Power Of SPL

Stephen Luedtke | Sr. Technical Marketing Manager

September 27, 2017 | Washington, DC
Forward-Looking Statements

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 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.

Splunk, Splunk>, Listen to Your Data, The Engine for Machine Data, Splunk Cloud, Splunk Light and SPL are trademarks and registered trademarks of Splunk Inc. in the United States and other countries. All other brand names, product names, or trademarks belong to their respective owners. © 2017 Splunk Inc. All rights reserved.
1. Overview & Anatomy of a Search
   • Quick refresher on search language and structure

2. SPL Commands and Examples
   • Searching, charting, converging, mapping, transactions, anomalies, exploring

3. Custom Commands
   • Extend the capabilities of SPL

4. Q&A
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 machine learning such as anomaly detection.
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
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
▶ Map geographic data in real time
▶ Identifying anomalies
▶ Transactions
▶ Data exploration & finding relationships between fields
▶ Custom commands
SPL Examples And Recipes

- Find the needle in the haystack
- Charting statistics and predicting values
- Enriching and converging data sources
- Map geographic data in real time
- Identifying anomalies
- Transactions
- Data exploration & finding relationships between fields
- Custom Commands
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)`
Keyword search: sourcetype=access* http

Filter: sourcetype=access* http host=webserver-02

Combined: sourcetype=access* http host=webserver-02 (503 OR 504)
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 = device." - ".clientip
```
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 = device." - ".clientip
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 = device.” - ”.clientip
# Eval – Just Getting Started!

**Splunk Search Quick Reference Guide**

## Common 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 some of the functions used with the eval command. You can also use basic arithmetic operators (+, -, /, *) string concatenation (e.g. "...") eval name = last "+" first", and Boolean operations (AND OR NOT XOR <= <= >= == LIKE).

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs(X)</td>
<td>Returns the absolute value of X.</td>
<td>abs(number)</td>
</tr>
<tr>
<td>case(X, &quot;Y&quot;,...)</td>
<td>Takes pairs of arguments X and Y, where X arguments are</td>
<td>case(error == 404, &quot;Not found&quot;, error == 500,&quot;Internal Server Error&quot;, error == 200, &quot;OK&quot;)</td>
</tr>
<tr>
<td>ceil(X)</td>
<td>Ceiling of a number X.</td>
<td>ceil(1.9)</td>
</tr>
<tr>
<td>cidrmatch(&quot;X&quot;,Y)</td>
<td>Identifies IP addresses that belong to a particular subnet.</td>
<td>cidrmatch(&quot;123.132.32.0/25&quot;,ip)</td>
</tr>
<tr>
<td>coalesce(X,...)</td>
<td>Returns the first value that is not null.</td>
<td>coalesce(null(), &quot;Returned val&quot;, null())</td>
</tr>
<tr>
<td>cos(X)</td>
<td>Calculates the cosine of X.</td>
<td>n=cos(0)</td>
</tr>
<tr>
<td>exact(X)</td>
<td>Evaluates an expression X using double precision floating point arithmetic.</td>
<td>exact(3.14*num)</td>
</tr>
<tr>
<td>exp(X)</td>
<td>Returns e^X.</td>
<td>exp(3)</td>
</tr>
<tr>
<td>if(X,Y,Z)</td>
<td>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.</td>
<td>if(error==200, &quot;OK&quot;, &quot;Error&quot;)</td>
</tr>
<tr>
<td>isbool(X)</td>
<td>Returns TRUE if X is Boolean.</td>
<td>isbool(field)</td>
</tr>
<tr>
<td>isint(X)</td>
<td>Returns TRUE if X is an integer.</td>
<td>isint(field)</td>
</tr>
<tr>
<td>isnull(X)</td>
<td>Returns TRUE if X is NULL.</td>
<td>isnull(field)</td>
</tr>
<tr>
<td>isstr()</td>
<td>Returns TRUE if X is a string.</td>
<td>isstr(field)</td>
</tr>
<tr>
<td>len(X)</td>
<td>This function returns the character length of a string X.</td>
<td>len(field)</td>
</tr>
<tr>
<td>like(X,&quot;Y&quot;)</td>
<td>Returns TRUE if and only if X is like the SQLite pattern in Y.</td>
<td>like(field, &quot;addr%&quot;)</td>
</tr>
</tbody>
</table>
SPL Examples And Recipes

- Find the needle in the haystack
- Charting statistics and predicting values
- Enriching and converging data sources
- Map geographic data in real time
- Identifying anomalies
- Transactions
- Data exploration & finding relationships between fields
- Custom commands
Stats, Timechart, Eventstats, Streamstats
Stats – Calculate Statistics Based on Field Values

Examples

▶ Calculate stats and rename

   Index=power_of_spl

   | stats avg(bytes) AS “Avg Bytes”

▶ Multiple statistics

   index=power_of_spl | stats avg(bytes) AS bytes
   sparkline(avg(bytes)) AS Bytes_Trend min(bytes) max(bytes)

▶ By another field

   index=power_of_spl
   | stats avg(bytes) AS avg_bytes
   sparkline(avg(bytes)) AS Bytes_Trend min(bytes) max(bytes) by clientip | sort - avg_bytes
Stats – Calculate Statistics Based on Field Values

Examples

► Calculate stats and rename
  Index=power_of_spl
  | stats avg(bytes) AS “Avg Bytes”

► Multiple statistics
  index=power_of_spl | stats avg(bytes) AS bytes
  sparkline(avg(bytes)) AS Bytes_Trend
  min(bytes) max(bytes)

► By another field
  index=power_of_spl
  | stats avg(bytes) AS avg_bytes
  sparkline(avg(bytes)) AS Bytes_Trend
  min(bytes) max(bytes) by clientip | sort -avg_bytes
Stats – Calculate Statistics Based on Field Values

Examples

- Calculate stats and rename
  Index=power_of_spl
  | stats avg(bytes) AS “Avg Bytes”

- Multiple statistics
  index=power_of_spl | stats avg(bytes) AS bytes
  sparkline(avg(bytes)) AS Bytes_Trend min(bytes) max(bytes)

- By another field
  index=power_of_spl
  | stats avg(bytes) AS avg_bytes
  sparkline(avg(bytes)) AS Bytes_Trend min(bytes) max(bytes) by clientip | sort - avg_bytes
**Timechart** – Visualize Statistics Over Time

**Examples**

- **Visualize stats over time**
  
  ```
  index=power_of_spl | timechart avg(bytes)
  ```

- **Add a trendline**
  
  ```
  index=power_of_spl | timechart avg(bytes) as bytes | trendline sma5(bytes)
  ```

- **Add a prediction overlay**
  
  ```
  index=power_of_spl | timechart avg(bytes) as bytes | predict future_timespan=5 bytes
  ```
Timechart – Visualize Statistics Over Time

Examples

- Visualize stats over time
  
  ```splunk
  index=power_of_spl
  | timechart avg(bytes)
  ```

- Add a trendline
  
  ```splunk
  index=power_of_spl
  | timechart avg(bytes) as bytes
  | trendline sma5(bytes)
  ```

- Add a prediction overlay
  
  ```splunk
  index=power_of_spl
  | timechart avg(bytes) as bytes
  | predict future_timespan=5 bytes
  ```
Timechart – Visualize Statistics Over Time

Examples

▶ Visualize stats over time

index=power_of_spl
| timechart avg(bytes)

▶ Add a trendline

index=power_of_spl
| timechart avg(bytes) as bytes
| trendline sma5(bytes)

▶ Add a prediction overlay

index=power_of_spl
| timechart avg(bytes) as bytes
| predict future_timespan=5 bytes
Streamstats – Cumulative/Running Totals Statistics

Examples

- Cumulative/Running Totals
  ```
  index=power_of_spl
  | reverse
  | streamstats sum(bytes) AS sum_bytes
  | timechart latest(sum_bytes) as "Total Bytes"
  ```

- Summary Statistics
  ```
  index=power_of_spl
  | eventstats avg(bytes) AS overall_avg_bytes
  | stats avg(bytes) as clientip_avg_bytes by clientip overall_avg_bytes
  ```
Streamstats – Cumulative/Running Totals Statistics

Examples

- **Cumulative/Running Totals**
  
  ```
  index=power_of_spl
  | reverse
  | streamstats sum(bytes) AS sum_bytes
  | timechart latest(sum_bytes) as "Total Bytes"
  ```

- **Summary Statistics**
  
  ```
  index=power_of_spl
  |
  | eventstats avg(bytes) AS overall_avg_bytes
  | stats avg(bytes) as clientip_avg_bytes by clientip overall_avg_bytes
  ```
Stats/Timechart – But Wait, There’s More!

Splunk Search Quick Reference Guide

<table>
<thead>
<tr>
<th>Common Stats Functions</th>
<th>Common statistical functions used with the chart, stats, and timechart commands. Field names can be wildcarded, so <code>avg(*delay)</code> might calculate the average of the delay and xdelay fields.</th>
</tr>
</thead>
<tbody>
<tr>
<td>avg(X)</td>
<td>Returns the average of the values of field X.</td>
</tr>
<tr>
<td>count(X)</td>
<td>Returns the number of occurrences of the field X. To indicate a specific field value to match, format X as eval(field=&quot;value&quot;).</td>
</tr>
<tr>
<td>dc(X)</td>
<td>Returns the count of distinct values of the field X.</td>
</tr>
<tr>
<td>earliest(X)</td>
<td>Returns the chronologically earliest seen value of X.</td>
</tr>
<tr>
<td>latest(X)</td>
<td>Returns the chronologically latest seen value of X.</td>
</tr>
<tr>
<td>max(X)</td>
<td>Returns the maximum value of the field X. If the values of X are non-numeric, the max is found from alphabetical ordering.</td>
</tr>
<tr>
<td>median(X)</td>
<td>Returns the middle-most value of the field X.</td>
</tr>
<tr>
<td>min(X)</td>
<td>Returns the minimum value of the field X. If the values of X are non-numeric, the min is found from alphabetical ordering.</td>
</tr>
<tr>
<td>mode(X)</td>
<td>Returns the most frequent value of the field X.</td>
</tr>
<tr>
<td>perc&lt;X&gt;(Y)</td>
<td>Returns the X-th percentile value of the field Y. For example, perc5(total) returns the 5th percentile value of a field &quot;total&quot;.</td>
</tr>
<tr>
<td>range(X)</td>
<td>Returns the difference between the max and min values of the field X.</td>
</tr>
<tr>
<td>stdev(X)</td>
<td>Returns the sample standard deviation of the field X.</td>
</tr>
<tr>
<td>stdevp(X)</td>
<td>Returns the population standard deviation of the field X.</td>
</tr>
<tr>
<td>sum(X)</td>
<td>Returns the sum of the values of the field X.</td>
</tr>
<tr>
<td>sumsq(X)</td>
<td>Returns the sum of the squares of the values of the field X.</td>
</tr>
<tr>
<td>values(X)</td>
<td>Returns the list of all distinct values of the field X as a multi-value entry. The order of the values is alphabetical.</td>
</tr>
<tr>
<td>var(X)</td>
<td>Returns the sample variance of the field X.</td>
</tr>
</tbody>
</table>
SPL Examples And Recipes

- Find the needle in the haystack
- Charting statistics and predicting values
- **Enriching and converging data sources**
- Map geographic data in real time
- Identifying anomalies
- Transactions
- Data exploration & finding relationships between fields
- Custom commands
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
Lookup – Converging Data Sources

Examples

- **Enrich data with lookups**

  ```
  index=power_of_spl status!=200 | lookup customer_info uid | stats count by customer_value
  ```

- **Search Inception!**

  ```
  index=power_of_spl [ search index=power_of_spl | stats sum(bytes) as total_bytes by clientip | sort - total_bytes | head 1 | return clientip ] | stats count by clientip status uri | sort - count
  ```

- **Append multiple searches**

  ```
  index=power_of_spl | timechart span=15s avg(bytes) as avg_bytes | appendcols [ search index=power_of_spl | stats stdev(bytes) as stdev_bytes | eval 2stdv_upper = avg_bytes + stdev_bytes*2 | filldown 2stdv_upper | eval 2stdv_lower = avg_bytes - stdev_bytes*2 | filldown 2stdv_lower | eval 2stdv_lower = if("2stdv_lower" <0,0,'2stdv_lower') | fields - stdev_bytes
  ```

---

**Lookup – Converging Data Sources**

**Examples**
Converging Data Sources

Examples

- **Enrich data with lookups**
  ```
  index=power_of_spl status!=200
  | lookup customer_info uid
  | stats count by customer_value
  ```

- **Search Inception!**
  ```
  index=power_of_spl
  [ search index=power_of_spl | stats sum(bytes) as total_bytes by clientip
  | sort - total_bytes | head 1 | return clientip ]
  | stats count by clientip status uri | sort - count
  ```

- **Append multiple searches**
  ```
  index=power_of_spl
  | timechart span=15s avg(bytes) as avg_bytes
  | appendcols [ search index=power_of_spl
  | stats stdev(bytes) as stdev_bytes] | eval 2stdv_upper = avg_bytes + stdev_bytes*2 | filldown 2stdv_upper | eval 2stdv_lower = avg_bytes - stdev_bytes*2 | filldown 2stdv_lower
  | eval 2stdv_lower = if('2stdv_lower' <0,0,'2stdv_lower') | fields - stdev_bytes
  ```
appendcols - Converging Data Sources

Examples

► Enrich data with lookups

```
index=power_of_spl status!=200
| lookup customer_info uid
| stats count by customer_value
```

► Search Inception!

```
index=power_of_spl
[ search index=power_of_spl | stats sum(bytes) as total_bytes by clientip
| sort - total_bytes | head 1 | return clientip ]
| stats count by clientip status uri | sort - count
```

► Append multiple searches

```
index=power_of_spl
| timechart span=15s avg(bytes) as avg_bytes
| appendcols [ search index=power_of_spl
| stats stdev(bytes) as stdev_bytes] | eval 2stdv_upper = avg_bytes + stdev_bytes*2 | filldown 2stdv_upper | eval 2stdv_lower = avg_bytes - stdev_bytes*2 | filldown 2stdv_lower
| eval 2stdv_lower = if('2stdv_lower' <0,0,'2stdv_lower') | fields - stdev_bytes
```
SPL Examples And Recipes

- Find the needle in the haystack
- Charting statistics and predicting values
- Enriching and converging data sources
- **Map geographic data in real time**
- Identifying anomalies
- Transactions
- Data exploration & finding relationships between fields
- Custom commands
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
Assign Lat/Lon to IP addresses
  … | iplocation clientip

Visualize statistics geographically
  … | geostats sum(price) by product

Use custom choropleths
  … | geom <featureCollection> <featureld>

Track object movements
  … | table _time latitude longitude vehicleld
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

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

- Find the needle in the haystack
- Charting statistics and predicting values
- Enriching and converging data sources
- Map geographic data in real time
- Identifying anomalies

Transactions

- Data exploration & finding relationships between fields
- Custom commands
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)
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)
SPL Examples And Recipes

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

| analyzefields |
| anomalies     |
| arules        |
| associate     |
| cluster       |
| contingency   |
| correlate     |
| fieldsummary  |
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
fieldsummary – 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
  sourctype=access_combined
  | fields – date* source* time*
  | fieldsummary maxvals=5

- Show patterns of co-occurring fields
  sourctype=access_combined
  | fields – date* source* time* | correlate

- View field relationships
  sourctype=access_combined
  | contingency uri status

- Find predictors of fields
  sourctype=access_combined
  | analyzefields classfield=statu
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
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
Predict Numeric Fields
Predict Categorical Fields
Detect Numerical Outliers
Detect Categorical Outliers
Forecast Time Series
Cluster Events
SPL Examples And Recipes

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

- Power of SPL App!
- Search Manual
- Blogs
- Answers
- Exploring Splunk
Join The Pony Poll

ponypoll.com/***
Q&A
Thank You

Don't forget to rate this session in the .conf2017 mobile app