Power Of Splunk SPL (Search Processing Language)

Stephen Luedtke

Sr. Technical Marketing Mgr, Splunk

.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.

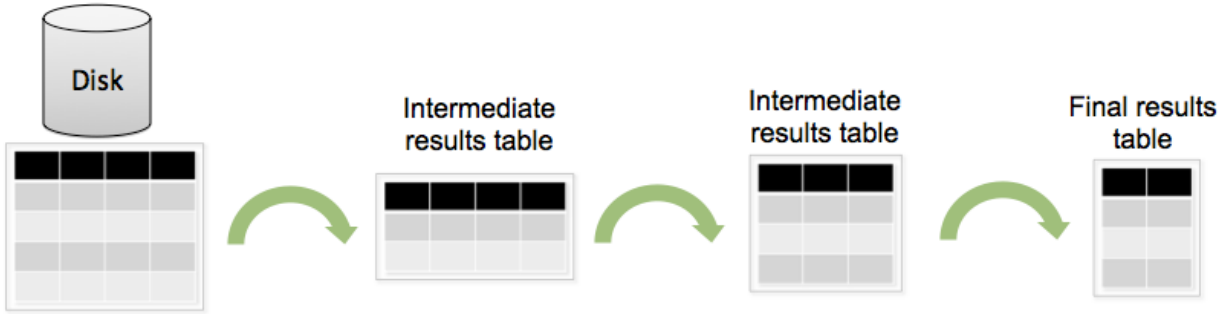
Agenda

- Overview & Anatomy of a Search
  - Quick refresher on search language and structure
- SPL Commands and Examples
  - Searching, *charting*, converging, mapping, transactions, anomalies, exploring
- Custom Commands
  - Extend the capabilities of SPL
- Q&A

SPL Overview

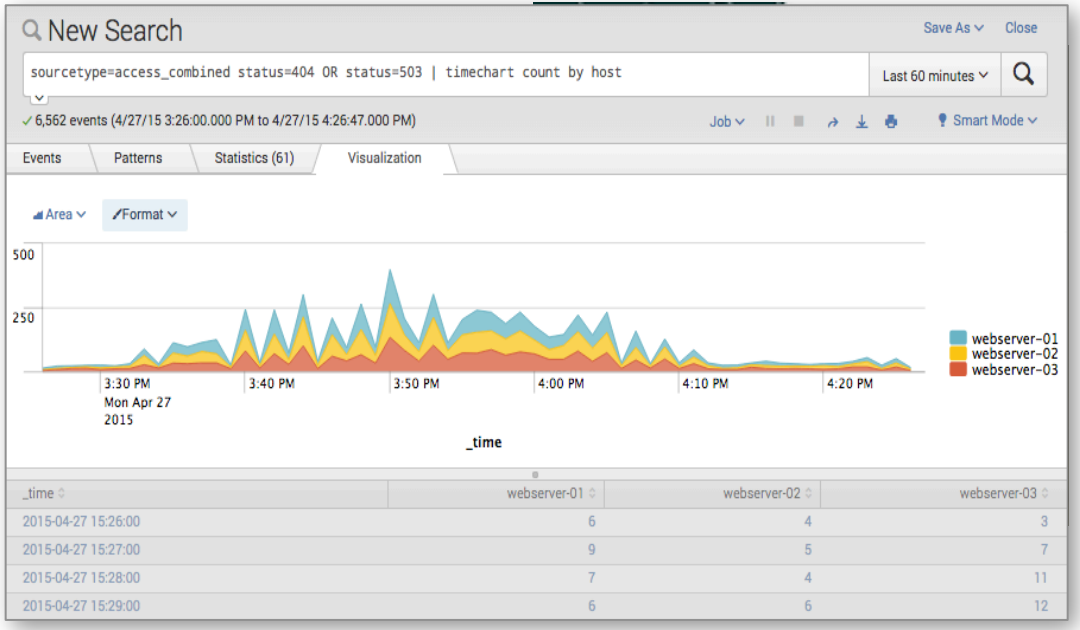
SPL Overview

- Over 140+ search commands
- Syntax was originally based upon the Unix pipeline and SQL and is optimized for time series data
- The scope of SPL includes data searching, filtering, modification, manipulation, enrichment, insertion and deletion
- Includes anomaly detection and machine learning



Why Create A New Query Language?

- Flexibility and effectiveness on *small* and ***big*** data
- Late-binding schema
- More/better methods of correlation
- Not just analyze, but visualize



SPL Basic Structure

search and filter | munge | report | cleanup

```
sourcetype=access*
| eval KB=bytes/1024
| stats sum(KB) dc(clientip)
| rename sum(KB) AS "Total KB" dc(clientip) AS "Unique Customers"
```

SPL Examples

SPL Examples And Recipes

- Find the needle in the haystack
- Charting statistics and predicting values
- Enriching and converging data sources
- Visualize geographic data in real time
- Identifying transactions and anomalies
- Data exploration & finding relationships between fields

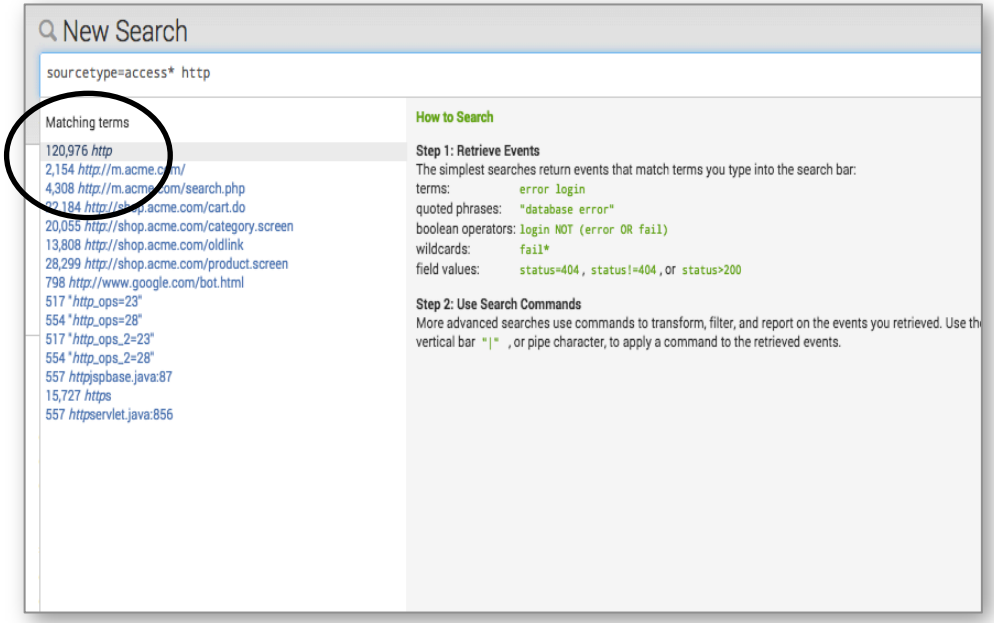
SPL Examples And Recipes

- **Find the needle in the haystack**
- Charting statistics and predicting values
- Enriching and converging data sources
- Visualize geographic data in real time
- Identifying transactions and anomalies
- Data exploration & finding relationships between fields

Search And Filter

Examples

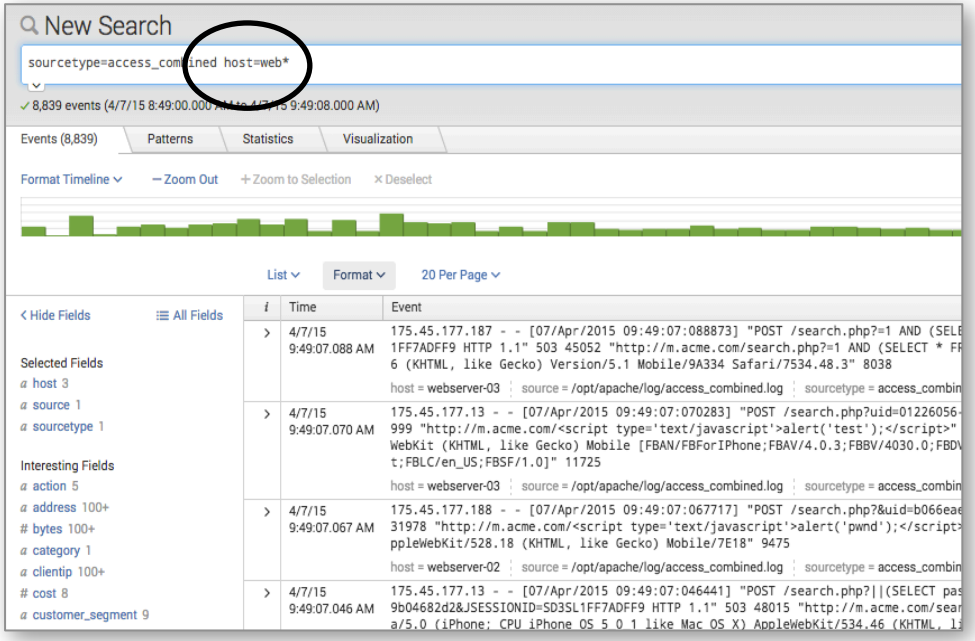
- **Keyword search:**
  sourcetype=access* http
- Filter:
  sourcetype=access* http
  host=webserver-02
- Combined:
  sourcetype=access* http
  host=webserver-02 (503 OR 504)



Search And Filter

Examples

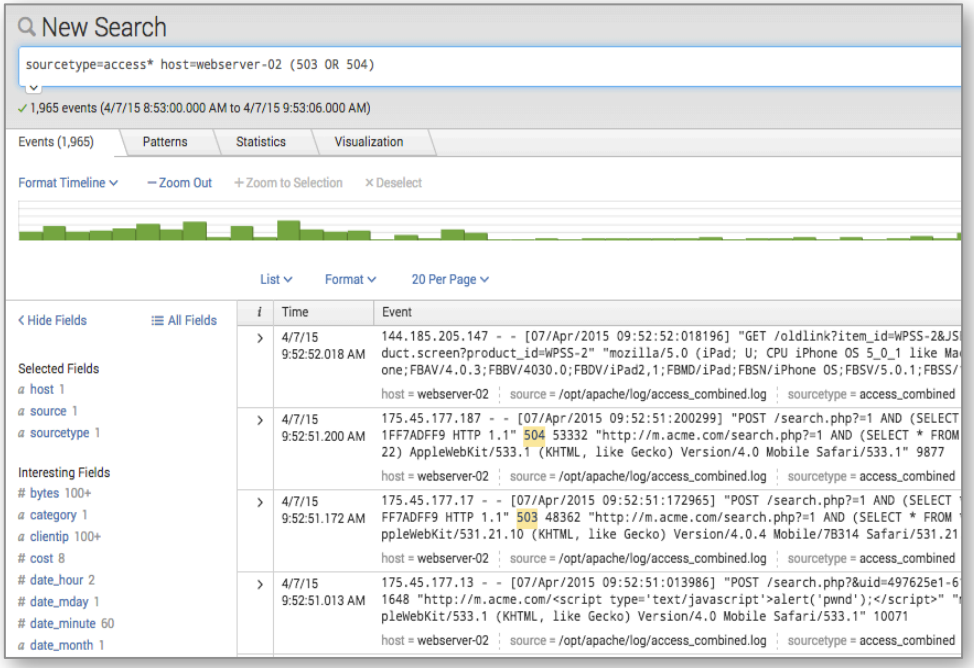
- Keyword search:
  sourcetype=access* http
- **Filter:**
  sourcetype=access* http
  host=webserver-02
- Combined:
  sourcetype=access* http
  host=webserver-02 (503 OR 504)



Search And Filter

Examples

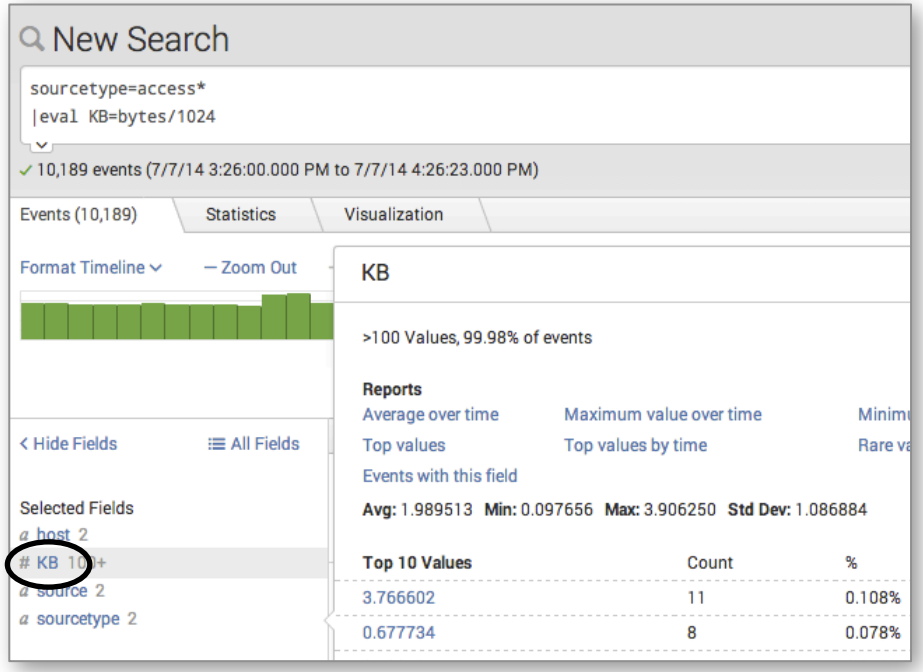
- Keyword search:
  sourcetype=access* http
- Filter:
  sourcetype=access* http
  host=webserver-02
- **Combined:**
  sourcetype=access* http
  host=webserver-02 (503 OR 504)



Eval – Modify Or Create New Fields And Values

Examples

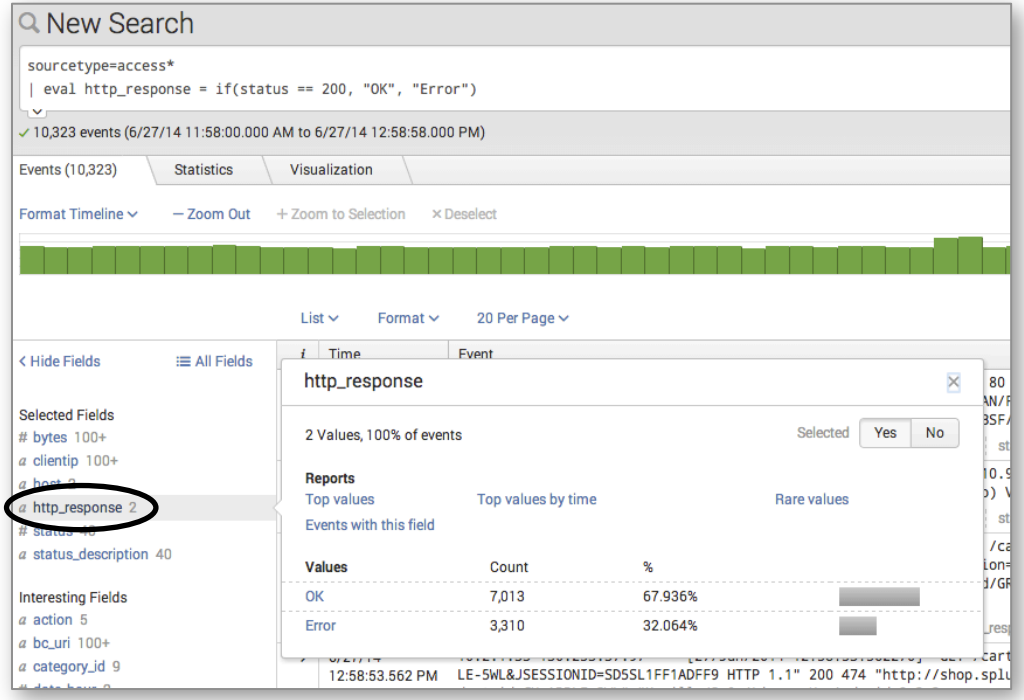
- **Calculation:**
  sourcetype=access*
  |eval KB=bytes/1024
- Evaluation:
  sourcetype=access*
  | eval http_response =
  if(status != 200, ”Error", ”OK”)
- Concatenation:
  sourcetype=access*
  | eval connection = clientip.":".port



Eval – Modify Or Create New Fields And Values

Examples

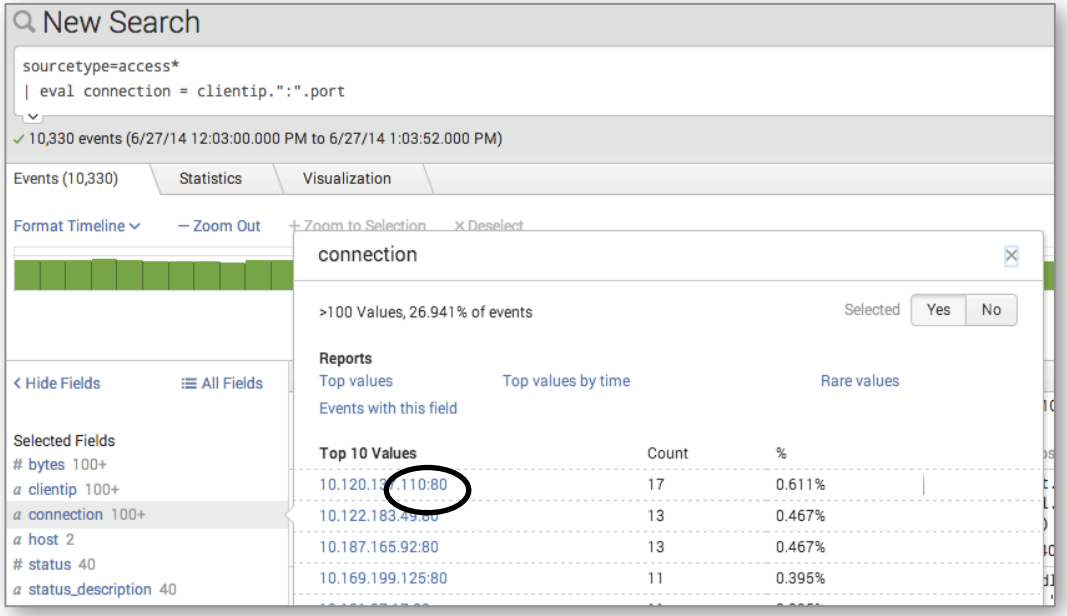
- Calculation:
  sourcetype=access*
  |eval KB=bytes/1024
- **Evaluation:**
  sourcetype=access*
  | eval http_response =
  if(status != 200, ”Error", ”OK”)
- Concatenation:
  sourcetype=access*
  | eval connection = clientip.":".port



Eval – Modify Or Create New Fields And Values

Examples

- Calculation:
  sourcetype=access*
  |eval KB=bytes/1024
- Evaluation:
  sourcetype=access*
  | eval http_response =
  if(status != 200, ”Error", ”OK”)
- **Concatenation:**
  sourcetype=access*
  | eval connection = clientip.":".port



Eval – Just Getting Started!

Splunk Search Quick Reference Guide

EVAL FUNCTIONS

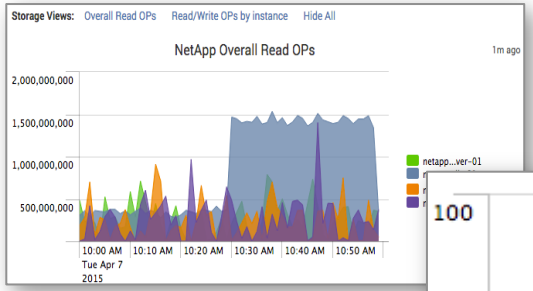
The eval command calculates an expression and puts the resulting value into a field (e.g. "...| eval force = mass * acceleration"). The following table lists the functions eval understands, in addition to basic arithmetic operators (+ - * / %), string concatenation (e.g., '...| eval name = last . ", " . last'), boolean operations (AND OR NOT XOR < > <= >= != = == LIKE).

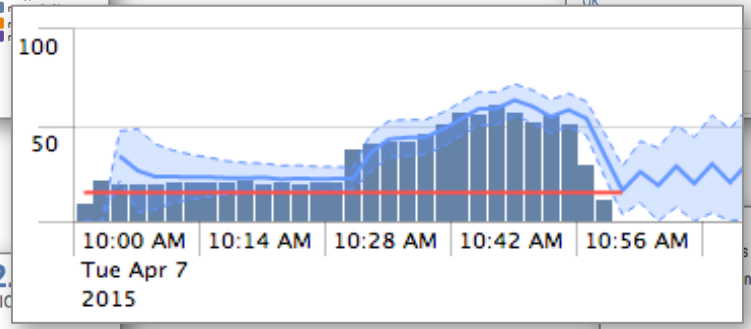
FUNCTION	DESCRIPTION	EXAMPLES
abs(X)	Returns the absolute value of X.	`abs(number)`
case(X,"Y",…)	Takes pairs of arguments X and Y, where X arguments are Boolean expressions that, when evaluated to TRUE, return the corresponding Y argument.	`case(error == 404, "Not found", error == 500,"Internal Server Error", error == 200, "OK")`
ceil(X)	Ceiling of a number X.	`ceil(1.9)`
cidrmatch("X",Y)	Identifies IP addresses that belong to a particular subnet.	`cidrmatch("123.132.32.0/25",ip)`
coalesce(X,…)	Returns the first value that is not null.	`coalesce(null(), "Returned val", null())`
exact(X)	Evaluates an expression X using double precision floating point arithmetic.	`exact(3.14*num)`
exp(X)	Returns e^X.	`exp(3)`
floor(X)	Returns the floor of a number X.	`floor(1.9)`
if(X,Y,Z)	If X evaluates to TRUE, the result is the second argument Y. If X evaluates to FALSE, the result evaluates to the third argument Z.	`if(error==200, "OK", "Error")`
isbool(X)	Returns TRUE if X is Boolean.	`isbool(field)`
isint(X)	Returns TRUE if X is an integer.	`isint(field)`
isnotnull(X)	Returns TRUE if X is not NULL.	`isnotnull(field)`
isnull(X)	Returns TRUE if X is NULL.	`isnull(field)`
isnum(X)	Returns TRUE if X is a number.	`isnum(field)`
isstr()	Returns TRUE if X is a string.	`isstr(field)`

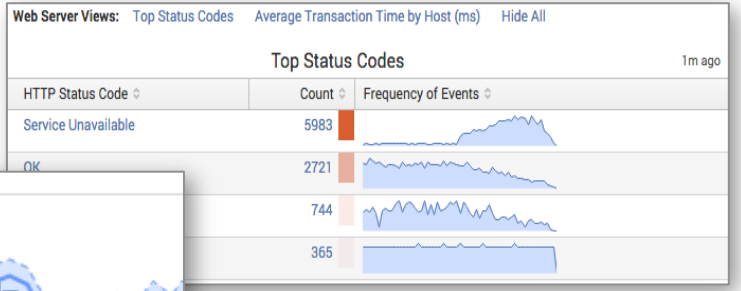
SPL Examples And Recipes

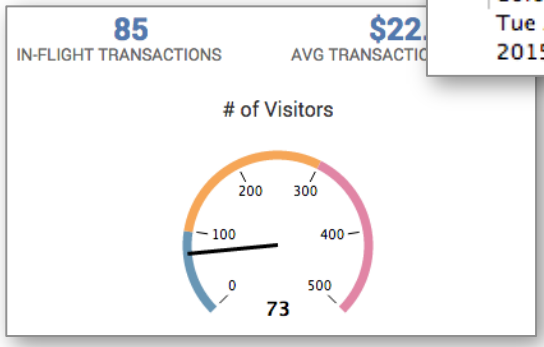
- Find the needle in the haystack
- **Charting statistics and predicting values**
- Enriching and converging data sources
- Visualize geographic data in real time
- Identifying transactions and anomalies
- Data exploration & finding relationships between fields

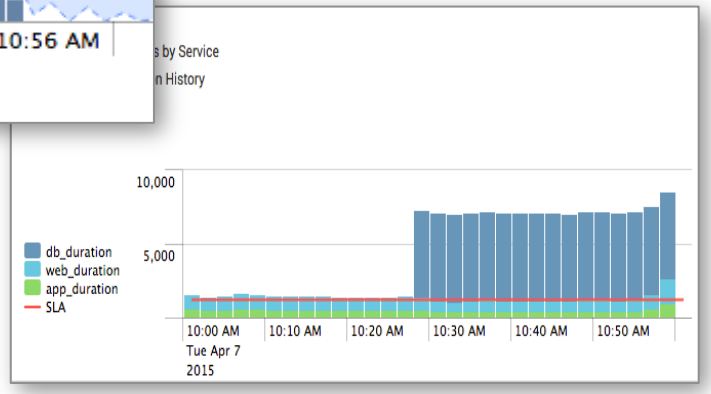
Stats, Chart, Timechart











Shake!!

Go to splunk.com/shake
on your mobile device.

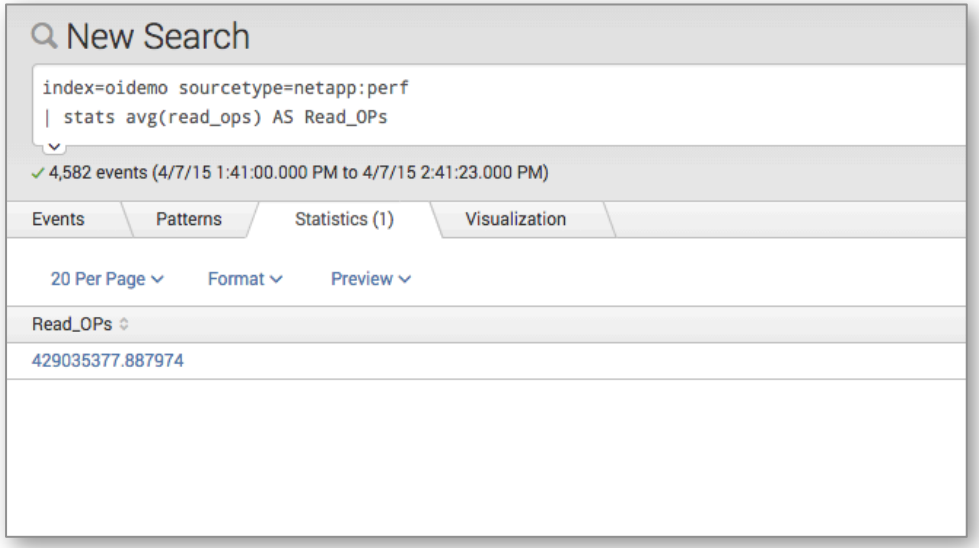
.conf2016

splunk>

Stats-Calculate Statistics Based On Field Values

Examples

- **Calculate stats and rename**
  sourcetype=netapp:perf
  | stats avg(read_ops) AS “Read OPs”
- Multiple statistics
  sourcetype=netapp:perf
  | stats avg(read_ops) AS Read_OPs
  sparkline(avg(read_ops)) AS Read_Trend
- By another field
  Sourcetype=netapp:perf
  | stats avg(read_ops) AS Read_OPs
  sparkline(avg(read_ops)) AS Read_Trend
  by instance



Stats-Calculate Statistics Based On Field Values

Examples

- Calculate stats and rename

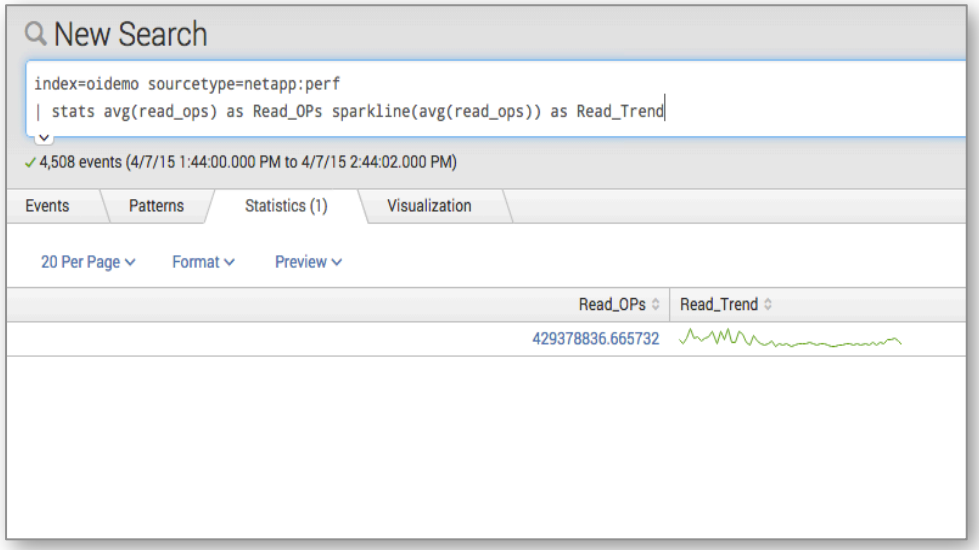
  sourcetype=netapp:perf
  | stats avg(read_ops) AS “Read OPs”

- **Multiple statistics**

  sourcetype=netapp:perf
  | stats avg(read_ops) AS Read_OPs
  sparkline(avg(read_ops)) AS Read_Trend

- By another field

  Sourcetype=netapp:perf
  | stats avg(read_ops) AS Read_OPs
  sparkline(avg(read_ops)) AS Read_Trend
  by instance



Stats-Calculate Statistics Based On Field Values

Examples

- Calculate stats and rename

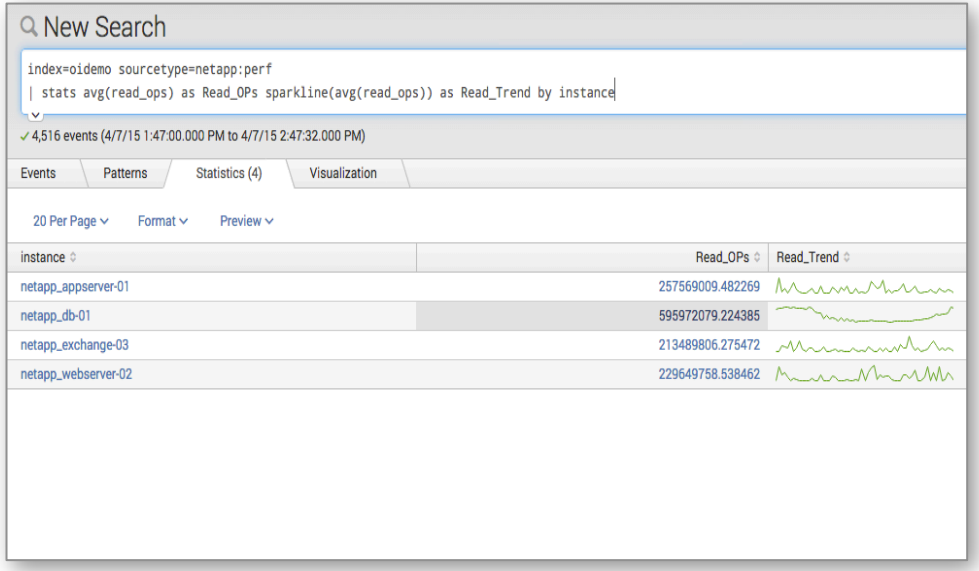
  sourcetype=netapp:perf
  | stats avg(read_ops) AS “Read OPs”

- Multiple statistics

  sourcetype=netapp:perf
  | stats avg(read_ops) AS Read_OPs
  sparkline(avg(read_ops)) AS Read_Trend

- **By another field**

  Sourcetype=netapp:perf
  | stats avg(read_ops) AS Read_OPs
  sparkline(avg(read_ops)) AS
  Read_Trend by instance



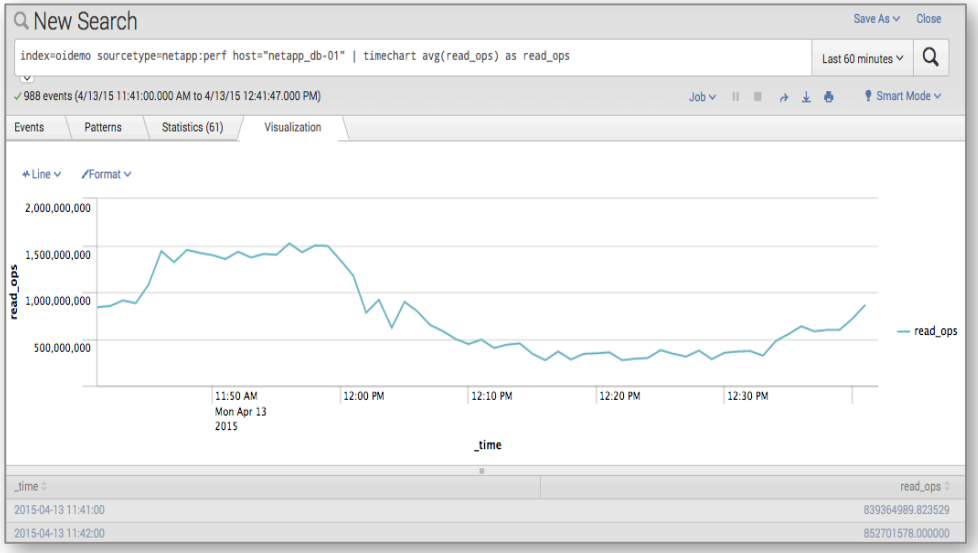
Timechart – Visualize Statistics Over Time

Examples

- **Visualize stats over time**
  sourcetype=netapp:perf
  | timechart avg(read_ops)

- Add a trendline
  sourcetype=netapp:perf
  | timechart avg(read_ops) as
  read_ops | trendline sma5(read_ops)

- Add a prediction overlay
  sourcetype=netapp:perf
  | timechart avg(read_ops) as
  read_ops | predict read_ops



Timechart – Visualize Statistics Over Time

Examples

- Visualize stats over time

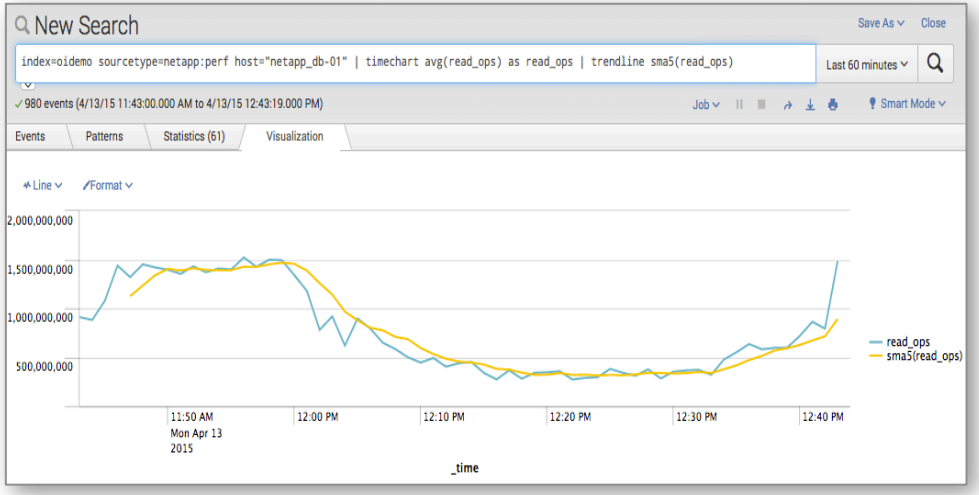
  sourcetype=netapp:perf
  | timechart avg(read_ops)

- **Add a trendline**

  sourcetype=netapp:perf
  | timechart avg(read_ops) as
  read_ops | trendline sma5(read_ops)

- Add a prediction overlay

  sourcetype=netapp:perf
  | timechart avg(read_ops) as
  read_ops | predict read_ops



Timechart – Visualize Statistics Over Time

Examples

- Visualize stats over time

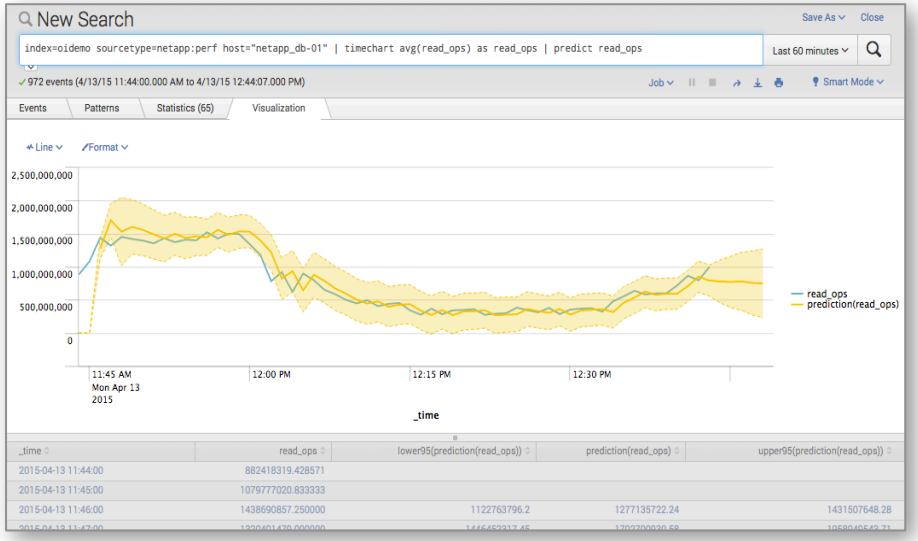
  sourcetype=netapp:perf
  | timechart avg(read_ops)

- Add a trendline

  sourcetype=netapp:perf
  | timechart avg(read_ops) as
  read_ops | trendline sma5(read_ops)

- **Add a prediction overlay**

  sourcetype=netapp:perf
  | timechart avg(read_ops) as
  read_ops | predict read_ops



Stats/Timechart – But Wait, There's More!

Splunk Search Quick Reference Guide

COMMON STATS FUNCTIONS

Common statistical functions used with the chart, stats, and timechart commands. Field names can be wildcarded, so avg(*delay) might calculate the average of the delay and xdelay fields.

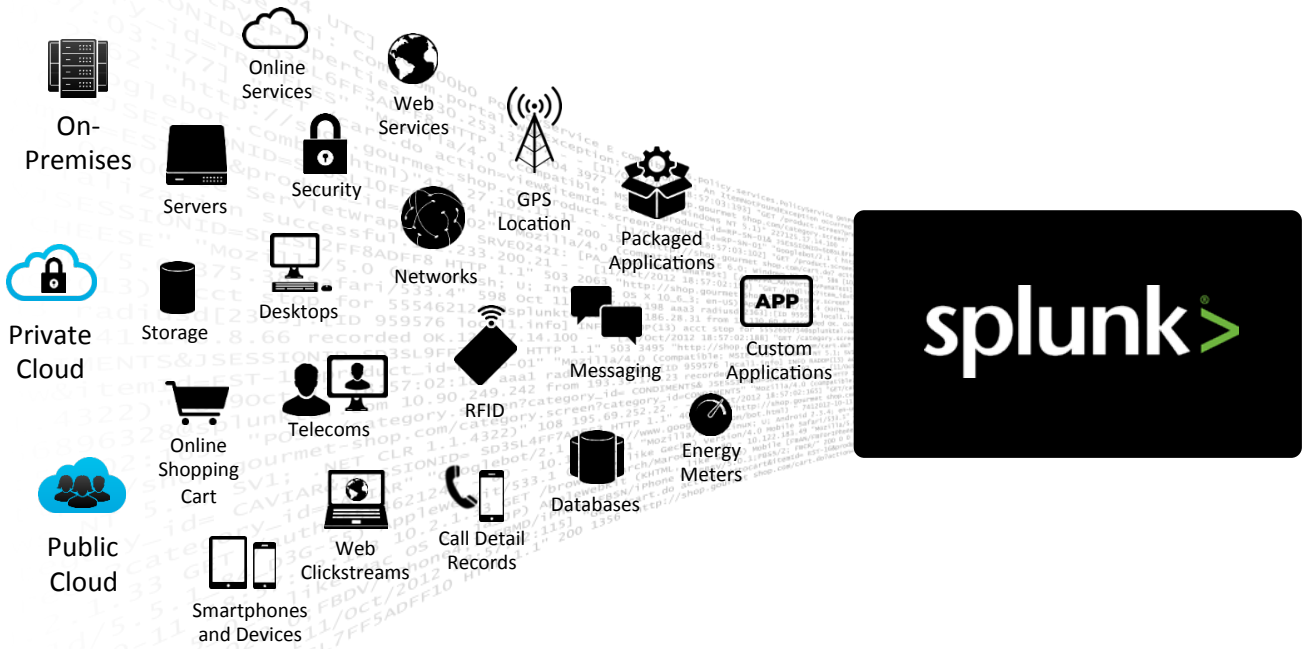
FUNCTION	DESCRIPTION
avg(X)	Returns the average of the values of field X.
count(X)	Returns the number of occurrences of the field X. To indicate a specific field value to match, format X as eval(field="value").
dc(X)	Returns the count of distinct values of the field X.
first(X)	Returns the first seen value of the field X. In general, the first seen value of the field is the chronologically most recent instance of field.
last(X)	Returns the last seen value of the field X.
list(X)	Returns the list of all values of the field X as a multi-value entry. The order of the values reflects the order of input events.
max(X)	Returns the maximum value of the field X. If the values of X are non-numeric, the max is found from lexicographic ordering.
median(X)	Returns the middle-most value of the field X.
min(X)	Returns the minimum value of the field X. If the values of X are non-numeric, the min is found from lexicographic ordering.
mode(X)	Returns the most frequent value of the field X.
perc<X>(Y)	Returns the X-th percentile value of the field Y. For example, perc5(total) returns the 5th percentile value of a field "total".
range(X)	Returns the difference between the max and min values of the field X.
stdev(X)	Returns the sample standard deviation of the field X.
stdevp(X)	Returns the population standard deviation of the field X.
sum(X)	Returns the sum of the values of the field X.
sumsq(X)	Returns the sum of the squares of the values of the field X.
values(X)	Returns the list of all distinct values of the field X as a multi-value entry. The order of the values is lexicographical.
var(X)	Returns the sample variance of the field X.

SPL Examples And Recipes

- Search and filter + creating/modifying fields
- Charting statistics and predicting values
- **Enriching and converging data sources**
- Visualize geographic data in real time
- Identifying transactions and anomalies
- Data exploration & finding relationships between fields

Converging Data Sources

Index Untapped Data: Any Source, Type, Volume



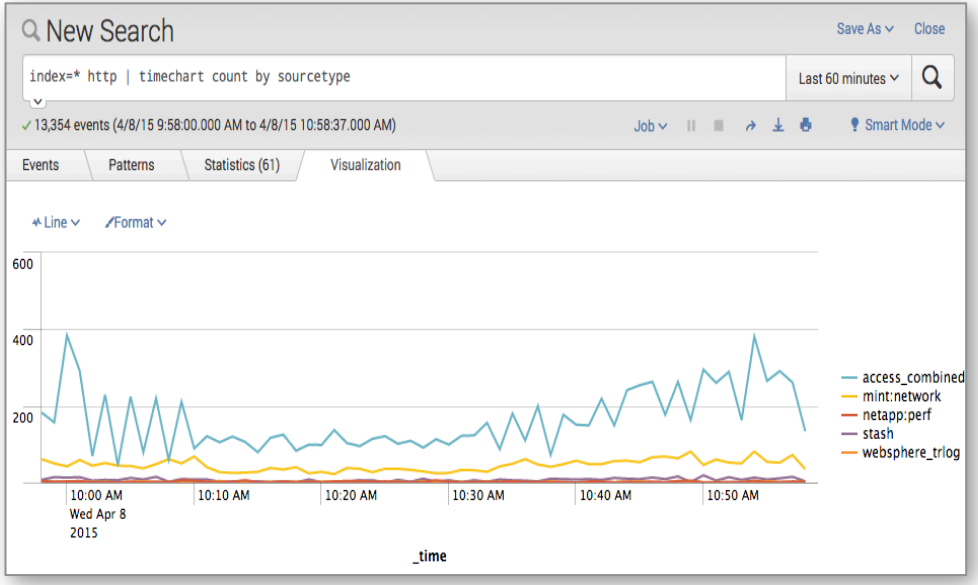
Ask Any Question

- Application Delivery
- IT Operations
- Security, Compliance and Fraud
- Business Analytics
- Industrial Data and the Internet of Things

Converging Data Sources

Examples

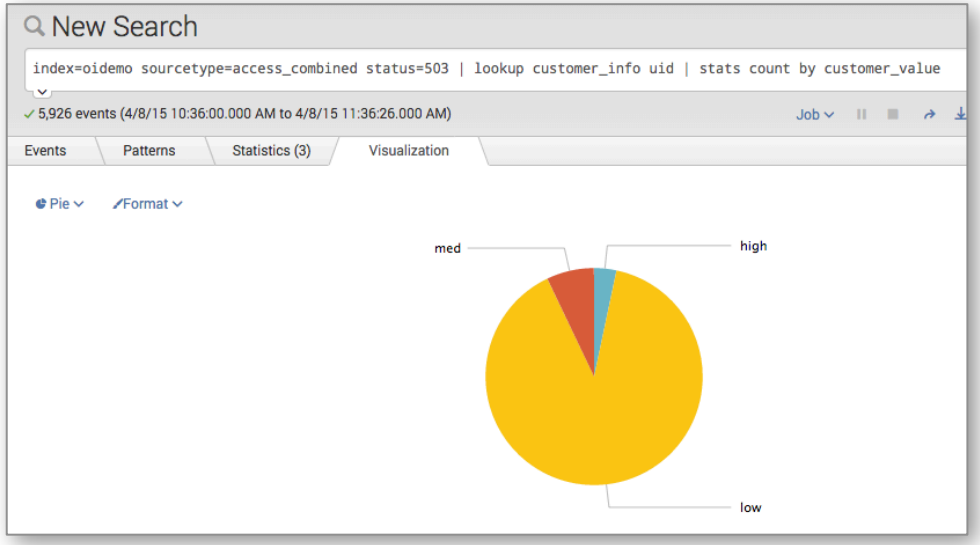
- **Implicit join on time**
  index=* http | timechart count by sourcetype
- Enrich data with lookup
  sourcetype=access_combined status=503
  | lookup customer_info uid |
  stats count by customer_value
- Append results from another search
  … | appendcols [search earliest=-1h
  sourcetype=Kepware units=W row=A
  | stats stdev(Value) as hr_stdev] …



Lookup – Converging Data Sources

Examples

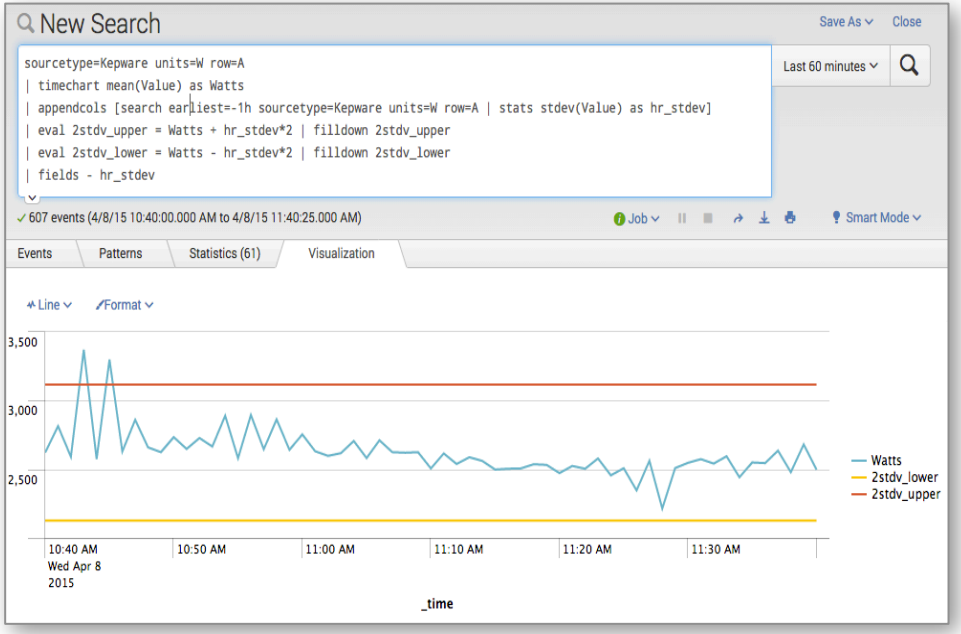
- Implicit join on time
  index=* http | timechart count by sourcetype
- **Enrich data with lookup**
  sourcetype=access_combined status=503 | lookup customer_info uid | stats count by customer_value
- Append results from another search
  … | appendcols [search earliest=-1h sourcetype=Kepware units=W row=A | stats stdev(Value) as hr_stdev] …



Appendcols – Converging Data Sources

Examples

- Implicit join on time
  index=* http | timechart count by sourcetype
- Enrich data with lookup
  sourcetype=access_combined status=503 | lookup customer_info uid | stats count by customer_value
- **Append results from another search**
  … | appendcols [search earliest=-1h sourcetype=Kepware units=W row=A | stats stdev(Value) as hr_stdev] …



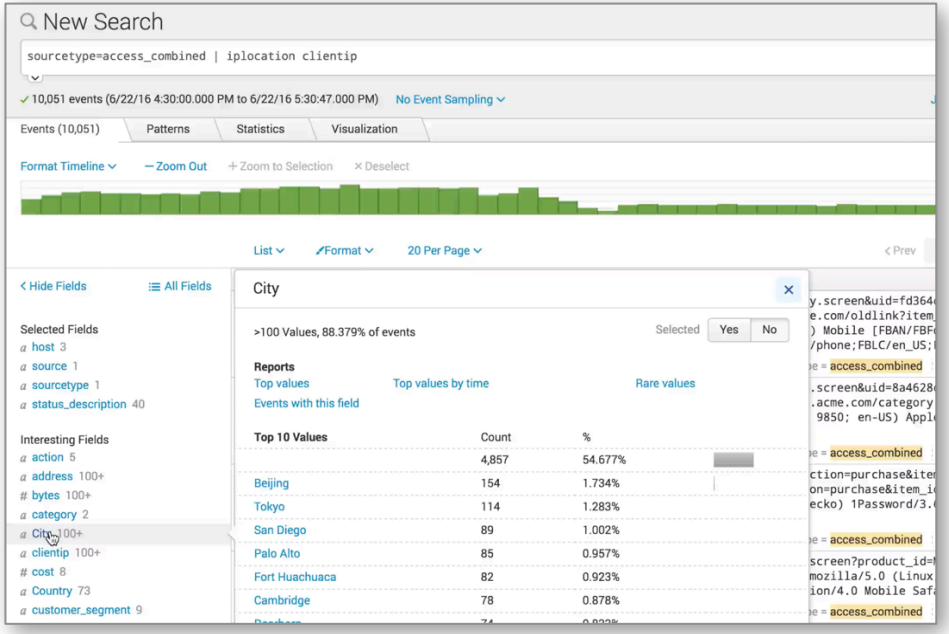
SPL Examples And Recipes

- Search and filter + creating/modifying fields
- Charting statistics and predicting values
- Enriching and converging data sources
- **Visualize geographic data in real time**
- Identifying transactions and anomalies
- Data exploration & finding relationships between fields

iplocation – Geographic Data

Examples

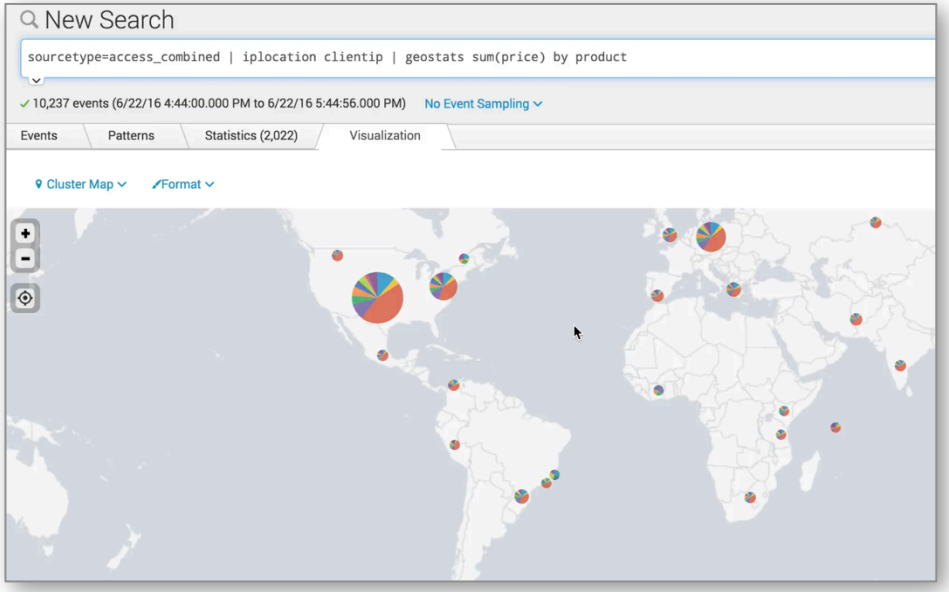
- **Assign Lat/Lon to IP addresses**
  … | iplocation clientip
- Visualize statistics geographically
  … | geostats sum(price) by product
- Use custom choropleths
  … | geom <featureCollection> <featureId>
- Track object movements
  … | table _time latitude longitude vehicleId



geostats – Geographic Data

Examples

- Assign Lat/Lon to IP addresses
  … | iplocation clientip
- **Visualize statistics geographically**
  … | geostats sum(price) by product
- Use custom choropleths
  … | geom <featureCollection> <featureId>
- Track object movements
  … | table _time latitude longitude vehicleId



geom – Geographic Data

Examples

- Assign Lat/Lon to IP addresses
  … | iplocation clientip
- Visualize statistics geographically
  … | geostats sum(price) by product
- **Use custom choropleths**
  … | geom <featureCollection> <featureId>
- Track object movements
  … | table _time latitude longitude vehicleId

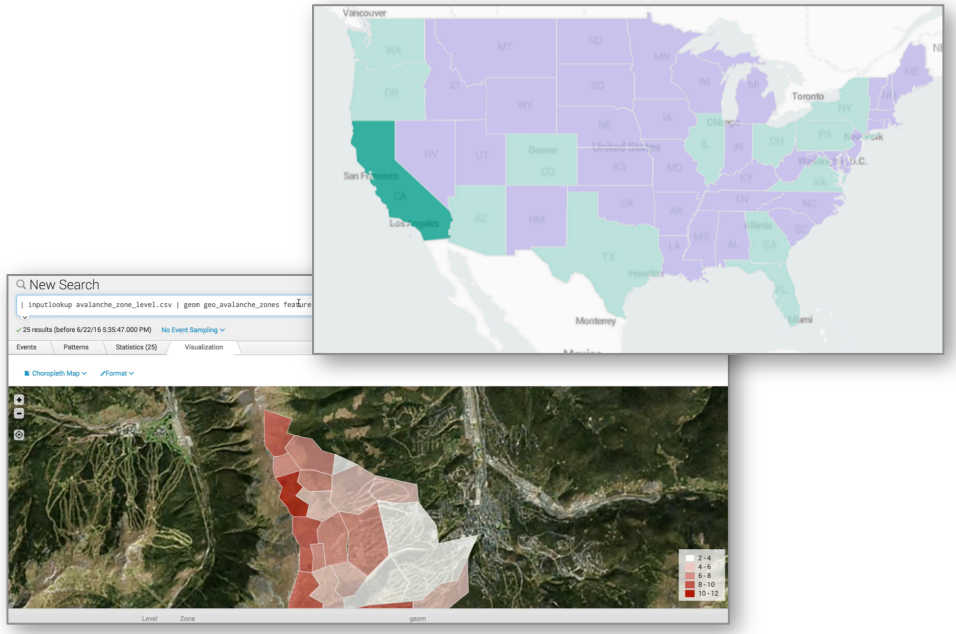
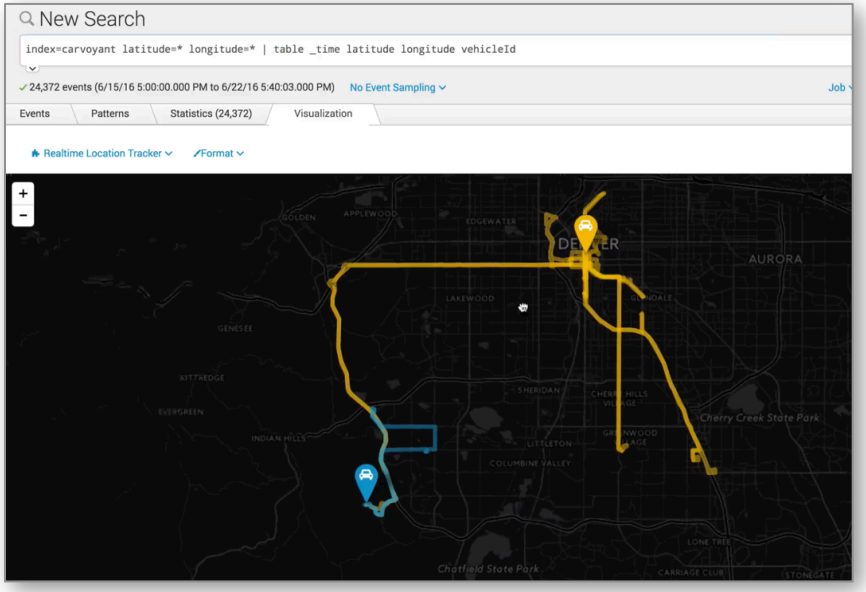


Table – Geographic Data

Examples

- Assign Lat/Lon to IP addresses
  … | iplocation clientip
- Visualize statistics geographically
  … | geostats sum(price) by product
- Use custom choropleths
  … | geom <featureCollection> <featureId>
- **Track object movements**
  … | table _time latitude longitude vehicleId



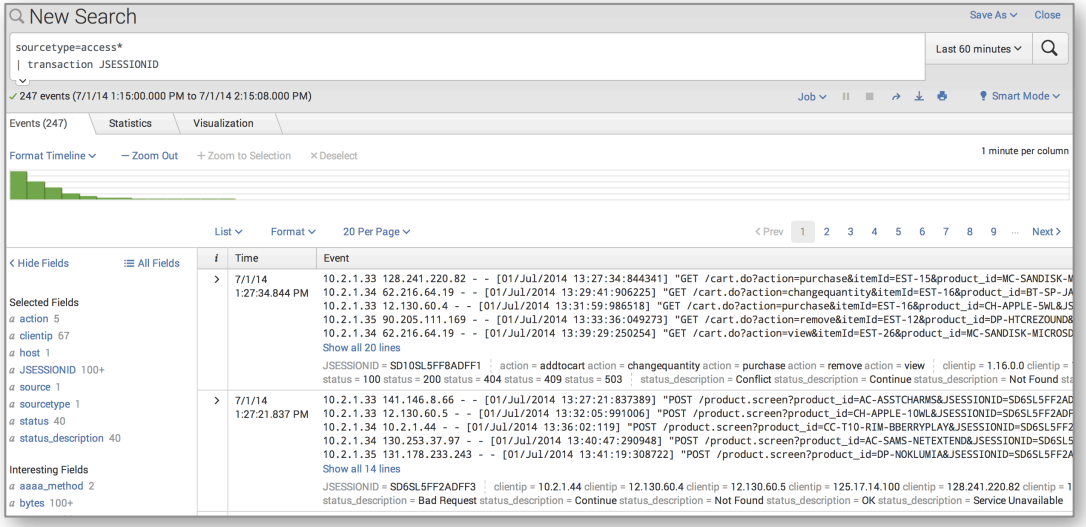
SPL Examples And Recipes

- Search and filter + creating/modifying fields
- Charting statistics and predicting values
- Enriching and Converging Data Sources
- Visualize Geographic data in real-time
- **Identifying transactions and anomalies**
- Data exploration & finding relationships between fields

Transaction – Group Related Events Spanning Time

Examples

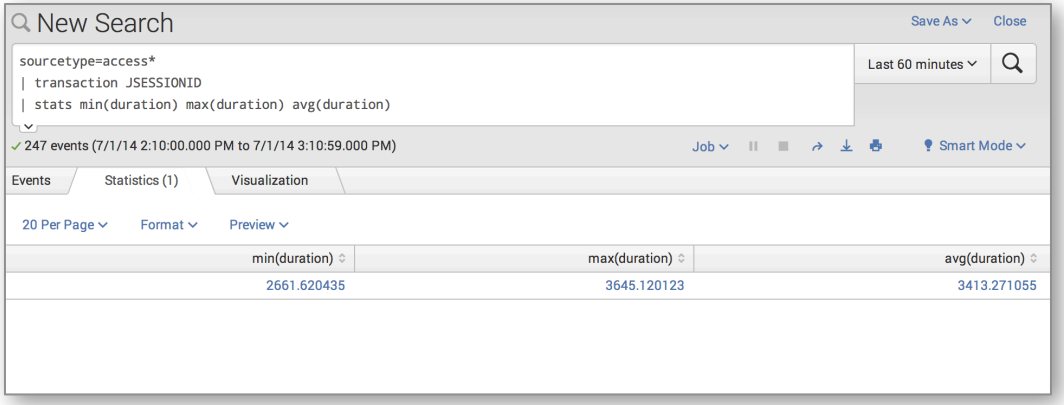
- **Group by session ID**
  sourcetype=access*
  | transaction JSESSIONID
- Calculate session durations
  sourcetype=access*
  | transaction JSESSIONID
  | stats min(duration) max(duration) avg(duration)
- Stats is better
  sourcetype=access*
  | stats min(_time) AS earliest max(_time) AS latest by JSESSIONID
  | eval duration=latest-earliest
  | stats min(duration) max(duration) avg(duration)



Transaction – Group Related Events Spanning Time

Examples

- Group by session ID
  sourcetype=access*
  | transaction JSESSIONID
- **Calculate session durations**
  sourcetype=access*
  | transaction JSESSIONID
  | stats min(duration) max(duration) avg(duration)
- Stats is better
  sourcetype=access*
  | stats min(_time) AS earliest max(_time) AS latest by JSESSIONID
  | eval duration=latest-earliest
  | stats min(duration) max(duration) avg(duration)

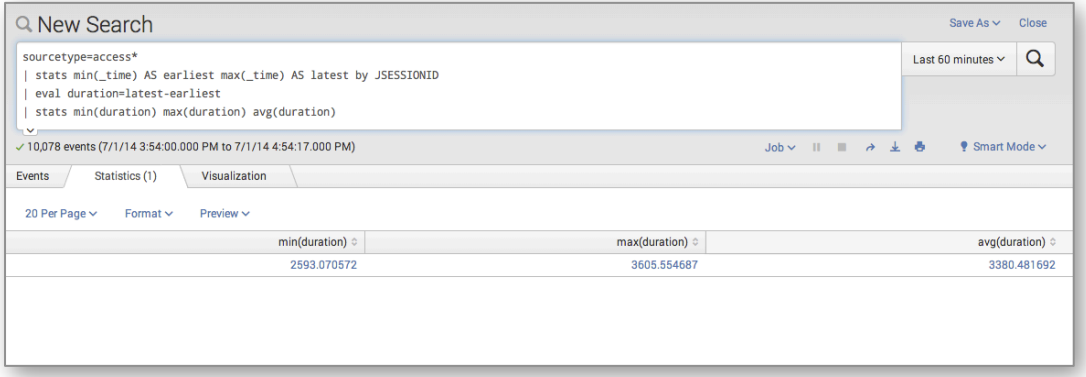


splunk> .conf2016

Transaction – Group Related Events Spanning Time

Examples

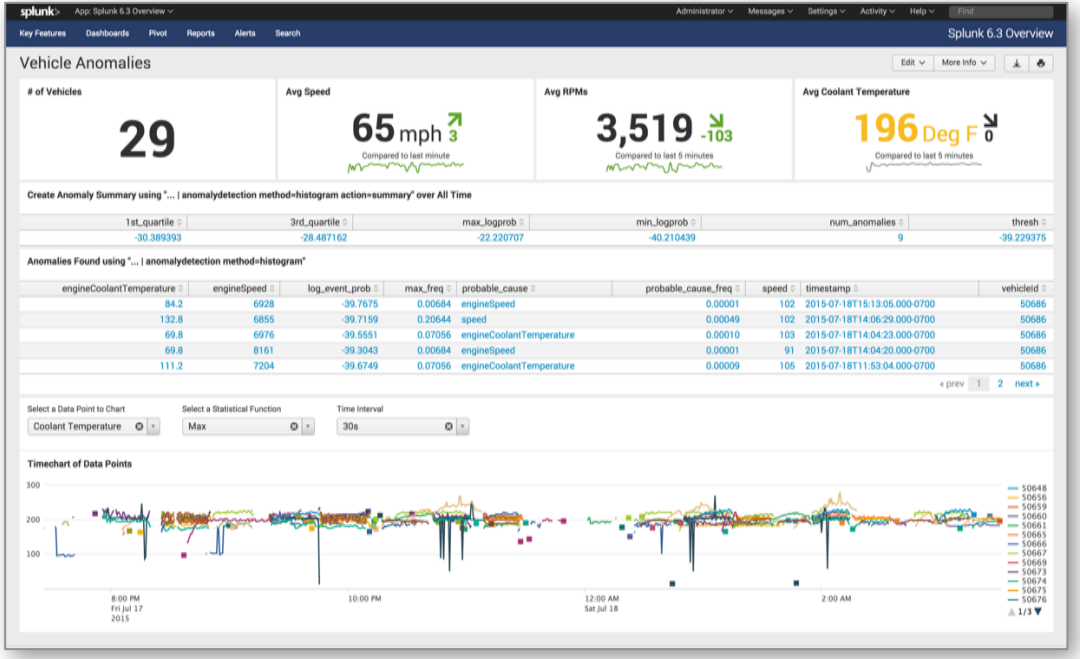
- Group by session ID
  sourcetype=access*
  | transaction JSESSIONID
- Calculate session durations
  sourcetype=access*
  | transaction JSESSIONID
  | stats min(duration) max(duration) avg(duration)
- **Stats is better**
  sourcetype=access*
  | stats min(_time) AS earliest max(_time) AS latest by JSESSIONID
  | eval duration=latest-earliest
  | stats min(duration) max(duration) avg(duration)



Anomaly Detection – Find Anomalies In Your Data

Examples

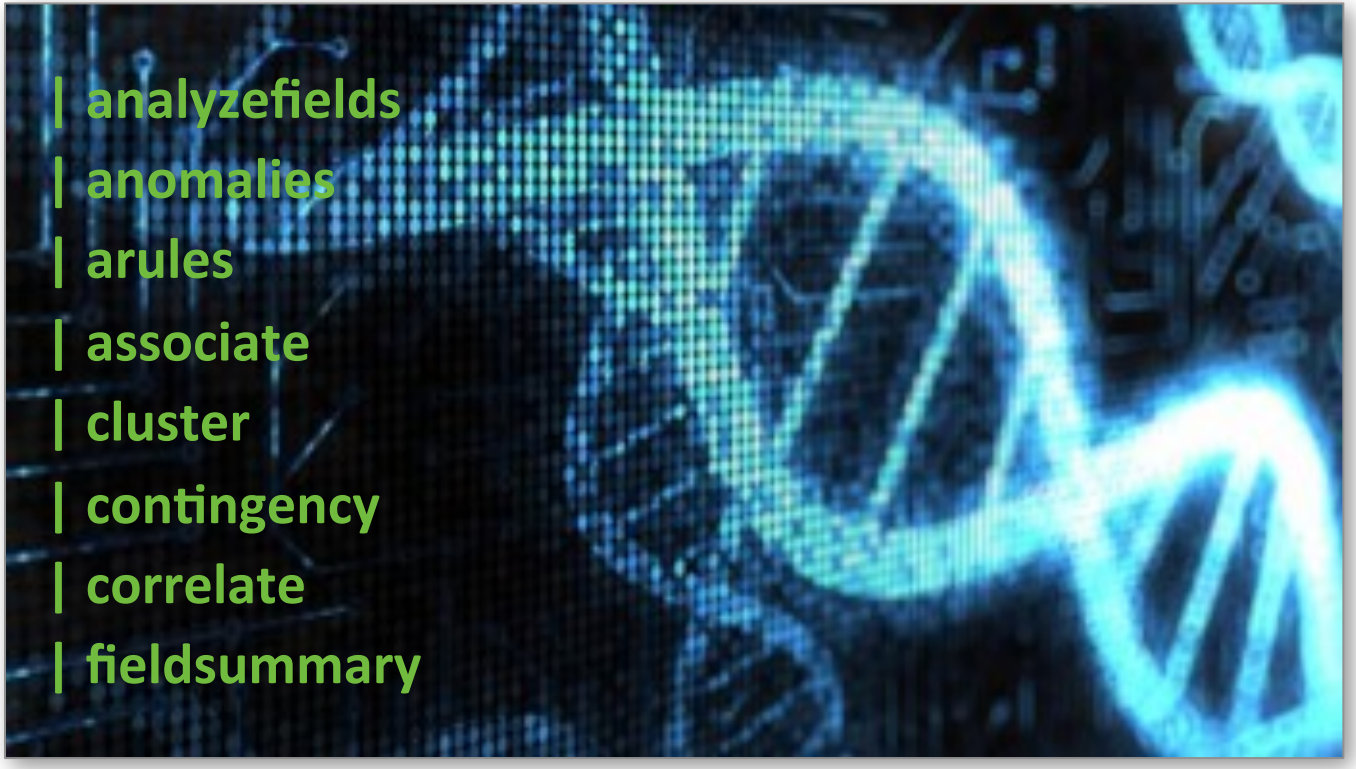
- Find anomalies
  | inputlookup car_data.csv | anomalydetection
- Summarize anomalies
  | inputlookup car_data.csv | anomalydetection action=summary
- Use IQR and remove outliers
  | inputlookup car_data.csv | anomalydetection method=iqr action=remove



SPL Examples And Recipes

- Search and filter + creating/modifying fields
- Charting statistics and predicting values
- Enriching and Converging Data Sources
- Visualize Geographic data in real-time
- Identifying transactions and anomalies
- **Data exploration & finding relationships between fields**

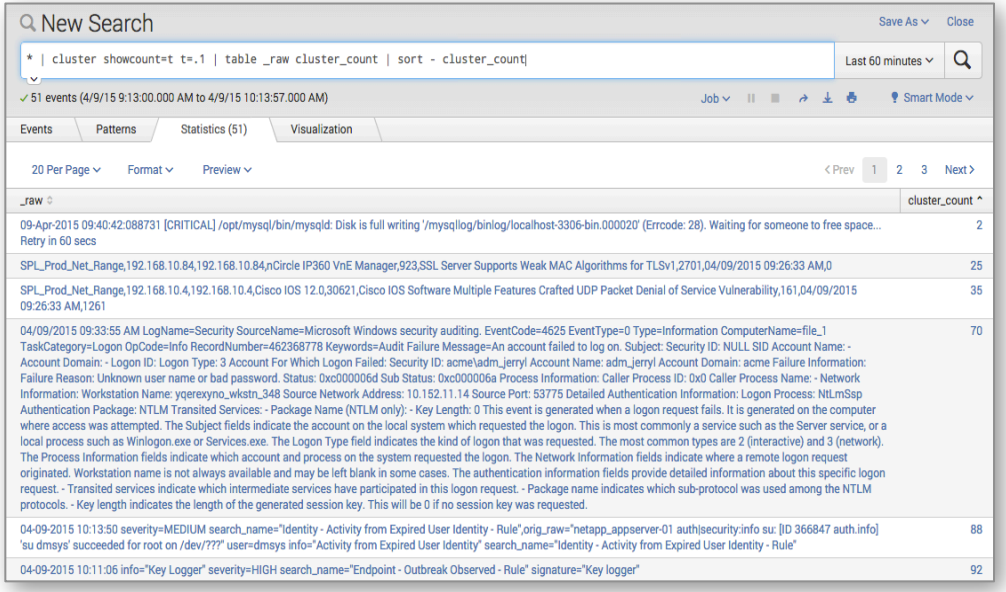
Data Exploration



Cluster – Exploring Your Data

Examples

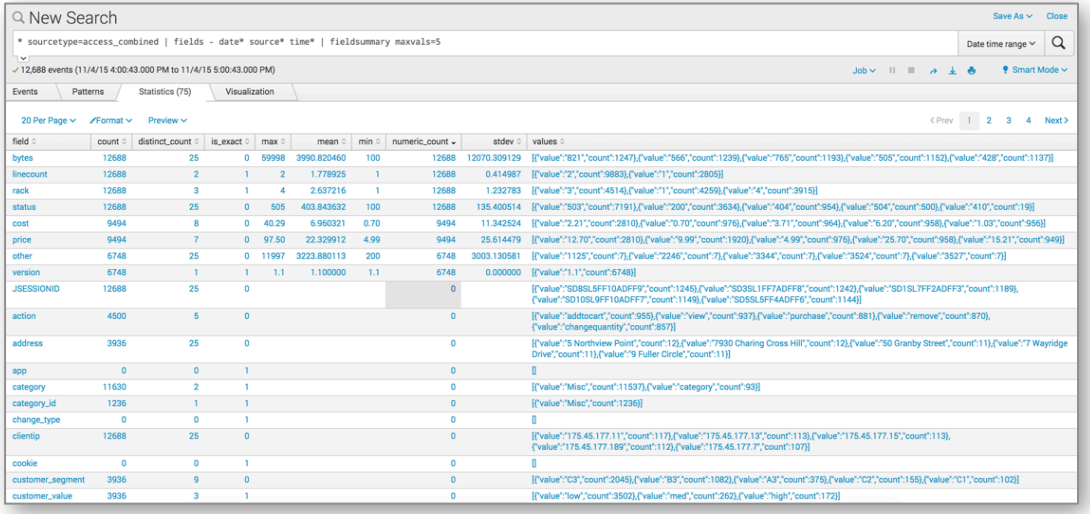
- **Find most/least common events**
  *** | cluster showcount=t t=.1**
  | table _raw cluster_count
- Display Summary of Fields.
  sourcetype=access_combined
  | fields – date* source* time*
  | fieldsummary maxvals=5
- Show patterns of co-occurring fields.
  sourcetype=access_combined
  | fields – date* source* time* | correlate
- View field relationships
  sourcetype=access_combined
  | contingency uri status
- Find predictors of fields
  sourcetype=access_combined
  | analyzefields classfield=status



Cluster – Exploring Your Data

Examples

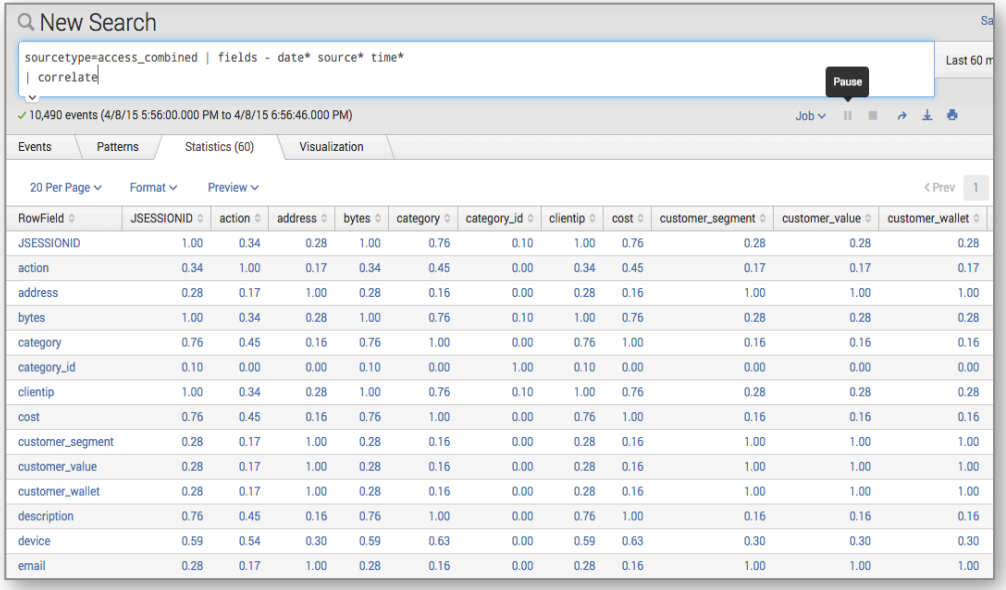
- Find most/least common events
  * | cluster showcount=t t=.1
  | table _raw cluster_count
- **Display Summary of Fields.**
  sourcetype=access_combined
  | fields – date* source* time*
  | fieldsummary maxvals=5
- Show patterns of co-occurring fields.
  sourcetype=access_combined
  | fields – date* source* time* | correlate
- View field relationships
  sourcetype=access_combined
  | contingency uri status
- Find predictors of fields
  sourcetype=access_combined
  | analyzefields classfield=status



Correlate – Exploring Your Data

Examples

- Find most/least common events
  * | cluster showcount=t t=.1
  | table _raw cluster_count
- Display Summary of Fields.
  sourcetype=access_combined
  | fields – date* source* time*
  | fieldsummary maxvals=5
- **Show patterns of co-occurring fields.**
  sourcetype=access_combined
  | fields – date* source* time* | correlate
- View field relationships
  sourcetype=access_combined
  | contingency uri status
- Find predictors of fields
  sourcetype=access_combined
  | analyzefields classfield=status

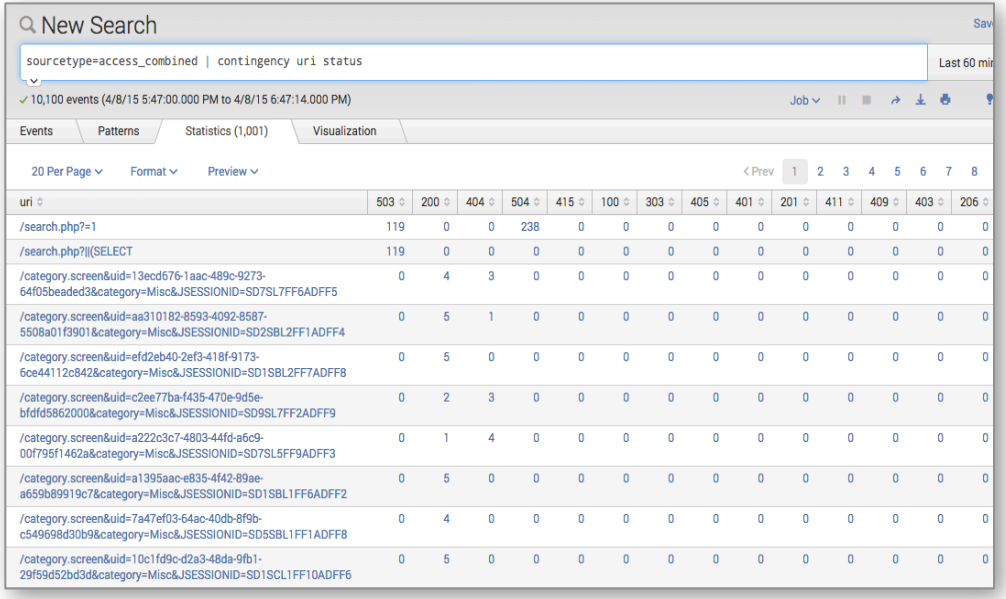


RowField	JSESSIONID	action	address	bytes	category	category_id	clientip	cost	customer_segment	customer_value	customer_wallet
JSESSIONID	1.00	0.34	0.28	1.00	0.76	0.10	1.00	0.76	0.28	0.28	0.28
action	0.34	1.00	0.17	0.34	0.45	0.00	0.34	0.45	0.17	0.17	0.17
address	0.28	0.17	1.00	0.28	0.16	0.00	0.28	0.16	1.00	1.00	1.00
bytes	1.00	0.34	0.28	1.00	0.76	0.10	1.00	0.76	0.28	0.28	0.28
category	0.76	0.45	0.16	0.76	1.00	0.00	0.76	1.00	0.16	0.16	0.16
category_id	0.10	0.00	0.00	0.10	0.00	1.00	0.10	0.00	0.00	0.00	0.00
clientip	1.00	0.34	0.28	1.00	0.76	0.10	1.00	0.76	0.28	0.28	0.28
cost	0.76	0.45	0.16	0.76	1.00	0.00	0.76	1.00	0.16	0.16	0.16
customer_segment	0.28	0.17	1.00	0.28	0.16	0.00	0.28	0.16	1.00	1.00	1.00
customer_value	0.28	0.17	1.00	0.28	0.16	0.00	0.28	0.16	1.00	1.00	1.00
customer_wallet	0.28	0.17	1.00	0.28	0.16	0.00	0.28	0.16	1.00	1.00	1.00
description	0.76	0.45	0.16	0.76	1.00	0.00	0.76	1.00	0.16	0.16	0.16
device	0.59	0.54	0.30	0.59	0.63	0.00	0.59	0.63	0.30	0.30	0.30
email	0.28	0.17	1.00	0.28	0.16	0.00	0.28	0.16	1.00	1.00	1.00

Contingency – Exploring Your Data

Examples

- Find most/least common events
  * | cluster showcount=t t=.1
  | table _raw cluster_count
- Display Summary of Fields.
  sourcetype=access_combined
  | fields – date* source* time*
  | fieldsummary maxvals=5
- Show patterns of co-occurring fields.
  sourcetype=access_combined
  | fields – date* source* time* | correlate
- **View field relationships**
  sourcetype=access_combined
  | contingency uri status
- Find predictors of fields
  sourcetype=access_combined
  | analyzefields classfield=status

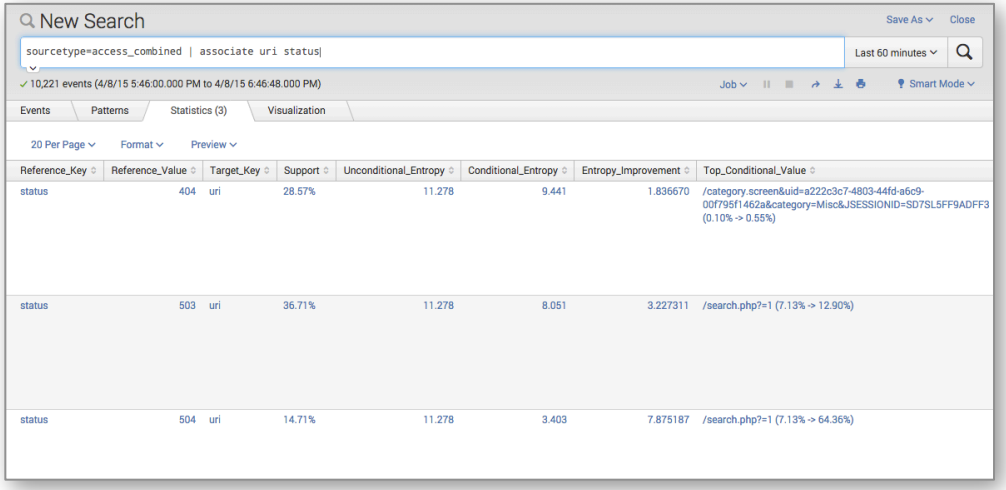


uri	503	200	404	504	415	100	303	405	401	201	411	409	403	206
/search.php?=1	119	0	0	238	0	0	0	0	0	0	0	0	0	0
/search.php?l(SELECT	119	0	0	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=13ecd676-1aac-489c-9273-64f05beaded3&category=Misc&JSESSIONID=SD7SL7FF6ADFF5	0	4	3	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=aa310182-8593-4092-8587-5508a01f3901&category=Misc&JSESSIONID=SD2SBL2FF1ADFF4	0	5	1	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=efd2eb40-2ef3-418f-9173-6ce44112c842&category=Misc&JSESSIONID=SD1SBL2FF7ADFF8	0	5	0	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=c2ee77ba-f435-470e-9d5e-bfdfd5862000&category=Misc&JSESSIONID=SD9SL7FF2ADFF9	0	2	3	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=a222c3c7-4803-44fd-a6c9-00f795f1462a&category=Misc&JSESSIONID=SD7SL5FF9ADFF3	0	1	4	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=a1395aac-e835-4f42-89ae-a659b89919c7&category=Misc&JSESSIONID=SD1SBL1FF6ADFF2	0	5	0	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=7a47ef03-64ac-40db-8f9b-c549698d30b9&category=Misc&JSESSIONID=SD5SBL1FF1ADFF8	0	4	0	0	0	0	0	0	0	0	0	0	0	0
/category.screen&uid=10c1fd9c-d2a3-48da-9fb1-29f59d52bd3d&category=Misc&JSESSIONID=SD1SCL1FF10ADFF6	0	5	0	0	0	0	0	0	0	0	0	0	0	0

Analyzefields – Exploring Your Data

Examples

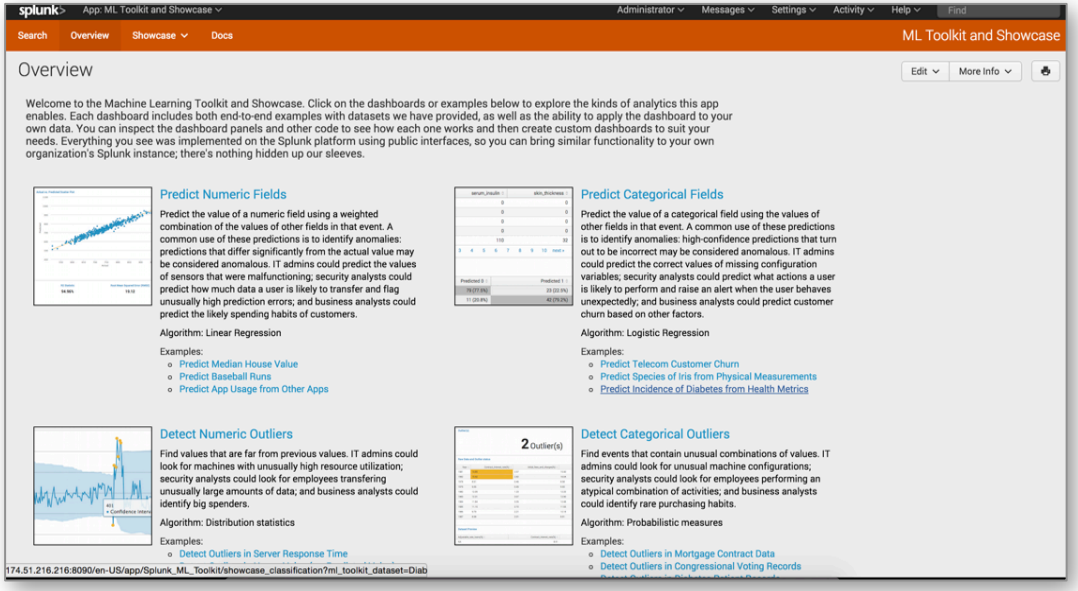
- Find most/least common events
  * | cluster showcount=t t=.1
  | table _raw cluster_count
- Display Summary of Fields.
  sourcetype=access_combined
  | fields – date* source* time*
  | fieldsummary maxvals=5
- Show patterns of co-occurring fields.
  sourcetype=access_combined
  | fields – date* source* time* | correlate
- View field relationships
  sourcetype=access_combined
  | contingency uri status
- **Find predictors of fields**
  sourcetype=access_combined
  | analyzefields classfield=status



Machine Learning Toolkit And Showcase

Examples

- Predict Numeric Fields
- Predict Categorical Fields
- Detect Numerical Outliers
- Detect Categorical Outliers
- Forecast Time Series
- Cluster Events



Custom Commands

Custom Commands

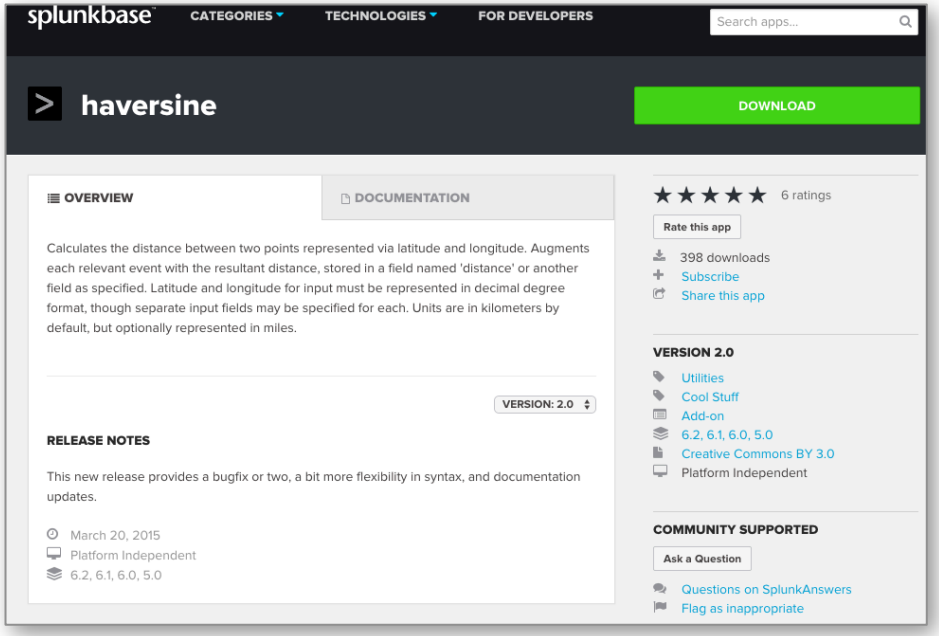
- What is a Custom Command?
  - “| **haversine** origin="47.62,-122.34" outputField=dist lat lon”
- Why do we use Custom Commands?
  - Run other/external algorithms on your Splunk data
  - Save time munging data (see Timewrap!)
  - Because you can!
- Create your own or download as Apps
  - Haversine (Distance between two GPS coords)
  - Timewrap (Enhanced Time overlay)
  - Levenshtein (Fuzzy string compare)
  - Base64 (Encode/Decode)

Custom Commands – Haversine

Examples

- **Download and install App Haversine**
- *Read documentation* then use in SPL!

```
sourcetype=access*
| iplocation clientip
| search City=A*
| haversine origin="47.62,-122.34"
units=mi outputField=dist lat lon
| table clientip, City, dist, lat, lon
```

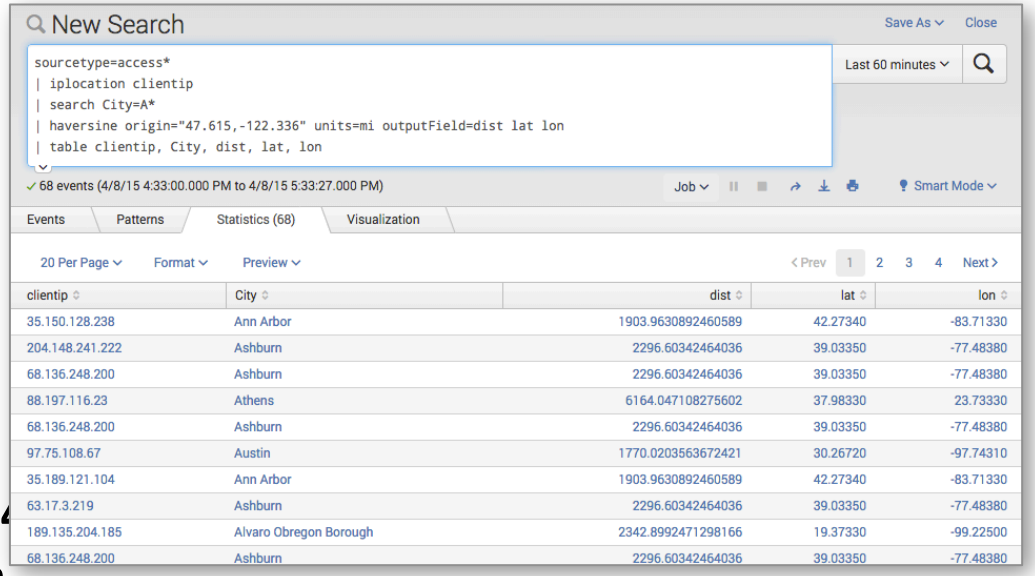


Custom Commands – Haversine

Examples

- Download and install App Haversine
- ***Read documentation*** **then use in SPL!**

```
sourcetype=access*
| iplocation clientip
| search City=A*
| haversine origin="47.62,-122.34
units=mi outputField=dist lat lon
| table clientip, City, dist, lat, lon
```



For More Information

- Additional information can be found in:
  - Search Manual
  - Blogs
  - Answers
  - Operational Intelligence Cookbook – available for purchase
  - Exploring Splunk

What Now?

Related breakout sessions and activities…

- Power of SPL Booth

Q & A

THANK YOU

splunk>