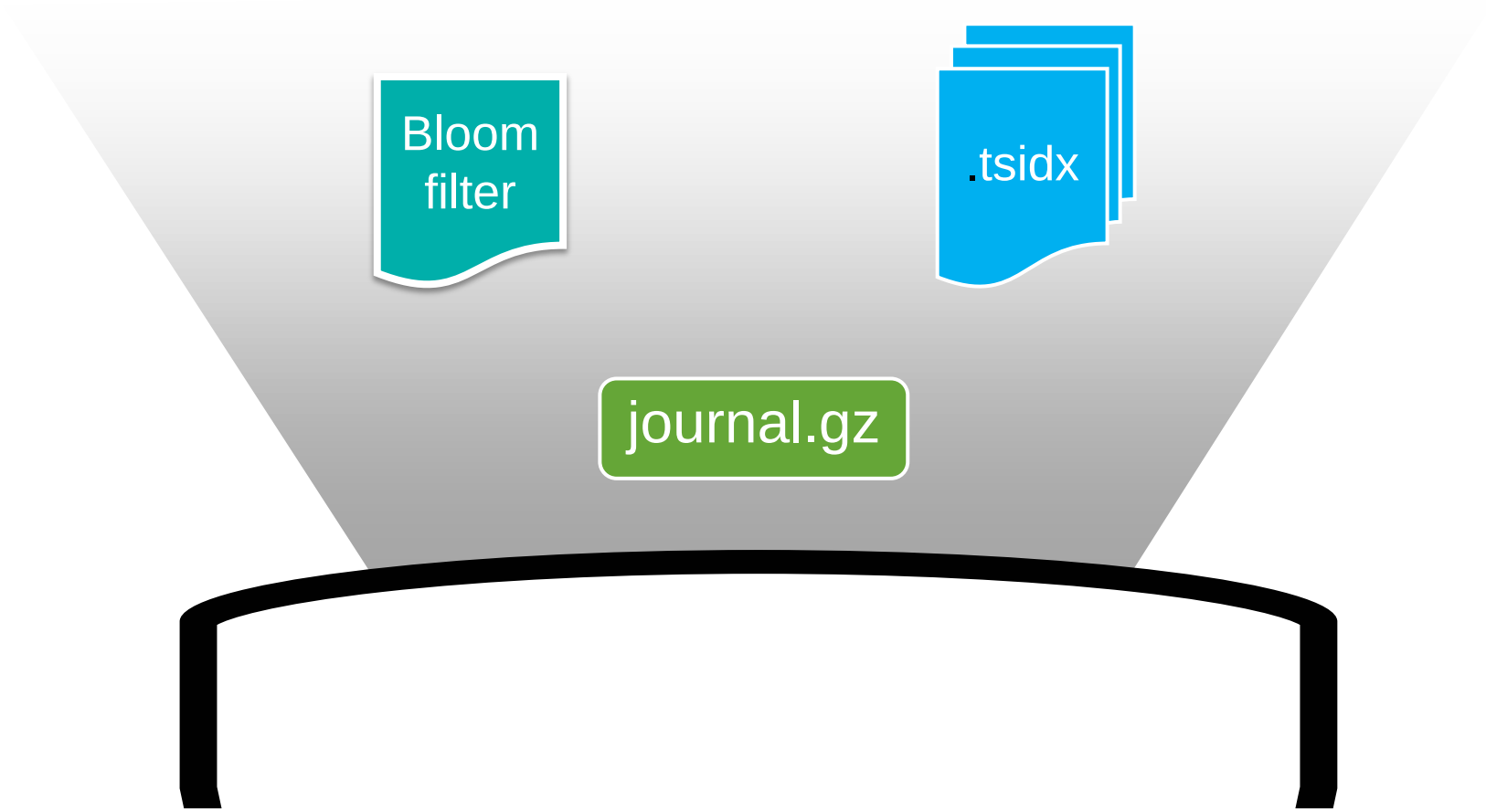
Object Storage

- Slower “bulk” storage
- All Non-Hot buckets

Archive Storage

- Historical/Compliance data
- Online (searchable)/Offline

What's in a Bucket?



What's in a Bucket?

Journal.gz

- Your events go here
- Journal.gz is made up of many smaller compressed slices
- Raw data is collected and saved into slices
  - ~128KB of uncompressed data make up a slice

splunk> .conf18

What's in a Bucket?

TSIDX

Raw Events
Jim likes Mickey
Suzie likes Donald
Pat likes Pluto

Unique terms from the raw events are written to the lexicon

Lexicon

Term	Postings List
Donald	1
Jim	0
likes	0,1,2
Mickey	0
Pat	2
Pluto	2
Suzie	1

The postings list tells us where we can find a specific term in the values array

Postings List

Posting Value	Seek Address
0	34
1	87
2	132

The seek address tells us where we can find the matching event(s) in the journal.gz slices

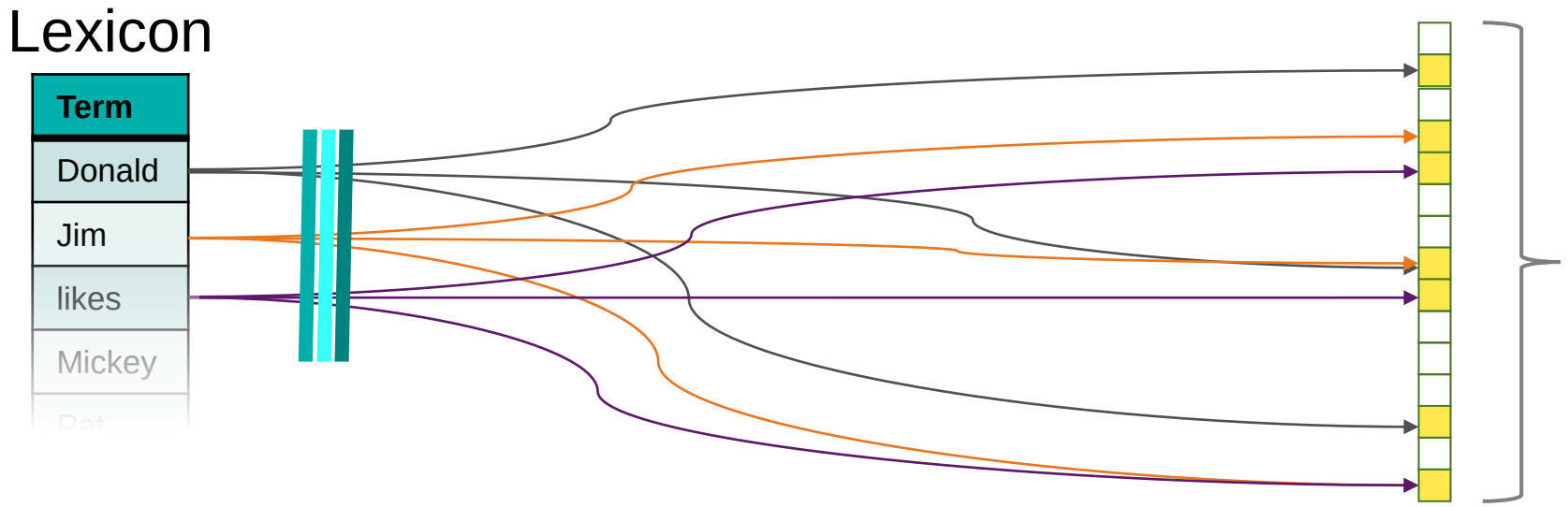
*The overall structure of a TSIDX file has been simplified for illustrative purposes

splunk> .conf18

What's in a Bucket?

Bloom Filter

- Determines whether a term is likely to exist in the TSIDX of a bucket
  - False positives are possible, false negatives are not
  - Interactive Example: https://www.jasondavies.com/bloomfilter/



- Regardless of the # of terms, bit array size remains fixed
- Binary format
- Fast to read vs. TSIDX, which grows with more unique terms

Each term from the lexicon is run through a set of hashing algorithms

The output of each hash sets a bit in the array to ON

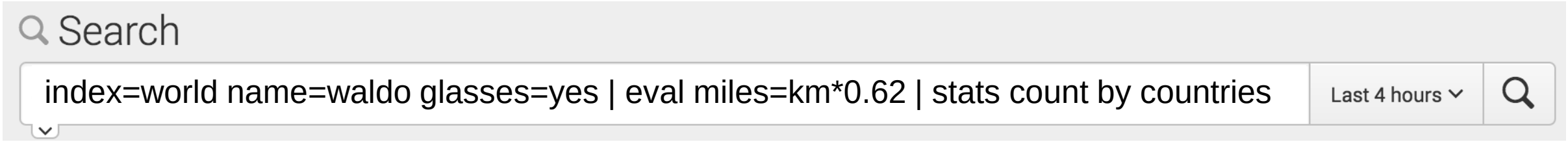
How Search Works…

An Example

splunk> .conf18

How Search Works

Components of a Search String



Base Search

Retrieves & filters events

SPL Commands

Evaluate, transform, and format events

Events are retrieved → Results move linearly through SPL commands

How Search Works

Where's Waldo?

index=world name=waldo 

How Search Works

Where's Waldo?

1

Search

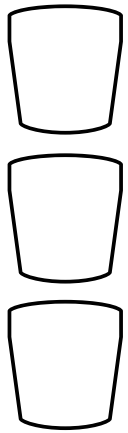
index=world name=waldo

Last 4 hours

2 Hash the value waldo to create a bloom filter for our search

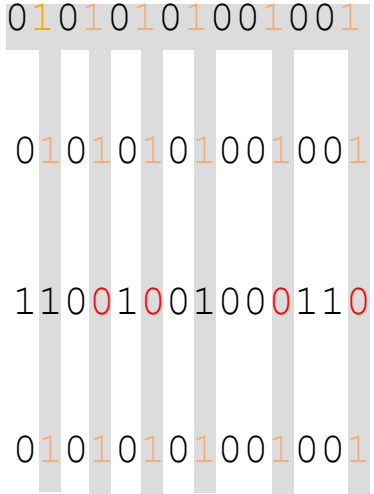
0101010101001001

3 Begin searching world buckets containing events from the Last 4 hours



4 Compare our filter to the one in each bucket

Bloom filter



5 Locate the value waldo in the TSIDX

.tsidx

find 0,1,3
Waldo 1
looking 0,1,2,4

The, 0,1,2,3,5,6
individual 0,2,4
you 0,1,2,3,4,5
are 1,2,5,6

Yeah 0,2,4
Waldo 0,3
comes 0,2,3,4,5

6 Retrieve events with waldo using the seek address in the TSIDX

journal.gz

I have been trying to find Waldo looking all over these books. I'm not sure I'll

The individual you are looking for does not exist in this dataset. We banished him. He isn't welcome.

Oh yeah, Waldo comes in this joint all the time. The last time I saw him was probably 6 months ago. He was wearing a fur coat from a bear that killed his brother.

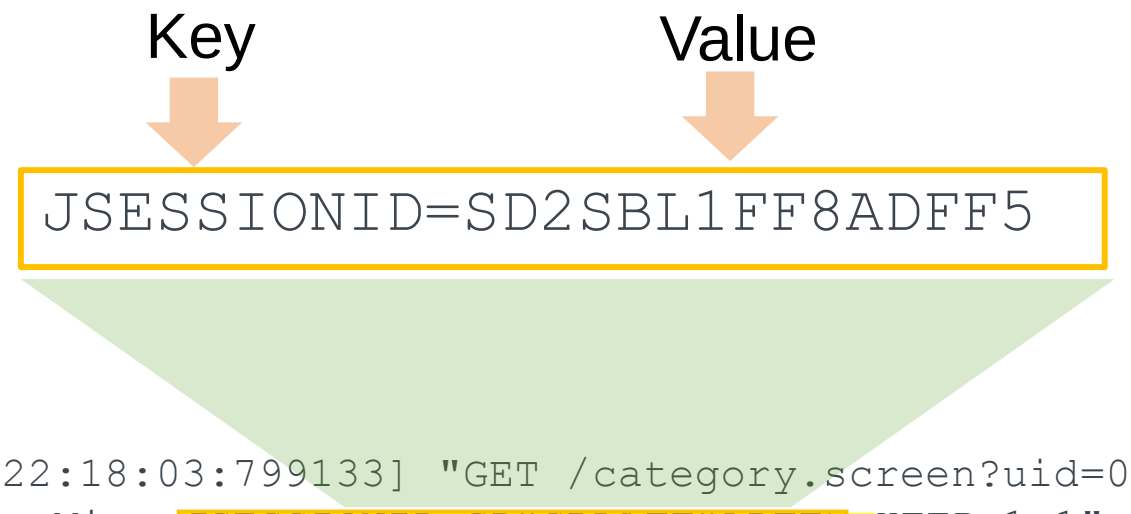
*The internal structure of Bloom filters, TSIDX, and Journal files has been simplified for illustrative purposes

How Search Works…

Schema on the Fly

splunk> .conf18

Schema on the Fly




```
147.31.14.76 - - [04/Sep/2018 22:18:03:799133] "GET /category.screen?uid=00b11ba0-3d81-
4195-8789-8d9c1fff1d8a&category=Misc&JSESSIONID=SD2SBL1FF8ADFF5 HTTP 1.1" 404 2480
"http://www.buttercupenterprises.com/category.screen?uid=00b11ba0-3d81-4195-8789-
8d9c1fff1d8a&category=Misc" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_2)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/56.0.2914.3 Safari/537.36 OPR/43.0.2431.0
(Edition developer)" 97
```

How Search Works…

Distributed Search

splunk> .conf18

How Search Works

Distributed Search

1. Search Head parses search into distributed and centralized parts
2. Distributed parts of search are sent to indexers
3. Indexers fetch events from disk
4. Schema is applied to events (Schema-on-the-fly)
5. Events are filtered based on KV pairs
6. Distributed commands are applied
7. Results are sent to Search Head
8. Search Head applies centralized streaming & transforming commands, then displays results

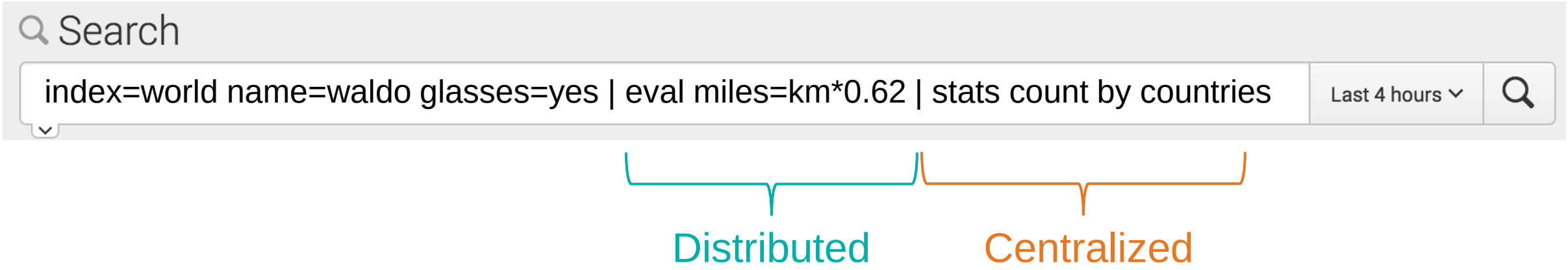
How Search Works

Types of Search Commands

- **Streaming Commands**
  - Distributable (Remote Streaming)
    - Operate on individual events
    - Run on indexers (distributed)
    - Ex: eval, rex, where, rename, fields…
  - Centralized (Stateful Streaming)
    - Operate on at least a sub-set of the entire result set
    - Run on Search Head (centralized)
    - Ex: head, streamstats
- **Transforming Commands**
  - Create a reporting data structure
  - Operate on the entire event set
    - Non-streaming
    - Typically run on the search head
  - Ex: transaction, stats, top, timechart…

How Search Works

Command Ordering



Events are retrieved → Results move linearly through SPL commands

- Commands are processed in the order you write them
- Placing centralized or transforming commands before distributable commands may force unnecessary data and/or processing to the Search Head

Job Inspector

Demo

- Search Pipeline Rendering
  - Streaming Pipeline = remoteSearch
  - Stateful & Events Pipelines = eventsSearch
  - Stream Report & Report Pipelines = reportSearch
- Searches
  - Streaming command
    - index=_internal | eval myCurrentSize=current_size+100
  - Transforming command with distributable component
    - index=_internal | stats count by component
  - Streaming command AFTER transforming command
    - index=_internal | stats count by component | eval myCount=count*100

Commands in Action

Command Abuse

Fields vs. Table

Goal: Remove fields I don't need from results

BAD:

```
index=myIndex field1=value1 | table field1, field2, field4 | head 10000
| table field2, field4
```

GOOD:

```
index=myIndex field1=value1 | fields field1, field2, field4 | head 10000
| table field2, field4
```

- Table is a formatting command NOT a filtering command
  - If used improperly, it will cause unnecessary data to be transferred to the search head from search peers
- Fields tells Splunk to explicitly drop or retain fields from your results

Command Abuse

Search Term	Status	Artifact Size	# of Events	Run Time
| table	Running (1%)	624.93MB	2,037,500	00:02:44
| fields	Done	9.95MB	10,000	00:00:13

Command Abuse

Stats vs. Transaction

Goal: Group multiple events by a common field value

NOT GREAT:

```
index=mail from=joe@schmoe.com| transaction message_id | table _time, to,
from, subject, message_id
```

GOOD:

```
index=mail from=joe@schmoe.com | stats latest(_time) AS mTime values(to)
AS to values(from) AS from values(subject) AS subject BY message_id
```

- If you're not using any of the Transaction command parameters, the same results can usually be accomplished using Stats
  - startswith, endswith, maxspan, maxpause, etc…

Command Abuse

Joins & Sub-searches

Goal: Return the latest JSESSIONID across two sourcetypes

NOT GREAT:

```
sourcetype=access_combined | join type=inner JSESSIONID
[search sourcetype=applogs | dedup JSESSIONID
| table JSESSIONID, clienip, othervalue]
```

GOOD:

```
sourcetype=access_combined OR sourcetype=applogs
| stats latest(*) AS * BY JSESSIONID
```

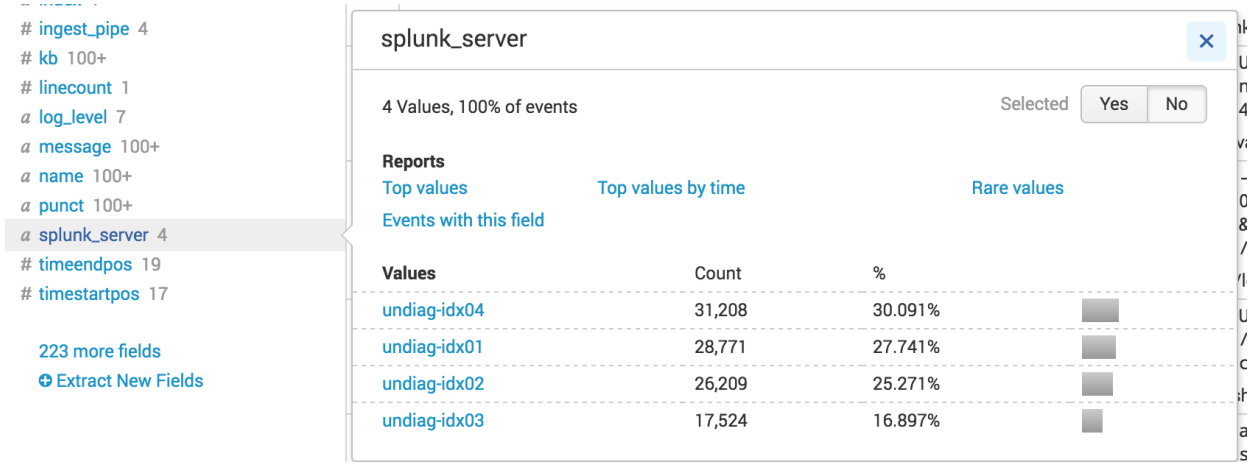
Search Tips

Just because you can…doesn't mean you should

Plan your search to leverage the power of Splunk!

Search Tips

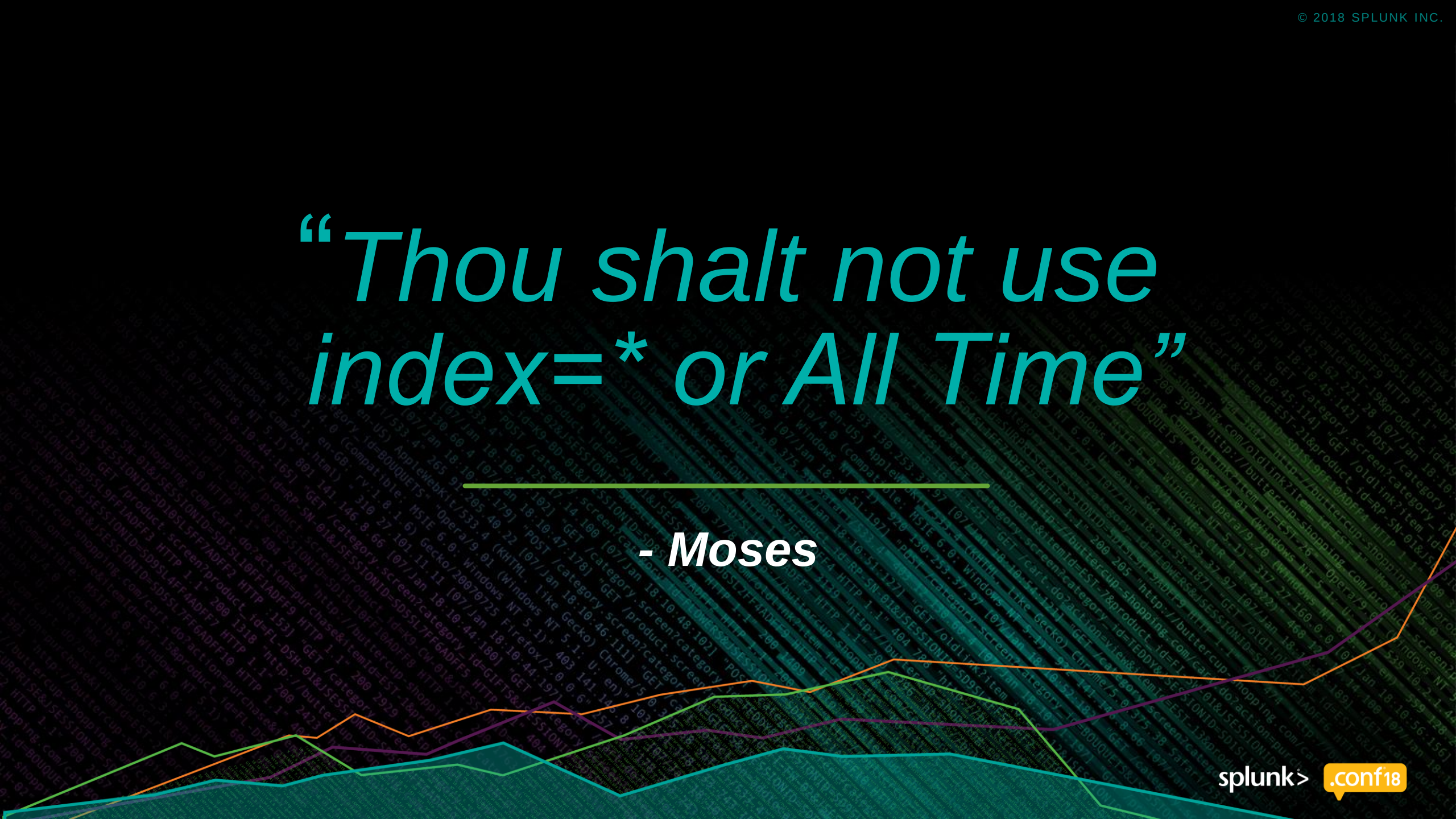
- Reduce the amount of data Splunk has to Search
  - Specify and limit the index(es)
  - Limit the time range
  - Search for values that are unique to your events where possible
    - Reduce the number of events filtered after schema-on-the-fly
- Distributed Search
  - Ensure events are well distributed
  - Place distributed commands before centralized commands



splunk> .conf18

“Thou shalt not use index= or All Time”*

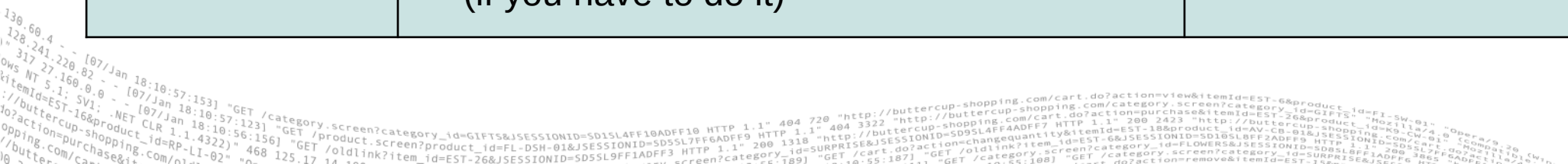
- Moses



splunk> .conf18

Search Tips

Avoid	Explanation	Suggested Alternative
All Time	• Events are grouped by time • Reduce searched buckets by being specific about time	• Use a specific time range • Narrow the time range as much as possible
index=*	• Events are grouped into indexes • Reduce searched buckets by specifying an index	• Always specify an index in your search
Wildcards	• Wildcards are not compatible with Bloom Filters • Wildcard matching of terms in the index takes time • Lexicon is structured by common prefixes, so appending an * is best (if you have to do it)	• Varying levels of suck-itude > myterm* → Not great > *myterm → Bad > *myterm* → Death • Use the OR operator i.e.: MyTerm1 OR MyTerm2



splunk> .conf18

Search Tips

Avoid	Explanation	Suggested Alternative
NOT !=	• Bloom filters & indexes are designed to quickly locate terms that exist • Searching for terms that don't exist takes longer	• Use the OR/AND operators (host=c OR host=d) (host=f AND host=h) vs. (host!=a host!=b) NOT host=a host=b
Verbose Search Mode	• Verbose search mode causes full event data to be sent to the search head, even if it isn't needed	• Use Smart Mode or Fast Mode
Real-time Searches	• RT Searches put an increased load on search head and indexers • The same effect can typically be accomplished with a 1 min. or 5 min. scheduled search	• Use a scheduled search that occurs more frequently • Use Indexed-Realtime searches (Set by Splunk admin)

splunk> .conf18

Search Tips

Avoid	Explanation	Suggested Alternative
Transaction	• Not distributed to indexers • Typically only needed if using additional parameters (maxSpan, startsWith, etc…)	• Use the stats command to link events where possible
Joins/Sub-searches	• Joins can be used to link events by a common field value, but this is an intensive search command	• Use the stats (preferred) or transaction command to link events
Search after first |	• Filtering search results using a second "| search" command in your query is inefficient	• As much as possible, add all filtering criteria before the first | i.e.: >index=main foo bar vs. >index=main foo | search bar

The TERM Directive

Why does it matter?

- Splunk breaks terms by Major and Minor Segmenters
  - When writing to the TSIDX and searching
  - Default minor segmenters:
    `/ : = @ . - $ # % \\ _`
- TERM prevents breaking on Minor segmenters

`[ AND 0 10 6 index::myindex ]`

Raw Events
10.0.0.6
9/28/2016
jeff@splunk.com

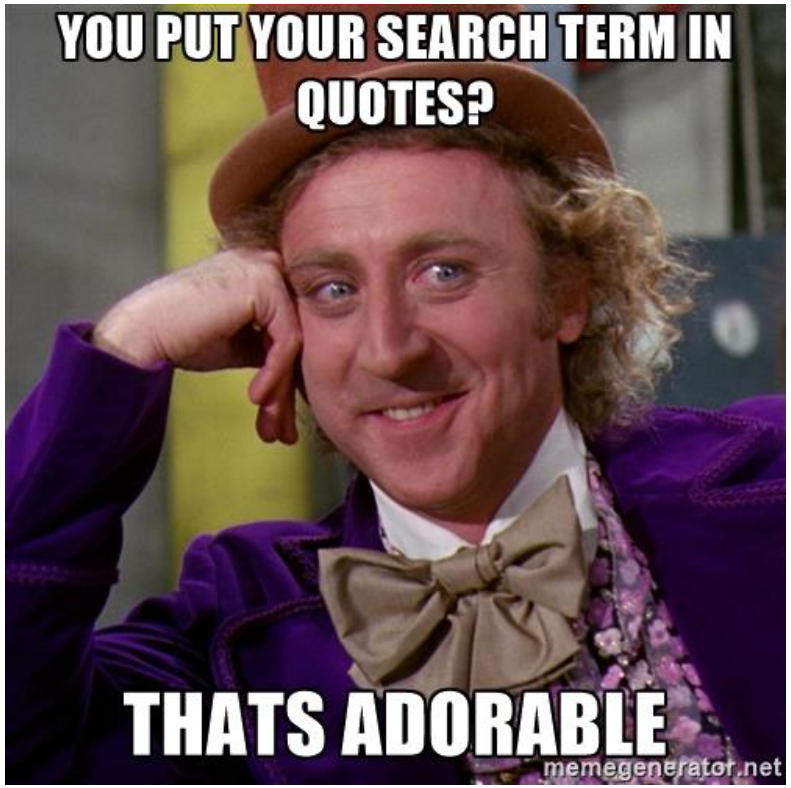
Lexicon

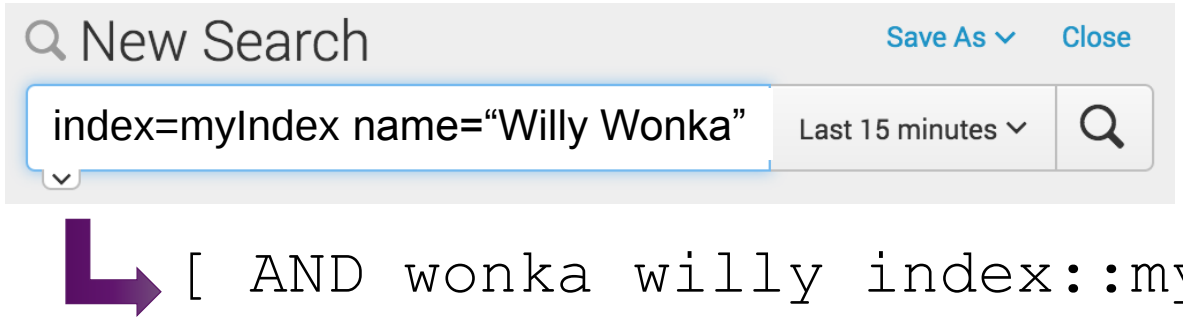
Term	Postings List
0	0
6	0
9	1
10	0
28	1
2016	1
10.0.0.6	0
9/28/2016	1
com	2
jeff	2
splunk	2
jeff@splunk.com	2

The TERM Directive

What about quotes?

- TERM controls how we search the lexicon and which events are retrieved from disk
- Quotes can help filter after the events are retrieved from disk
- Use quotes when the *value* in your key-value pair has major breakers

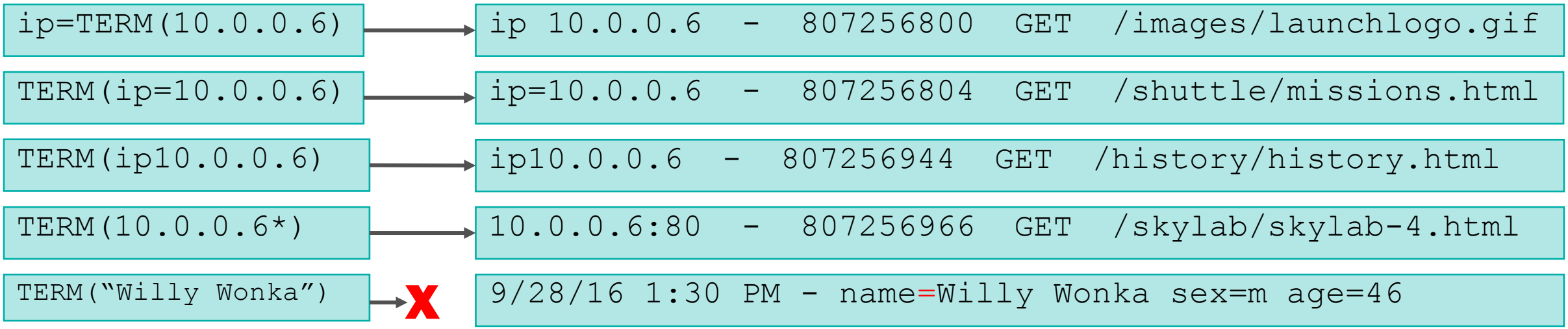





```
[ AND wonka willy index::myindex ]
```

The TERM Directive

How do I use it?



- Your term MUST be bounded by major segmenters
  - Example: Spaces, tabs, carriage returns
    - See Segmenters.conf spec for full details
  - Your term cannot contain major segmenters

Search Tips

Indexed Extractions

- Special Key-Value pairs that are stored in the TSIDX file
- Default Extractions
  - source, host, sourcetype
  - Use these whenever possible
- TSTATS
  - Super-fast command
  - Doesn’t search or return raw data
  - Can be used on report/data model accelerations AND indexed extractions

Resources

- Splunk Docs
  - Write Better Searches
    http://docs.splunk.com/Documentation/Splunk/latest/Search/Writebettersearches
  - Wiki: How Distributed Search Works
    http://wiki.splunk.com/Community:HowDistSearchWorks
  - Splunk Search Types
    http://docs.splunk.com/Documentation/Splunk/latest/Capacity/HowsearchtypesaffectSplunkEnterpriseperformance
  - Search Commands by Type (Centralized vs. Distributed)
    http://docs.splunk.com/Documentation/Splunk/latest/SearchReference/Commandsbytype
  - Blog: When to use Transaction and when to use Stats
    http://blogs.splunk.com/2012/11/29/book-excerpt-when-to-use-transaction-and-when-to-use-stats/
  - Segmenters.conf Spec
    http://docs.splunk.com/Documentation/Splunk/latest/Admin/Segmentersconf
  - Splunk Book: Exploring Splunk
    http://www.splunk.com/goto/book
- How Bloom Filters Work: An Interactive Demo
  https://www.jasondavies.com/bloomfilter/

Questions?

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