Splunk Performance Reloaded
Best Practices For Optimal Performance

Stefan Sievert

Staff Architect, Splunk Inc.

.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

- Setting The Stage, Why Is This Important
- Collection/Forwarding
- Indexing
- Search
- UI/Dashboards
- Summary

Setting The Stage

.conf2016

splunk>

The Goal

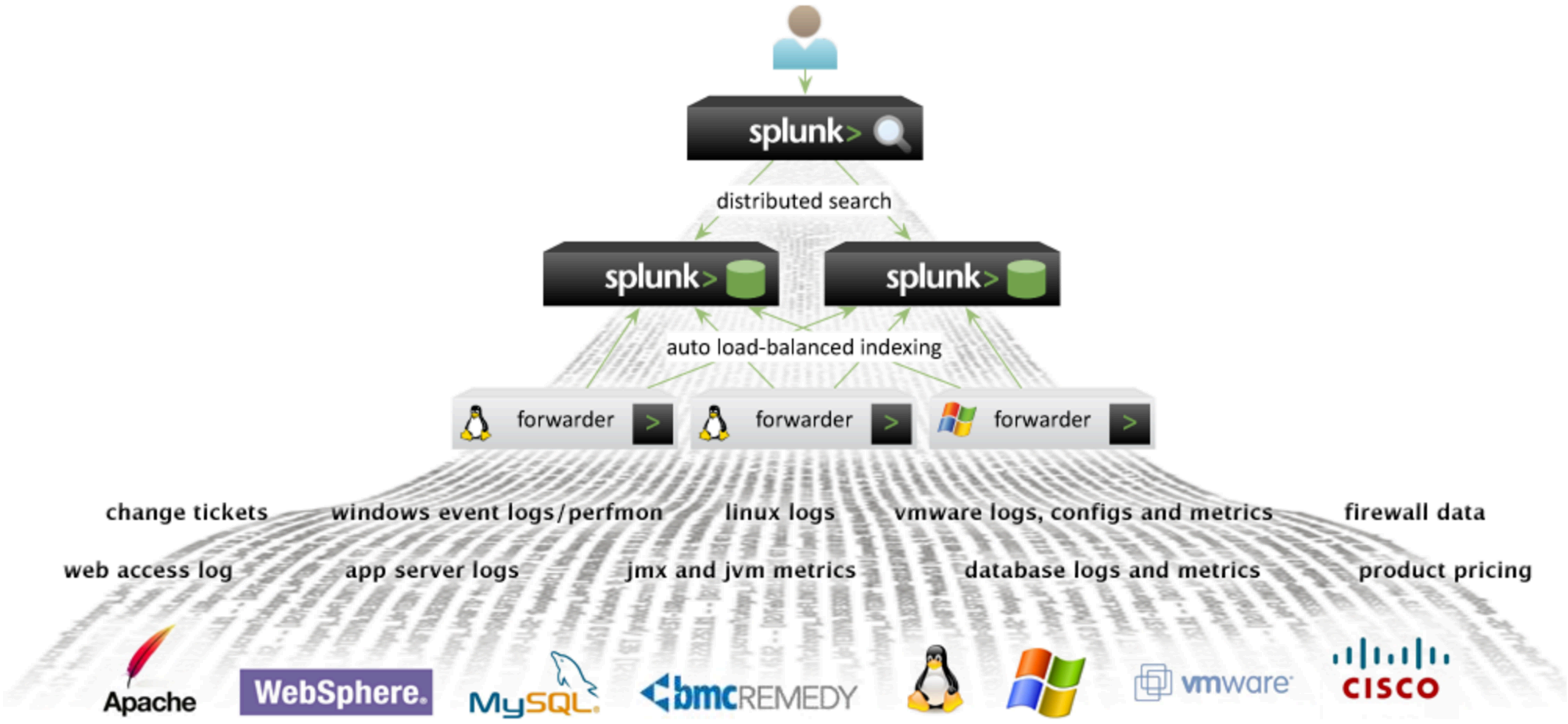
> "Let our advance worrying become advance thinking and planning."
>
> - *Winston Churchill*

General Architecture Considerations

.conf2016

splunk>

A Quick Refresher



Architecture Considerations

- Remember: Indexers are search peers and handle the bulk of the search workload
  - More indexers = less data per indexer = higher concurrency = more searches per time unit
  - Indexer processing capacity needs to be > SH capacity, top-heavy deployments can overwhelm the search peers
- Address search performance issues at the search peer tier first, i.e. when in doubt, add an indexer
- Avoid complex architectures, keep it simple (intermediary forwarders, over-building for every failure scenario, etc.)

Collection/Forwarding

.conf2016

splunk>

Collection/Forwarding Performance

- Forwarder configuration can affect…
  - **Event distribution** across indexers, which negatively affects search performance
    - High-velocity log source can cause stickiness (see: **forceTimebasedAutoLB**)
  - **Event throughput**, which may affect index time latency, causing events to not be searchable for extended periods of time
    - UF has Default MaxKBps of 256kbps
    - Keep number of monitored sources low
    - New in 6.4: **parallelIngestionPipelines** (server.conf)
- Use UF vs. HF for intermediary FWD tier if possible
- Consider HTTP Event Collector for forwarder-less collection

Indexing

.conf2016

splunk>

Indexer Resources

- Storage performance is single most critical factor
  – Splunk doesn't care which supported storage technology you use as long as it meets minimum IO performance requirements
  – Locally attached storage almost always wins over shared SAN
- Indexing itself is streaming write IO, but indexers do double duty!
  →Random Seek performance is critical for searching
- Slow storage for COLDDB can slow down indexing
- Indexers need resources (cores, memory, IO); constrained resources are the #1 cause for performance issues

Recommended Approach

- Separate HOT/WARM from COLD and limit HOT/WARM to the minimum required to fulfil ~80% of your search use cases

  This allows you to economically use SSDs for HOT/WARM and cheaper storage for the remaining search use cases (assuming search performance is less critical there)

Indexer Configuration

- Keep number of indices reasonable, create new index to address retention and access control requirements
- Separate high-velocity log sources from low-velocity sources
- Take advantage of parallel indexing pipelines if you can
- Combine things frequently searched together in the same index

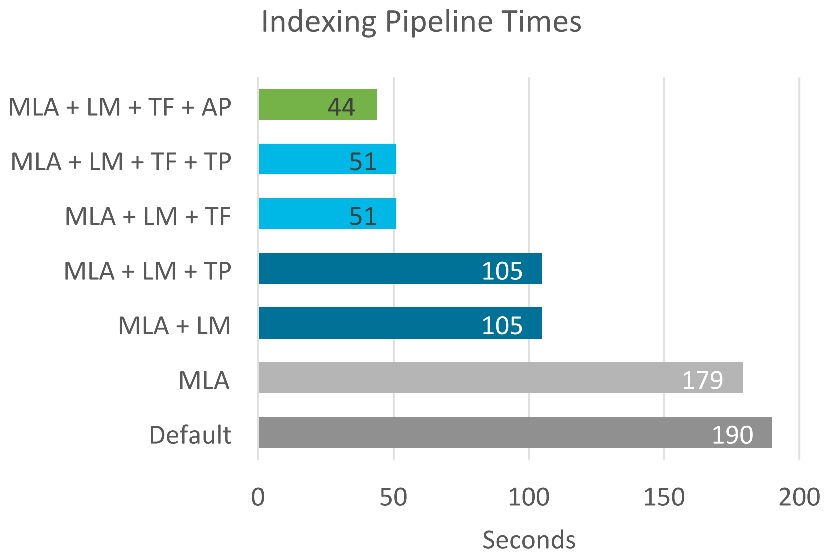
- Turn that Hyperthreading ON, it does not hurt!
- Turn CPU power-safe OFF!
- If you are on RedHat Linux, turn THP off!

“Although the default Splunk configurations are typically appropriate, certain high-performance environments can benefit from tuning various parameters.”

- John F. Kennedy

Data Source Configuration

- For each sourcetype, always set:
  - TIME_FORMAT (TF)
  - TIME_PREFIX (TP)
  - MAX_TIMESTAMP_LOOKAHEAD (MLA)
  - LINE_BREAKER
  - TRUNCATE
  - SHOULD_LINEMERGE=false
  - ANNOTATE_PUNCT=false (AP)



Searching

.conf2016

splunk>

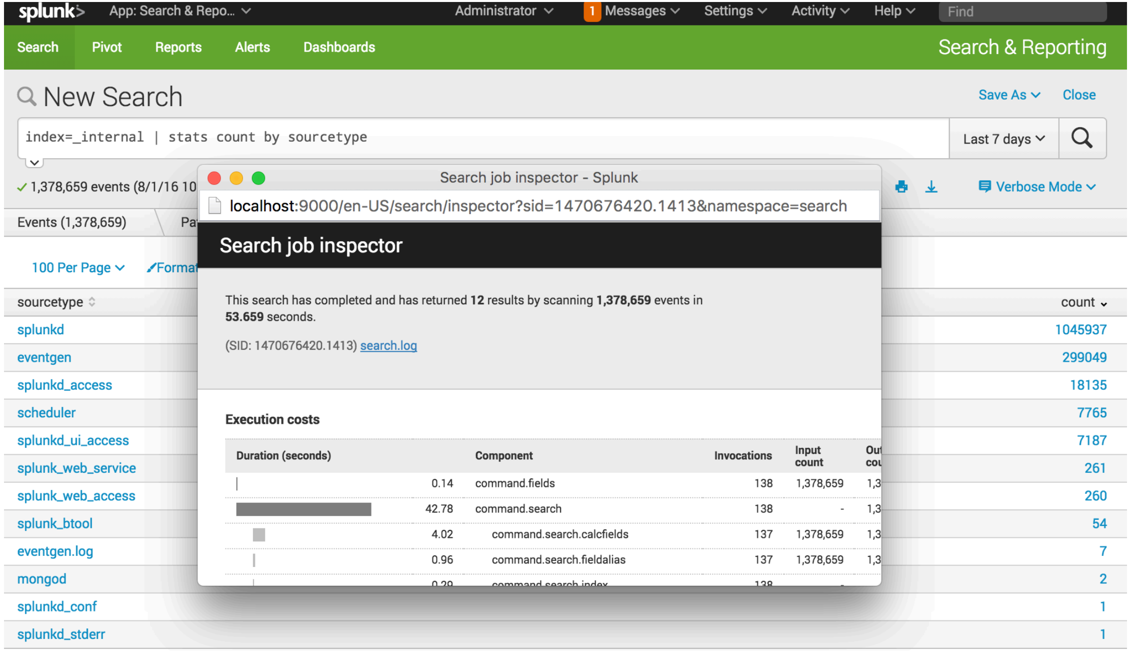
Searching – Part 1

- Search time field extractions
  - Use DELIMS based field extractions when you can (KV, comma, pipe)
  - Anchor RegExs, Avoid RegEx lookbehind if you can
- Be as specific as you can when writing searches
  - Pick the smallest search timerange that meets your needs (Default!=All time)
  - Use indexed fields (host/source/sourcetype)
  - Specify index explicitly, e.g. index=firewall
- Don't use **|table** in the middle of a search, use **|fields** instead
- Avoid realtime searches (use indexed_realtime if you can't)
- Avoid verbose mode, unless you are exploring

Searching – Part 2

- When reporting on indexed fields, consider using **| tstats** to search index files only
- Exploit acceleration options where it makes sense
  - Report acceleration
  - Data model acceleration
- Got extra indexer cores? Use parallel search pipelines (see D)
- Stay current on Splunk releases, we continuously focusing on performance improvements

Example: Search Vs. | tstats



```
index=_internal | stats count by sourcetype
```

✓ 1,378,659 events (8/1/16 10

Events (1,378,659)

sourcetype	count
splunkd	1045937
eventgen	299049
splunkd_access	18135
scheduler	7765
splunkd_ui_access	7187
splunk_web_service	261
splunk_web_access	260
splunk_btool	54
eventgen.log	7
mongod	2
splunkd_conf	1
splunkd_stderr	1

localhost:9000/en-US/search/inspector?sid=1470676420.1413&namespace=search

Search job inspector

This search has completed and has returned **12** results by scanning **1,378,659** events in **53.659** seconds.

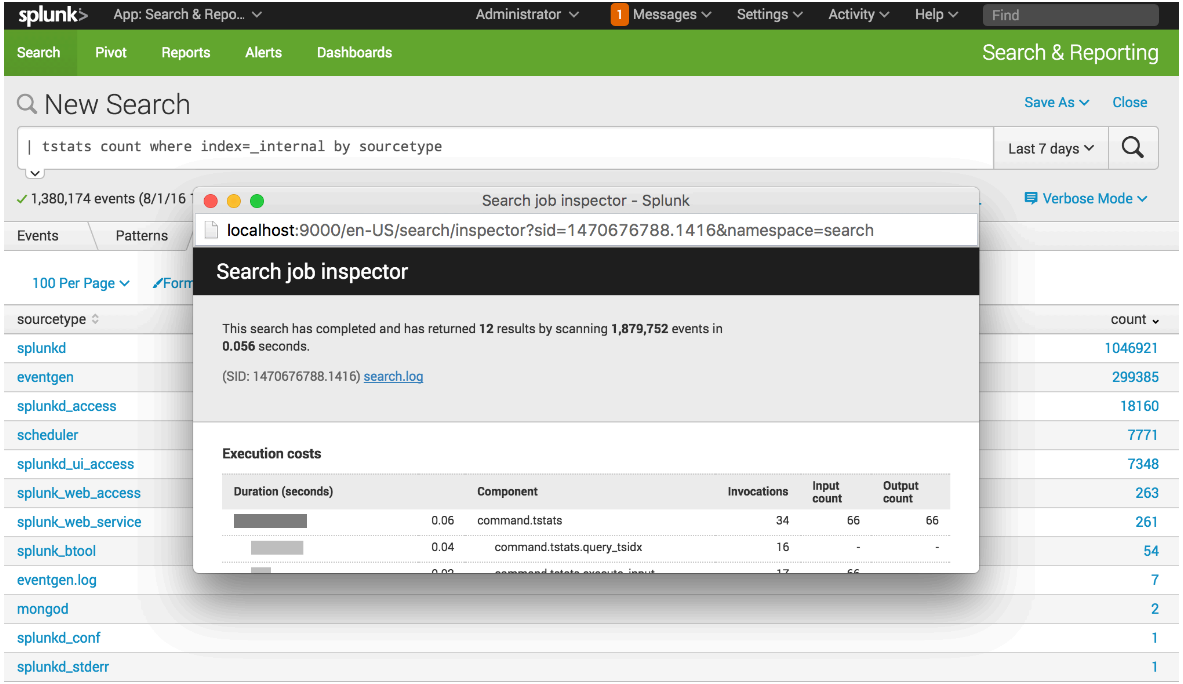
(SID: 1470676420.1413) search.log

Execution costs

Duration (seconds)	Component	Invocations	Input count	Ou cou
0.14	command.fields	138	1,378,659	1,3
42.78	command.search	138	-	1,3
4.02	command.search.calcfields	137	1,378,659	1,3
0.96	command.search.fieldalias	137	1,378,659	1,3
0.29	command.search.index	138	-	

index=_internal | stats count by sourcetype: 1.37MM events, 53.66secs

Example: Search Vs. | tstats



```
| tstats count where index=_internal by sourcetype
```

✓ 1,380,174 events (8/1/16

sourcetype	count
splunkd	1046921
eventgen	299385
splunkd_access	18160
scheduler	7771
splunkd_ui_access	7348
splunk_web_access	263
splunk_web_service	261
splunk_btool	54
eventgen.log	7
mongod	2
splunkd_conf	1
splunkd_stderr	1

Search job inspector - Splunk

localhost:9000/en-US/search/inspector?sid=1470676788.1416&namespace=search

Search job inspector

This search has completed and has returned **12** results by scanning **1,879,752** events in **0.056** seconds.

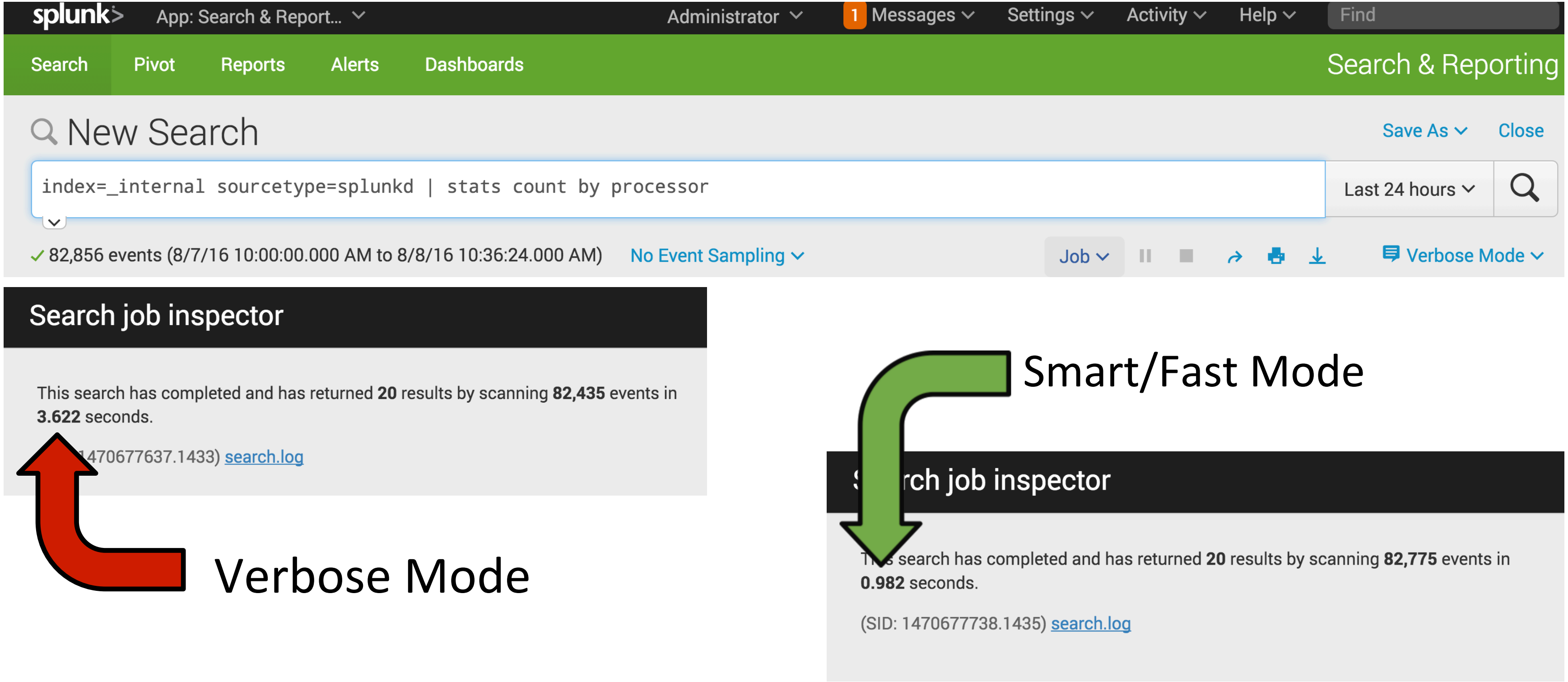
(SID: 1470676788.1416) search.log

Execution costs

Duration (seconds)	Component	Invocations	Input count	Output count
0.06	command.tstats	34	66	66
0.04	command.tstats.query_tsidx	16	-	-
0.02	command.tstats.execute_input	17	66	

| tstats count where index=_internal by sourcetype: 1.88MM events, 0.056secs

Example: Verbose Vs. Smart/Fast Mode



```
index=_internal sourcetype=splunkd | stats count by processor
```

✓ 82,856 events (8/7/16 10:00:00.000 AM to 8/8/16 10:36:24.000 AM)

Search job inspector

This search has completed and has returned **20** results by scanning **82,435** events in **3.622** seconds.

1470677637.1433) search.log

Verbose Mode

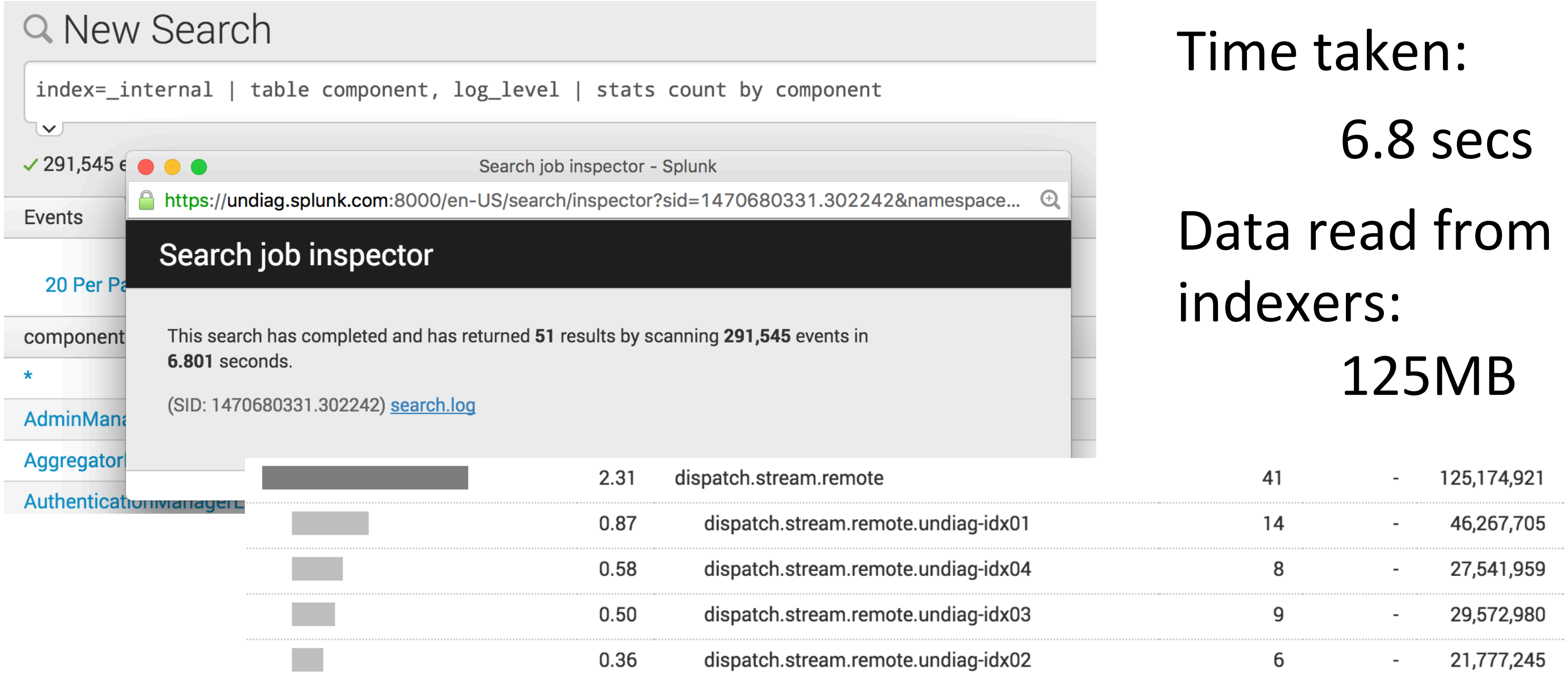
Smart/Fast Mode

Search job inspector

This search has completed and has returned **20** results by scanning **82,775** events in **0.982** seconds.

(SID: 1470677738.1435) search.log

Example: Table Vs. Fields



```
index=_internal | table component, log_level | stats count by component
```

Search job inspector - Splunk

https://undiag.splunk.com:8000/en-US/search/inspector?sid=1470680331.302242&namespace...

Search job inspector

This search has completed and has returned **51** results by scanning **291,545** events in **6.801** seconds.

(SID: 1470680331.302242) search.log

▬	2.31	dispatch.stream.remote	41	-	125,174,921
▬	0.87	dispatch.stream.remote.undiag-idx01	14	-	46,267,705
▬	0.58	dispatch.stream.remote.undiag-idx04	8	-	27,541,959
▬	0.50	dispatch.stream.remote.undiag-idx03	9	-	29,572,980
▬	0.36	dispatch.stream.remote.undiag-idx02	6	-	21,777,245

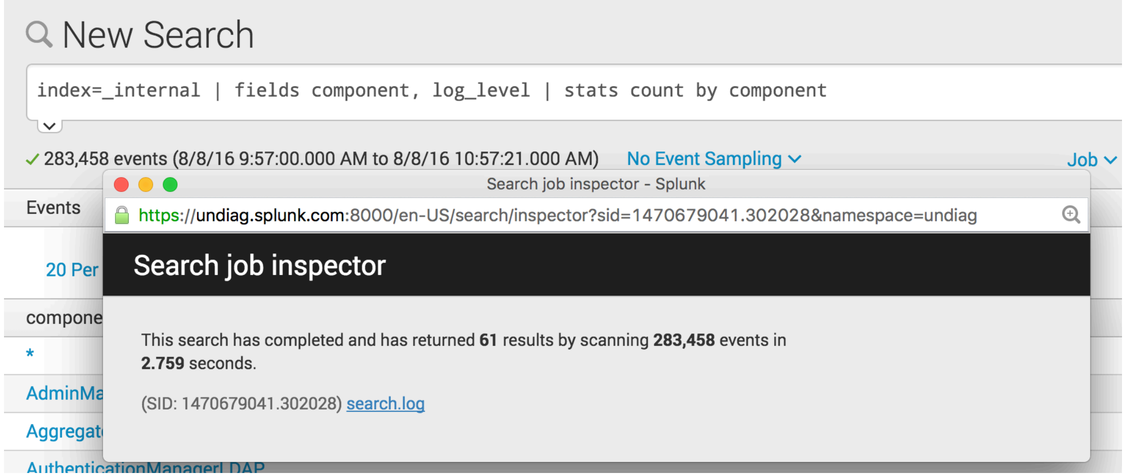
Time taken:

6.8 secs

Data read from indexers:

125MB

Example: Table Vs. Fields



```
index=_internal | fields component, log_level | stats count by component
```

✓ 283,458 events (8/8/16 9:57:00.000 AM to 8/8/16 10:57:21.000 AM) No Event Sampling Job

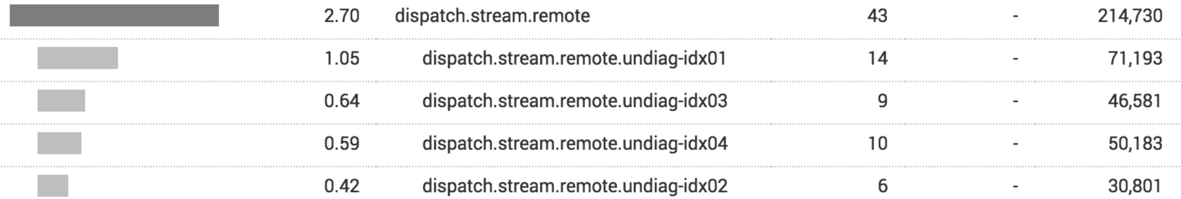
Search job inspector - Splunk

https://undiag.splunk.com:8000/en-US/search/inspector?sid=1470679041.302028&namespace=undiag

Search job inspector

This search has completed and has returned **61** results by scanning **283,458** events in **2.759** seconds.

(SID: 1470679041.302028) search.log



2.70	dispatch.stream.remote	43	-	214,730
1.05	dispatch.stream.remote.undiag-idx01	14	-	71,193
0.64	dispatch.stream.remote.undiag-idx03	9	-	46,581
0.59	dispatch.stream.remote.undiag-idx04	10	-	50,183
0.42	dispatch.stream.remote.undiag-idx02	6	-	30,801

Time taken:

2.76 secs

Data transferred from indexers:

214KB

UI/Dashboarding

.conf2016

splunk>

UI/Dashboarding

- Use saved/scheduled searches in dashboards (reuse search results across users)
- Use summary indices for long-term, aggregated metrics (don’t recalculate from raw)
- Restrict time-range picker options to minimum req’d for use case
- Use base searches and PostProcess for panels that are based on the same raw event search
- Minimize the number of panels that require individual searches
- Avoid auto-refresh if you can (kiosk/NOC use-case only)
- Don’t use real-time searches or at least use indexed_realtime

Conclusions/References

Conclusions

- Architecture choices affect performance. KISS!
- Pick the fastest storage you can afford for HOT/WARM
- Configurations at all tiers can affect performance
- Inefficient use of SPL affects performance
- Concurrent searches is the critical metric for search capacity planning
- Always consider search impact on ‘indexers’
- Enjoy your well-performing Splunk deployment!

Where To Go From Here

- Docs on search performance:
  - Optimize Splunk for Peak performance: http://docs.splunk.com/Documentation/Splunk/6.1.4/Admin/OptimizeSplunkforpeakperformance
  - Splunk performance checklist: http://docs.splunk.com/Documentation/Splunk/6.4.2/Capacity/Performancechecklist
  - How search types affect performance: http://docs.splunk.com/Documentation/Splunk/6.4.2/Capacity/HowsearchtypesaffectSplunkEnterpriseperformance

Related Sessions Of Interest

- **Observations and Recommendations on Splunk Performance**
  Wednesday, September 28, 2016 | 12:05 PM-12:50 PM
- **Behind the Magnifying Glass: How Search Works**
  Wednesday, September 28, 2016 | 1:10 PM-1:55 PM
- **Fields, Indexed Tokens and You**
  Wednesday, September 28, 2016 | 11:00 AM-11:45 AM
- **Indexer Clustering Internals, Scaling, and Performance**
  Tuesday, September 27, 2016 | 3:15 PM-4:00 PM
- **Worst Practices... and How to Fix Them**
  Tuesday, September 27, 2016 | 10:30 AM-11:15 AM
- **Jiffy Lube Quick Tune-up for Your Splunk Environment**
  Wednesday, September 28, 2016 | 11:00 AM-11:45 AM
- **Architecting Splunk for Epic Performance at Blizzard Entertainment**
  Tuesday, September 27, 2016 | 12:40 PM-1:25 PM
- **Lesser Known Search Commands**
  Wednesday, September 28, 2016 | 3:30 PM-4:15 PM

THANK YOU

.conf2016

splunk>