Architecting Splunk for Epic Performance at Blizzard Entertainment

Mason Morales

Sr. Security Engineer, Blizzard Entertainment

SPLUNK TRUST 2015-2016

.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

- Introduction
- Performance Issues
- Splunk Re-Design Project
- Q&A
- Bonus Content

Introduction

About Me

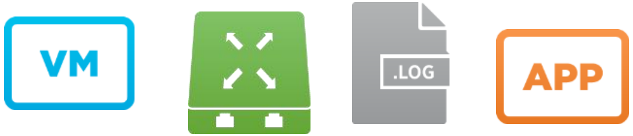
- SplunkTrust Member 2015-2016
- Splunk Certified Architect
- Experience
  - 5 years of Splunk XP
  - Sole developer of the Utilization Monitor for Splunk App
  - Published 126 Splunk training videos with Skillsoft
- Community
  - Splunk Answers @masonmorales
  - #Splunk IRC @Mason
- Started at Blizzard Entertainment in October 2015

About Blizzard

Blizzard Use Cases

- Security
- Game Fraud Detection

- IT Operations
  - Troubleshooting
  - Monitoring
  - Reporting
  - Alerting



- Provide Data to Other Internal Applications

History of Splunk at Blizzard

- Three separate Splunk deployments
- Nobody owned Splunk, no SME
- Serious performance issues on-prem
- Indexes with default settings
- Forwarders dating back to v4.3.3
- Indexers and search heads running v5.0.1
- Largest user group moved to Splunk Cloud because the on-prem deployment wasn't being maintained

October 2015

Battle Plan

* = What we'll cover in this talk

1. Upgrade all the things, implement DS, train users, and more…
2. *Fix performance issues with existing deployment
3. *Implement new infrastructure to meet business needs
4. Migrate forwarders and users to the new Splunk instance
5. Continue to add more data, users, and apps to Splunk

Performance Issues

Historical Causes at Blizzard

1. Too many accelerated searches
2. Too many real-time searches
3. Bad search schedules
4. Inefficient searches
5. IOPS-constrained hardware
6. Too many users in the same role

Splunk Performance

Addressing Performance Issues

Reactive

- Delete orphaned and unused scheduled searches
- Revoke search acceleration and real-time capabilities from role(s)
- Modifying scheduled searches
  - Disable search acceleration
  - Disable real-time
  - Convert fixed search schedules to the new “schedule window”

Proactive

- Perform capacity planning
- Implement role-based access control
- On-board data to different indexes
- Change default time range for timepicker
- User training

Using Roles

- Blizzard creates separate Splunk roles for each department
- Advantages
  1. Limit concurrent jobs
     - User-level
     - Role-level
  2. Limit disk usage on SHs
  3. Enforce search restrictions
  4. Separate knowledge objects when each role also has their own app
  5. Limit capabilities for each role
     - http://docs.splunk.com/Documentation/Splunk/latest/Security/Rolesandcapabilities

- Disadvantages
  - Slightly more administrative overhead

Tips for Configuring Roles

- Empty Index Trick
  1. Create an empty index (e.g. index=nothing)
  2. Assign index=nothing as the **index searched by default** for **every** role
  3. Inform users that they must always specify an index in their searches
- Limit Advanced Capabilities
  - We **do not** give these capabilities to all users:
    - accelerate_search
    - accelerate_datamodel
    - rtsearch
    - schedule_rtsearch
  - Evaluate the need for each capability on a case-by-case basis

Default Time Range

- Default time range in the time picker for search is “All Time”
- Users often forget to specify time range, but we can limit the damage

  – Edit $SPLUNK_HOME/etc/system/local/ui-prefs.conf

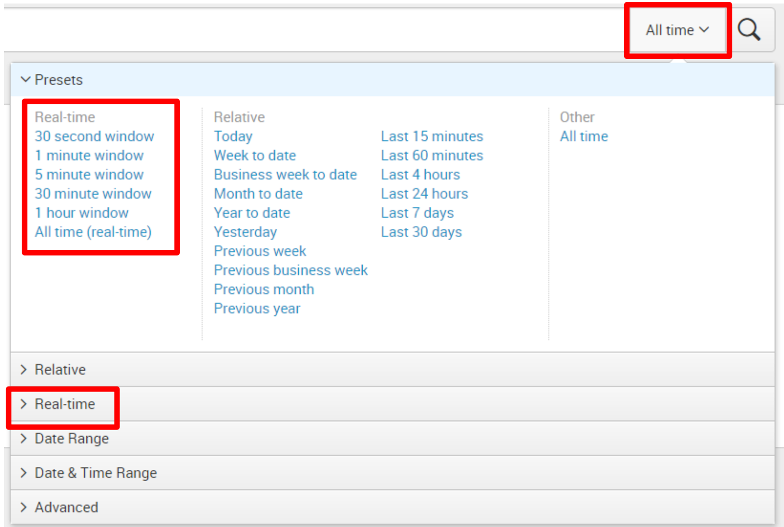
```
[search]
dispatch.earliest_time = -5m
dispatch.latest_time = now
```

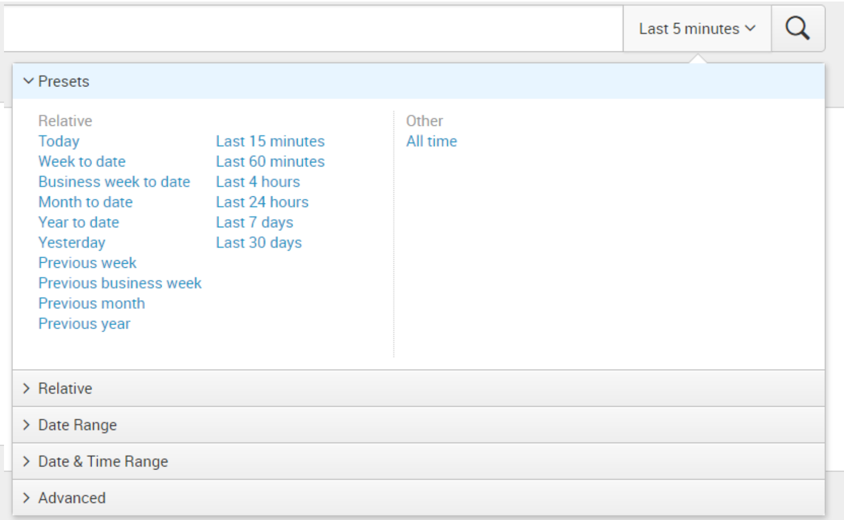
  – Or configure it through Splunk Web

    Server settings -> Search preferences -> Default search time range

Time Picker Customization

- Customize the time picker
  - Copy $SPLUNK_HOME/etc/system/default/times.conf
    To $SPLUNK_HOME/etc/system/local/times.conf
  - Edit as desired!





Scheduled Searches

- Long-running scheduled searches can waste system resources
  - Can cause the concurrent search limit to be hit
  - System-wide impact if everyone has the same role!
- Tip: Limit the amount of time searches can run for at the role level in authorize.conf

```
srchMaxTime = <number><unit>
* Maximum amount of time that searches of users from this role will be
  allowed to run.
* Once the search has been ran for this amount of time it will be auto
  finalized, If the role
* Inherits from other roles, the maximum srchMaxTime value specified in the
  included roles.
* This maximum does not apply to real-time searches.
* Examples: 1h, 10m, 2hours, 2h, 2hrs, 100s
* Defaults to 100days
```

Scheduled Searches

- How long are your scheduled searches *really* running for?

splunk> App: Utilization Monitor for Splunk | Mason Morales | Messages | Settings | Activity | Help | Find

Home | License Usage Analysis | Capacity Planning | Search Head Utilization | Forwarder Reports | Troubleshooting | Search | Utilization Monitor for Splunk

Search Head Scheduled Search Utilization

Edit | More Info

Time: Last 24 hours | App: All | User: All | Splunk Server: All

Total Run Time and Count of Scheduled Searches by App, User, Scheduled Search Name

App	Saved Search Name	User	Count	Total Run Time (sec)	% of Total Run Time
[redacted]			126	74956.8	21.51
			126	68641.8	19.70
			96	38302.9	10.99
			36	31848.4	9.14
			291	18841.8	5.41

Indexes

- Many of our source types get their own index
  - Why? Efficiency
    - Each index has its own directory and buckets on the file system
  - When should you separate sourcetypes into additional indexes?
    - Different retention times
    - Different access requirements
    - Different applications generating the data
    - One set of data searched more often than another set
- Tip: Create a data catalog for users

User Training

- Blizzard has an internal Splunk User Group that does training at least once/month, along with recurring workshops to help users learn Splunk
- When Blizzard on-boards new users to Splunk, they are invited to the User Group and given the following list of learning resources
  - Splunk Cheat Sheet: http://docs.splunk.com/images/4/4f/Splunk_Quick_Reference_Guide_6.x.pdf
  - Community Forum: https://answers.splunk.com/
  - Free Splunk eBook: http://www.splunk.com/web_assets/v5/book/Exploring_Splunk.pdf
  - Free Splunk Course: http://www.splunk.com/view/SP-CAAAHSM
  - Splunk Education Videos: http://www.splunk.com/view/education-videos/SP-CAAAGB6
  - Splunk Docs: http://docs.splunk.com/Documentation/Splunk
  - Splunk Wiki: https://wiki.splunk.com/Main_Page
  - Splunk Apps: https://splunkbase.splunk.com/
  - Splunk YouTube Channel: https://www.youtube.com/channel/UCjwOFZzLPnji1EstaVyyvAw
  - <Internal Wiki Links>

Tips for User Training

- Document examples of good searches on your internal Wiki
- Ask your rep for Splunk swag, like query mugs!
- Distribute Splunk quick reference cards
- Hold your own SPLing bee to encourage hands-on practice with Splunk
- Look at use cases in your environment and help users implement things like summary indexing, accelerated data models, and report acceleration

Splunk Re-Design Project

Project Summary

- Goals
  - "One Splunk" experience at Blizzard
  - Awesome performance
  - High availability
  - 1-year data retention
- Bonus
  - Retire the two on-prem Splunk instances
  - Level-up configuration management
  - Standardize on one hardware platform

Approach

1. Determine hardware requirements
2. Procure hardware
3. Benchmark various configurations
4. Deploy new Splunk cluster

Hardware Selection

- Storage Requirements
  - Various use cases required fast random read
  - 1-year data retention + indexer clustering = **MANY DISKS!! NOW HANDLE IT!**

- Cost of SSD evaluated against 15k HDD
  - SAS 15K Enterprise Drive: **$0.81/GB**
  - SAS SSD Enterprise Drive: **$0.91/GB**
  - SATA SSD Enterprise Drive: **$0.49/GB**
  - SATA SSD was the clear winner in terms of cost
  - Additionally, SSD drives had 640% more storage density than the 15k drives
- Splunk Performance with SSD

  http://blogs.splunk.com/2012/05/10/quantifying-the-benefits-of-splunk-with-ssds/

Evaluating SSDs for Splunk

SATA vs SAS Technical Comparison

Enterprise SATA 3.84 TB SSD

- 540 MB/s Seq. Read
- 480 MB/s Seq. Write
- 99,000 IOPS Random Read
- 18,000 IOPS Random Write
- MTTF: 2,000,000 Hours
- Cost: $1,900

SAS 3.84 TB SSD

- 1500 MB/s Seq. Read
- 750 MB/s Seq. Write
- 270,000 IOPS Random Read
- 22,000 IOPS Random Write
- MTTF 2,000,000 Hours
- Cost: $3,500

Blizzard Conclusion: SAS was 84% more expensive for the same amount of storage while Splunk would likely be CPU constrained with a sufficiency quantity of either drive

CPU and Memory

- Reference machine for distributed deployments
  - 16 cores @ 2+ Ghz/core
  - 12 GB RAM (really more like 64+ GB)
- Ultimately a business decision
  - Memory is cheap, better too much than too little
  - Many options for CPU, just stay within cost constraint

Blizzard Indexer Hardware

- 1U dual-socket enterprise servers
- Dual Intel Xeon E5 v4 @ 3.4 GHz
- 256 GB ECC DDR4 2400 MHz
- 20x 2.5” external hot-swap bays (data)
- 2x 2.5” internal bays (OS) (RAID1)
- 2x HBAs + on-board storage controller

Scaling Splunk for Performance

- Splunk scales horizontally, so we distributed pretty heavily
- Tip: Always add indexers before adding search heads
  - More indexers = greater search distribution = faster search completion time
  - Faster search completion time = less search concurrency
- Each search uses <u>one core</u> on each indexer
  - Frequency-optimized CPUs can offer better search performance but at the cost of less concurrency (since they typically have a lower core count)

Doubling Performance

- Blizzard deployed twice as many indexers with only 20% additional cost by purchasing SSDs with *half* the capacity of the max available
- This gave us <u>twice the compute</u> and the <u>same amount of storage</u>
- Other benefits
  - Double the available disk throughput
  - Lower CPU contention
  - Lower memory contention
  - Reduction in concurrency factor
  - Substantially better search performance

System Configurations

- Settings
  - BIOS
    - Enabled hyper-threading
    - Disabled CPU power saving in BIOS
  - OS
    - Partitions were aligned to erase blocks on SSDs
    - Swap file was disabled
    - Linux IO scheduler was set to deadline
    - Queue depth was set to 32 for each drive
    - Disabled Transparent Huge Pages (THP)
    - ulimit
      - Core file size (ulimit -c) to unlimited
      - Data segment size (ulimit -d) to unlimited
      - Max open files (ulimit -n) to 65536
      - Max user processes (ulimit -u) to 258048

Testing Methodology

- Tested random read, sequential read, and Splunk search performance
  - Scope included different file systems, RAID levels, and Splunk journal compression algorithms (GZIP vs LZ4)
  - Goal was to determine the best performing configuration
- RAID
  - mdadm used for the EXT4 and XFS tests
  - BTRFS used built-in RAID functionality
- Same indexer used for all testing

Testing Process

- Synthetic benchmarks performed with FIO on Ubuntu 14.04.4 (x64)
  - Flexible I/O (FIO) is available at https://github.com/axboe/fio
  - Syntax at https://github.com/axboe/fio/blob/master/HOWTO
  - Disk cache invalidated at the start of each test and used non-buffered IO
- Splunk benchmarks performed using a large static data set
  - Splunk v6.4.1 with parallelization settings enabled
  - Ran the same searches under each configuration
  - Recorded search completion times

Synthetic Benchmark Results

Sequential Read at 1M Block Size

RAID 10

FS	**Throughput**
BTRFS	4,594 MB/s
EXT4	10,266 MB/s
XFS	10,310 MB/s

RAID 5

FS	**Throughput**
BTRFS	5,346 MB/s
EXT4	10,345 MB/s
XFS	10,390 MB/s

```
fio --time_based --name=4k_benchmark --size=100G --runtime=30 --filename=/splunkdata/test --ioengine=libaio
--iodepth=128 --direct=1 --invalidate=1 --verify=0 --verify_fatal=0 --numjobs=12 --rw=read --blocksize=1M --group_reporting
```

Synthetic Benchmark Results

Random Read at 4k Block Size

RAID 10

FS	IOPS	Throughput
BTRFS	443,000	1,733 MB/s
EXT4	389,000	1,533 MB/s
XFS	1,228,000	5,032 MB/s

RAID 5

FS	IOPS	Throughput
BTRFS	427,000	1,670 MB/s
EXT4	448,000	1,750 MB/s
XFS	2,794,000	10,915 MB/s

```
fio --time_based --name=4k_benchmark --size=100G --runtime=30 --filename=/splunkdata/test --ioengine=libaio
--randrepeat=0 --iodepth=128 --direct=1 --invalidate=1 --verify=0 --verify_fatal=0 --numjobs=12 --rw=randread --blocksize=4k
--group_reporting
```

Synthetic Benchmark Results

- The numbers for XFS on RAID 5 seemed “too good”
  - Retested without a time limit and set FIO to random read 1 TB per process
- Final result was 1,295,400 IOPS and 5,058 MB/s at a 4k

```
benchmark: (g=0): rw=randread, bs=4K-4K/4K-4K/4K-4K, ioengine=libaio,
iodepth=128
 fio-2.1.3 Starting 12 processes
 benchmark: (groupid=0, jobs=12):
 read : io=12000GB, bw=5058.8MB/s, iops=1295.4K, runt=2429074msec
```

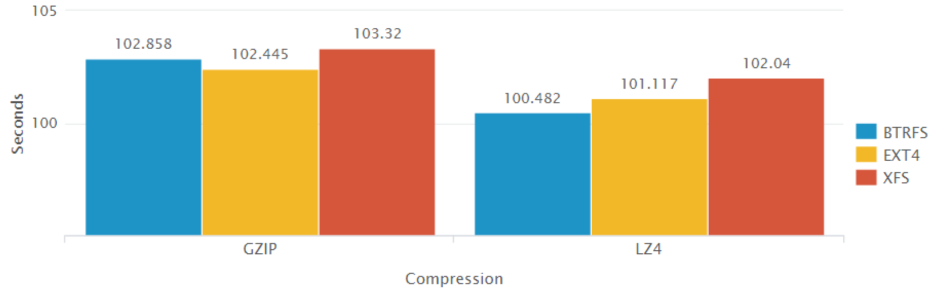
- Sequential read throughput was 10,294 MB/sec at a 1M block size

```
 fio-2.1.3 Starting 64 processes
 Run status group 0 (all jobs):
 READ: io=617713MB, aggrb=10294MB/s, minb=10294MB/s, maxb=10294MB/s
```

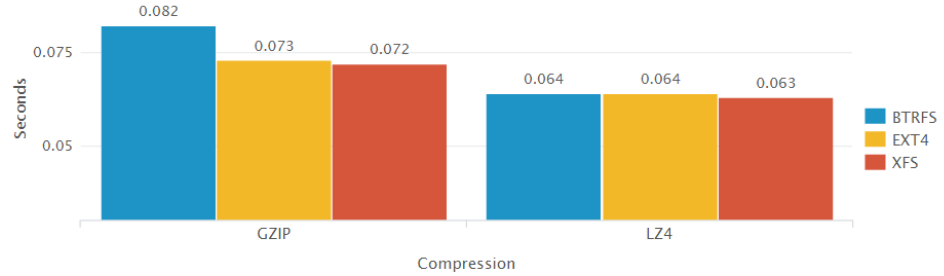
Splunk Benchmark Results

Single Indexer

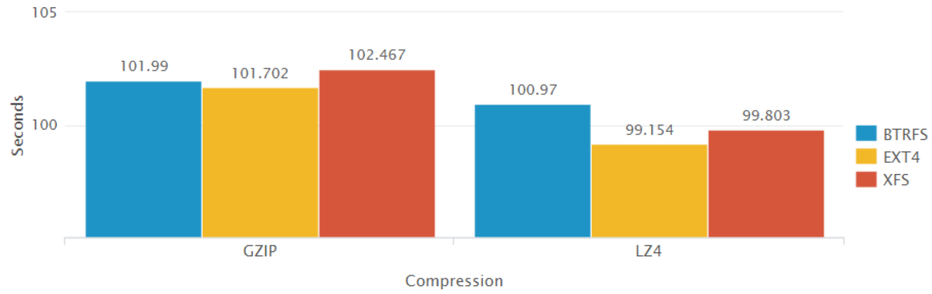
RAID 10 - Dense Search Completion Time



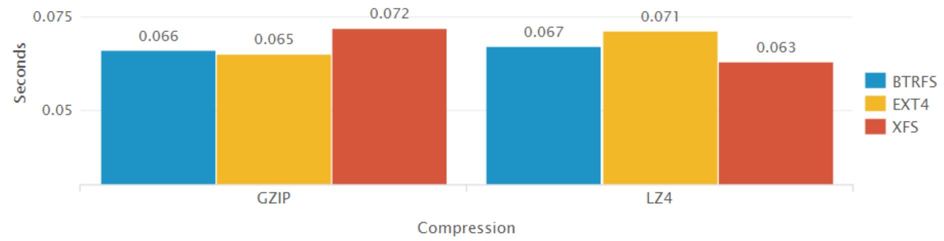
RAID 10 - Rare Search Completion Time



RAID 5 - Dense Search Completion Time



RAID 5 - Rare Search Completion Time



Splunk Benchmark Results

Dense Search Test

Seconds ^	FS	RAID	Compression
99.154	EXT4	R5	LZ4
99.803	XFS	R5	LZ4
100.482	BTRFS	R10	LZ4
100.97	BTRFS	R5	LZ4
101.117	EXT4	R10	LZ4
101.702	EXT4	R5	GZIP
101.99	BTRFS	R5	GZIP
102.04	XFS	R10	LZ4
102.445	EXT4	R10	GZIP
102.467	XFS	R5	GZIP
102.858	BTRFS	R10	GZIP
103.32	XFS	R10	GZIP

Rare Search Test

Seconds ^	FS	RAID	Compression
0.063	XFS	R5	LZ4
0.063	XFS	R10	LZ4
0.064	BTRFS	R10	LZ4
0.064	EXT4	R10	LZ4
0.065	EXT4	R5	GZIP
0.066	BTRFS	R5	GZIP
0.067	BTRFS	R5	LZ4
0.071	EXT4	R5	LZ4
0.072	XFS	R5	GZIP
0.072	XFS	R10	GZIP
0.073	EXT4	R10	GZIP
0.082	BTRFS	R10	GZIP

Splunk Benchmark Conclusion

- Search speed was nearly identical in all tests

<u>CPU will always be the bottleneck for adhoc searches in Splunk once you have a sufficiently fast disk subsystem</u>

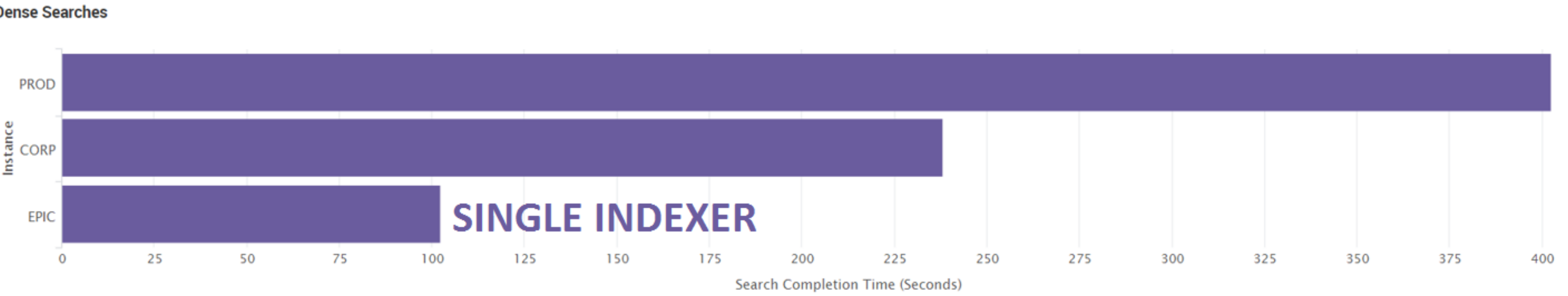
  – Bonus finding: LZ4 does not yield any substantial gains in performance that would be worth the tradeoff in extra storage vs. GZIP

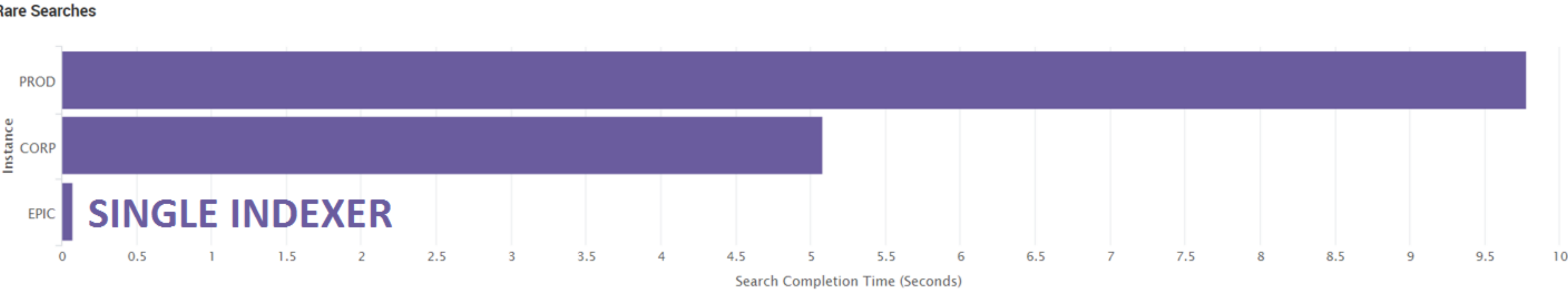
Wrap-up

.conf2016

splunk>

Performance Comparison





Splunk Features for Faster Searching

- Summary indexing
  - http://docs.splunk.com/Documentation/Splunk/latest/Knowledge/Usesummaryindexing
- Data model acceleration
  - http://docs.splunk.com/Documentation/Splunk/latest/Knowledge/Acceleratedatamodels
- Report acceleration
  - http://docs.splunk.com/Documentation/Splunk/latest/Report/Acceleratereports
- Post-process searches
  - http://docs.splunk.com/Documentation/Splunk/latest/Viz/Savedsearches#Post-process_searches
- Batch mode search parallelization
  - http://docs.splunk.com/Documentation/Splunk/latest/Knowledge/Configurebatchmodesearch

Q&A

.conf2016

splunk>

What Now?

Related breakout sessions and activities…

- How Splunkd Works
- Notes on Optimizing Splunk Performance
- Architecting Splunk for High Availability and Disaster Recovery
- Architecting and Sizing Your Splunk Deployment
- Harnessing Performance and Scalability in the Next Version of Splunk
- Onboarding Data Into Splunk
- Splunk User Groups: More Than Pints and Pizza

THANK YOU

Bonus Content

Optimizations for Data On-Boarding

- Splunk's flexibility to perform automatic sourcetype recognition, timestamp recognition, etc. come at the expense of performance
- To maximize CPU efficiency on indexers, always configure:
  - LINE_BREAKER
  - SHOULD_LINEMERGE
  - MAX_TIMESTAMP_LOOKAHEAD
  - TIME_PREFIX
  - TIME_FORMAT

Bonus Content

Enabling Parallelization

- Enable parallelization settings (v6.3+)

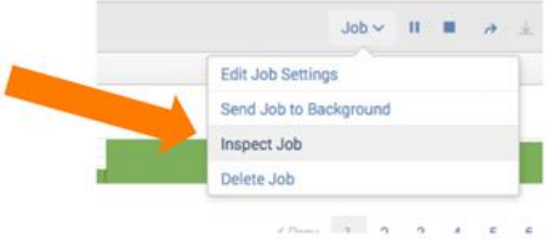
Setting	Description
Batch mode search parallelization	Allows a batch mode search to open additional search pipelines on each indexer, processing multiple buckets simultaneously.
Parallel summarization for data models	Allows the scheduler to run concurrent data model acceleration searches on the indexers.
Parallel summarization for report accelerations	Allows the scheduler to run concurrent report acceleration searches on the indexers.
Index parallelization	Allows concurrent data processing pipelines on indexers and forwarders.

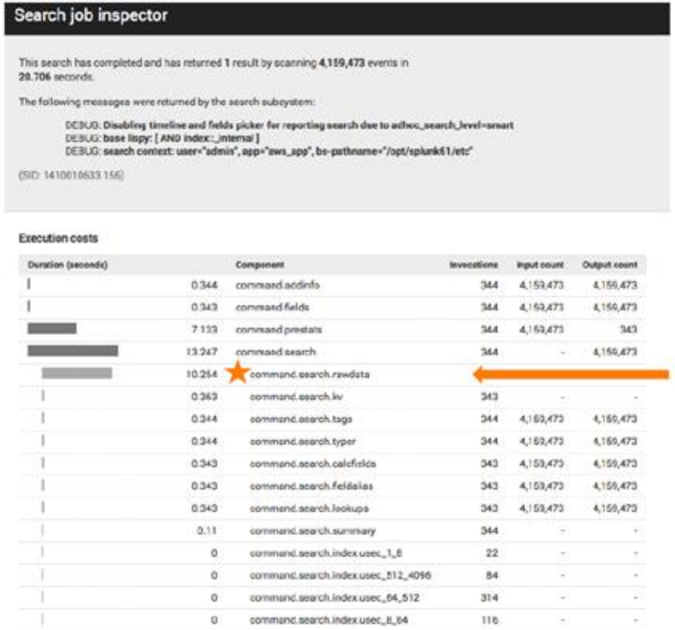
http://docs.splunk.com/Documentation/Splunk/latest/Capacity/Parallelization

Bonus Content

Identifying Bottlenecks

- Search Job Inspector





Search job inspector

This search has completed and has returned **1** result by scanning **4,159,473** events in **20.706** seconds.

The following messages were returned by the search subsystem:

DEBUG: **Disabling timeline and fields picker for reporting search due to adhoc_search_level=smart**
DEBUG: **base lispy: [AND index::_internal]**
DEBUG: **search context: user="admin", app="aws_app", bs-pathname="/opt/splunk61/etc"**

(SID: 1410010633.156)

Execution costs

Duration (seconds)	Component	Invocations	Input count	Output count
0.344	command.addinfo	344	4,159,473	4,159,473
0.343	command.fields	344	4,159,473	4,159,473
7.133	command.prestats	344	4,159,473	343
13.247	command.search	344	-	4,159,473
10.254	command.search.rawdata			
0.363	command.search.kv	343	-	-
0.344	command.search.tags	344	4,159,473	4,159,473
0.344	command.search.typer	344	4,159,473	4,159,473
0.343	command.search.calcfields	343	4,159,473	4,159,473
0.343	command.search.fieldalias	343	4,159,473	4,159,473
0.343	command.search.lookups	343	4,159,473	4,159,473
0.11	command.search.summary	344	-	-
0	command.search.index.usec_1_8	22	-	-
0	command.search.index.usec_512_4096	84	-	-
0	command.search.index.usec_64_512	314	-	-
0	command.search.index.usec_8_64	116	-	-

Guideline in absence of full instrumentation

- **command.search.rawdata** ~ CPU Bound
  - Others: .kv, .typer, .calcfields,
- **command.search.index** ~ IO Bound

Bonus Content

Apps for Splunk Performance Management

- Distributed Management Console (DMC)
  - http://docs.splunk.com/Documentation/Splunk/latest/DMC/DMCoverview
- Utilization Monitor for Splunk (SUM)
  - https://splunkbase.splunk.com/app/2678/
- Search Activity
  - https://splunkbase.splunk.com/app/2632/