Searching & Reporting with Splunk

Class Goals

- Create efficient, well-formed searches
- Perform calculations and evaluations on search results
- Generate reports and charts and vizualizations
- Analyze and format results
- Correlate events with transactions

Course Outline

Module 1: Introduction

Module 2: Search Fundamentals

Module 3: Transforming Commands, Part 1 Deriving Statistics

Module 4: Transforming Commands, Part 2 Creating Visualizations

Module 5: Transforming Commands, Part 3 Enriching Visualizations

Module 6: Manipulating and Filtering Results

Module 7: Correlating Events

Module 1: Introduction

Buttercup Games, Inc.

- Buttercup Games, Inc.
  - Is a multinational company with its HQ in San Francisco and offices in Boston and London
  - Sells product mainly through its worldwide chain of third party stores, but also sells through its online store
- For more information about Buttercup Games, please see Appendix C

Course Scenario

- Use cases used in this course are based on Buttercup Games, a gaming company
- Searches and reports are based on:
  - IT operations information from mail and internal network data
  - Security operations information from internal network and badge reader data
  - Business analytics from the web access logs and vendor data

Callouts

Scenarios

- Many of the examples in this course relate to a specific scenario
- For each example, a question is posed from a colleague or manager at Buttercup Games

Scenario

For failed logins into the network during the last 60 minutes, display the `IP` and `user` name.

Notes & Tips

- References for more information on a topic and tips for best practices

Note

Lookups are discussed in the *Creating knowledge Objects* course.

Your Role at Buttercup Games

- You are a Splunk power user
- Your responsibility is to provide information to users throughout the company
- You gather data and statistics and report on:
  - Security
  - IT operations
  - Business intelligence
  - Etc.

Useful References

- Search Reference:
  http://docs.splunk.com/Documentation/Splunk/latest/SearchReference
- Search Quick Reference:
  http://www.splunk.com/content/dam/splunk2/pdfs/solution-guides/splunk-quick-reference-guide.pdf
- Search Cheat Sheet
  http://docs.splunk.com/images/a/a3/Splunk_4.x_cheatsheet.pdf

Module 2: Search Fundamentals

Module Objectives

- Become familiar with the source types used during the course
- Review basic search commands and general search practices
- Examine the search pipeline
- Use the following commands to perform searches:
  - `table`
  - `rename`
  - `fields`
  - `dedup`
  - `sort`

Buttercup Games Environment

Data	host	sourcetype
AD/DNS data	adldapsv1	WinEventLog:Security
Badge reader data	badgesv1	history_access
BI server data	ecommsv1	sales_entries
Email data	cisco_router1	cisco_esa
Online transactions & Web server	www1	access_combined linux_secure
	www2	
	www3	
Retail sales data	vendorUS1	vendor_sales
Splunk indexer data	splunk1	ps
Web appliance data	cisco_router1	cisco_wsa_squid

Basic Search Review

- **Keywords**

  search for error, password

- **Booleans**

  OR, AND, NOT; AND is implied; MUST be uppercase; can use ( )'s to force precedence

  sourcetype=vendor_sales OR (sourcetype=access_combined action=purchase)

- **Phrases**

  "web error" (different than web AND error)

- **Field searches**

  status=404, user=admin

- **Wildcards**

  status=40* matches 40, 40a, 404, etc., starting keywords with a wildcard is very inefficient, e.g. *dmin

- **Comparisons**

  =, !=, <, <=, >=, > status>399, user!=admin

General Search Practices

- Time is the most efficient filter
- Be specific
  - Searching for "access denied" is always better than searching for "denied"
  - To make searches more efficient, include as many terms as possible
    - If you want to find events with `"error"` and `"sshd"` and 90% of the events include "error", but only 5% "sshd", include both values in the search
- Inclusion is generally better than exclusion
  - Searching for `"access denied"` is faster than `NOT "access granted"`
- Filter as early as possible
  - Removing duplicate events then sorting is faster than sorting then removing duplicate events

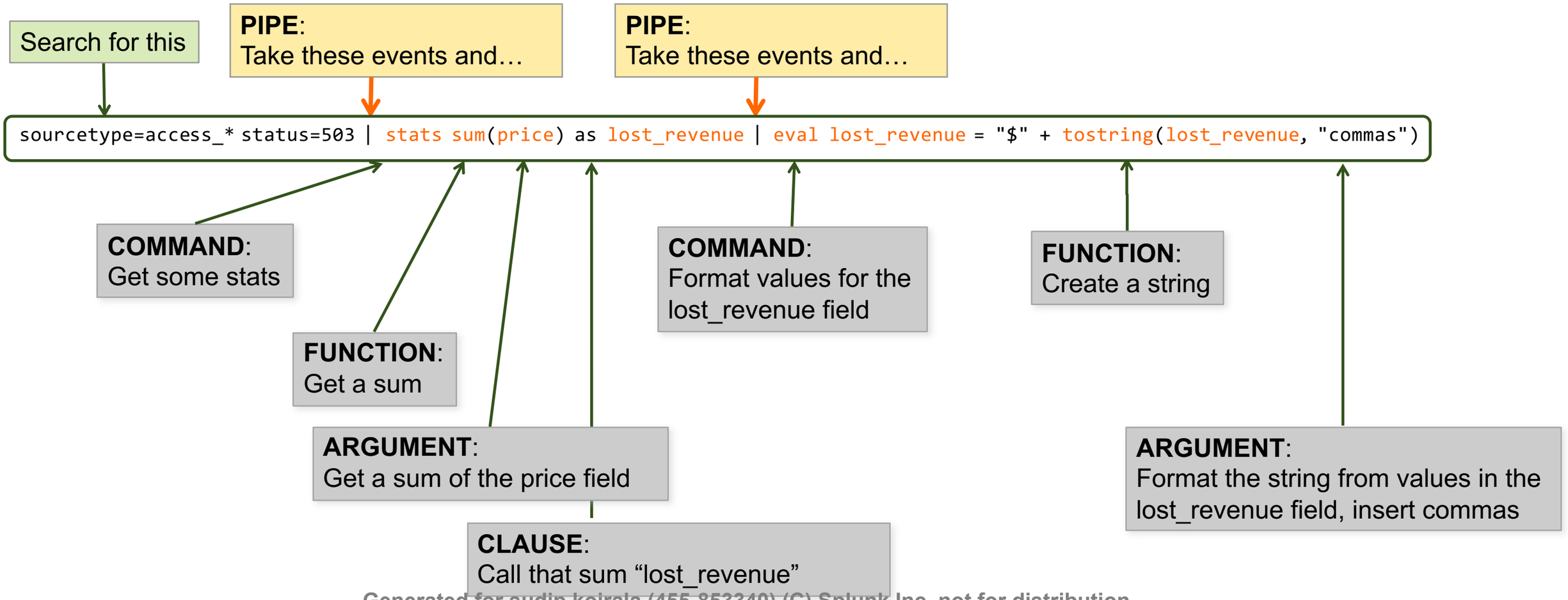
Search Language Syntax Concepts

Searches are made up of 5 basic components

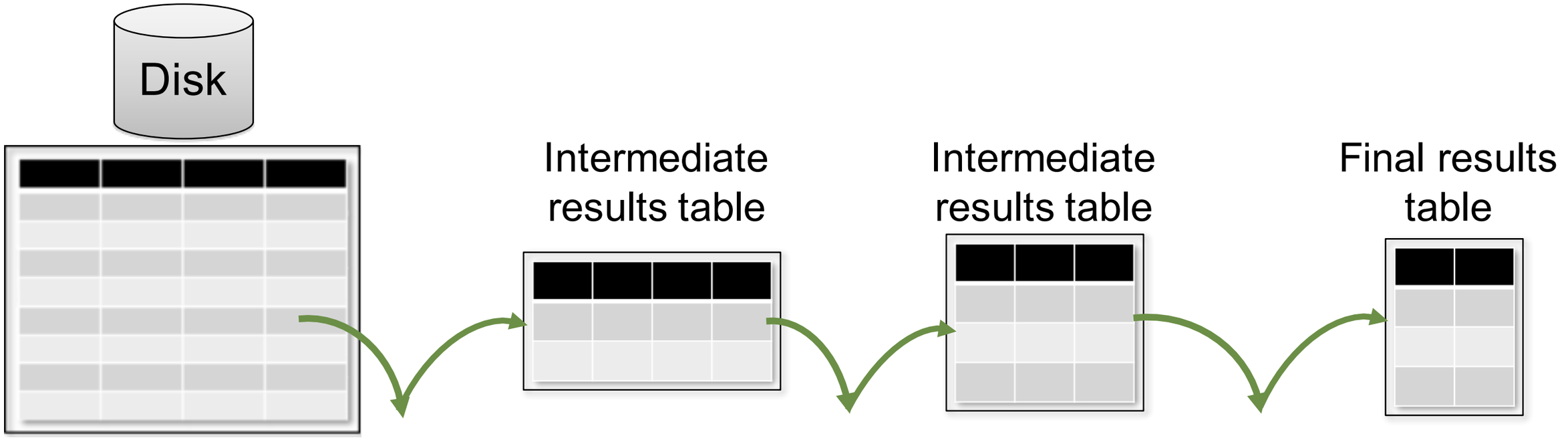
- **Search terms** – what are you looking for?
  - Keywords, phrases, Booleans, etc.
- **Commands** – what do you want to do with the results?
  - Create a chart, compute statistics, evaluate and format, etc.
- **Functions** – how do you want to chart, compute, or evaluate the results?
  - Get a sum, get an average, transform the values, etc.
- **Arguments** – are there variables you want to apply to this function?
  - Calculate average value for a specific field, convert milliseconds to seconds, etc.
- **Clauses –** how do you want to group or rename the fields in the results?
  - Give a field another name or group values by or over

Search Pipeline Example

This diagram represents a search, broken into its syntax components



The Search Pipeline



Creating a Table

- `table` command returns a table formed by only fields in the argument list
- Columns are displayed in the order given in the command
  - Column headers are field names
  - Each row is an event
  - Rows are field values

Scenario

Display the `clientip`, `action`, `productId`, and `status` of customer interactions in the online store.

```
sourcetype=access_combined
| table clientip, action, productId, status
```

clientip	action	productId	status
223.205.219.67			200
69.80.0.18	view	WC-SH-A02	200
69.80.0.18		SF-BVS-01	408
91.205.189.15	view	FS-SG-G03	200
91.205.189.15	view	CU-PG-G06	200
91.205.189.15	view	WC-SH-A02	200
91.205.189.15	remove	WC-SH-A01	200
91.205.189.15			200

Note

To make searches more readable, you can break a line in the search bar with shift-enter.

Renaming Fields

- To change the name of a field, use the `rename` command
- Useful for giving fields more meaningful names
- When including spaces or special characters in field names, use double straight quotes:
  - A `rename productId as ProductID`
  - B `rename action as "Customer Action"`
  - C `rename status as "HTTP Status"`

Scenario

Display the `clientip`, `action`, `productId`, and `status` of customer interactions in the online store.

```
sourcetype=access_combined
| table clientip, action, productId, status
| rename productId as ProductID, A
  action as "Customer Action", B
  status as "HTTP Status" C
```

clientip	Customer Action B	ProductID A	HTTP Status C
141.146.8.66		MB-AG-T01	200
141.146.8.66		WC-SH-A01	200
195.80.144.22		DC-SG-G02	200
141.146.8.66		WC-SH-A02	200
195.80.144.22		SC-MG-G10	200
141.146.8.66		PZ-SG-G05	200
195.80.144.22	purchase		200
195.80.144.22	purchase	SC-MG-G10	200

`fields` Command

- Field extraction is one of the most costly parts of a search
- `fields` command allows you to include or exclude specified fields in your search or report
- To include, use `fields +(default)`
  - Occurs before field extraction
  - Improved performance
- To exclude, use `fields -`
  - Occurs after field extraction
  - No performance benefit
  - Exclude fields used in search to make the table/display easier to read

fields Command – Examples

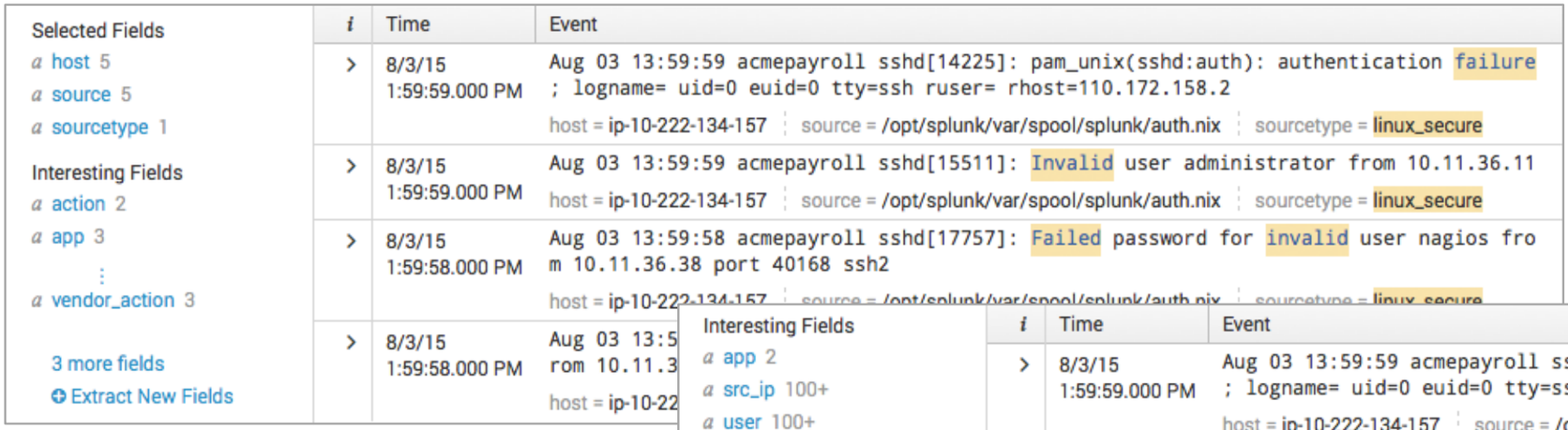
- Improves performance – only the fields you specify are extracted

Scenario

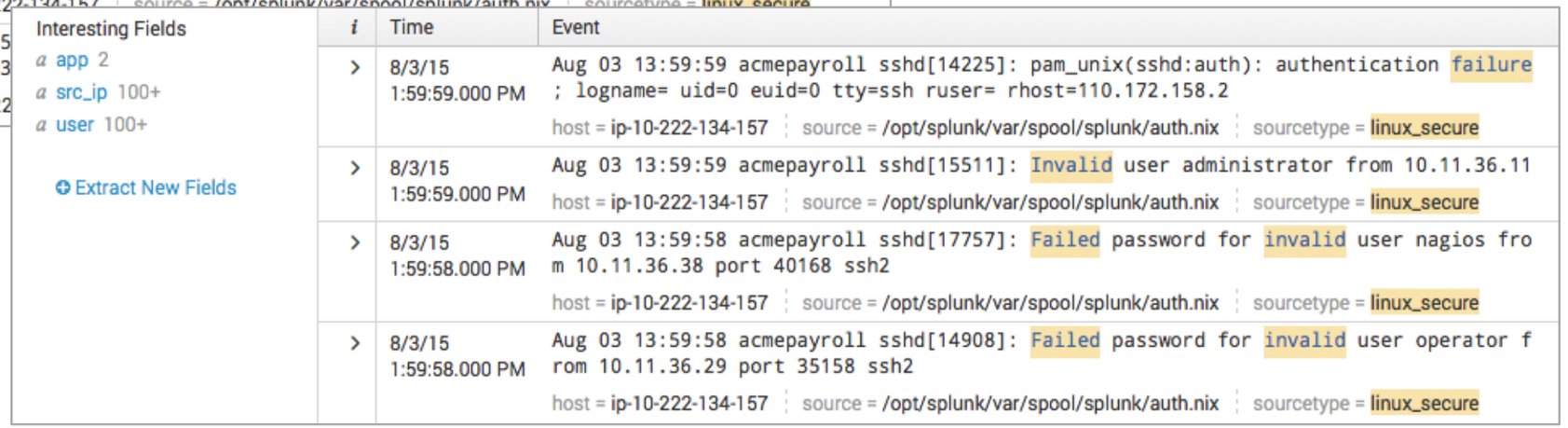
Display network failures during the previous week. Retrieve only user, app, and src_ip.

```
sourcetype=linux_secure
  (fail* OR invalid)
| fields user, app, src_ip
```

A



Returned **6,567** results by scanning **6,567** events in **1.425** seconds.



Returned **6,567** results by scanning **6,567** events in **0.753** seconds.

dedup Command

- Use dedup to remove duplicates from your results

```
sourcetype=vendor_sales | table VendorCountry, VendorStateProvince, VendorCity, Vendor
```

VendorCountry	VendorStateProvince	VendorCity	Vendor
United States	Texas	Waco	Wow Games
United States	Utah	Cedar City	Woody's Games
United States	Virginia	Staunton	Woody's Games
United States	Utah	Cedar City	Woody's Games
United States	Utah	Cedar City	Woody's Games
Australia	Western Australia	Perth	Wonderland Hobbies
Australia	Western Australia	Perth	Wonderland Hobbies

```
...| dedup Vendor | table ...
```

VendorCountry	VendorStateProvince	VendorCity	Vendor
United States	Texas	Waco	Wow Games
United States	Utah	Cedar City	Woody's Games
Australia	Western Australia	Perth	Wonderland Hobbies

```
...| dedup VendorCity, Vendor | table ...
```

VendorCountry	VendorStateProvince	VendorCity	Vendor
United States	Texas	Waco	Wow Games
United States	Utah	Cedar City	Woody's Games
United States	Virginia	Staunton	Woody's Games
Australia	Western Australia	Perth	Wonderland Hobbies

sort Command

- Use sort to order your results in
  + ascending (default) or
  – descending
- To limit the returned results, use the limit option

```
... | sort limit=20 -categoryId, product_name
... | sort 20 count
```

sort Help More »

Sorts search results by the specified fields.

Examples

Sort results by "ip" value in ascending order and then by "url" value in descending order.
... | sort ip, -url

Sort results by the "_time" field in ascending order and then by the "host" value in descending order.
... | sort _time, -host

Sort first 100 results in descending order of the "size" field and then by the "source" value in ascending order.
... | sort 100 -size, +source

sort Command (cont.)

`sort -/+<fieldname>`	sign followed by fieldname sorts results in the sign's order
`sort -/+ <fieldname>`	sign followed by space and then fieldname applies sort order to all following fields without a different explicit sort order

```
sourcetype=vendor_sales
| dedup Vendor
| sort - VendorCountry, +VendorStateProvince, VendorCity, Vendor
| table VendorCountry, VendorStateProvince, VendorCity, Vendor
```

VendorCountry	VendorStateProvince	VendorCity	Vendor
United States	Arizona	Yuma	Yumster Games
United States	Arizona	Tucson	Boothill Games
United States	Arizona	Phoenix	Rising Games
United States	Arizona	Phoenix	Phoenix Games
United States	Arizona	Flagstaff	Flaggin Games

Adding Knowledge to Data through Lookups

- There are use cases where you need additional data for the search that is not available in the index
- In this example, `Username` and `Department` are not in the event data

Aug30 2015 23:50:21 Address=1.1.1.R2 Address_Description=San Francisco Device=Proximity Reader Event_Description=Access Granted: Door Used rfid=341402271288

Scenario

Display badge-ins during the last 4 hours to include location, badge ID, user name and department.

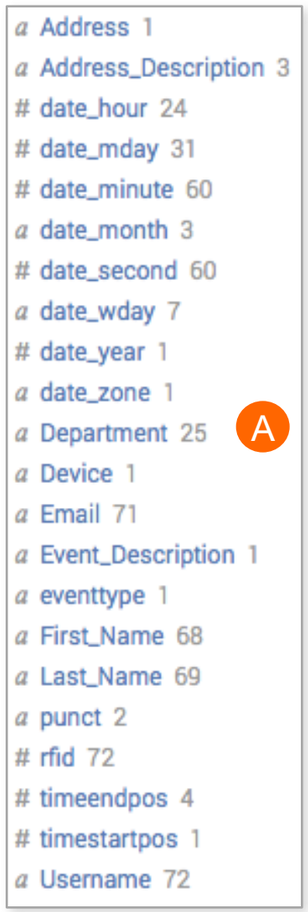
```
sourcetype=history_access
| table Address_Description, rfid,
  Username, Department
```

Address_Description	rfid	Username	Department
London	890313901800	bhussain	ITOps
London	890313901800	bhussain	ITOps
London	890313901800	bhussain	ITOps
London	862417886973	fbryan	Sales
Boston	249772079712	lsagers	SecOps
Boston	398009643042	pbunch	ITOps
Boston	672903009231	dhale	Sales
London	963871339460	rjayaraman	Engineering

Note

Lookups are discussed in the *Creating knowledge Objects* course.

Adding Knowledge to Data through Lookups (cont.)



- Lookups allow you to add more fields to your events, such as:
  - Associate RFIDs with user names , IP addresses, and workstation IDs
  - Provide descriptions for http status codes (“file not found”, “service unavailable”)
  - Reveal sale prices and descriptions for products
- Most lookups are automatic and performed in the background
- Lookups fields appear in the Fields sidebar A
- Lookups are Knowledge Objects and typically created by the Knowledge Manager

Module 3: Transforming Commands, Part 1 Deriving Statistics

Module Objectives

Use the following commands and their functions:

- `top`
- `rare`
- `stats`

Transforming Commands

- Orders search results into a data table that Splunk can use for statistical purposes
- Required to transform search results into visualizations

Getting Top Values

The `top` command finds the most common values of a given field in the result set

- By default, returns top 10 results

Scenario

During the last 60 minutes, which IPs generated the most attacks?

src_ip	count	percent
3.0.0.44	31	52.542373
175.45.176.223	8	13.559322
175.45.176.98	5	8.474576
41.32.0.85	4	6.779661
2.144.0.210	4	6.779661

top Command

- By default, output displays in table format
- Automatically returns **count** and **percent** columns
- `limit=#` returns this number of results
  - By default, 10 results are displayed
  - `limit=0` returns unlimited results

top Help More »

Displays the most common values of a field.

Examples

Return the 20 most common values of the "url" field.

```
... | top limit=20 url
```

Return top URL values.

```
... | top url
```

Return top "user" values for each "host".

```
... | top user by host
```

- `countfield=<string>` provides the name of a new field to write the value of count, default is `"count"`
- `showperc=f` specifies whether to create field called `"percent"`, default is true

Note

Refer to the search assistant or Splunk docs for the other available options.

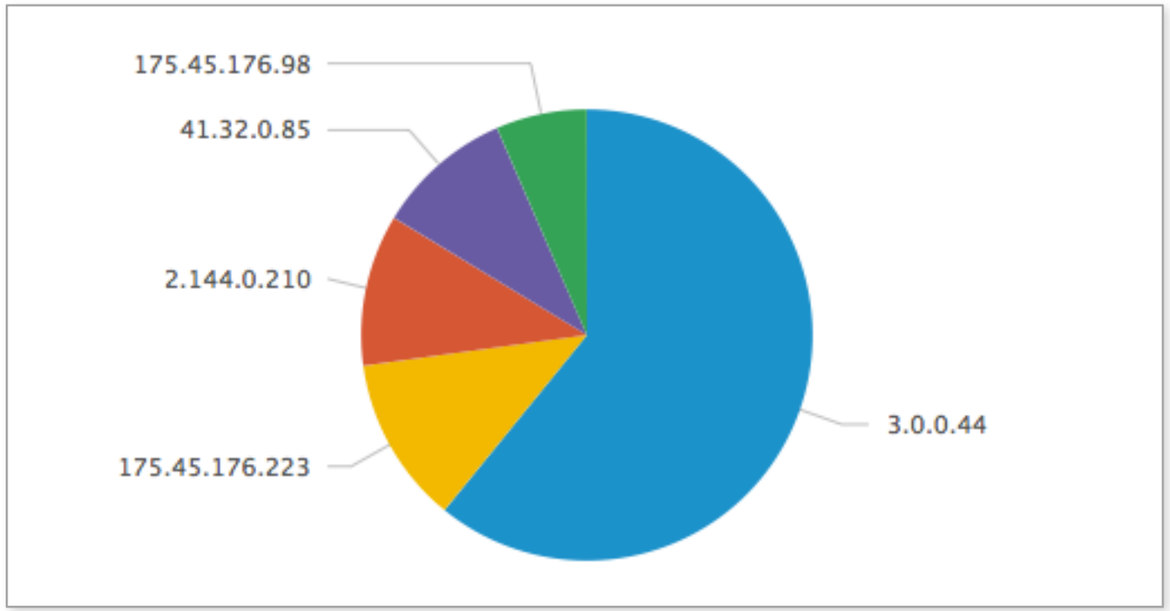
top Command – Single Field Examples

Scenario

During the last hour, display the top 5 IPs that generated the most attacks.

```
sourcetype=linux_secure
(fail* OR invalid)
| top limit=5 src_ip
```

src_ip	count	percent
3.0.0.44	56	54.901961
175.45.176.223	11	10.784314
2.144.0.210	10	9.803922
41.32.0.85	9	8.823529
175.45.176.98	6	5.882353



top Command – Multiple Field Examples

Scenario

Display the top 3 subjects for all users on the network during the last 24 hours. Rename the count field and show count, but not percentage.

```
sourcetype=WinEventLog:Security
| top user, A subject limit=3 B
  countfield=Attempts C showperc=f
```

user A	subject	C Attempts
Administrator	Account Used for Logon by	56
B Hax0r	Unknown user name or bad password	53
administrator	Logon attempt using explicit credentials	27

Note

A Boolean can be `t/f`, `true/false` as well as `1/0`.

Scenario

Display the top 3 subjects on the network for each user during the last 24 hours. Rename the count field and show count, but not percentage.

```
sourcetype=WinEventLog:Security
| top subject by user D limit=3 E
  countfield=Attempts showperc=f
```

user	subject E	Attempts
@@user	Successful Logon	9
@@user D	Unknown user name or bad password	8
@@user	Successful Network Logon	8
ACMEDC01$	The logon attempt failed for other reasons.	8
Hax0r	Unknown user name or bad password	54
S-1-5-21-4095465814-4193276578-644264660-1000	System security access was granted to an account	9
S-1-5-32-544	System security access was removed from an account	8
WIN-L25DGSHI03K$	A privileged service was called	9
WIN-L25DGSHI03K$ D	A process has exited	8
WIN-L25DGSHI03K$	A new process has been created	8
administrator	Logon attempt using explicit credentials	25
bamboo	The audit log was cleared	1

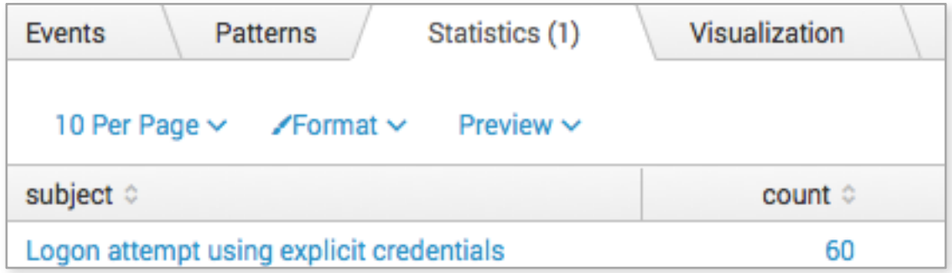
rare Command

- The rare command returns the least common field values of a given field in the results
- Options are identical to the top command

Scenario

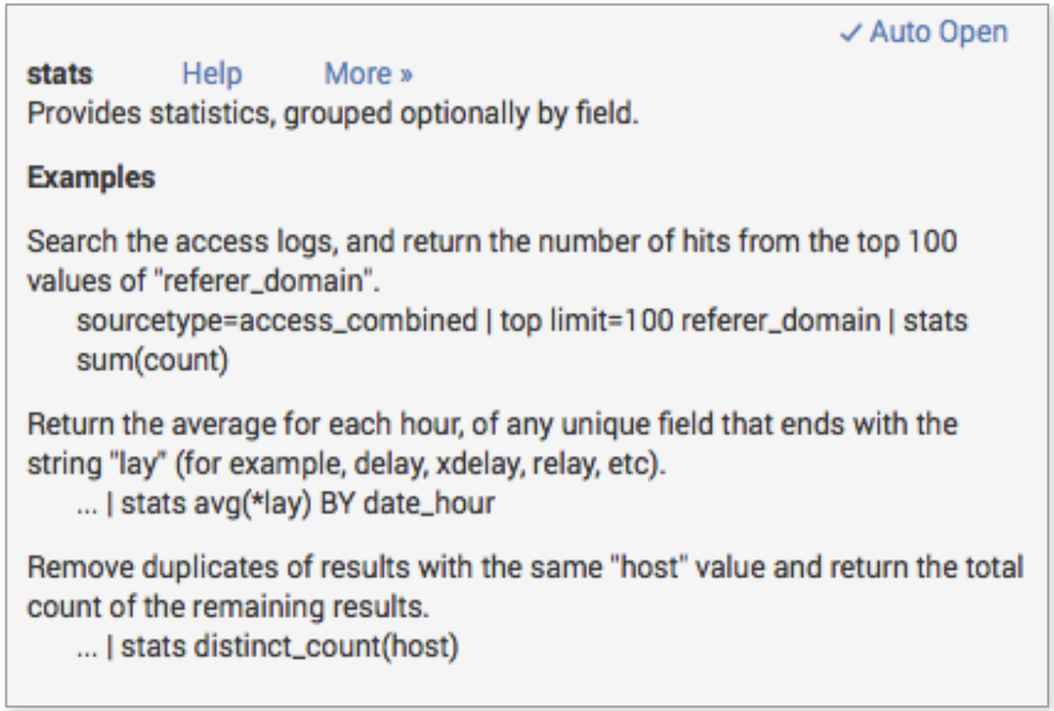
Which action was the least used on the network during the past 60 minutes?

```
sourcetype=WinEventLog:Security
  | rare showperc=f limit=1 subject
```



stats Command

- stats allows you to calculate statistics on data that matches your search criteria
- Common functions include:
  - count – returns the number of events that match the search criteria
  - distinct_count, dc – returns a count of unique values for a given field
  - sum – returns a sum of numeric values
  - avg – returns an average of numeric values
  - list – lists all values of a given field
  - values – lists unique values of a given field



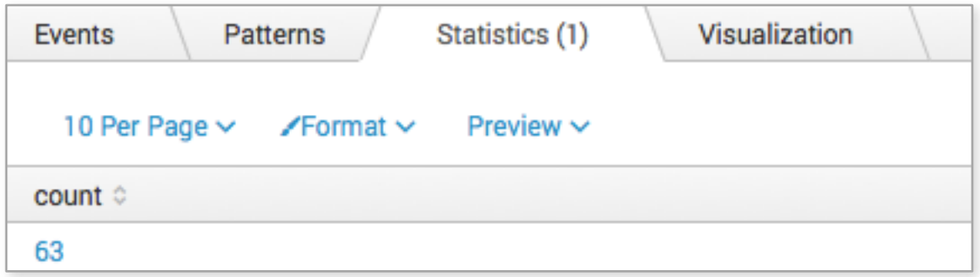
stats Command – count

- count returns the number of matching events based on the current search criteria
- Use the as clause to rename the count field

Scenario

Display invalid or failed login attempts during the last 60 minutes.

```
sourcetype=linux_secure (invalid OR failed)
| stats count
```




```
sourcetype=linux_secure (invalid OR failed)
| stats count as "Potential Issues"
```

stats Command – count(*field*)

Adding a *field* as an argument to the count function returns the number of events where a value is present

Scenario

Count the number of events during the last 15 minutes that contain a vendor action field. Also count the total events.

```
sourcetype=linux_secure
| stats count(vendor_action) as ActionEvents, A
  count as TotalEvents B
```

Events | Patterns | Statistics (1) | Visualization

10 Per Page | Format | Preview

A ActionEvents	B TotalEvents
97	250

stats Command – by *fields*

Scenario

Display the number of vendor actions by user and application during the last 15 minutes.

```
sourcetype=linux_secure
| stats count by user, app, vendor_action
```

- by clause returns a count for each value of a named field or set of fields
- Can use any number of fields in the by *field* list
  - Fundamental difference between stats and chart/timechart for which the limit is 2

user	app	vendor_action	count
arangel	sshd	Failed	1
bhussain	cron	session opened	54
eminem	sshd	Failed	1
ftpuser	ftpd	FTP LOGIN	4
jdoe	ftpd	FTP LOGIN	4
lsagers	sshd	Failed	1
madeyemi	sshd	Accepted	2
oracle	sshd	Failed	1
pdabbeville	sshd	Failed	1
root	ftpd	FTP LOGIN	4

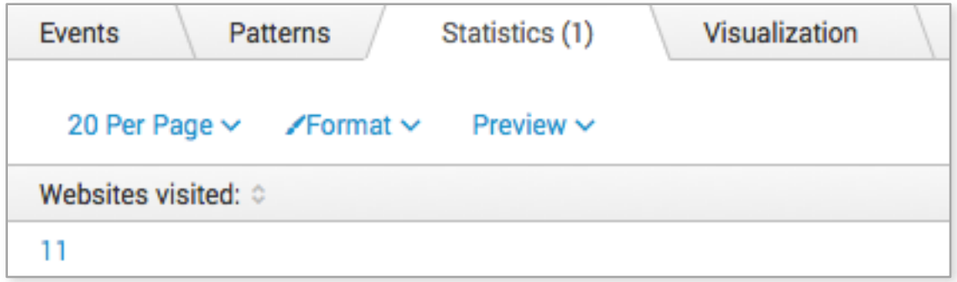
`stats` Command – `distinct_count(`*`field`*`)`

- `distinct_count()` or `dc()` provides a count of how many unique values there are for a given field in the result set
- This example counts how many unique values for `s_hostname`

Scenario

How many unique websites have our employees visited?

```
sourcetype=cisco_wsa_squid
 | stats dc(s_hostname) as "Websites visited:"
```



stats Command – sum(*field*)

Scenario

How much bandwidth did employees spend at each website during the past week?

```
sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as Bandwidth by s_hostname
| sort -Bandwidth
```

A B C

For fields with a numeric value, you can sum the actual values of that field

Events | Patterns | Statistics (54) | Visualization

20 Per Page | Format | Preview

s_hostname	Bandwidth
www.gctsindia.in	4866091
www.heals.co.uk	1233522
www.animationmagazine.net	1203433
www.kare11.com	1202753
www.finedinings.com	747407

stats Command – sum(*field*) – (cont.)

Scenario

Report the number of retail units sold and sales revenue for each product during the past week.

```
sourcetype=vendor_sales
| stats count(price) as "Units Sold" A
  sum(price) as "Total Sales" B by product_name C
| sort -"Total Sales" D
```

- A A single `stats` command can
- B have multiple functions
- C The by clause is applied to both functions
- D `sort` in descending order

product_name	A Units Sold	B Total Sales
Dream Crusher	100	3999.00
Manganiello Bros.	83	3319.17
World of Cheese	116	2898.84
Orvil the Wolverine	60	2399.40
SIM Cubicle	111	2218.89
Mediocre Kingdoms	81	2024.19
Final Sequel	65	1624.35
Curling 2014	53	1059.47
World of Cheese Tee	83	829.17
Manganiello Bros. Tee	82	819.18

C D

stats Command – avg(*field*)

- The avg function provides the average numeric value for the given field
- You can only use avg on numeric fields
  - If an event does not have the field or has an invalid value for the field, it is not considered in the calculation

Scenario

What is the average bandwidth used for each website usage type?

```
sourcetype=cisco_wsa_squid
| stats avg(sc_bytes) as "Average Bytes" A
  by usage B
```

usage	Average Bytes
Borderline	13709.812627
Business	13166.405498
Personal	17105.070920
Unknown	13528.095825
Violation	7139.151515

B

A

stats Command – list(*field*)

- list function lists all field values for a given field
- This example lists the websites visited by each employee
  - Since the security logs generate an event for each network request, the same hostname appears multiple times
  - If you want a list of "unique" field values, use the values function

Scenario

Which websites have our employees accessed during the last 60 minutes?

```
sourcetype=cisco_wsa_squid
| stats list(s_hostname) as "Websites visited:"
  by cs_username
```

cs_username	Websites visited:
basselin@buttercupgames.com	- -
blu@buttercupgames.com	www.lowermybills.com
cquinn@buttercupgames.com	- static.pochta.ru
dhale@buttercupgames.com	-
dpiazza@buttercupgames.com	www.ayles.com www.ayles.com www.ayles.com www.ayles.com www.ayles.com www.ayles.com
gbowser@buttercupgames.com	www.fftoday.com www.fftoday.com www.fftoday.com www.fftoday.com www.fftoday.com

stats Command – values(*field*)

Scenario

Display the user names of failed attempts by IP in the last 60 minutes.

```
sourcetype=linux_secure fail*
| stats values(user) as "User Names",
  count(user) as Attempts by src_ip
```

values function lists “unique” values for the specified field

src_ip	User Names	Attempts
1.0.32.67	root	2
10.232.44.142	twilliam	1
10.232.44.71	jsimon1	1
175.45.176.223	gbottazzi oracle root scanner user	37
175.45.176.98	abc andrew cvs enquiries logs michael test test3	8

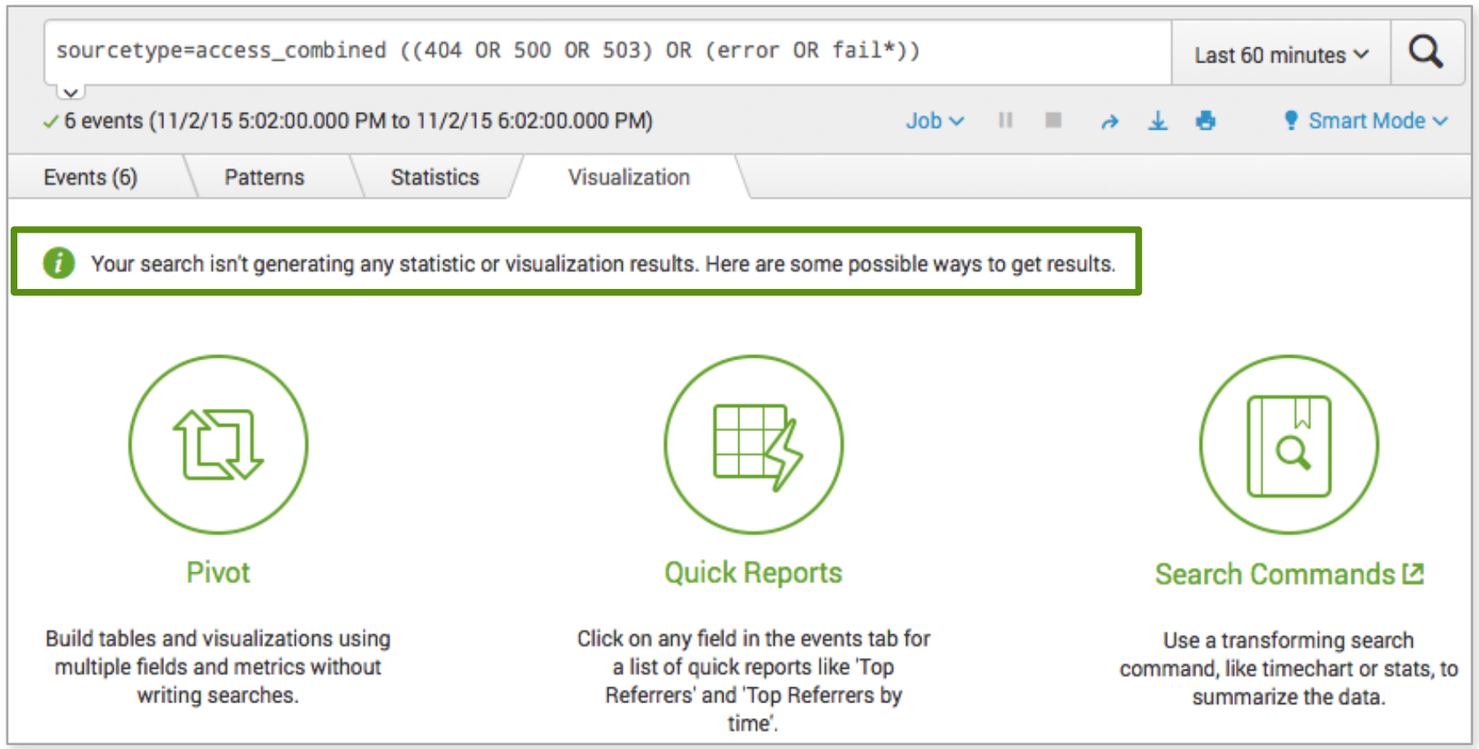
Module 4: Transforming Commands, Part 2 Creating Visualizations

Module Objectives

- Explore data structure requirements
- Explore visualization types
- Create and format charts
- Create and format `timecharts`
- Explain when to use each type of reporting command

Viewing Results as a Visualization

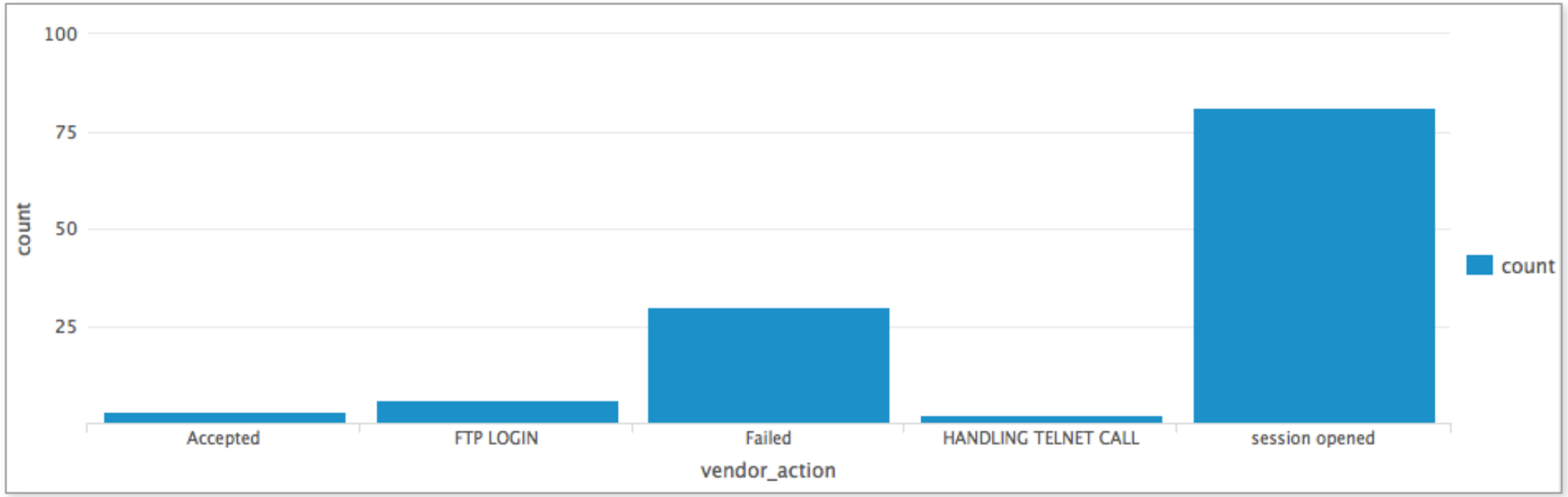
- Not all searches can be visually represented
- A data series is a sequence of related data points that are plotted in a visualization
- Data series can generate any statistical or visualization results



Data Structure Requirements – Single Series

- Most visualizations require search results structured as tables with at least two columns, a **single series**
  - **first column** provides x-axis values
  - **subsequent columns** provide y-axis values for each series in the chart

vendor_action	count
Accepted	2
FTP LOGIN	6
Failed	29
HANDLING TELNET CALL	2
Invalid user	3
session opened	59

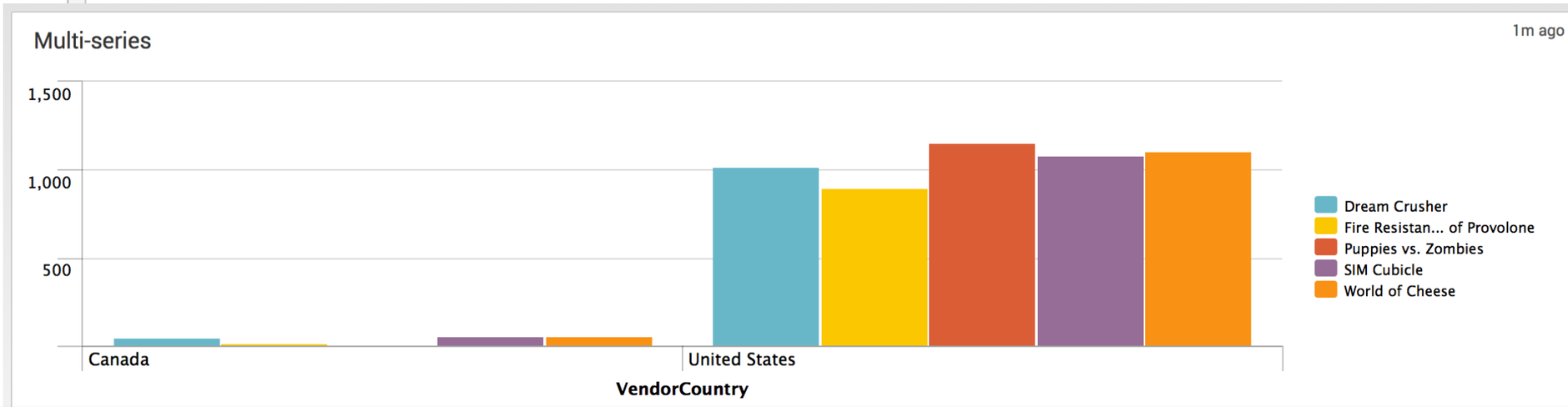


Data Structure Requirements – Multi-Series

To get **multi-series** tables you need to set up the underlying search with reporting search commands like **chart** or **timechart**

Multi-series

VendorCountry	Dream Crusher	Fire Resistance Suit of Provolone	Puppies vs. Zombies	SIM Cubicle	World of Cheese
Canada	54	19	15	71	64
United States	1189	1014	1284	1214	1258




```
sourcetype=vendor_sales VendorID<4000 | chart count over VendorCountry by product_name limit=5 useother=f
```

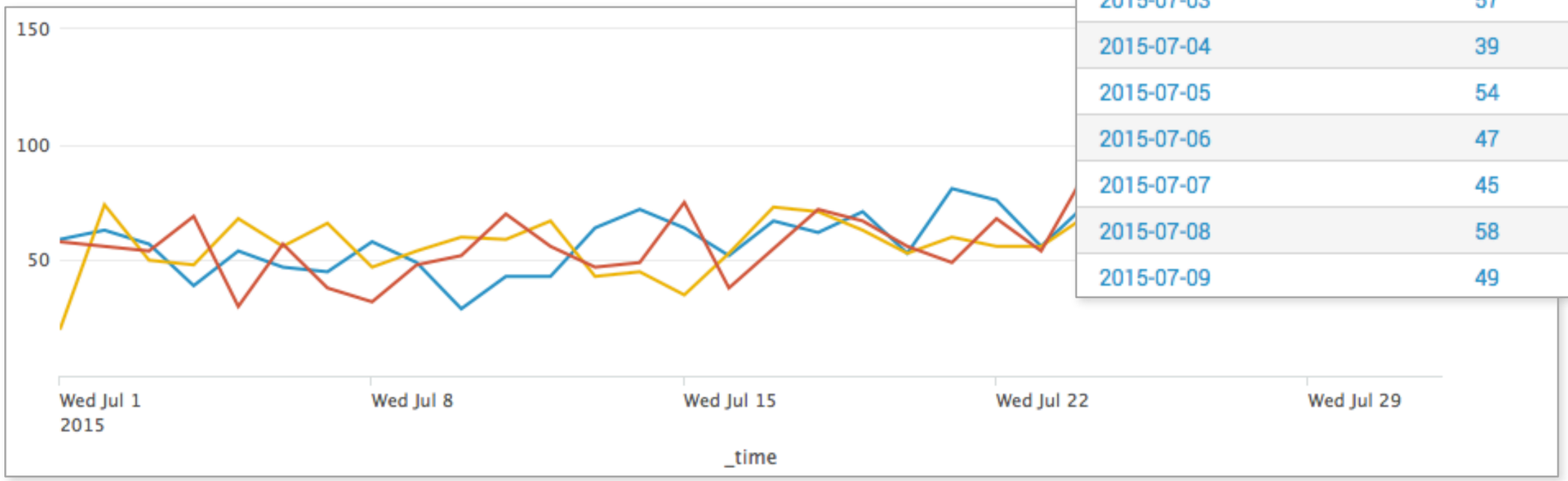
Last 30 days

 | 30 June 2016

Data Structure Requirements – Time Series

Time series display statistical trends over time

_time	www1	www2	www3
2015-07-01	59	20	58
2015-07-02	63	74	56
2015-07-03	57	50	54
2015-07-04	39	48	69
2015-07-05	54	68	30
2015-07-06	47	56	57
2015-07-07	45	66	38
2015-07-08	58	47	32
2015-07-09	49	54	48

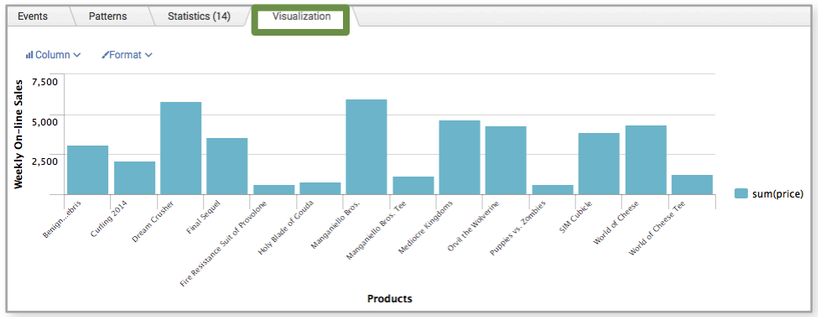



```
sourcetype=access_combined action=purchase status=200 | timechart count by host
```

Previous month

Visualization Types

- When a search returns statistical values, you can view results as a visualization:

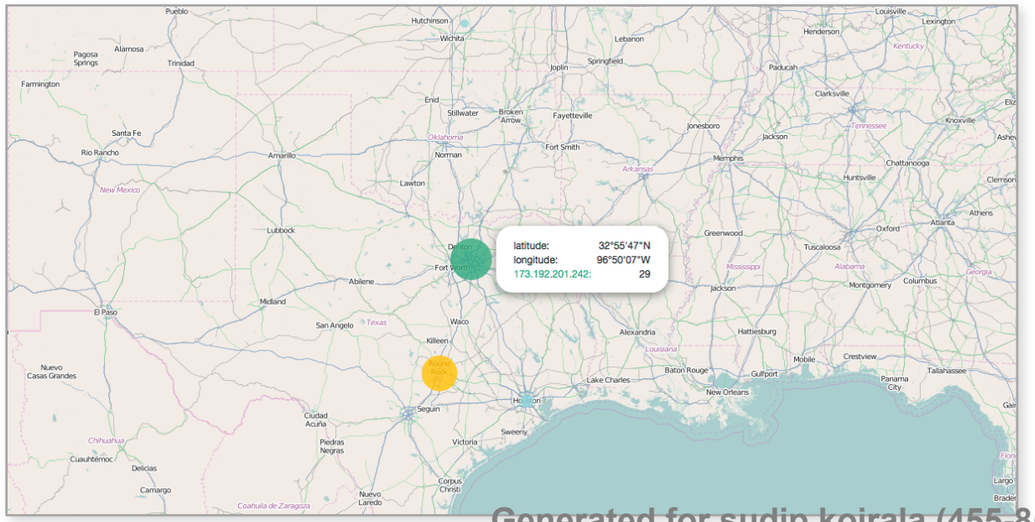


Chart

Events	Patterns	Statistics (14)	Visualization

10 Per Page | Format | Preview

product_name	sum(price)
Benign Space Debris	3073.77
Curling 2014	2118.94
Dream Crusher	5838.54
Final Sequel	3573.57
Fire Resistance Suit of Provolone	626.43
Holy Blade of Gouda	766.72
Manganiello Bros.	5998.50
Manganiello Bros. Tee	1168.83
Mediocre Kingdoms	4648.14
Orvil the Wolverine	4278.93

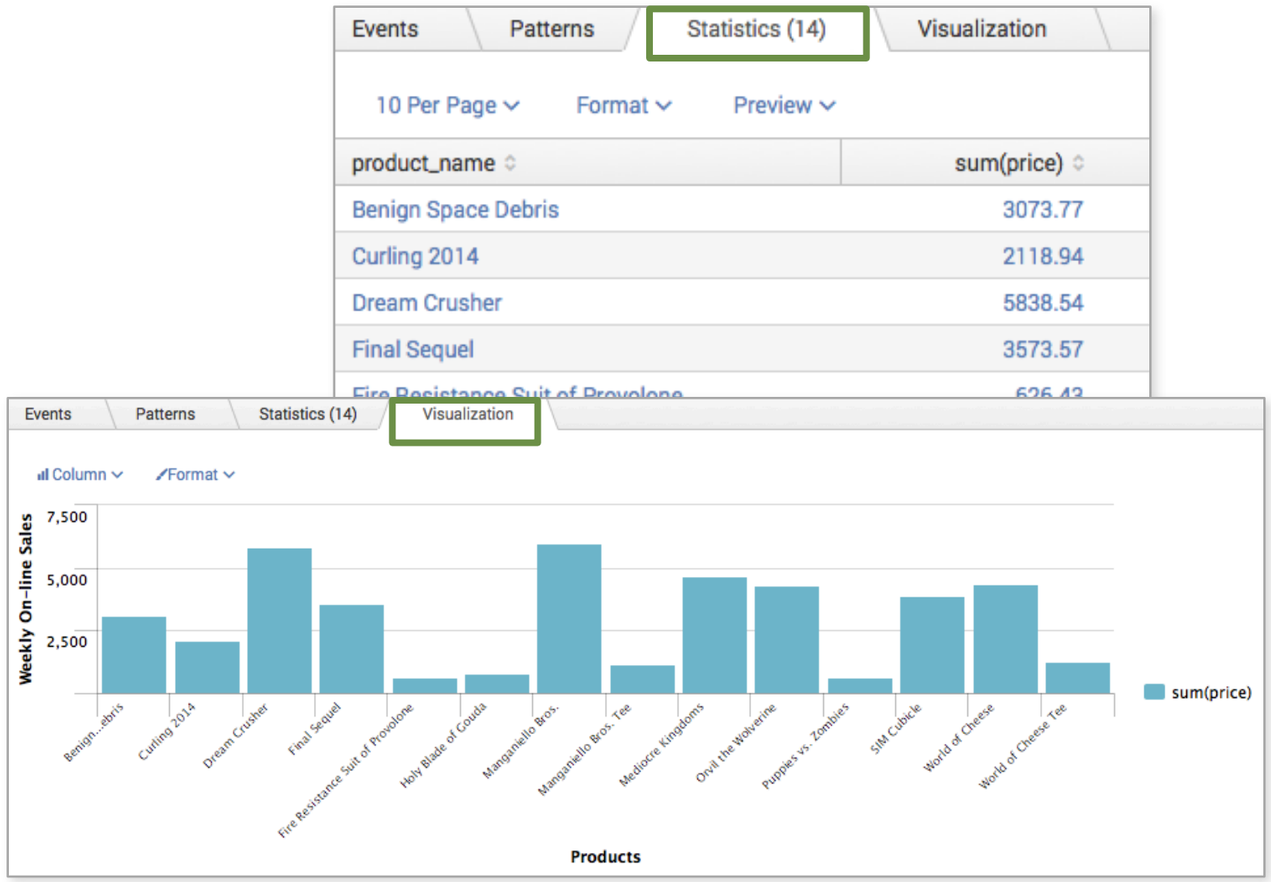


Map

Single value

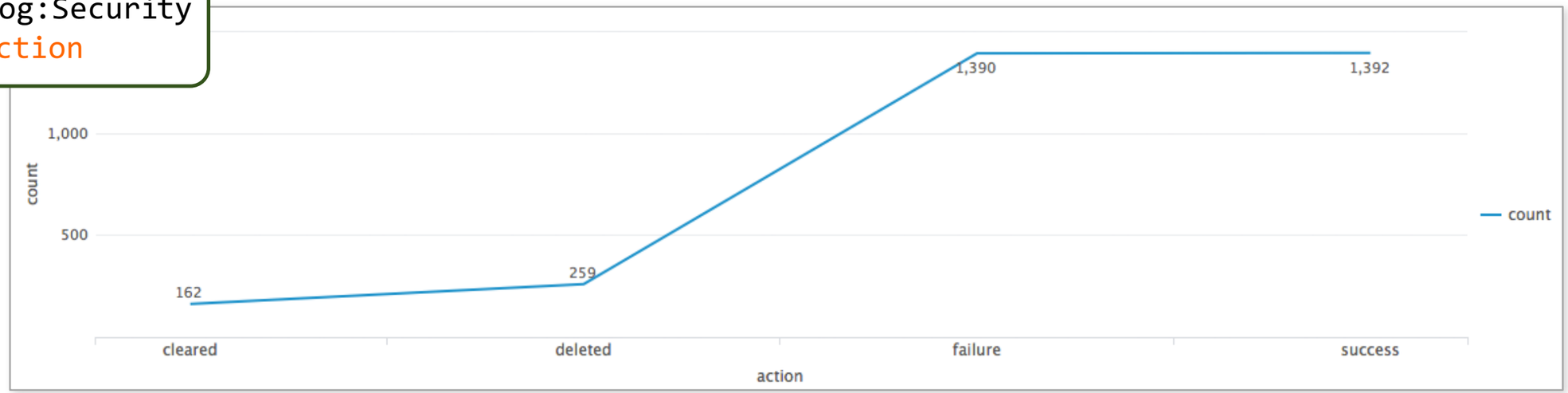
Viewing Results as a Chart

- There are seven chart types:
  - Line
  - Area
  - Column
  - Bar
  - Bubble
  - Scatter
  - Pie

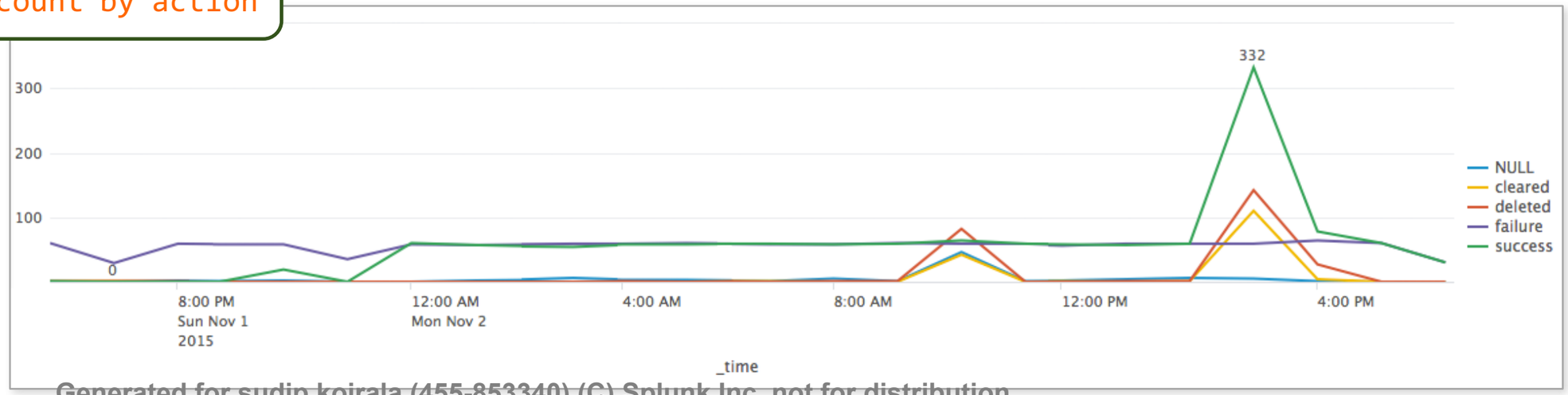


Charts – Line

```
sourcetype=WinEventLog:Security
| chart count over action
```

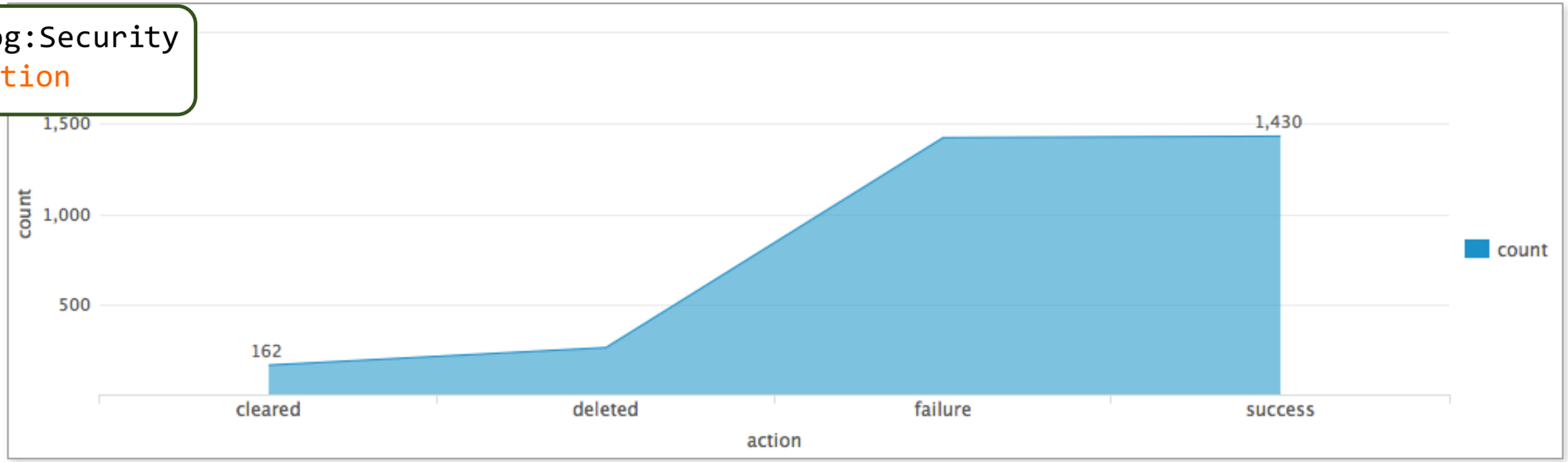



```
sourcetype=WinEventLog:Security
| timechart span=1h count by action
```

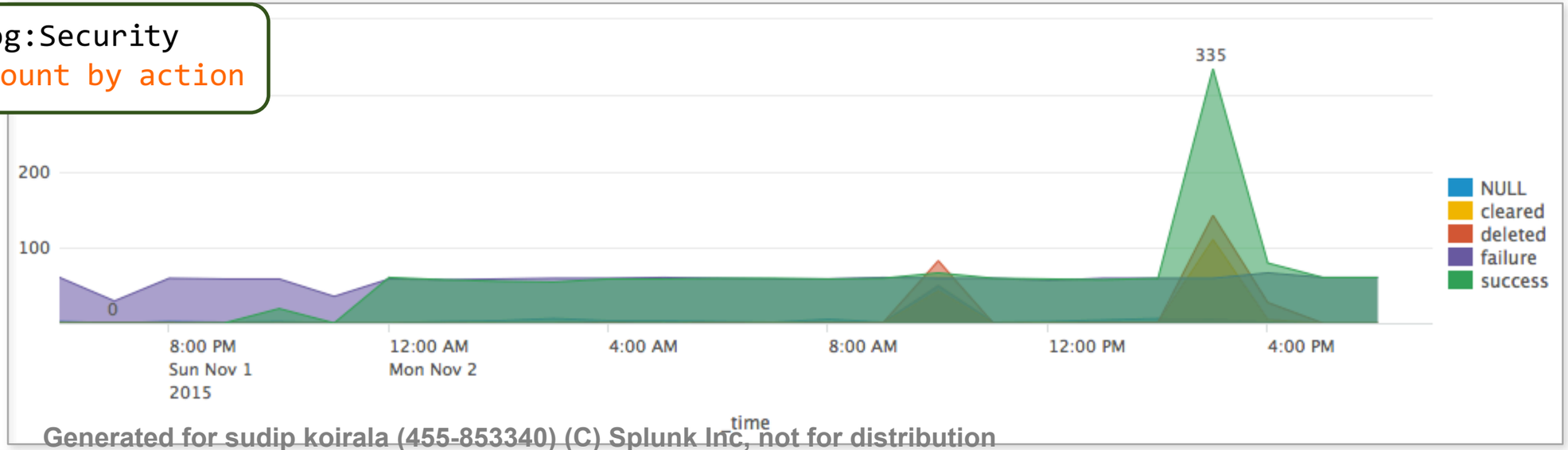


Charts – Area

```
sourcetype=WinEventLog:Security
| chart count over action
```

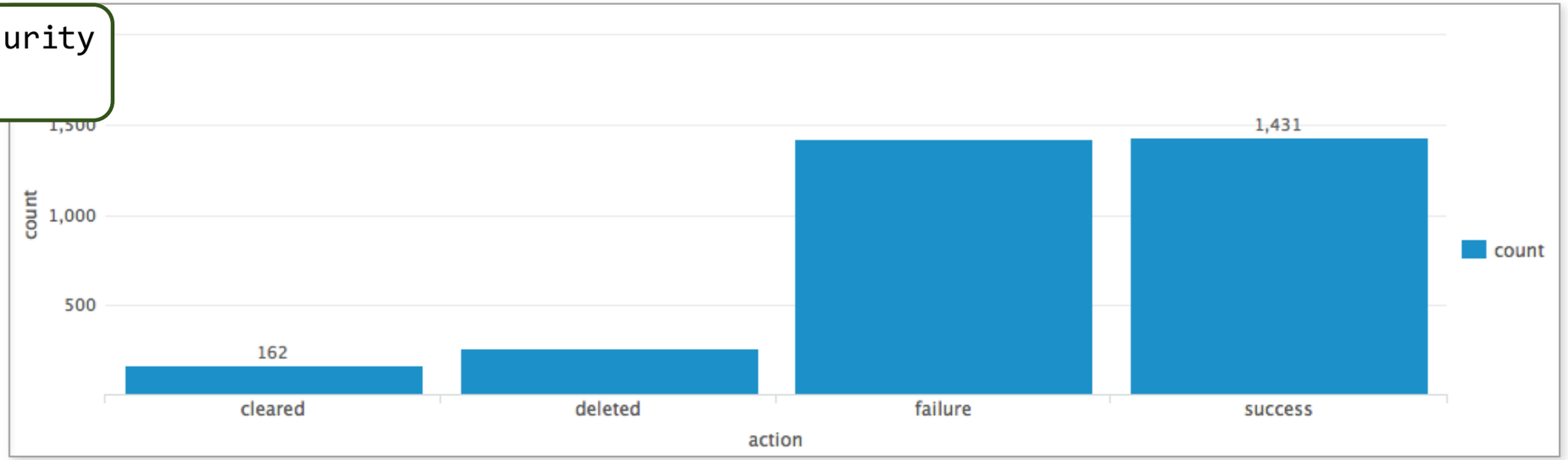



```
sourcetype=WinEventLog:Security
| timechart span=1h count by action
```

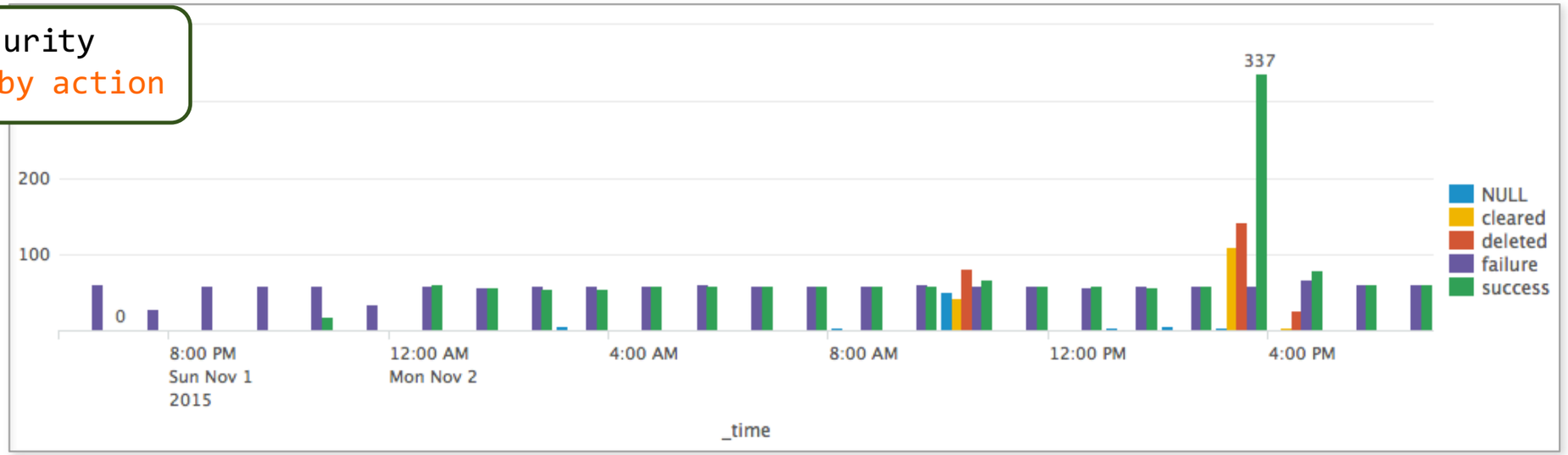


Charts – Column

```
sourcetype=WinEventLog:Security
| chart count over action
```

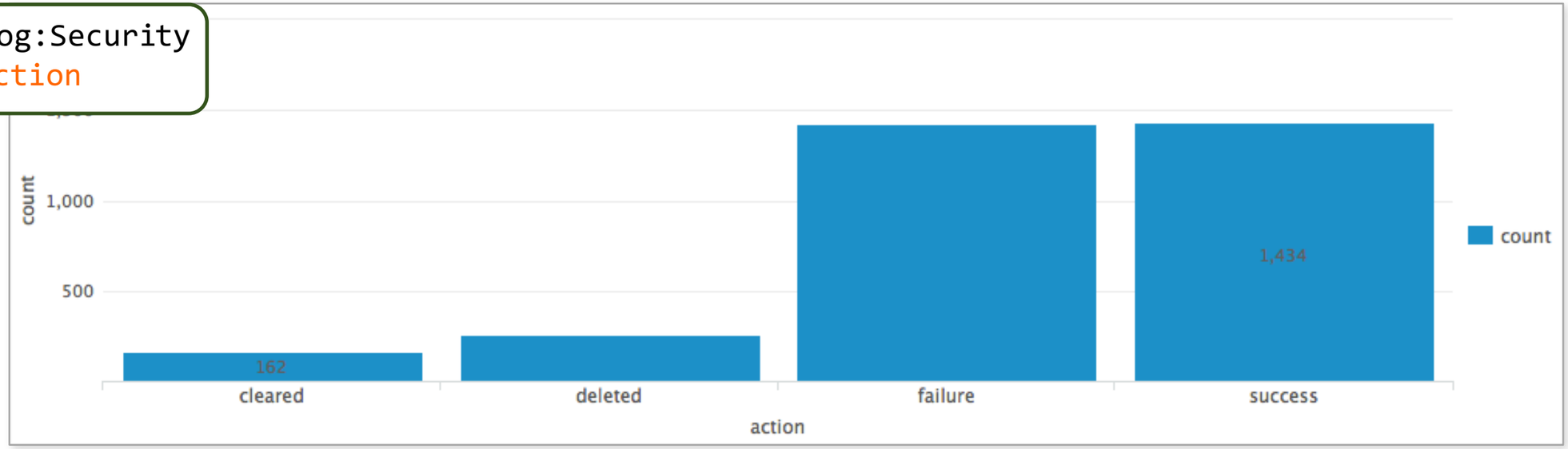



```
sourcetype=WinEventLog:Security
| timechart span=1h count by action
```

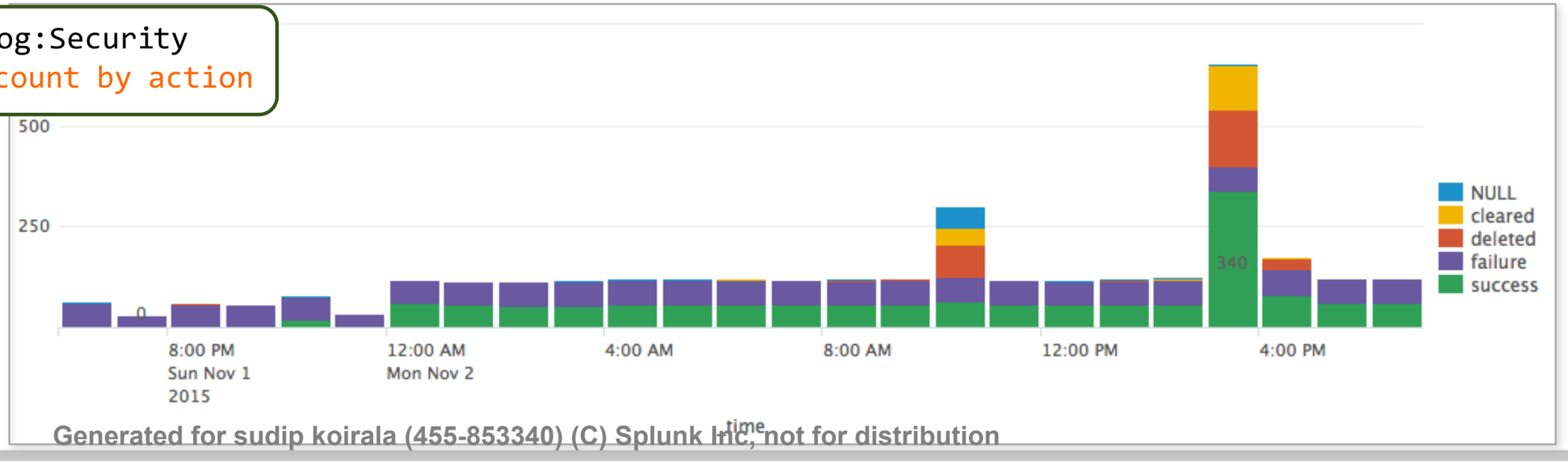


Charts – Column (Formatted as Stacked)

```
sourcetype=WinEventLog:Security
| chart count over action
```

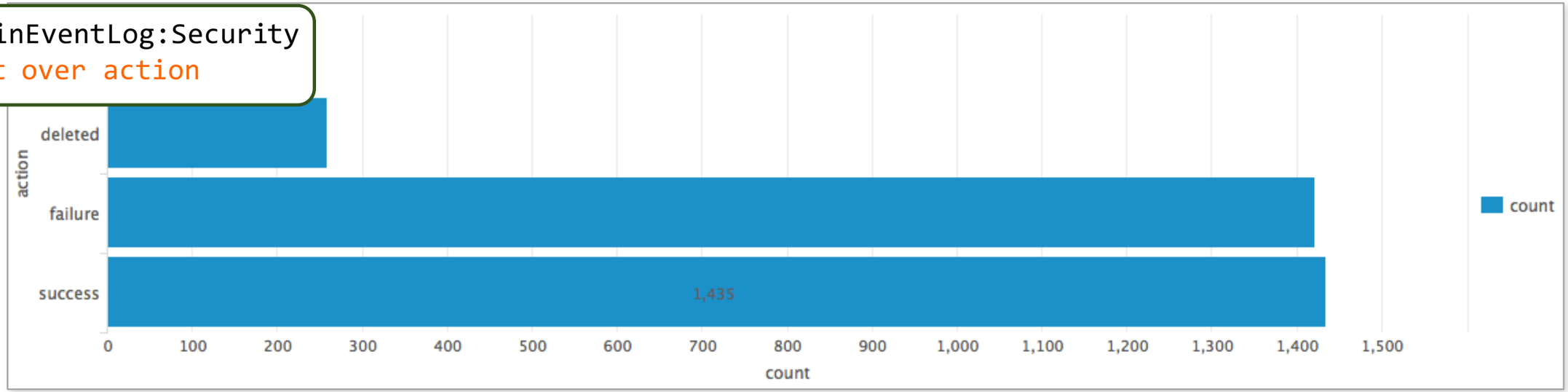



```
sourcetype=WinEventLog:Security
| timechart span=1h count by action
```

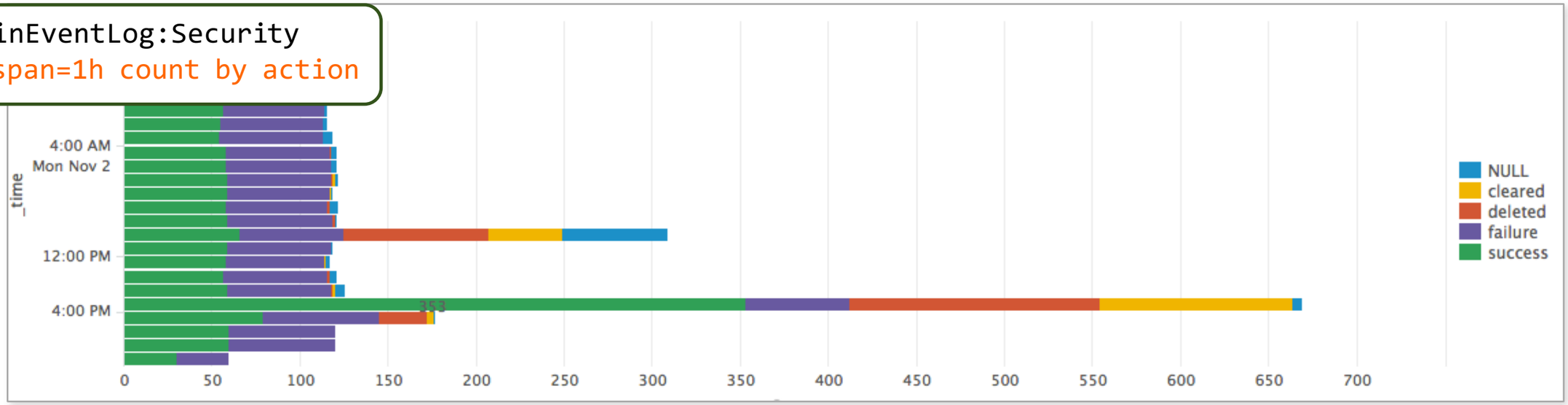


Charts – Bar

```
sourcetype=WinEventLog:Security
| chart count over action
```

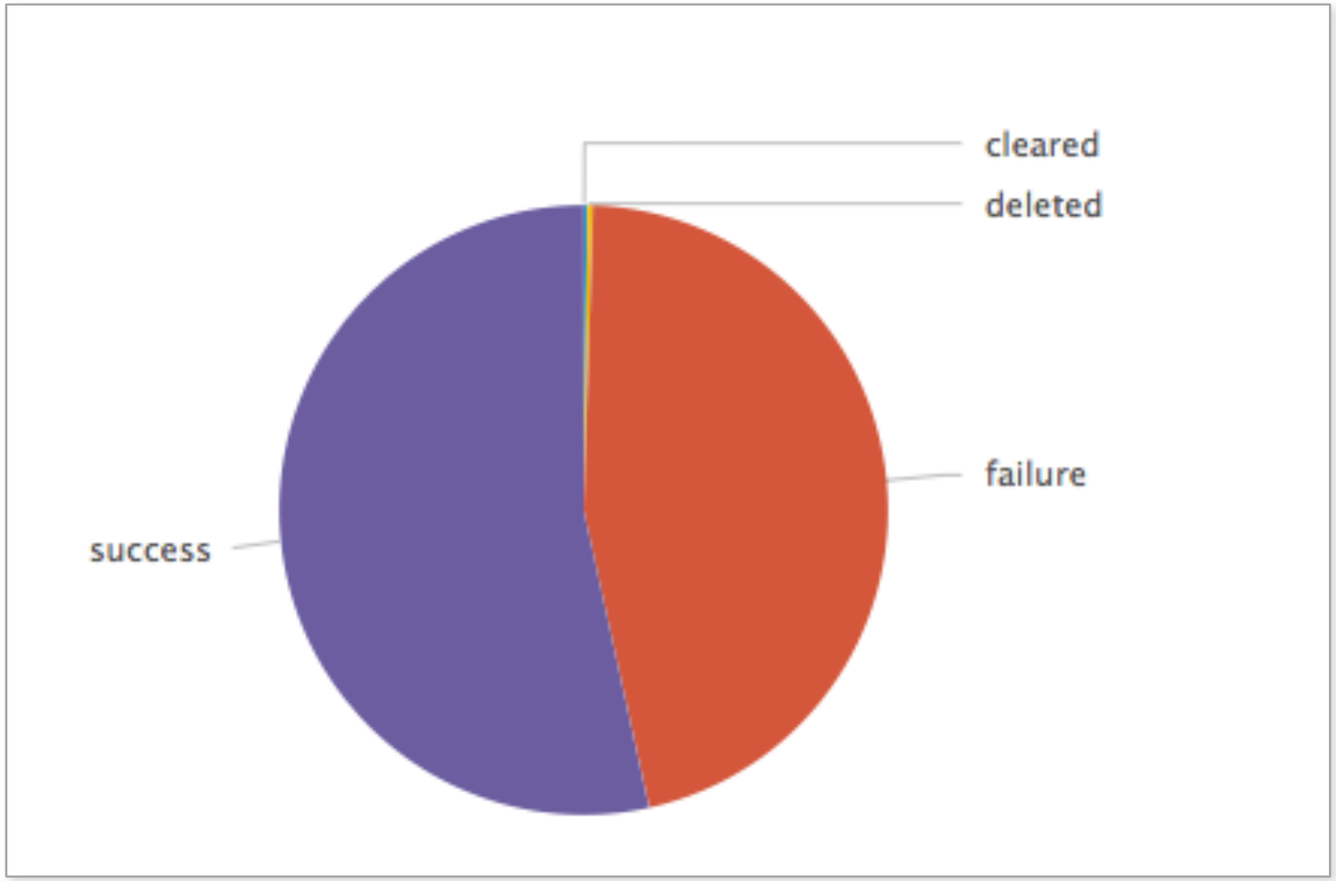



```
sourcetype=WinEventLog:Security
| timechart span=1h count by action
```



Charts – Pie

```
sourcetype=WinEventLog:Security
| chart count over action
```



Charts – Scatter

- Scatter chart shows trends in the relationships between discrete data values

```
sourcetype=vendor_sales
VendorID >=7000 AND VendorID <=8999
| stats sum(price) as "Total Sales",
  avg(price) as "Average Price", count as Count
  by VendorCountry, product_name
```

- Generally, it shows discrete values that do not occur at regular intervals or belong to a series

Charts – Bubble

- Bubble chart provides a visual way to view a three dimensional series
- Each bubble plots against two dimensions on the X and Y axes
- The size of the bubble represents the value for the third dimension

```
sourcetype=vendor_sales
VendorID >=7000 AND VendorID <=8999
| stats sum(price) as "Total Sales",
  avg(price) as "Average Price", count as Count
  by VendorCountry, product_name
```

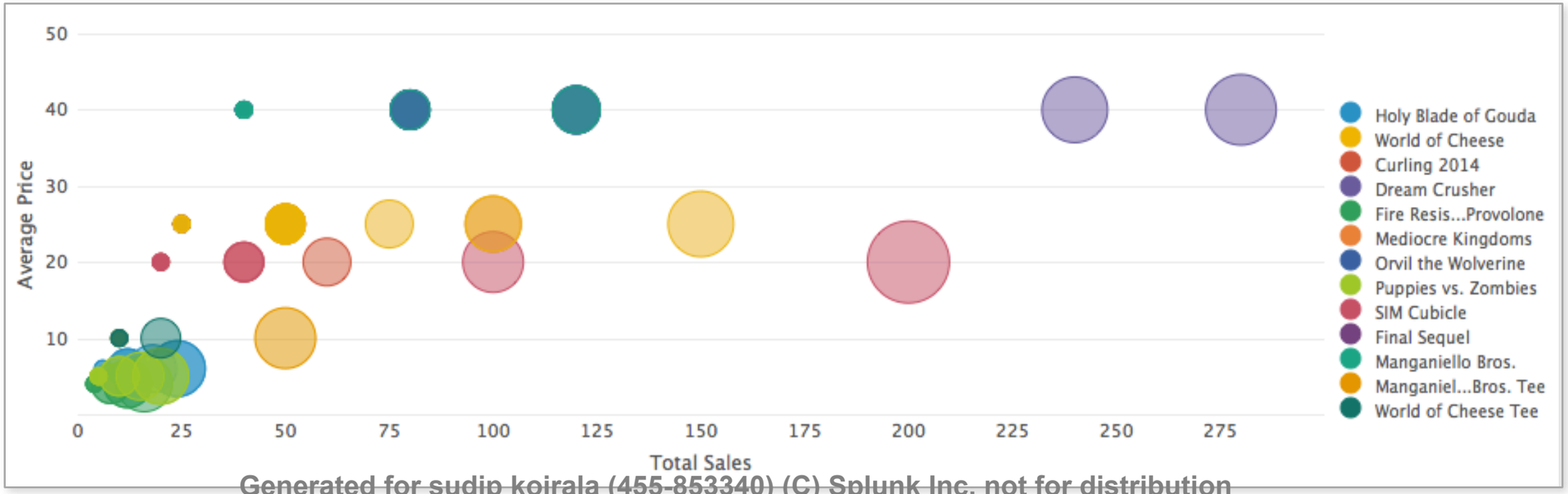


chart Command

- chart command can display any series of data that you want to plot
- You decide which field to plot on the x-axis
  - the function defines the value of the y-axis, therefore should be numeric
  - the first field after the over is the x-axis
  - using the over and by clauses divides the data into sub-groupings, the values from by will display in the legend

```
chart avg(bytes) over host
```

  - the host values will display over the x-axis

```
chart avg(bytes) over host by product_name
```

  - the host field is the x-axis and the series is further split by product_name

chart Command – over <field> Example

- This example shows a basic chart
- count function tallies the number of events for each value in the result set

Scenario

Display a count of vendor actions over the last 24 hours.

```
sourcetype=linux_secure
| chart count over vendor_action
```

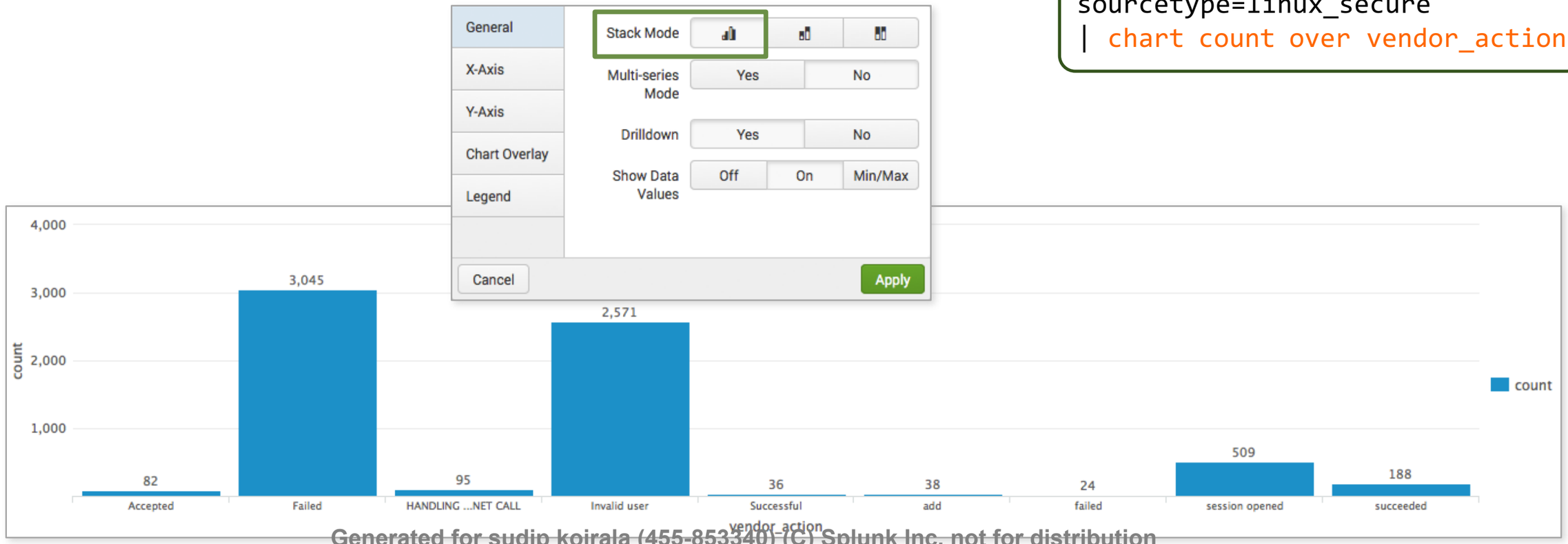


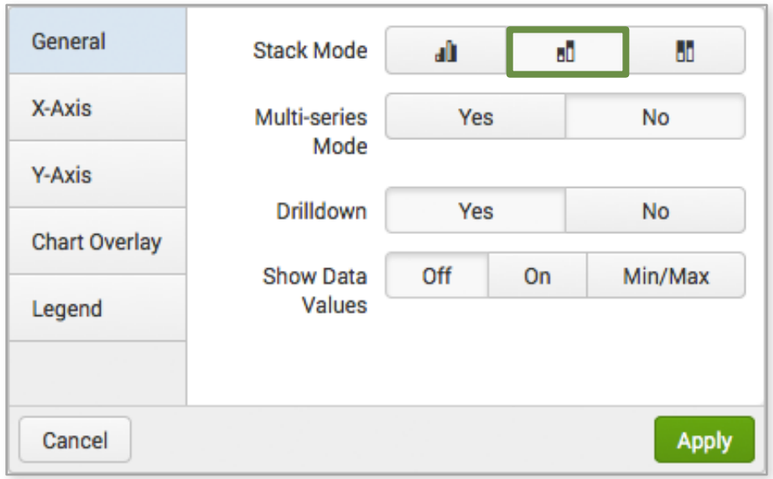
chart Command – over <field> by <field>

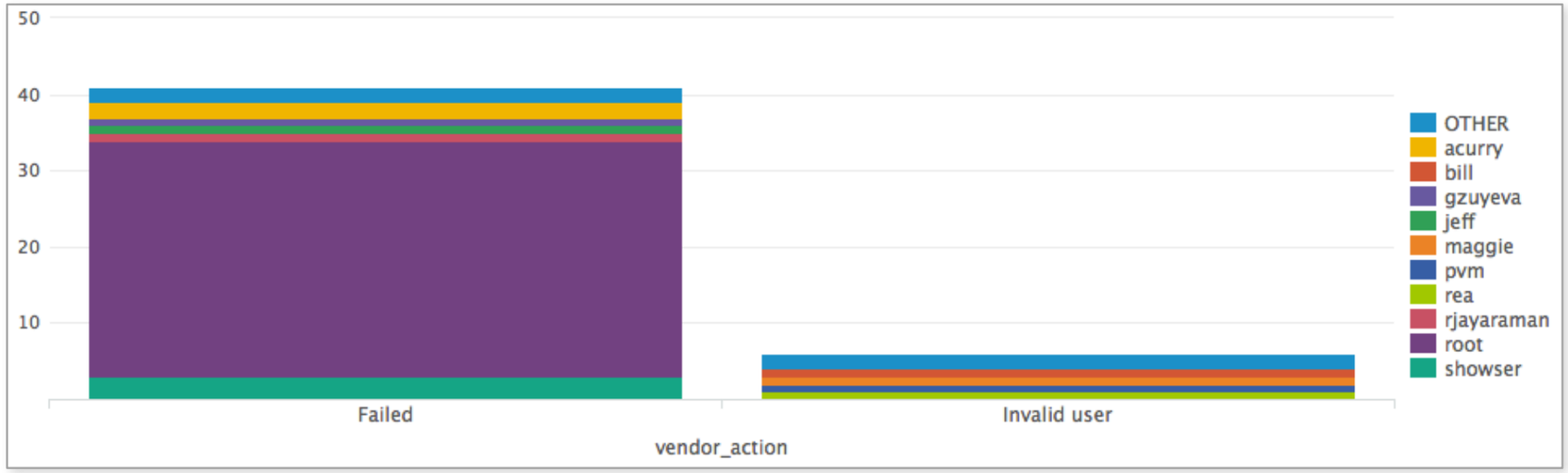
- In this example, results are grouped by vendor_action, then split by user
- This example displays stacked columns

Scenario

Display a count of vendor actions over the last 60 minutes.

```
sourcetype=linux_secure (invalid OR fail*)
| chart count over vendor_action by user
```





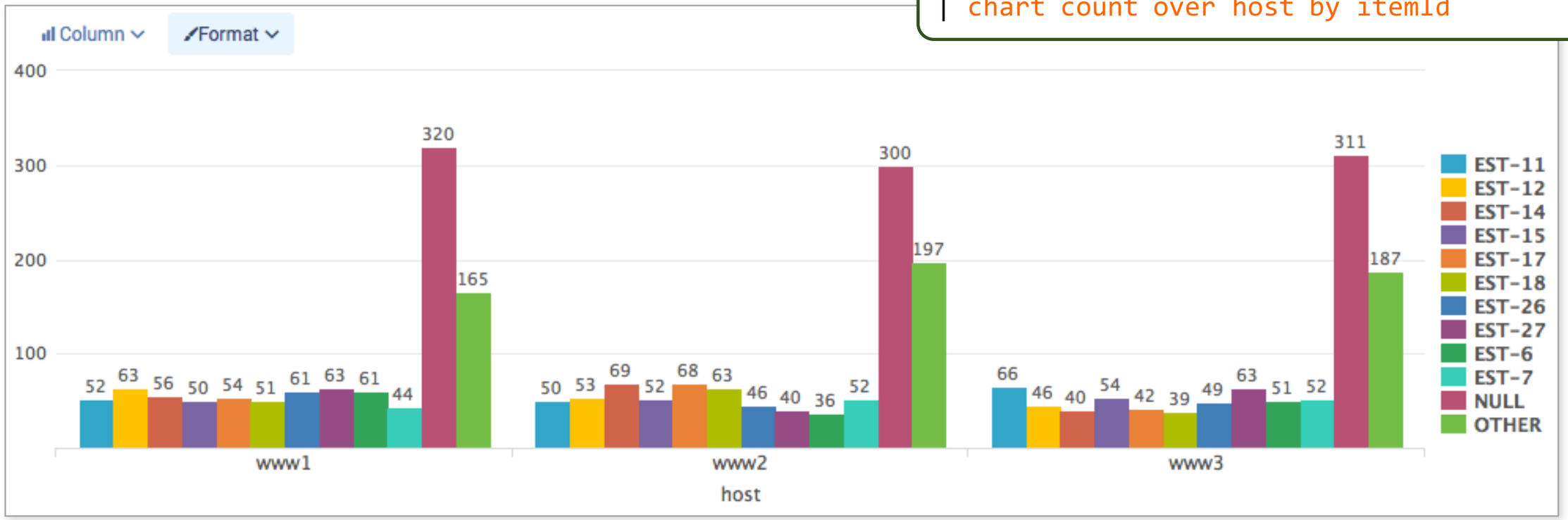
Including Null and Other Values

Notice the results are skewed by NULL and OTHER values, shown by default, that we may not want to show

Scenario

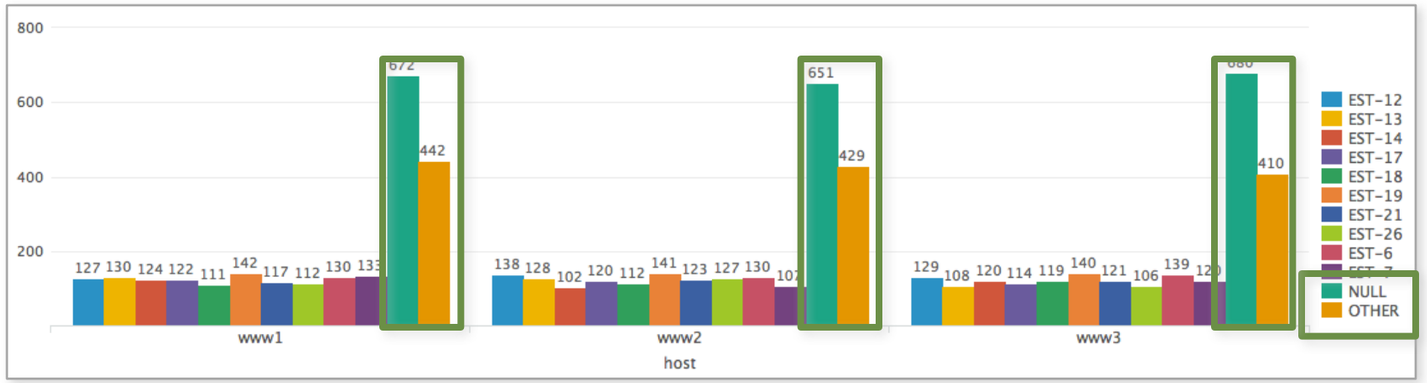
Display a count of unsuccessful web transactions by host for each item over the last 7 days.

```
sourcetype=access_combined status>399
| chart count over host by itemId
```

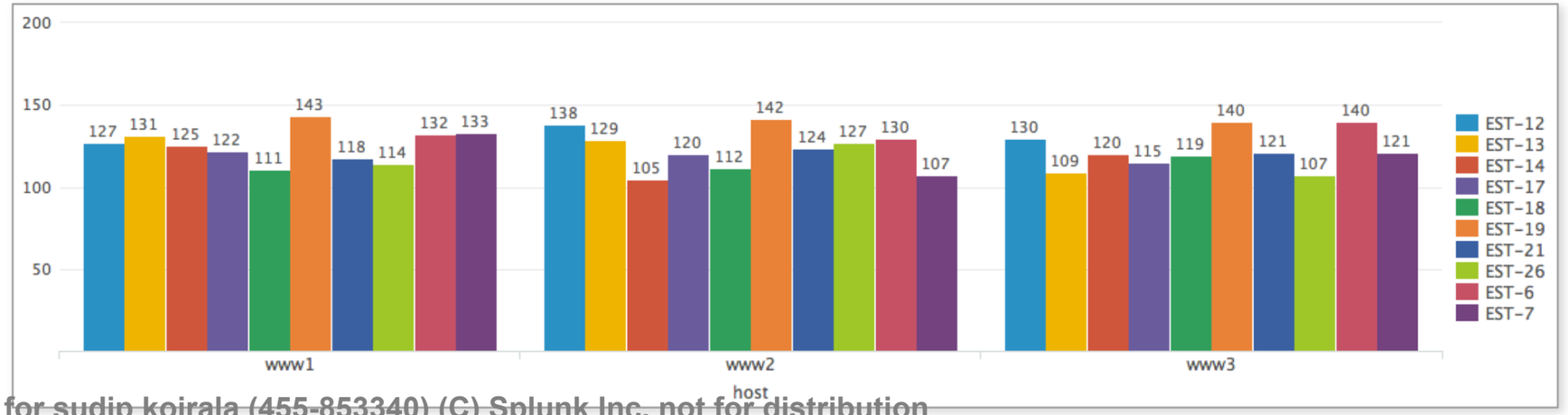


Omitting Null and Other Values

- `chart` and `timechart` commands automatically filter the series they plot to the 10 highest values
  - Surplus values are grouped into OTHER
- To remove empty (`null`) and other field values from the display, use these options:
  - `useother=f`
    `usenull=f`




```
sourcetype=access_combined status>399
| chart count over host by itemId
  useother=f usenull=f
```



Best Practice

To remove null values, add `itemId=*` to base search.

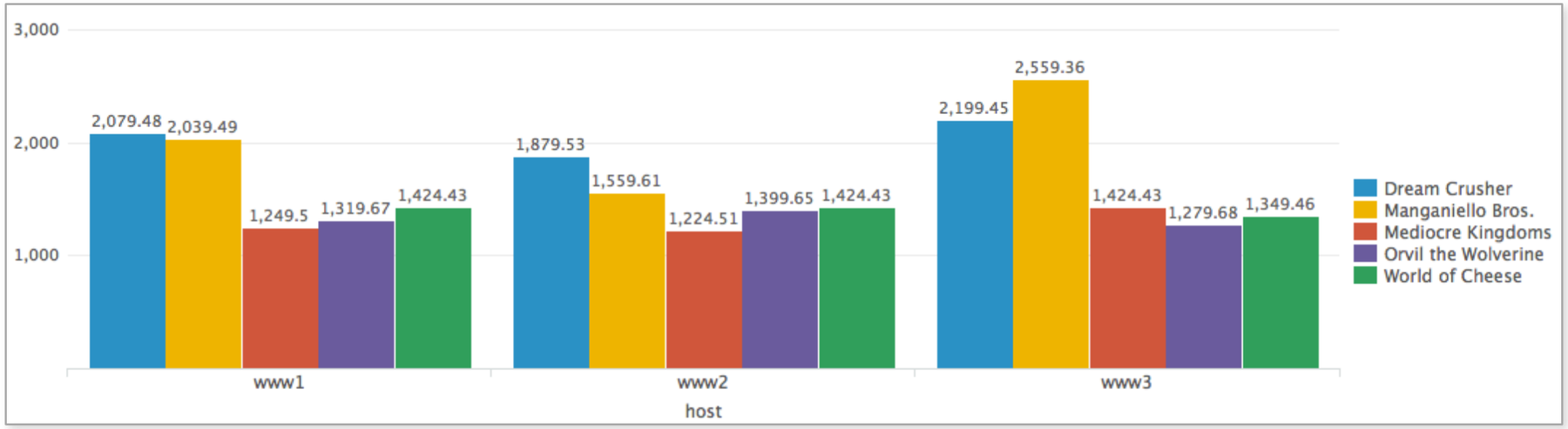
Limiting the Number of Values

- To adjust the number of plotted series, use the `limit` argument, limit=0 for unlimited

Scenario

Display sales per host for each product over the last 7 days.

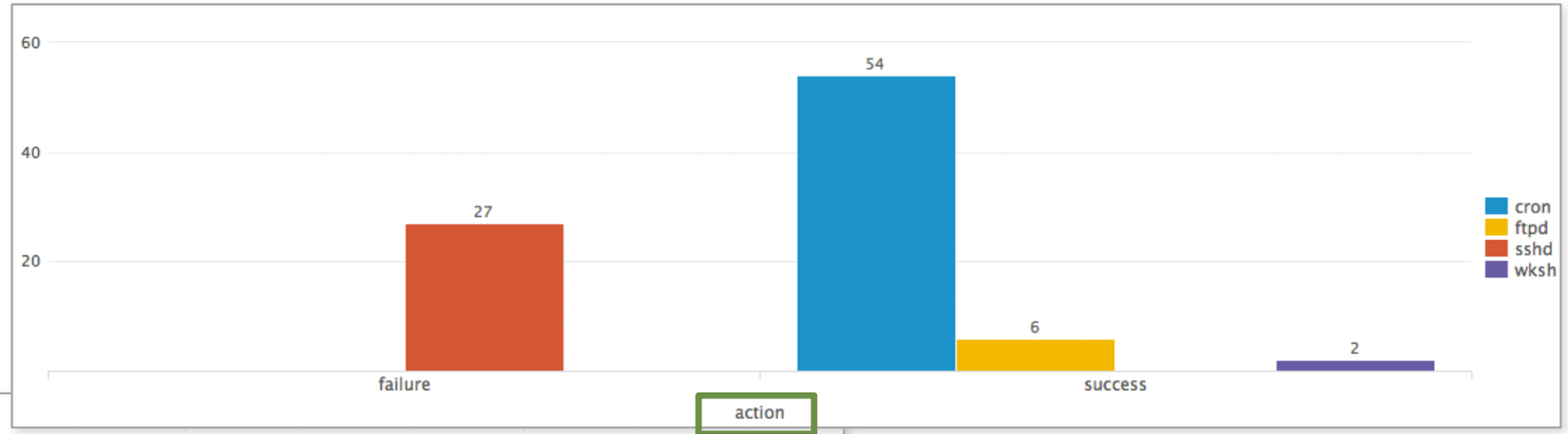
```
sourcetype=access_combined
  action=purchase status=200
| chart sum(price) over host
  by product_name limit=5 useother=f
```



Visualizations – x and y Axes

- For line, area, and column charts, the x axis is horizontal

```
sourcetype=linux_secure fail*
| chart count over src_country by src_city
```



- For bar chart, the x axis is vertical

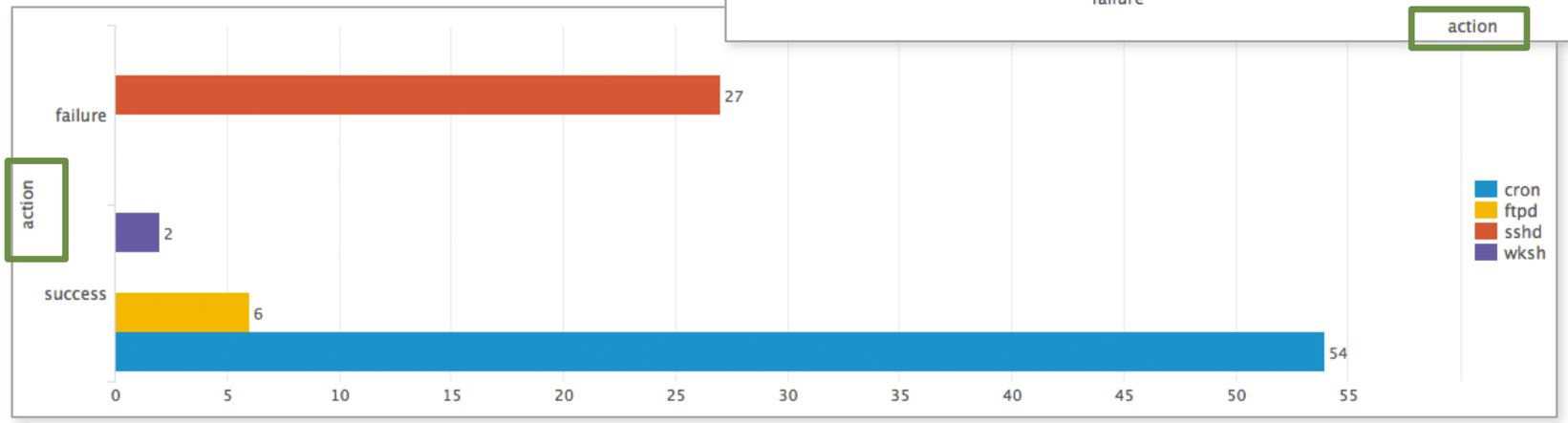
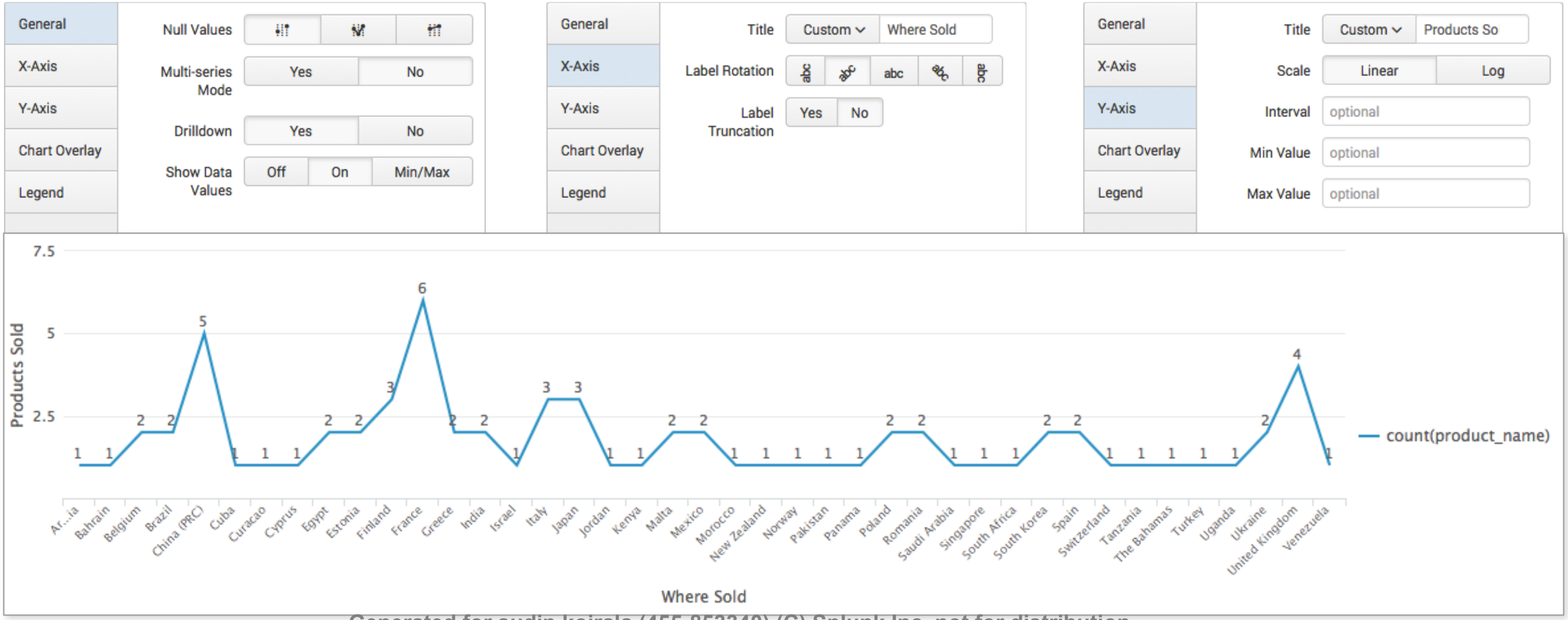


chart Command – Labeling

```
sourcetype=vendor_sales VendorID > 4000
| chart count(product_name) by VendorCountry
```



 | 30 June 2016

`timechart` Command – Overview

- `timechart` command performs statistical aggregations against time
- Plots and trends data over time
- `_time` is always the x-axis
- You can optionally split data using the “by” clause for one other field
  - Each distinct value of the "`split by`" field is a separate series in the chart
- Timecharts are best represented as line or area charts

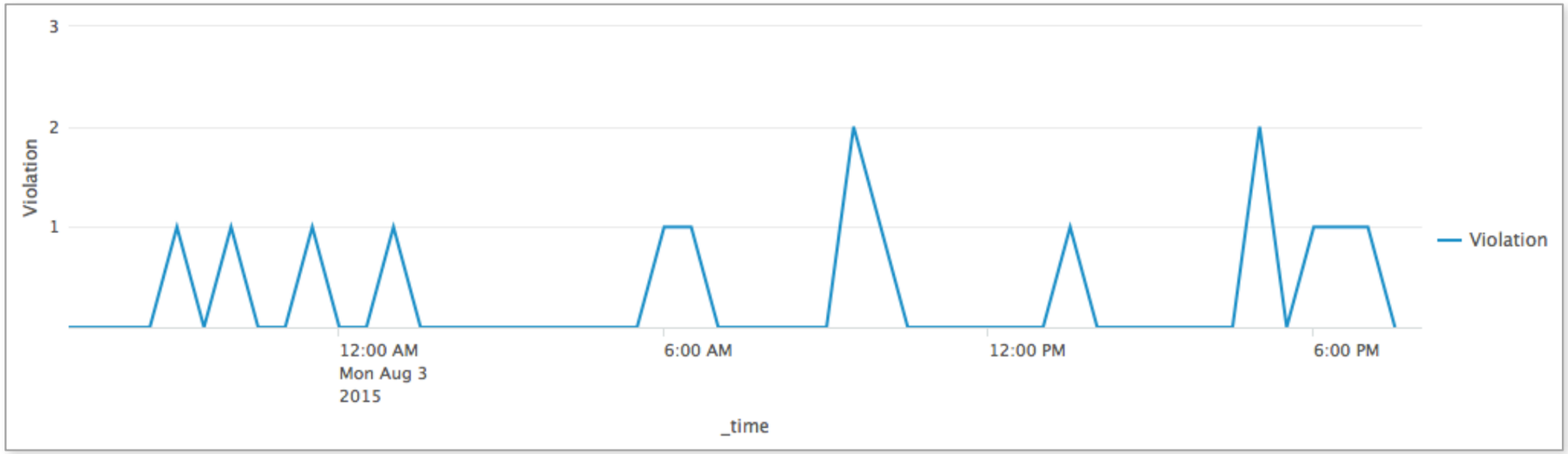
`timechart` Command – Example

This basic timechart displays the number of usage violations

Scenario

How many usage violations have occurred in the last 24 hours?

```
sourcetype=cisco_wsa_squid usage=Violation
| timechart count
```



Note

Functions and arguments used with stats and chart can also be used with timechart.

timechart Command – Multiple Values

- Splitting by the usage field, each line represents a unique field value
- y-axis represents the count for each field value

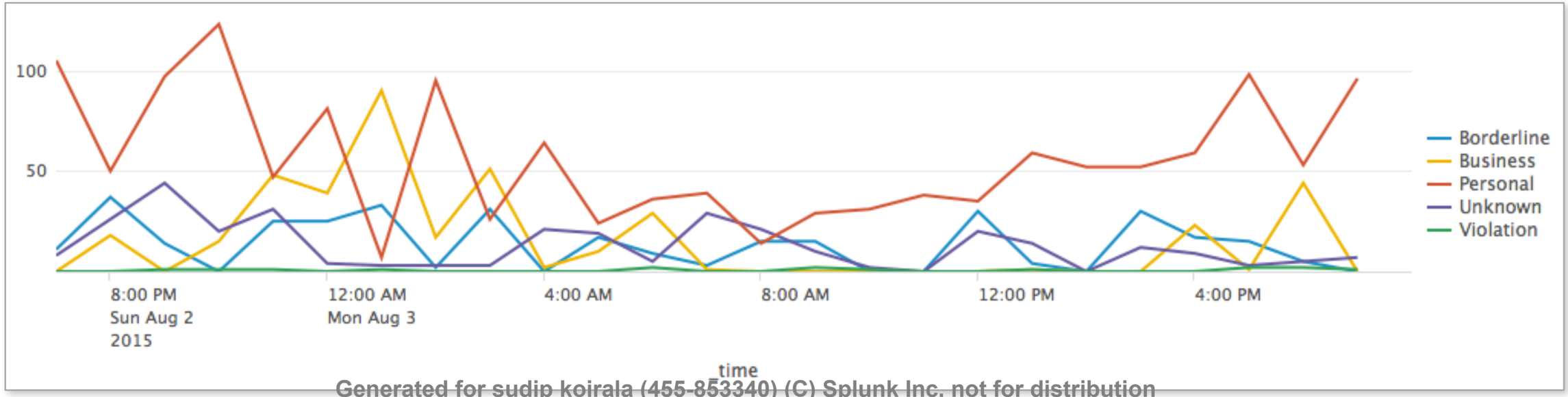
Scenario

What's the overall usage trend for the last 24 hours?

```
sourcetype=cisco_wsa_squid
| timechart count by usage
```

Note

Using timechart, you can split by a maximum of one field because _time is the implied first by field.



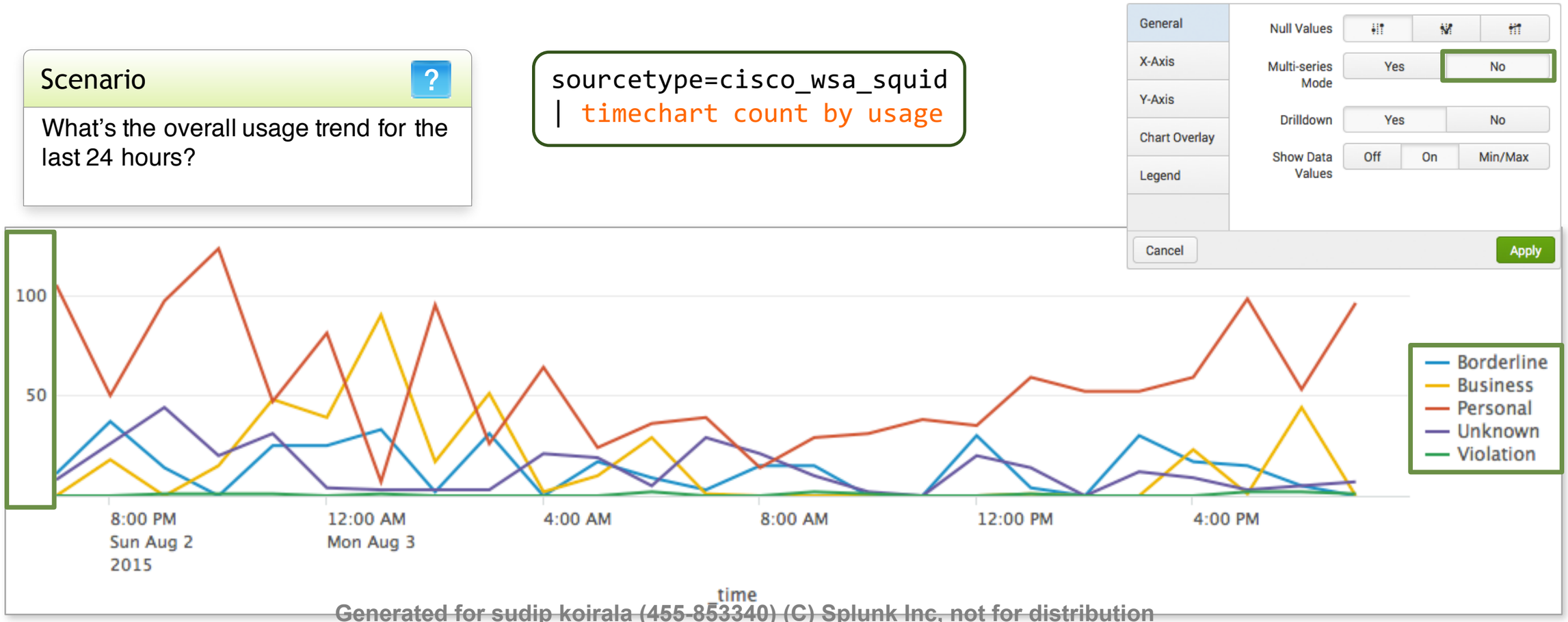
timechart Command – Multi-series: No

When the Multi-series mode is set to **No**, all fields share the y-axis

Scenario

What's the overall usage trend for the last 24 hours?

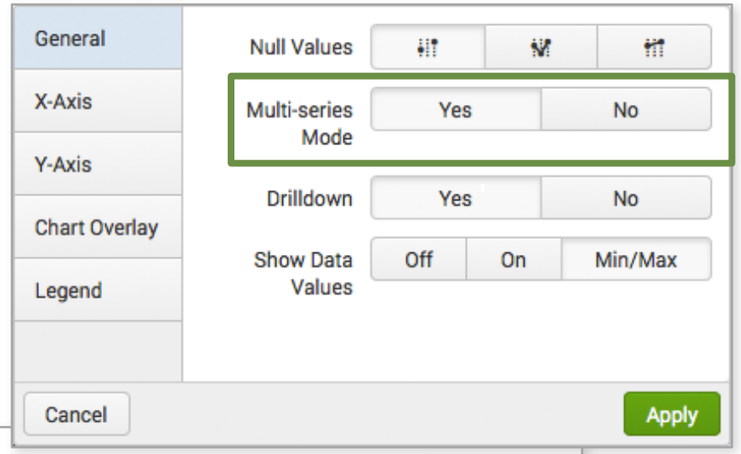
```
sourcetype=cisco_wsa_squid
| timechart count by usage
```

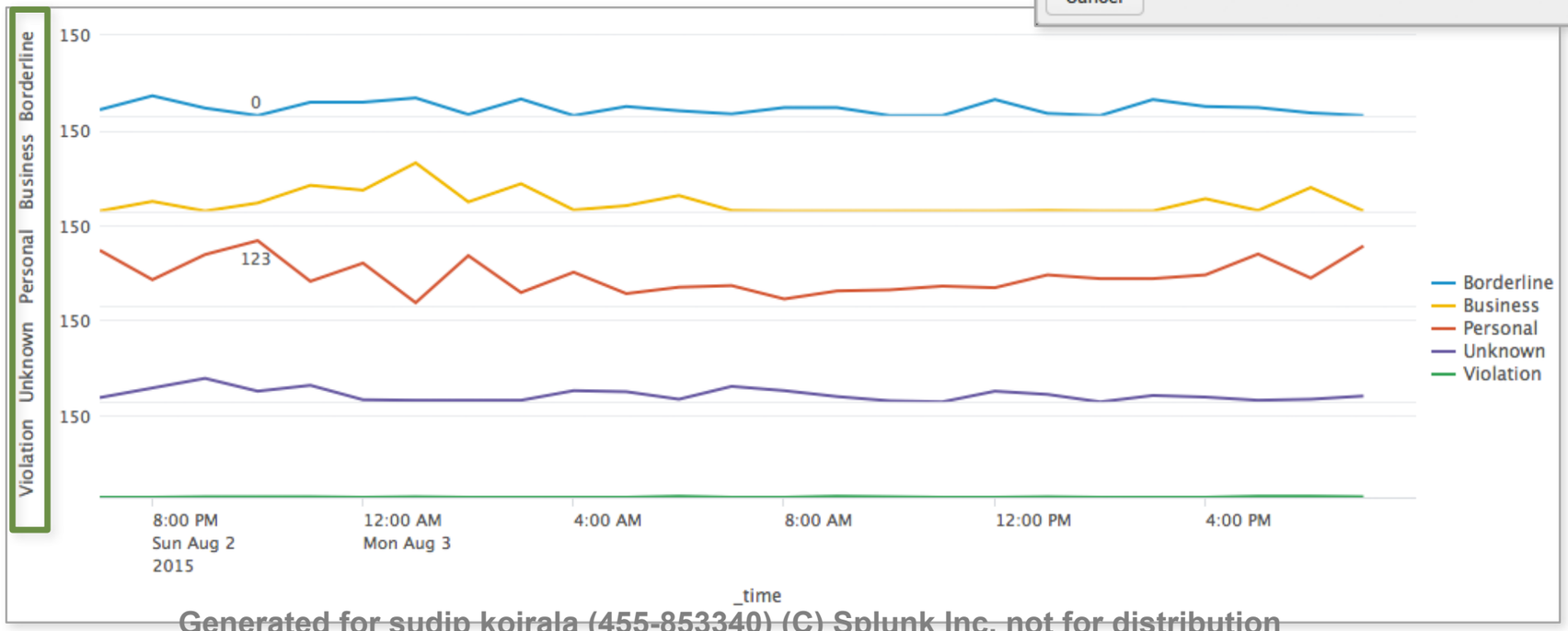


 | 30 June 2016

`timechart` Command – Multi-series: Yes

- Setting Multi-series mode to **Yes** causes the y-axis to split for each field value
- y-axis is divided into sections, each spanning the same max and `min` count





timechart Command – Adjusting the Sampling Interval

- The timechart command "buckets" the values of the _time field to provide dynamic sampling intervals based upon the time range of the search
- Example defaults:
  - Last 60 minutes uses span=1m
  - Last 24 hours uses span=30m
- Adjust the interval using the span argument, i.e. span=15m

```
sourcetype=linux_secure (invalid OR fail*)
vendor_action=*
| timechart span=15m count by vendor_action
```

✓ 428 events (10/27/15 12:46:00.000 PM to 10/27/15 4:46:12.000 PM)

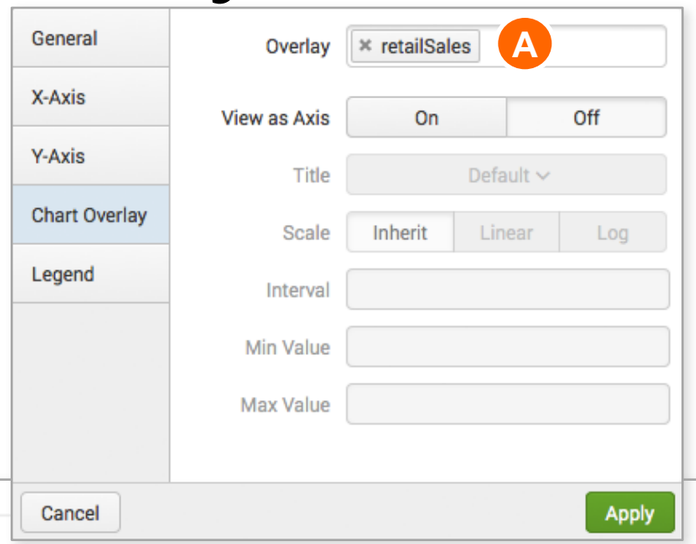
Events | Patterns | Statistics (17) | Visualization

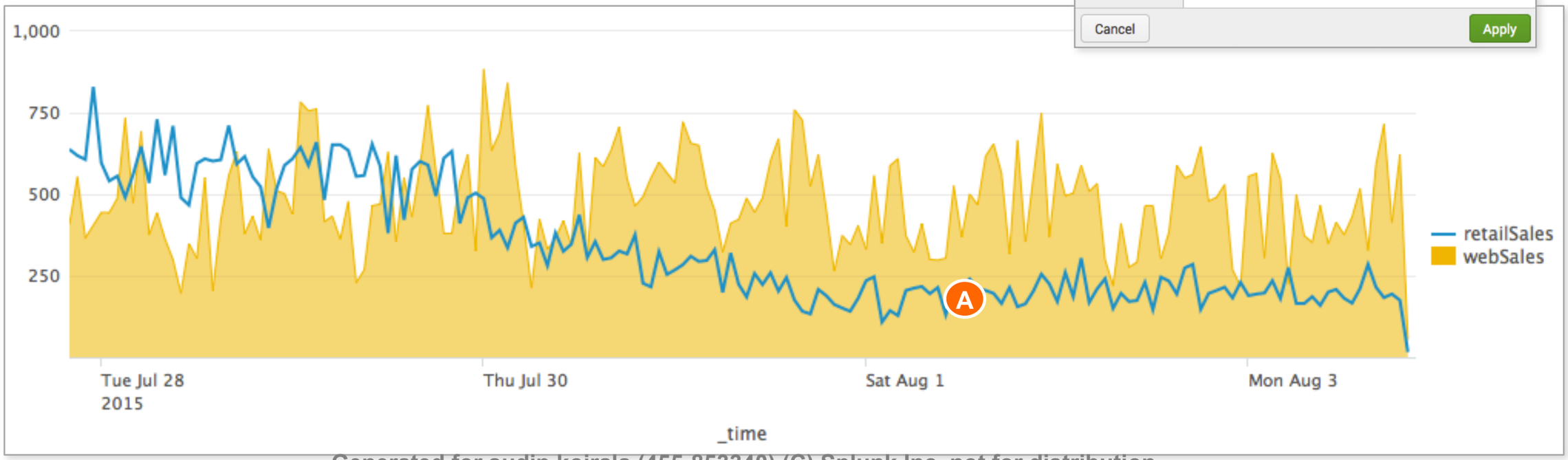
10 Per Page | Format | Preview

_time	Failed	Invalid user
2015-10-27 12:45:00	21	2
2015-10-27 13:00:00	26	2
2015-10-27 13:15:00	26	8
2015-10-27 13:30:00	20	3
2015-10-27 13:45:00	20	3
2015-10-27 14:00:00	32	2
2015-10-27 14:15:00	29	3
2015-10-27 14:30:00	36	2
2015-10-27 14:45:00	26	4
2015-10-27 15:00:00	36	1

Formatting – Chart Overlay

```
(sourcetype=access_combined action=purchase status<400)
 OR sourcetype=vendor_sales
| timechart span=1h sum(price) by sourcetype
| rename access_combined as webSales, vendor_sales as retailSales
```





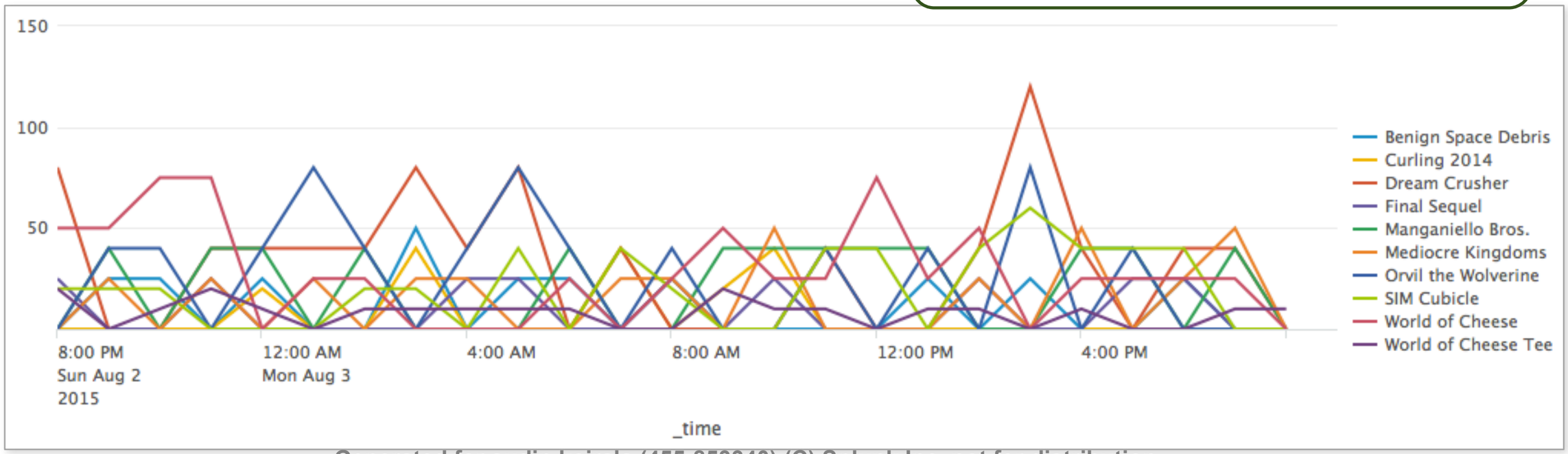
timechart Command – Statistical Functions

As with the stats and chart commands, you can apply statistical functions to the timechart command

Scenario

How much retail revenue did we receive from each product during the last 24 hours?

```
sourcetype=vendor_sales
| timechart sum(price) by product_name
  useother=f usenull=f
```



Transforming Command Summary

Feature	stats	chart	timechart
Multi-level breakdown [by clause]	Many	2	1
Limit # series shown	NA	limit=*n* *Default=10*	limit=*n* *Default=10*
Filter other series	NA	useother=f	useother=f
Filter null values	NA	usenull=f	usenull=f
Set time value on x axis	NA	NA	span

Transforming Command Summary (cont.)

To count the frequency of a field(s), use top/rare

```
sourcetype=linux_secure
 | top src_ip, user, vendor_action, app
```

src_ip	user	vendor_action	app	count	percent
10.16.1.6	ftpuser	FTP LOGIN	ftpd	168	0.391691
10.16.1.6	root	FTP LOGIN	ftpd	166	0.387028
10.16.1.6	jdoe	FTP LOGIN	ftpd	166	0.387028
10.11.36.5	naughtyuser	Failed	sshd	48	0.111912
10.11.36.41	naughtyuser	Failed	sshd	47	0.109580

```
sourcetype=linux_secure
 | rare src_ip, vendor_action, app
```

src_ip	user	vendor_action	app	count	percent
10.1.10.172	administrator	Failed	sshd	1	0.002336
10.1.10.172	art	Failed	sshd	1	0.002336
10.1.10.172	db2inst1	Failed	sshd	1	0.002336
10.1.10.172	db4	Failed	sshd	1	0.002336
10.1.10.172	desktop	Failed	sshd	1	0.002336

Transforming Command Summary (cont.)

Use `stats` to calculate statistics for two or more by fields (non time-based)

```
sourcetype=linux_secure
 | stats count by src_ip, user, vendor_action, app
```

src_ip	user	vendor_action	app	count
10.1.10.172	admin	Failed	sshd	2
10.1.10.172	administrator	Failed	sshd	1
10.1.10.172	apache	Failed	sshd	3
10.1.10.172	art	Failed	sshd	1
10.1.10.172	db	Failed	sshd	3
10.1.10.172	db2inst1	Failed	sshd	1
10.1.10.172	db4	Failed	sshd	1
10.1.10.172	desktop	Failed	sshd	1
10.1.10.172	email	Failed	sshd	1
10.1.10.172	games	Failed	sshd	1

Transforming Command Summary (cont.)

To calculate statistics with an arbitrary field as the x-axis (not `_time`), use `chart`

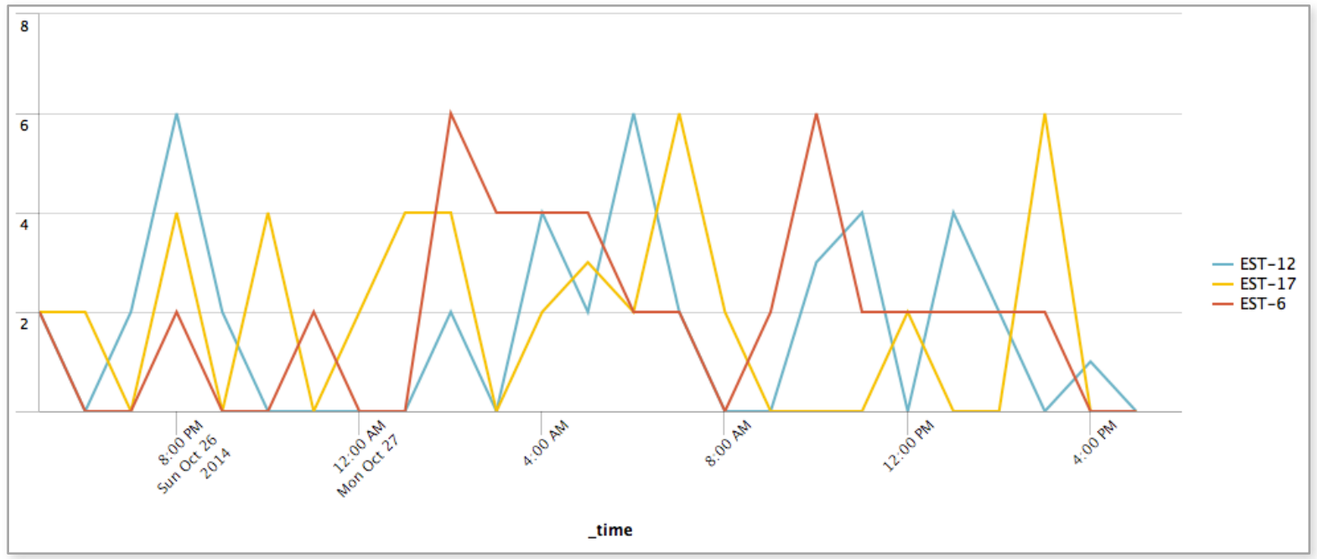
- When you use a by field, the output is a table where each column represents a distinct value of the split-by field

```
sourcetype=access_combined
  action=purchase
| chart sum(price) over host
  by itemId limit=5 useother=f
```

host	EST-13	EST-14	EST-16	EST-17	EST-7
www1	39.99		64.98	69.97	
www2	39.99	74.97	49.98		54.97
www3	9.99		39.99	49.98	24.99

Transforming Command Summary (cont.)

- Use `timechart` to calculate statistics with `_time` as the x-axis
- If a by field is used, the output is a table where each column represents a distinct value of the split-by field




```
... |  timechart span=1h count by itemId limit=3 useother=f
```

_time	EST-12	EST-17	EST-6
2014-10-26 17:00	2	2	2
2014-10-26 18:00	0	2	0
2014-10-26 19:00	2	0	0
2014-10-26 20:00	6	4	2
2014-10-26 21:00	2	0	0
2014-10-26 22:00	0	4	0
2014-10-26 23:00	0	0	2

Module 5: Transforming Commands, Part 3 Enriching Visualizations

Module Objectives

- Use the `trendline` command
- Create `maps`
  - `iplocation`
  - `geostats`
  - `geom`
- Create and format `single values`
- Use the `addtotals` command

trendline Command

- trendline computes the moving averages of a field

```
trendline <trendtype><period> (<field>) [AS <newfield>]
```

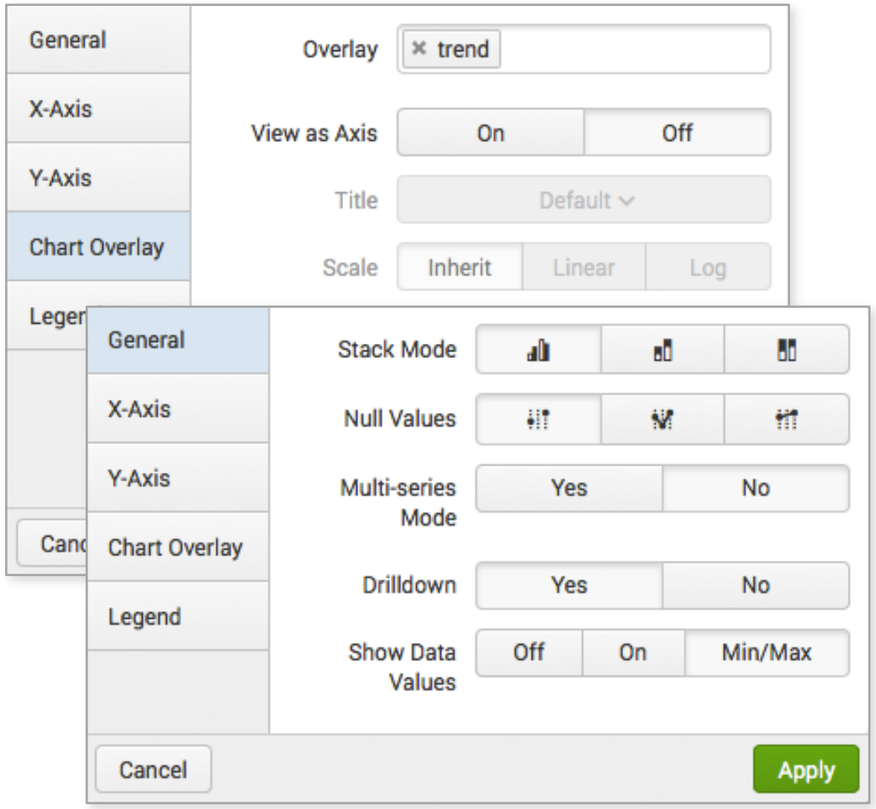
- trendtype:
  - sma - simple moving average
  - ema - exponential moving average
  - wma - weighted moving average
- Define the period over which to compute the trend; an integer between 2 and 10000, e.g., sma2
- trendtype requires the period parameter; for example, sma(sales) would fail as it is missing an integer, the defining period

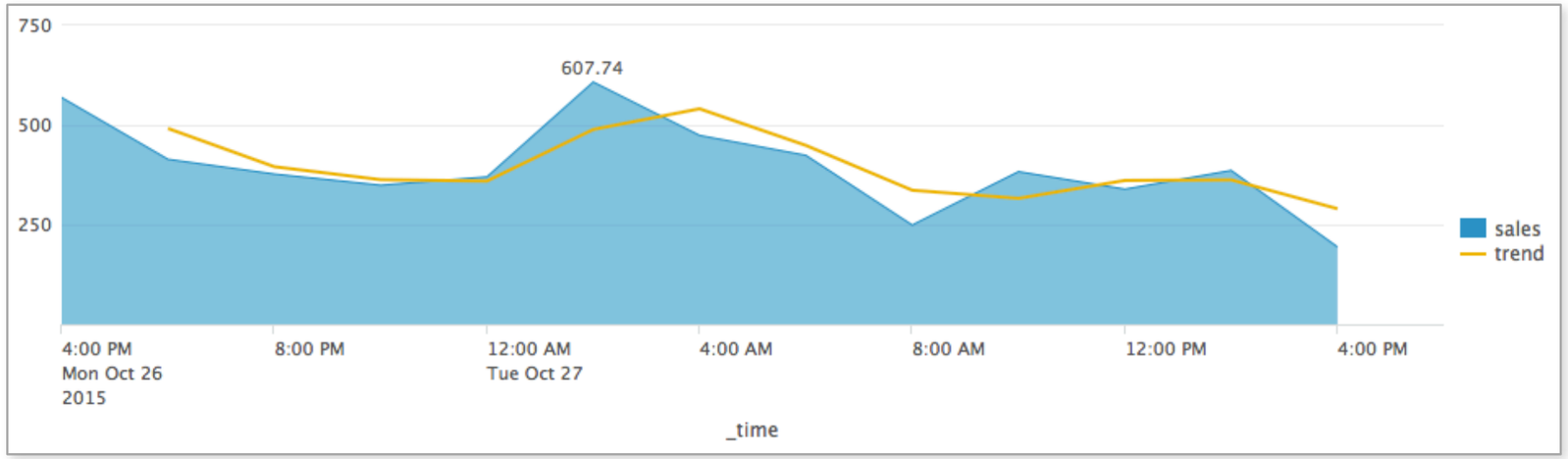
trendline Command – Example

Scenario

Display total sales and sales trend over the 24 hours.

```
sourcetype=access_combined action=purchase status=200
| timechart span=2h sum(price) as sales
| trendline sma2(sales) as trend
```



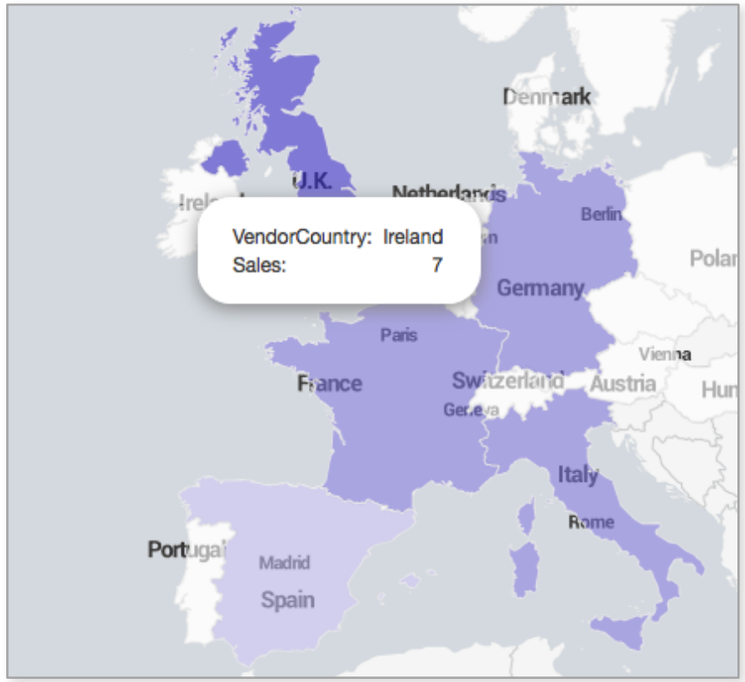


Viewing Results as a Map

- There are two map types:

Map

Choropleth



iplocation Command

Scenario

Failed logins to the network during the last 60 minutes.

```
sourcetype=linux_secure (fail* OR invalid)
| iplocation src_ip
```

- Use iplocation to look up and add location information (city, country, metro code, region, timezone, latitude and longitude) to an event
- Not all of the information is available for all ip address ranges
- Automatically defines the default lat and lon fields required by geostats

 | 30 June 2016

geostats Command

- Use geostats to compute statistical functions and render a world map

```
geostats [latfield=string] [longfield=string] [stats-agg-term]*
[by-clause]
```

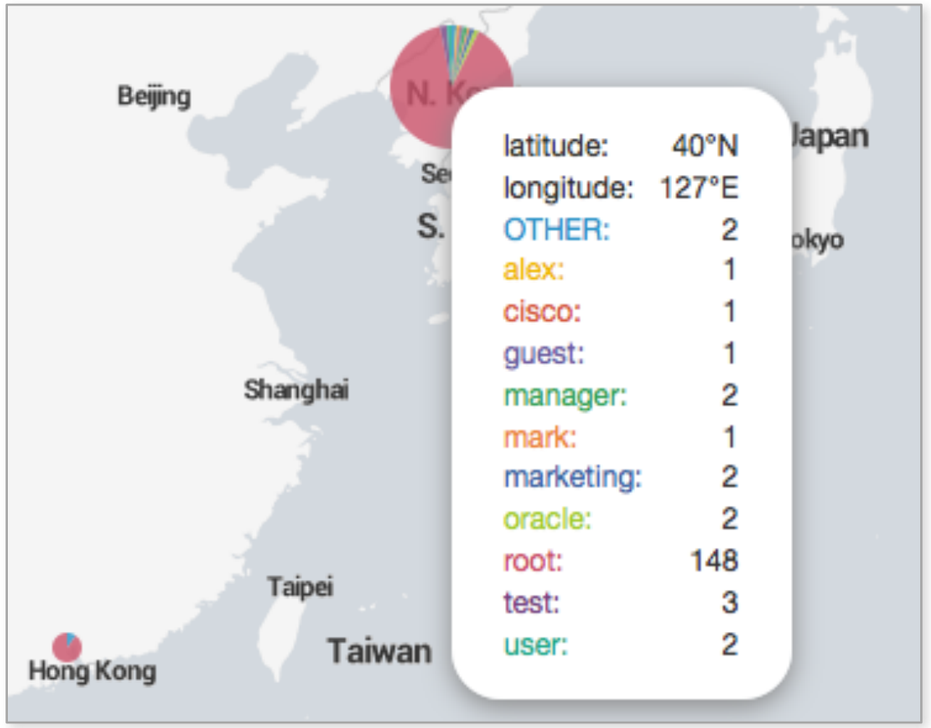
- Data must include latitude and longitude values
- Define the latfield and longfield only if they differ from the default lat and lon fields
- Use with the iplocation command to define lat and lon fields
- To control the column count, use the globallimit argument

geostats Command – Example

Scenario

Map the users of failed actions on the network worldwide during the last 24 hours.

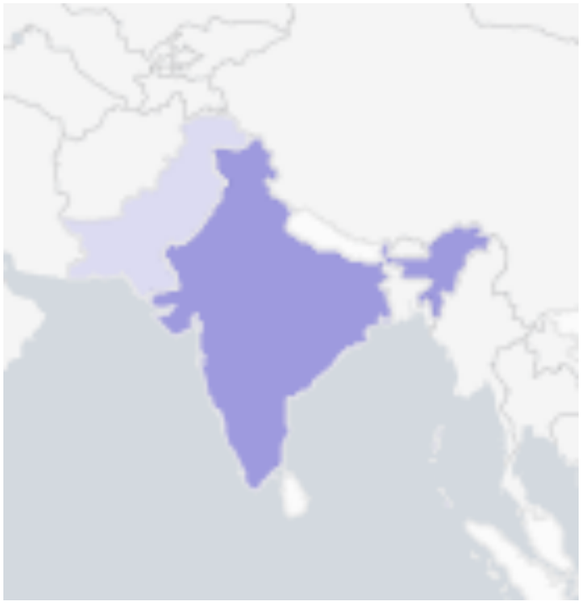
```
sourcetype=linux_secure (fail* OR invalid)
| iplocation src_ip
| geostats globallimit=20 count by user
```



geobin	latitude	longitude	OTHER	alex	cisco	guest	manager	mark	marketing	oracle	root	test	user
bin_id_zl_0_y_2_x_4	-29.00000	24.00000	2							2		1	4
bin_id_zl_0_y_5_x_2	41.14120	-73.26370	14								348		
bin_id_zl_0_y_5_x_4	30.26289	31.31977	3	2		5	2			2		2	2
bin_id_zl_0_y_5_x_5	35.69610	51.42310	3		1			1	1	5	104	1	2
bin_id_zl_0_y_5_x_6	39.29653	126.42708	2	1	1	1	2	1	2	2	157	3	2

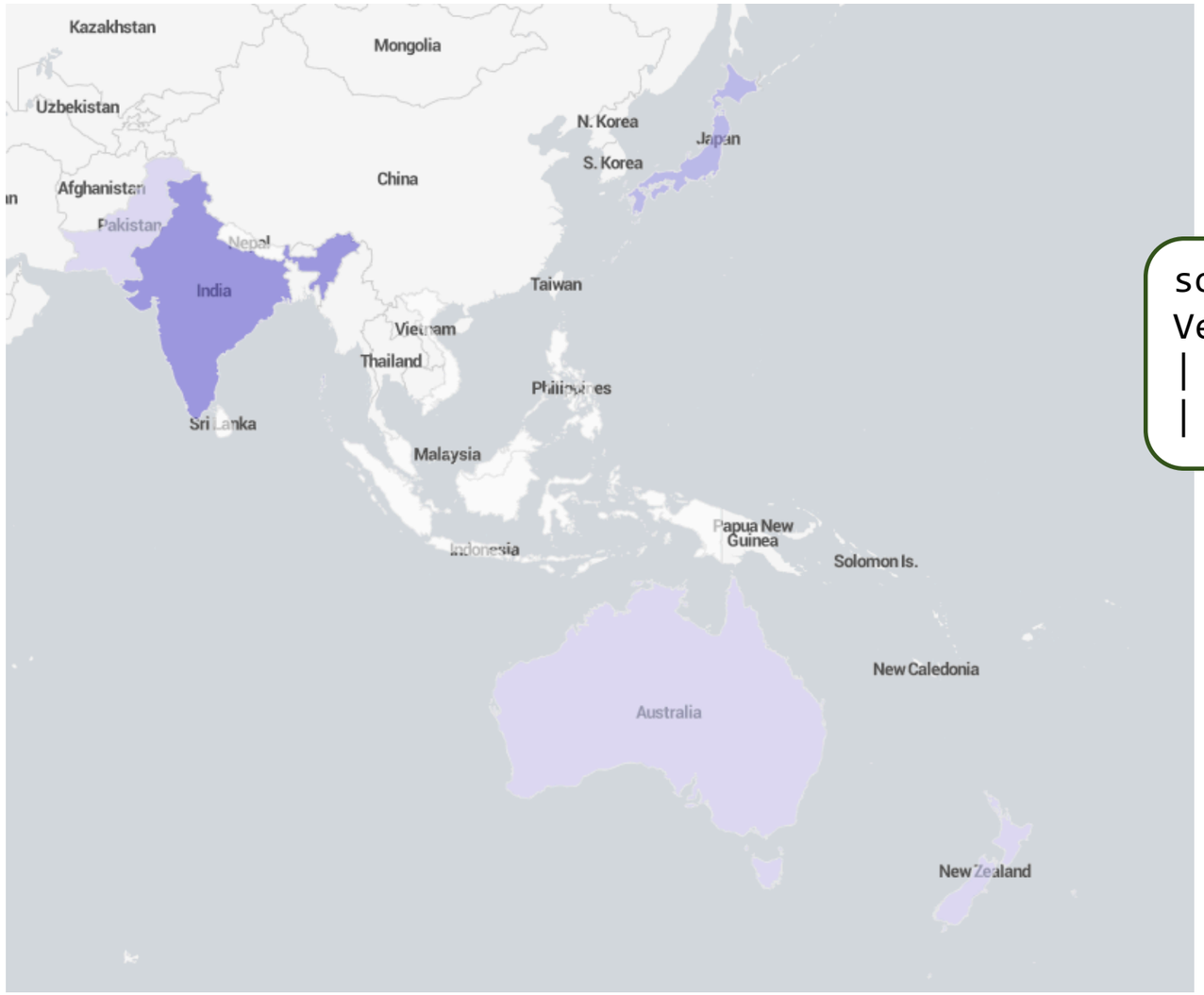
Choropleth Map

- Uses shading to show relative metrics, such as sales, network intruders, etc, for predefined geographic regions
- Must have a KMZ, or compressed Keyhole Markup Language, file that defines region boundaries
- Splunk ships with:
  - `geo_us_states`, United States
  - `geo_countries`, countries of the world



```
...| geom [<featureCollection>] [featureIdField=<string>]
```


geom Command



Scenario

Display previous week's retail sales in APAC.

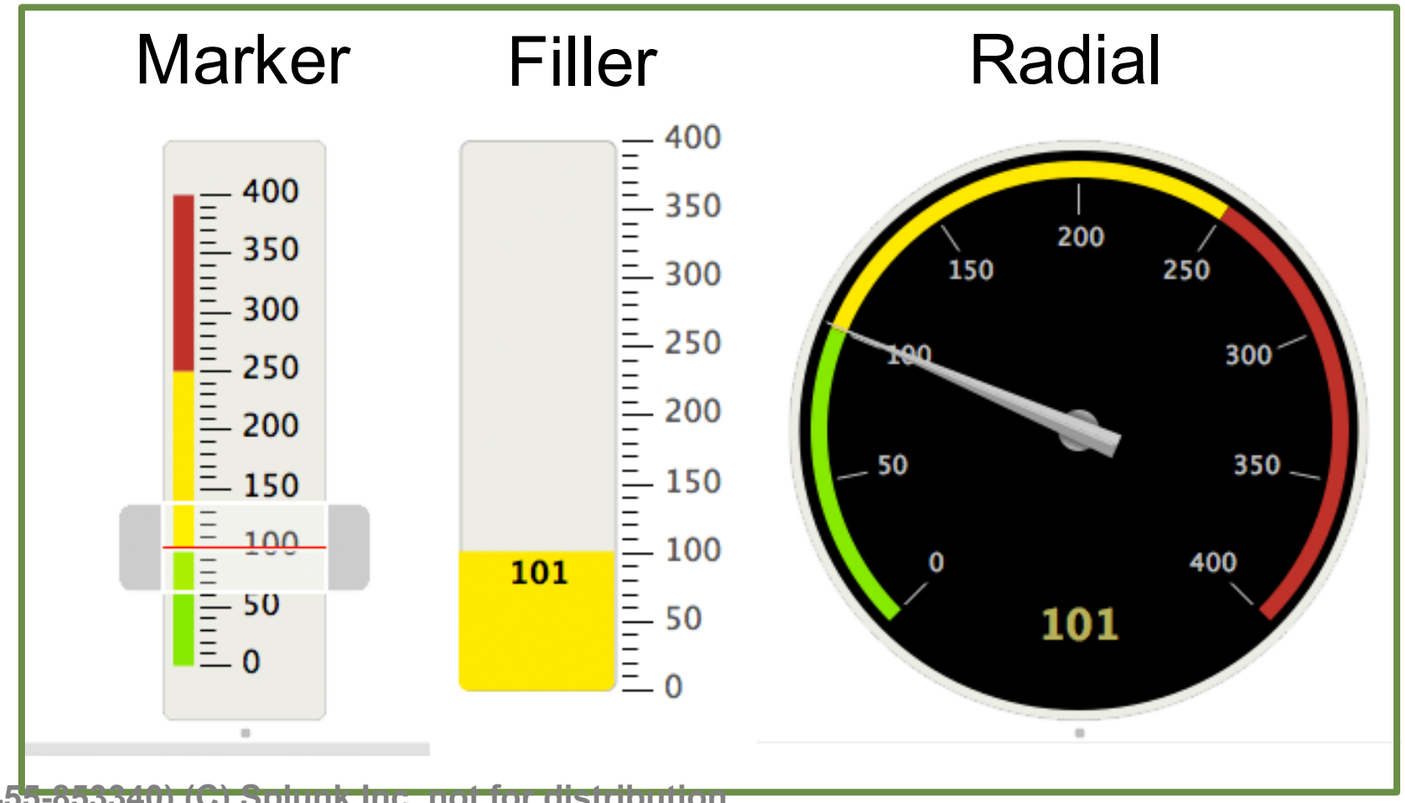
```
sourcetype=vendor_sales
VendorID > 7000 AND VendorID < 8999
| stats count as Sales by VendorCountry
| geom geo_countries featureIdField=VendorCountry
```

Viewing Results as a Single Value

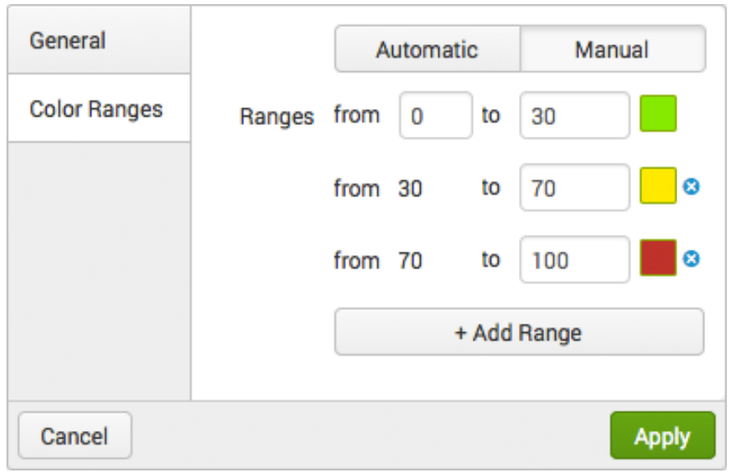
- There are two Single Value types:
  - Single Value
  - Gauge

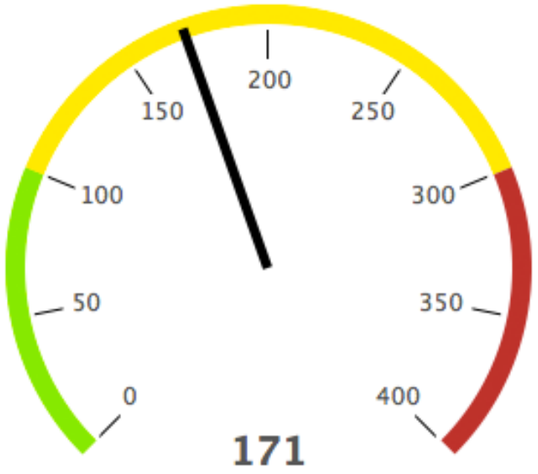
```
sourcetype=linux_secure vendor_action=failed
| stats count as count
| gauge count 0 100 250 400
```

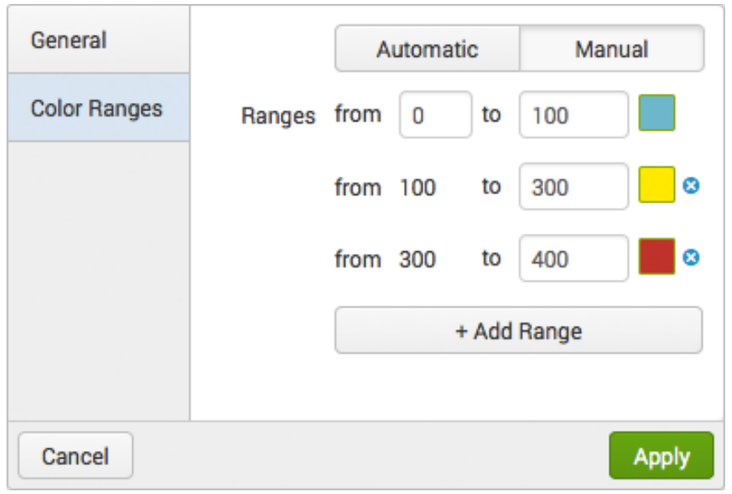
101

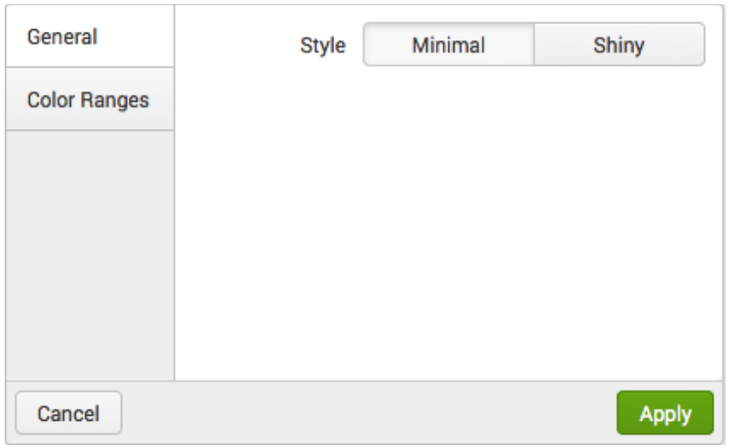


Single Value Visualizations: Format

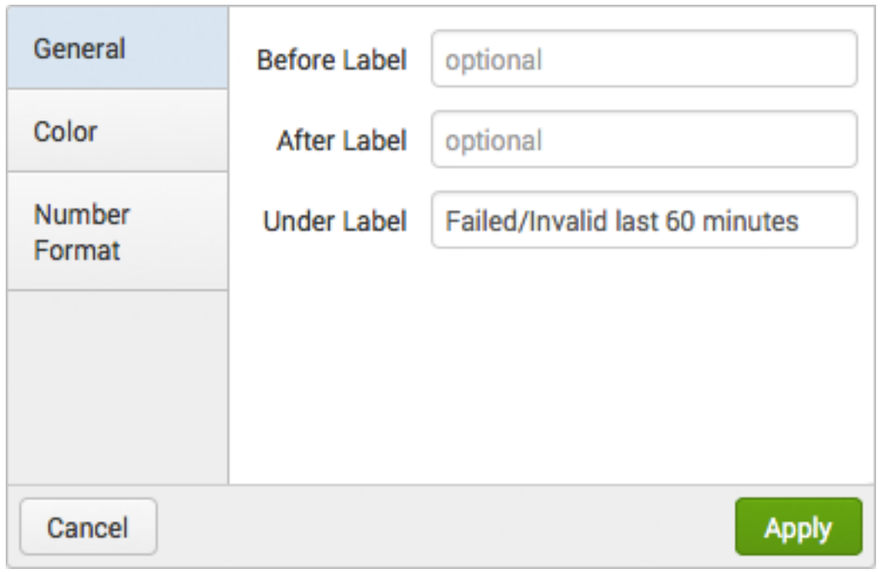






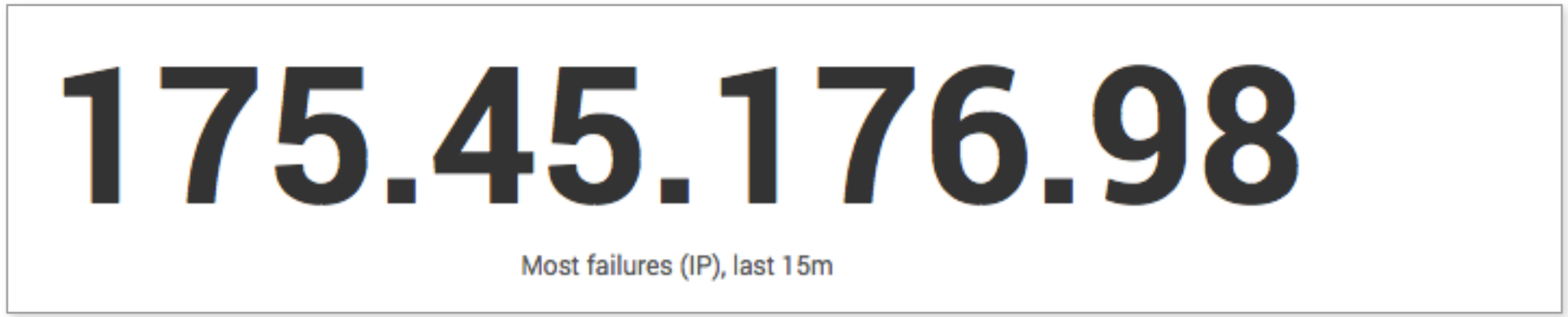


Single Value Visualizations: Single Value



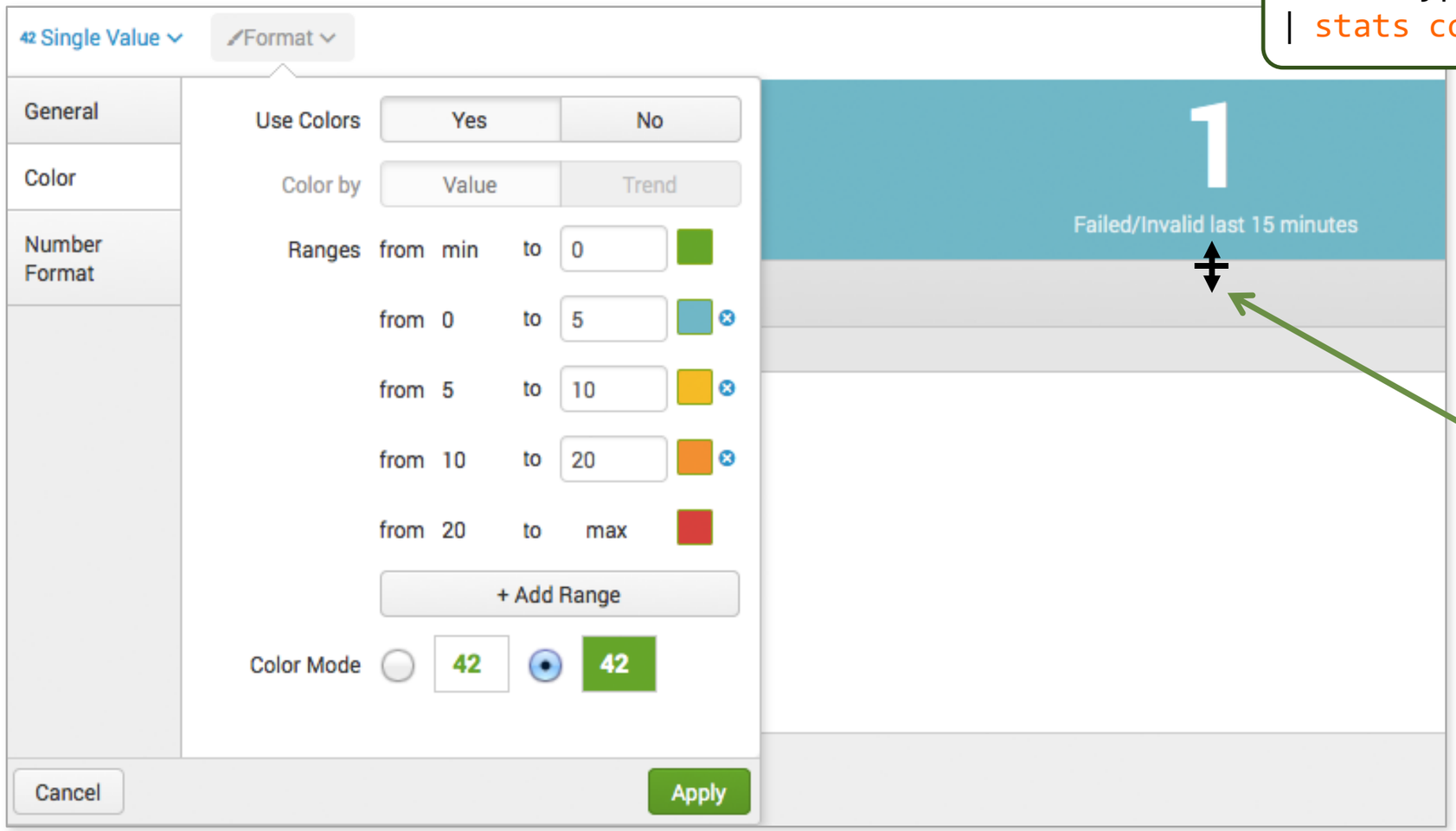

```
sourcetype=linux_secure (fail* OR invalid)
| stats count(vendor_action)
```

```
sourcetype=linux_secure (fail* OR invalid)
| chart count by src_ip
| sort -count
```



Single Value Visualizations: Single Value (cont)

```
sourcetype=linux_secure fail* OR invalid
| stats count
```

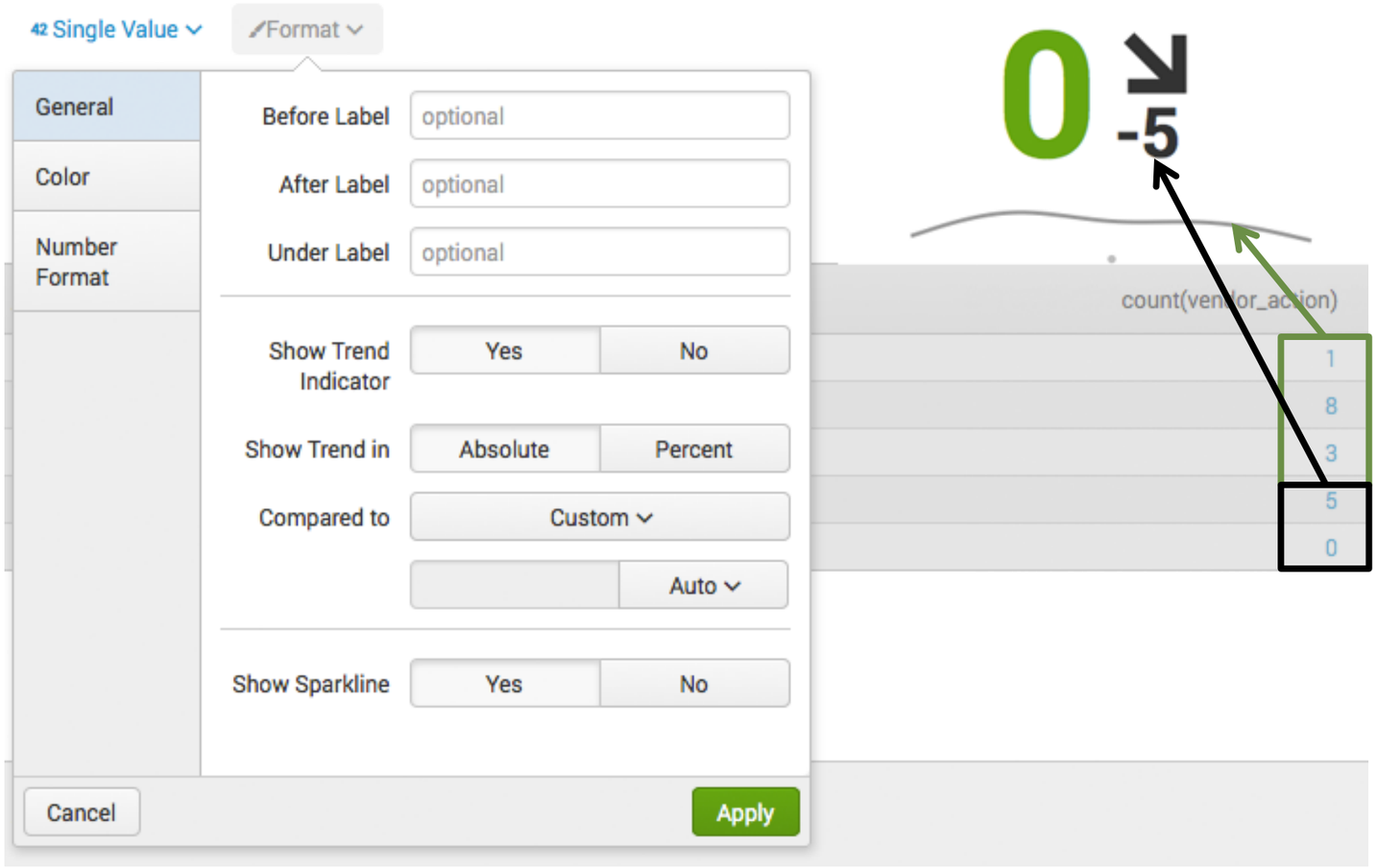


To resize the font, resize the pane

Single Value Visualizations: `timechart`

- With the `timechart` command, you can add a `sparkline` and a `trend`

```
sourcetype=linux_secure fail* OR invalid
| timechart span=15m count(vendor_action)
```



addtotals Command

- Use the addtotals command to compute the sum of **all** numeric fields for each row and column:

  addtotals [row=*bool*] [fieldname=*field*] [col=*bool*] [labelfield=*field*] [label=*string*] *field-list*

- Options:
  - By default, adds the total of the numeric rows in a field called Total; row=true
  - Use fieldname=<field> to rename the Total field
  - You can specify to total each numeric column; col=true
  - To name the column total row, use label=<string> and then identify where to display the label by using labelfield=<field>
  - If fields are not specified, then all numeric columns are totaled

addtotals Command – Example

Scenario

Display the retail products sold by country with totals by product and by country during the last 4 hours.

- row=t (default) counts the fields in each row under a column named "Total Per Product"
- col=t counts the fields in each row in a row named "Total Per Country"

```
sourcetype=vendor_sales
| chart count over product_name by VendorCountry
| addtotals
  fieldname="Total Per Product" A
  col=t B
  label="Total Per Country”  labelfield=product_name C
```

product_name C	Indonesia	Norway	B United States	A Total Per Product
Curling 2014	0	0	1	1
Dream Crusher	0	0	4	A 4
Fire Resistance Suit of Provolone	0	0	2	2
Manganiello Bros.	0	0	1	1
Puppies vs. Zombies	0	0	3	3
SIM Cubicle	0	1	3	4
World of Cheese	1	0	2	3
Total Per Country C	1	1	B 16	18

addtotals Command – Example 2

Scenario

Display the total number of events with total and average size (in bytes) by web server, and total the Bytes

```
sourcetype=access_combined
| stats sum(bytes) as Bytes,
  avg(bytes) as avgBytes,
  count as totalEvents by host
| addtotals row=f A col=t B label=totalBytes C
  labelfield=host D Bytes E
```

- A Do not total rows
- B Total columns
- C Add the label totalBytes
- D Place the label under the host column
- E Only total the Bytes column

D host	B Bytes	avgBytes	totalEvents
www1	444653	2039.692661	218
www2	470741	2120.454955	222
www3	537626	2133.436508	252
C totalBytes	E 1453020		

Module 6: Manipulating and Filtering Results

Module Objectives

- Use the `eval` command to:
  - Perform calculations
  - Convert values
  - Round values
  - Format values
  - Use conditional statements
- Use the `search` and `where` commands to filter calculated results
- Use `fillnull` command

`eval` Command – Overview

- `eval` allows you to calculate and manipulate field values in your report
  - Useful for calculations such as add, subtract, multiply, divide
  - Does not re-write event data into the index
- Supports a variety of functions
- Results of `eval` are written to a specified field
  - Can be a new or existing field
  - If the destination field exists, the values of the field are replaced by the results of `eval`
  - Field values are treated in a case-sensitive manner

eval Command

- The eval command allows you to:
  - Calculate expressions
  - Place the results in a field
  - Use that field in searches or other expressions

eval Help More »

Calculates an expression and puts the resulting value into a field.

Examples

Set velocity to distance / time.

```
... | eval velocity=distance/time
```

Set lowuser to the lowercase version of username.

```
... | eval lowuser = lower(username)
```

Set full_name to the concatenation of first_name, a space, and last_name.

```
... | eval full_name = first_name." ".last_nameSearch
```

Type	Operators
Arithmetic	+ - * / %
Concatenation	+ .
Boolean	AND OR NOT XOR
Comparison	< > <= >= != = == LIKE

eval Command – Convert Values

- This example report displays the sum of bytes used for each usage category
  - It's hard to determine how much bandwidth is being used by looking at bytes
- First, we'll use `eval` to convert the `bytes` value into megabytes…

Scenario

What types of websites used the most bandwidth in bytes during the previous month?

```
sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as Bytes by usage
```

Events | Patterns | Statistics (5) | Visualization

10 Per Page | Format | Preview

usage	Bytes
Borderline	3032638
Business	2151552
Personal	33467017
Unknown	4825323
Violation	198318

eval Command – Convert Values (cont.)

- Results of `eval` must be set to a new or existing field
- In this example:
  - A Calculate the number of bytes for each usage type
  - B Create a new field named `bandwidth`
  - C Convert the values of the Bytes field into MB by dividing `Bytes` field values by `(1024*1024)`

Scenario

What types of websites used the most bandwidth in megabytes during the previous month?

```
sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as Bytes A by usage
| eval bandwidth B = Bytes/(1024*1024) C
```

Events | Patterns | Statistics (5) | Visualization

10 Per Page | Format | Preview

usage	A Bytes	B bandwidth
Borderline	3032638	C 2.892149
Business	2151552	2.051880
Personal	33467017	31.916635
Unknown	4825323	4.601787
Violation	198318	0.189131

eval Command – Round Values

- However, the results are hard to read with so many decimal points
- round(field/number, decimals) function sets the value of a field to the number of decimals you specify
  - In this example, divide the value of the Bytes field by (1024*1024)
  - Then round to 2 decimal points
- If number of decimals is not specified, then the result is a whole number

Scenario

What types of websites used the most bandwidth in megabytes, rounded to 2 decimal places, during the previous month? Sort by bandwidth.

```
sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as Bytes by usage
| eval bandwidth = round(Bytes/(1024*1024), 2)
| sort -bandwidth
| rename bandwidth as "Bandwidth (MB)"
```

A

Events | Patterns | Statistics (5) | Visualization

10 Per Page | Format | Preview

usage	Bytes	Bandwidth (MB)
Personal	33467017	31.92
Unknown	4825323	4.60
Borderline	3032638	2.89
Business	2151552	2.05
Violation	198318	0.19

Removing Fields

Now that we've calculated and formatted our results in the new "Bandwidth (MB)" field, we can remove the bytes field from the report as a final command

- It is safe to remove fields after their values have been used in previous parts of the search string

Scenario

What types of websites used the most bandwidth in megabytes, rounded to 2 decimal places, during the previous month? Sort by bandwidth.

```
sourcetype=cisco_wsa_squid
| stats sum(sc_bytes) as Bytes by usage
| eval bandwidth = round(Bytes/(1024*1024), 2)
| sort -bandwidth
| rename bandwidth as "Bandwidth (MB)"
| fields - Bytes
```

A

Events | Patterns | Statistics (5) | Visualization

10 Per Page | Format | Preview

usage	Bandwidth (MB)
Personal	31.92
Unknown	4.60
Borderline	2.89
Business	2.05
Violation	0.19

eval Command – Calculating Values

Can perform mathematical functions against fields with numeric field values

- A In this example, stats calculates the total list price and total sale price by product_name
- B eval calculates the discount percentage and formats the discount field
- C sort lists the highest discounted items first
- D rename provides user friendly headings

Scenario

Calculate total online sales for last week, include price, sales price, and discount percentage. Sort by descending discount value.

```
sourcetype=access_combined product_name=* action=purchase
A | stats sum(price) as tp, sum(sale_price) as tsp by product_name
B | eval Discount = round(((tp - tsp)/ tp)*100)
  | sort -Discount
C | eval Discount = Discount."%"
  | rename tp as "Total List Price", tsp as "Total Sale Price",
    product_name as Product
```

Product	Total List Price	Total Sale Price	Discount
Puppies vs. Zombies	698.60	278.60	60%
Fire Resistance Suit of Provolone	602.49	300.49	50%
Holy Blade of Gouda	808.65	403.65	50%
Dream Crusher	7078.23	4423.23	38%
Manganiello Bros.	6798.30	4248.30	38%
Orvil the Wolverine	4638.84	2898.84	38%

stats Command – eval function

To count the number of events that contain a specific field value, use the count and eval functions

- Requires an as clause
- Double quotes are required for character field values
- Field values **are** case-sensitive

Scenario

Count the number of vendor action events during yesterday.

```
sourcetype=linux_secure vendor_action=*
| stats
  count(eval(vendor_action="Accepted")) as Accepted, A
  count(eval(vendor_action="Failed")) as Failed, B
  count(eval(vendor_action="session opened")) as
  SessionOpened C
```

Events | Patterns | Statistics (1) | Visualization

10 Per Page | Format | Preview

A Accepted	B Failed	C SessionOpened
314	6219	351

eval Command – tostring Function

- tostring converts a numeric field value to a string

  tostring(field,"option")

- Options:
  - "commas" - applies commas and, if the number includes decimals, rounds to two decimal places
  - "duration" - formats the number as "hh:mm:ss"
  - "hex" - formats the number in hexadecimal

Scenario

How much potential online sales revenue was lost the previous week, due to 503 server errors?

```
sourcetype=access_combined action=purchase status=503
| stats count(price) as NumberOfLostSales,  A
  avg(price) as AverageLostSales,
  sum(price) as TotalLostRevenue
| eval AverageLostSales =
  "$" + tostring(AverageLostSales, "commas")  B
| eval TotalLostRevenue =
  "$" + tostring(TotalLostRevenue, "commas")  C
```

Events | Patterns | Statistics (1) | Visualization

20 Per Page | Format | Preview

A NumberOfLostSales	AverageLostSales B	TotalLostRevenue C
188	$22.19	$4,172.12

tostring Function – duration Option

This example shows "duration" option of tostring function

- A stats calculates sessionTime and JSESSIONID
- B sort 5 displays the top 5 most frequent values
- C The duration option formats the time as "hh:mm:ss"

Scenario

Identify the five longest client sessions over the last 4 hours in HH:MM:SS format

```
sourcetype=access_combined
| stats range(_time) as sessionTime by JSESSIONID   A
| sort 5 -sessionTime   B
| eval duration = tostring(sessionTime,"duration")   C
```

Events | Patterns | Statistics (5) | Visualization

10 Per Page | Format | Preview

JSESSIONID A	A sessionTime	duration
SD4SL1FF3ADFF4960	5935	01:38:55
SD0SL9FF9ADFF4952	160	00:02:40
SD6SL10FF5ADFF4965	149	00:02:29
SD9SL9FF5ADFF4954	139	00:02:19
SD1SL2FF7ADFF4964	138	00:02:18

B C

Formatting and Sorting Values

- `eval` with added characters converts numeric field values to strings
- To order numerically, first `sort`, then use `eval`

```
sourcetype=access_combined price=*
| stats values(price) as price by product_name
| eval price = "$".price
| sort -price
```

product_name	price
Manganiello Bros. Tee	$9.99
World of Cheese Tee	$9.99
Holy Blade of Gouda	$5.99
Puppies vs. Zombies	$4.99
Dream Crusher	$39.99
Manganiello Bros.	$39.99
Orvil the Wolverine	$39.99
Fire Resistance Suit of Provolone	$3.99
Benign Space Debris	$24.99
Final Sequel	$24.99

Alpha

```
sourcetype=access_combined price=*
| stats values(price) as price by product_name
| sort -price
| eval price = "$".price
```

product_name	price
Dream Crusher	$39.99
Manganiello Bros.	$39.99
Orvil the Wolverine	$39.99
Benign Space Debris	$24.99
Final Sequel	$24.99
Mediocre Kingdoms	$24.99
World of Cheese	$24.99
Curling 2014	$19.99
SIM Cubicle	$19.99
Manganiello Bros. Tee	$9.99

Numeric

Multiple eval Commands

Each subsequent command references the results of previous commands

- A Based on the values of list_price and current_sale_price, calculate the current_discount percentage
- B Calculate the new_discount value by subtracting 5 from current_discount
- C Calculate the new_sale_price by applying the new_discount percentage

Scenario

Calculate a new sale price that is 5% less than the current discount percentage.

```
sourcetype=access_combined price=*
| stats values(price) as list_price, values(sale_price)
  as current_sale_price by product_name
| eval current_discount = round((list_price -  A
  current_sale_price)/list_price*100,2)
| eval new_discount = (current_discount - 5)  B
| eval new_sale_price = list_price - (list_price *
  (new_discount/100))  C
```

product_name	list_price	current_sale_price	current_discount	new_discount	new_sale_price
Benign Space Debris	24.99	19.99	20.01	15.01	21.24
Curling 2014	19.99	16.99	15.01	10.01	17.99
Dream Crusher	39.99	24.99	37.51	32.51	26.99
Final Sequel	24.99	16.99	32.01	27.01	18.24
Fire Resistance Suit of Provolone	3.99	1.99	50.13	45.13	2.19

`eval` Command – `if` Function Syntax

`if(X,Y,Z)`

- The `if` function takes three arguments
- The first argument, `X`, is a Boolean expression
  - If it evaluates to TRUE, the result evaluates to the second argument, `Y`
  - If it evaluates to FALSE, the result evaluates to the third argument, `Z`
- Non-numeric values must be enclosed in `"double quotes"`
- Field values are treated in a case-sensitive manner

eval Command – if Function

- Create a new field, SalesTerritory
- Evaluate VendorID
  - If < 4000 is TRUE, set result to "North America"
    - Remember, arguments must be enclosed in quotes
  - If it evaluates to FALSE, set result to "Rest of the World"

Scenario

Display retail sales for the previous week, broken down by North America and the Rest of the World.

```
sourcetype=vendor_sales
| eval SalesTerritory =
  if(VendorID < 4000,"North America", "Rest of the World")
          X                  Y                  Z
| stats sum(price) as TotalRevenue by SalesTerritory
| eval TotalRevenue = "$" + tostring(TotalRevenue, "commas")
```

Events | Patterns | Statistics (2) | Visualization

10 Per Page | Format | Preview

SalesTerritory	TotalRevenue
North America	$10,263.06
Rest of the World	$4,950.60

Filtering Results

The `search` and `where` commands can be used at any point in the search pipeline to filter results

- `search` command
  - May be easier because you are familiar with basic search syntax
  - Treats field values in a case-insensitive manner
  - Can use the * (asterisk) as wildcard
  - Allows searching on keyword
- `where` command
  - Can compare values from two different fields
  - Can do a wildcard search on multiple characters (%) or simply on one character (_); must use the like operator with wildcards
  - Functions are available, example `isnotnull()`
  - Field values are case-sensitive

search Command

- To filter results, use search at any point in the search pipeline
- Behaves exactly like search strings before the first pipe
  - search uses the "*" wildcard and treats field values in a case-insensitive manner

Scenario

Report which products during the last 24 hours have sold more than $500 on-line.

```
sourcetype=access_combined
action=purchase status=200
| stats sum(price) as sales by product_name
| search sales>500
| sort -sales
| eval sales="$"+sales
| rename sales as "Popular Products",
  product_name as "Product Name"
```

A

product_name	sales
Manganiello Bros.	719.82
Dream Crusher	679.83
Mediocre Kingdoms	649.74
Orvil the Wolverine	439.89
Benign Space Debris	399.84
World of Cheese	399.84
SIM Cubicle	399.80

Product Name	Popular Products
Manganiello Bros.	$719.82
Dream Crusher	$679.83
Mediocre Kingdoms	$649.74

A

eval and where Command

<eval-expression>

- Both commands use the same expression syntax
- Uses Booleans to filter search results and only keeps results that are True
- Quoted strings are interpreted as field values
  - Unquoted or single-quoted strings are treated as fields
  - Treats field values in a case-sensitive manner

Note

To view all of where's functions, please see: docs.splunk.com/Documentation/Splunk/latest/SearchReference/Where

where Command - Example

- where <eval-expression>
- Compare results of calculated fields
- Keep rows where the number of removal actions exceeds the number change quantity actions

Scenario

Report which days over the previous week have seen more remove actions than change quantity actions.

```
sourcetype=access_combined
| timechart count(eval(action="changequantity"))
  as changes, count(eval(action="remove")) as removals
| where removals > changes
```

A

_time	changes	removals
2014-10-24	121	127
2014-10-28	88	135
2014-10-29	121	130

A

`fillnull` Command

Use `fillnull` to replace null values in the specified fields

- Default is 0
- `value=string` is the string you want to display

fillnull Help More »

Replaces null values with a specified value.

Examples

For the current search results, fill all empty fields with NULL.

```
... | fillnull value=NULL
```

For the current search results, fill all empty fields with zero.

```
... | fillnull
```

Build a time series chart of web events by host and fill all empty fields with NULL.

```
sourcetype="web" | timechart count by host | fillnull value=NULL
```


fillnull Command – Examples

Scenario

Evaluate vendor sales by country for the last hour.

```
sourcetype= vendor_sales
| chart sum(price) over product_name by VendorCountry
| fillnull
```

product_name	Hungary	Norway	United States
Curling 2014	0	0	59.97
Dream Crusher	0	0	199.95
Fire Resistance Suit of Provolone	0	0	3.99
Holy Blade of Gouda	0	0	5.99
Manganiello Bros.	0	0	39.99
Manganiello Bros. Tee	9.99	0	9.99
Mediocre Kingdoms	0	0	24.99

```
sourcetype= vendor_sales
| chart sum(price) over product_name by VendorCountry
| fillnull value="No Value"
```

Events | Patterns | Statistics (4) | Visualization

10 Per Page | Format | Preview

product_name	India	United States
Benign Space Debris	No Value	24.99
Fire Resistance Suit of Provolone	3.99	No Value
Holy Blade of Gouda	No Value	5.99
SIM Cubicle	No Value	19.99

Module 7: Correlating Events

Module Objectives

- Identify transactions
- Group events using fields
- Group events using fields and time
- Search with transactions
- Report on transactions
- Determine when to use `transaction` vs. `stats`

What is a Transaction?

- A transaction is any group of related events that span time
- Events can come from multiple applications or hosts
  - Events related to a single purchase from an online store can span across an application server, database, and e-commerce engine
  - One email message can create multiple events as it travels through various queues
  - Each event in the network traffic logs represents a single user generating a single http request
  - Visiting a single website normally generates multiple `http` requests
    - HTML, JavaScript, CSS files
    - Flash, Images, etc.

transaction Command

- <field-list>
  - One field or a list of field names
  - The events are grouped into transactions based on the values of this field list
    ... | transaction field1, field2
- Common constraints:
  | <maxspan> | <maxpause> | <startswith> | <endswith>

transaction Help « Less

Groups events into transactions.

Syntax More »

transaction [*field-list*] name= [*string*] [(maxspan=*int* [s|m|h|d])|*maxpause-opt*|*maxevents-opt*|*field-list*|*start-opt*|*end-opt*|*connected-opt*|*unify-ends-opt*|*keeporphans-opt*]* [*memcontrol-opt*]* [*rendering-opt*]*

Examples

Group search results that have the same "host" and "cookie", occur within 30 seconds of each other, and do not have a pause greater than 5 seconds between each event into a transaction.

... | transaction host cookie maxspan=30s maxpause=5s

transaction Command – Example

- Here you can see a number of events that share the same JSESSIONID (SD4SL9FF9ADFF4961)
- However, it is difficult to view as a group or to gain insight to what is happening or know if there are others scattered in the results set

Scenario

Display customer transactions in the online store during the last 60 minutes.

```
sourcetype=access_combined
```

i	Time	Event
>	9/15/15 2:39:25.000 PM	128.241.220.82 - - [15/Sep/2015:14:39:25] "GET /cart.do?action=view&itemId=EST-12&productId=WC-SH-T02&JSESSIONID=SD3SL6FF8ADFF4954 HTTP 1.1" 200 3686 "http://www.buttercupgames.com/cart.do?action=view&itemId=EST-12&productId=WC-SH-T02" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 171 JSESSIONID = SD3SL6FF8ADFF4954 host = www1 source = /opt/log/www1/access.log sourcetype = access_combined
>	9/15/15 2:39:08.000 PM	128.241.220.82 - - [15/Sep/2015:14:39:08] "GET /category.screen?categoryId=STRATEGY&JSESSIONID=SD3SL6FF8ADFF4954 HTTP 1.1" 200 2139 "http://www.buttercupgames.com/oldlink?itemId=EST-21" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 121 JSESSIONID = SD3SL6FF8ADFF4954 host = www1 source = /opt/log/www1/access.log sourcetype = access_combined
>	9/15/15 2:39:01.000 PM	128.241.220.82 - - [15/Sep/2015:14:39:01] "GET /oldlink?itemId=EST-13&JSESSIONID=SD3SL6FF8ADFF4954 HTTP 1.1" 406 2722 "http://www.buttercupgames.com/cart.do?action=view&itemId=EST-13" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 483 JSESSIONID = SD3SL6FF8ADFF4954 host = www1 source = /opt/log/www1/access.log sourcetype = access_combined

transaction Command – Example 1

- Use the transaction command to create a single event from a group of events that share the same value in a given field
- Transactions can cross multiple tiers (i.e., web server, application server) using a common field(s), in this example, JSESSIONID
- You can easily view the events for JSESSIONID SD4SL9FF9ADFF4961

Scenario

Display network login activity by IP during the last 60 minutes.

```
sourcetype=access_combined
| transaction JSESSIONID
```

Time	Event
9/15/15 2:50:24.000 PM	95.130.170.231 - - [15/Sep/2015:14:50:24] "GET /product.screen?productId=MB-AG-G07&JSESSIONID=SD2SL10FF4ADFF4961 HTTP 1.1" 200 1040 "http://www.buttercupgames.com" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 859 95.130.170.231 - - [15/Sep/2015:14:50:40] "GET /category.screen?categoryId=NULL&JSESSIONID=SD2SL10FF4ADFF4961 HTTP 1.1" 408 1281 "http://www.buttercupgames.com/cart.do?action=remove&itemId=EST-16" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 950 95.130.170.231 - - [15/Sep/2015:14:50:48] "GET /cart.do?action=view&itemId=EST-17&productId=FI-AG-G08&JSESSIONID=SD2SL10FF4ADFF4961 HTTP 1.1" 200 3683 "http://www.buttercupgames.com/category.screen?categoryId=ARCADE" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 379 95.130.170.231 - - [15/Sep/2015:14:51:06] "GET /product.screen?productId=WC-SH-G04&JSESSIONID=SD2SL10FF4ADFF4961 HTTP 1.1" 200 2211 "http://www.buttercupgames.com/category.screen?categoryId=SHOOTER" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 664 95.130.170.231 - - [15/Sep/2015:14:51:10] "POST /cart.do?action=addtocart&itemId=EST-27&productId=WC-SH-G04&JSESSIONID=SD2SL10FF4ADFF4961 HTTP 1.1" 200 1815 "http://www.buttercupgames.com/product.screen?productId=WC-SH-G04" "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/536.5 (KHTML, like Gecko) Chrome/19.0.1084.46 Safari/536.5" 738 Show all 7 lines JSESSIONID = SD2SL10FF4ADFF4961 ¦ host = www1 ¦ source = /opt/log/www1/access.log ¦ sourcetype = access_combined

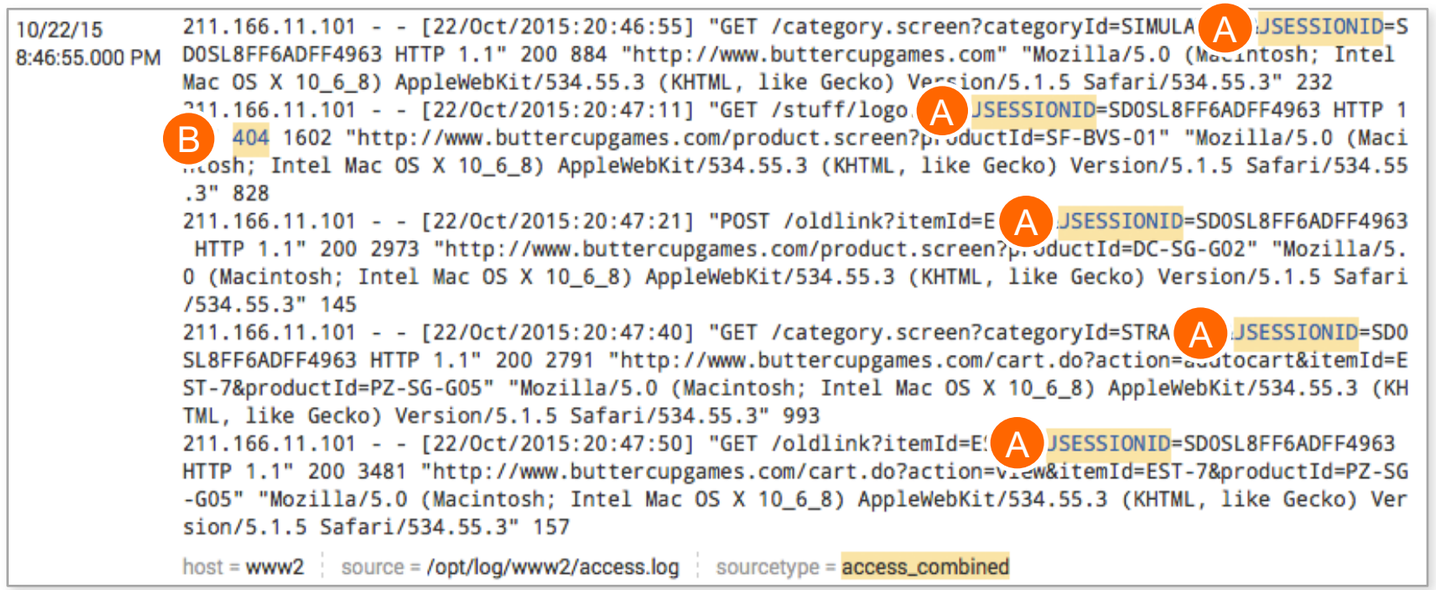
transaction Command – Example 2

- Use the search command at any point in the search pipeline to filter results
- Behaves exactly like search strings before the first pipe
  - search uses the "*" wildcard and treats field values in a case-insensitive manner
  - status=404 finds the errors
  - highlight highlights the terms you specify

Scenario

Display transactions that included a 404 error during the last 60 minutes.

```
sourcetype=access_combined
| transaction JSESSIONID A
| search status=404
| highlight JSESSIONID, 404 B
```



transaction Command – Example 3

Scenario

For failed network logins, display different users from the same IP during the last 15 minutes.

```
sourcetype=linux_secure failed
| transaction src_ip
```

i	Time	Event
>	8/3/15 10:22:07.000 PM	Mon Aug 03 2015 22:22:07 www3 sshd[2155]: Failed password for invalid user irc from 27.35.11.11 port 2723 ssh2 Mon Aug 03 2015 22:22:16 www3 sshd[4811]: Failed password for invalid user mailman from 27.35.11.11 port 4989 ssh2 Mon Aug 03 2015 22:22:22 www3 sshd[1938]: Failed password for invalid user administrator from 27.35.11.11 port 3547 ssh2 host = www3 ¦ source = /opt/log/www3/secure.log ¦ sourcetype = linux_secure
>	8/3/15 10:20:59.000 PM	Mon Aug 03 2015 22:20:59 www3 sshd[4919]: Failed password for invalid user robbie from 141.146.8.66 port 3244 ssh2 Mon Aug 03 2015 22:21:09 www3 sshd[1283]: Failed password for invalid user harrison from 141.146.8.66 port 3634 ssh2 Mon Aug 03 2015 22:21:19 www3 sshd[5390]: Failed password for madonna from 141.146.8.66 port 4475 ssh2 Mon Aug 03 2015 22:21:39 www3 sshd[5086]: Failed password for games from 141.146.8.66 port 1938 ssh2 Mon Aug 03 2015 22:21:54 www3 sshd[5333]: Failed password for invalid user system from 141.146.8.66 port 1613 ssh2 host = www3 ¦ source = /opt/log/www3/secure.log ¦ sourcetype = linux_secure
>	8/3/15 10:18:40.000 PM	... 1 line omitted ... Mon Aug 03 2015 22:18:52 www2 sshd[1313]: Failed password for invalid user local from 67.170.226.218 port 4948 ssh2 Mon Aug 03 2015 22:19:05 www2 sshd[4595]: Failed password for invalid user jabber from 67.170.226.218 port 4641 ssh2 Mon Aug 03 2015 22:19:34 www2 sshd[1850]: Failed password for invalid user agushto from 67.170.226.218 port 3742 ssh2 Mon Aug 03 2015 22:19:43 www2 sshd[5282]: Failed password for invalid user ubuntu from 67.170.226.218 port 4670 ssh2 Mon Aug 03 2015 22:19:54 www2 sshd[4728]: Failed password for beyonce from 67.170.226.218 port 2172 ssh2 Show all 7 lines host = www2 ¦ source = /opt/log/www2/secure.log ¦ sourcetype = linux_secure

transaction Command – Specific Fields

The transaction command produces additional fields, such as:

- duration – the difference between the timestamps for the first and last event in the transaction
- eventcount – the number of events in the transaction

transaction Command – maxspan/maxpause

You can also define a max overall time span and max gap between events

- maxspan=10m
  - Maximum total time between the *earliest* and *latest* events
  - If not specified, default is -1 (or no limit)
- maxpause=1m
  - Maximum total time *between* events
  - If not specified, default is -1 (or no limit)

Scenario

Display customer actions on the website during the last 4 hours.

```
sourcetype=access_combined
| transaction clientip maxspan=10m maxpause=1m
| eval duration = tostring(duration,"duration")
| sort -duration
| table clientip duration action
| rename  clientip as "Client IP",
  action as "Client Actions"
```

Client IP	duration	Client Actions
198.35.2.120	00:02:19	addtocart view
175.44.1.172	00:02:15	addtocart changequantity purchase view
217.15.20.146	00:02:15	addtocart changequantity purchase remove

Note

Assumptions: Transactions spanning more than 10 minutes with the same client IP are considered unrelated, nor can there be more than one 1 minute between any two related events.

transaction Command – startswith/endswith

- To form transactions based on terms, field values, or evaluations, use `startswith` and `endswith` options
- In this example, the first event in the transaction includes `addtocart` and the last event includes `purchase`

Scenario

Determine the length of time spent by customers in the online store to purchase.

```
sourcetype=access_combined
| transaction clientip JSESSIONID
  startswith=eval(action="addtocart")
  endswith=eval(action="purchase")
| table clientip, JSESSIONID, duration, eventcount
```

clientip	JSESSIONID	duration	eventcount
192.162.19.179	SD7SL7FF9ADFF4957	1	2
203.223.0.20	SD3SL2FF8ADFF4963	3	2
65.19.167.94	SD7SL3FF8ADFF4959	1	2
173.44.37.226	SD2SL7FF5ADFF4953	3	2
74.82.57.172	SD8SL6FF5ADFF4960	1	2
110.138.30.229	SD7SL1FF4ADFF4951	1	2

Investigating with Transactions

- Transactions can be useful when a single event does not provide enough information
- This example searches email logs for the term “REJECT”
- Events that include the term don’t provide much information about the rejection

Scenario

Find emails that were rejected during the last 24 hours.

```
sourcetype=cisco_esa REJECT
```

i	Time	Event
>	8/3/15 10:26:55.000 PM	Mon Aug 03 22:26:55 2015 Info: ICID 744203 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 6.2 host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa
>	8/3/15 10:21:44.000 PM	Mon Aug 03 22:21:44 2015 Info: ICID 744202 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 10.0 host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa
>	8/3/15 10:09:56.000 PM	Mon Aug 03 22:09:56 2015 Info: ICID 744201 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 4.3 host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa
>	8/3/15 9:54:01.000 PM	Mon Aug 03 21:54:01 2015 Info: ICID 744200 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 4.3 host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa
>	8/3/15 9:04:27.000 PM	Mon Aug 03 21:04:27 2015 Info: ICID 744203 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 6.2 host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa

Investigating with Transactions (cont.)

- By creating a transaction, we can then search and see additional events related to the rejection, such as:
  - IP address of sender
  - Reverse DNS lookup results
  - Action taken by the mail system following the rejection
- `mid` – Message ID
- `dcid` – Delivery Connection ID
- `icid` – Incoming Connection ID

Scenario

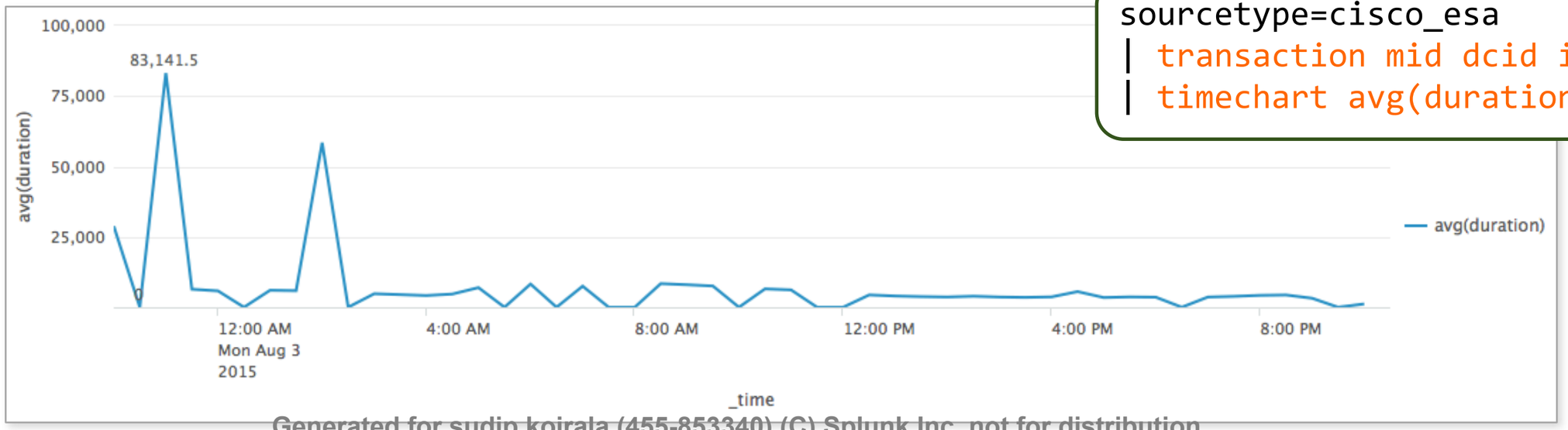
Find emails that were rejected.

```
sourcetype=cisco_esa
| transaction mid dcid icid
| search REJECT
```

i	Time	Event
>	8/2/15 11:20:13.000 PM	Mon Aug 03 21:04:20 2015 Info: New SMTP ICID 744203 interface Management (192.168.3.120) address 190.148.95.173 reverse dns host 173.95.148.190.dsl.intelnet.net.gt verified yes Mon Aug 03 21:04:27 2015 Info: ICID 744203 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 6.2 Mon Aug 03 21:04:36 2015 Info: ICID 744203 close Mon Aug 03 22:26:41 2015 Info: New SMTP ICID 744203 interface Management (192.168.3.120) address 190.148.95.173 reverse dns host 173.95.148.190.dsl.intelnet.net.gt verified yes Mon Aug 03 22:26:55 2015 Info: ICID 744203 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 6.2 Show all 6 lines host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa
>	8/2/15 11:03:39.000 PM	Mon Aug 03 21:00:55 2015 Info: New SMTP ICID 744202 interface Management (192.168.3.120) address 84.61.83.14 reverse dns host dslb 084 061 083 014.pools.arcor ip.net verified yes Mon Aug 03 21:01:01 2015 Info: ICID 744202 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 10.0 Mon Aug 03 21:01:06 2015 Info: ICID 744202 close Mon Aug 03 22:21:31 2015 Info: New SMTP ICID 744202 interface Management (192.168.3.120) address 84.61.83.14 reverse dns host dslb 084 061 083 014.pools.arcor ip.net verified yes Mon Aug 03 22:21:44 2015 Info: ICID 744202 REJECT SG BLACKLIST match sbrs[10.0: 3.0] SBRS 10.0 Show all 6 lines host = cisco_router1 source = /opt/log/cisco_router1/cisco_ironport_mail.log sourcetype = cisco_esa

Reporting on Transactions

- You can use statistics and reporting commands with transactions
- This example takes advantage of the `duration` field
  - It shows a trend of the mail queue slowing, then correcting, then slowing again
  - Adding events to the transaction from additional hosts or sources can uncover the cause of the slowdown




```
sourcetype=cisco_esa
| transaction mid dcid icid
| timechart avg(duration)
```

transaction vs. stats

- Use `transaction` when you
  - Need to see events correlated together
  - Must define event grouping based on `start/end` values or chunk on time
  - Have less than 1,000 events for each correlated transaction
    - By default, `transaction` displays a maximimum event count of 1,000
    - admins can configure `max_events_per_bucket` in `limits.conf`
- Use `stats` when you:
  - Want to see the results of a calculation
  - Can group events based on a field value (e.g. "`by src_ip`")
  - Have more than 1,000 events for each grouped set of events
- When you have a choice, always use `stats` as it is faster and more efficient, especially in large Splunk environments

transaction vs. stats: Example

```
sourcetype=linux_secure
| transaction src_ip
| table src_ip, eventcount
| sort - eventcount
```

Note

1. `transaction` has a limit of 1,000
2. Count of transactions vs count of IPs

```
sourcetype=linux_secure
| stats count as eventcount
  by src_ip
| sort - eventcount
```

src_ip	eventcount
3.0.0.44	1000
3.0.0.44	1000
3.0.0.44	1000
175.45.176.223	1000
2.144.0.210	784
3.0.0.44	608
41.32.0.85	473
23.16.0.181	316
175.45.176.98	233
175.45.176.223	121

src_ip	eventcount
3.0.0.44	3608
175.45.176.223	1121
2.144.0.210	784
41.32.0.85 .23	473
23.16.0.181	316
175.45.176.98	233
41.0.0.142	57
2.144.0.22	44
1.0.32.67	30
41.32.0.27	29

Thank You