splunk>

Search Under the Hood – Lab Guide

Overview

Welcome to the Splunk Education lab environment. These lab exercises will test your knowledge of the Job Inspector, your ability to optimize searches, the **`makeresults`** command, the **`fieldsummary`** command and informational functions.

Scenario

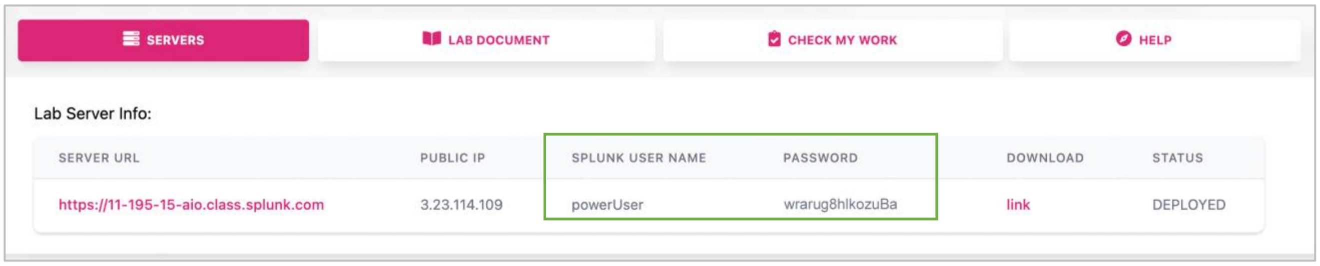
You will use data from the international video game company, Buttercup Games. A list of source types is provided below.

NOTE: This is a lab environment driven by data generators with obvious limitations. This is not a production environment. Screenshots approximate what you should see, not the **exact** output.

Index	Type	Sourcetype	Interesting Fields
`web`	Online sales	**`access_combined`**	`action`, `bytes`, `categoryId`, `clientip`, `itemId`, `JSESSIONID`, `price`, `productId`, `product_name`, `referer`, `referer_domain`, `sale_price`, `status`, `user`, `useragent`
`security`	Active Directory	**`winauthentication_security`**	`LogName`, `SourceName`, `EventCode`, `EventType`, `User`
	Web server	**`linux_secure`**	`action`, `app`, `dest`, `process`, `src_ip`, `src_port`, `user`, `vendor_action`
`sales`	Business Intelligence server	**`sales_entries`**	`AcctCode`, `CustomerID`, `TransactionID`
`network`	Web security appliance data	**`cisco_wsa_squid`**	`action`, `cs_method`, `cs_mime_type`, `cs_url`, `cs_username`, `sc_bytes`, `sc_http_status`, `sc_result_code`, `severity`, `src_ip`, `status`, `url`, `usage`, `x_mcafee_virus_name`, `x_wbrs_score`, `x_webcat_code_abbr`
	Firewall data	**`cisco_firewall`**	`bcg_ip`, `dept`, `Duration`, `fname`, `IP`, `lname`, `location`, `rfid`, `splunk_role`, `splunk_server`, `Username`
`games`	Game logs	**`SimCubeBeta`**	`date_hour`, `date_mday`, `date_minute`, `date_month`, `date_second`, `data_wday`, `data_year`, `date_zone`, `eventtype`, `index`, `linecount`, `punct`, `splunk_server`, `timeendpos`, `timestartpos`

Lab Connection Info

Access labs using the server URL, user name, and password shown in your lab environment.



Lab Server Info:

SERVER URL	PUBLIC IP	SPLUNK USER NAME	PASSWORD	DOWNLOAD	STATUS
https://11-195-15-aio.class.splunk.com	3.23.114.109	powerUser	wrarug8hlkozuBa	link	DEPLOYED

splunk>

Common Commands and Functions

These commands and statistical functions are commonly used in searches but may not have been explicitly discussed in the module. Please use this table for quick reference. Click on the hyperlinked SPL to be taken to the Search Manual for that command or function.

SPL	Type	Description	Example
`sort`	command	Sorts results in descending or ascending order by a specified field. Can limit results to a specific number.	*Sort the first 100 `src_ip` values in descending order* `| sort 100 -src_ip`
`where`	command	Filters search results using eval-expressions.	*Return events with a `count` value greater than 30* `| where count > 30`
`rename`	command	Renames one or more fields.	*Rename `SESSIONID` to 'The session ID'* `| rename SESSIONID as "The session ID"`
`fields`	command	Keeps (+) or removes (-) fields from search results.	*Remove the `host` field from the results* `| fields - host`
`stats`	command	Calculates aggregate statistics over the results set.	*Calculate the total sales, i.e. the sum of `price` values* `| stats sum(price)`
`eval`	command	Calculates an expression and puts the resulting value into a new or existing field.	*Concatenate `first_name` and `last_name` values with a space to create a field called "`full_name`"* `| eval full_name=first_name." ".last_name`
`table`	command	Returns a table.	*Output `vendorCountry`, `vendor`, and `sales` values to a table* `| table vendorCountry, vendor, sales`
`sum()`	statistical function	Returns the sum of the values of a field. Can be used with **`stats`**, **`timechart`**, and **`chart`** commands.	*Calculate the sum of the `bytes` field* `| stats sum(bytes)`

`count` or `count()`	statistical function	Returns the number of occurrences of all events or a specific field. Can be used with **`stats`**, **`timechart`**, and **`chart`** commands.	*Count all events as "events" and count all events that contain a value for `action` as "action"*
			`| stats count as events,`
			`count(action) as action`

Refer to the Search Reference Manual for a full list of commands and functions.

Lab Exercises

Description

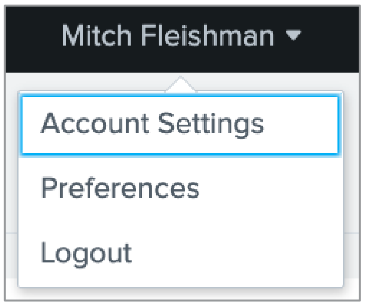
Configure the lab environment user account. Then, evaluate **`lispy`** expressions and optimize searches, use the **`makeresults`** command to test a regex expression, generate summary statistics with the **`fieldsummary`** command, and use informational functions to gain insights about search results.

Steps

Task 1: Log into Splunk and change the account name and time zone.

Set up your lab environment to fit your time zone. This also allows the instructor to track your progress and assist you if necessary.

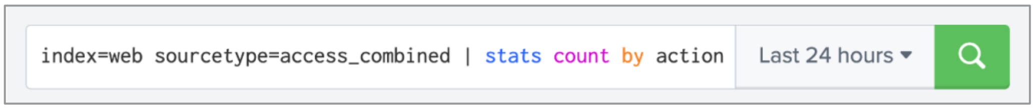
1. Log into your Splunk lab environment using the username and password provided to you.
2. You may see a pop-up window welcoming you to the lab environment. You can click **Continue to Tour** but this is not required. Click **Skip** to dismiss the window.
3. Click on the username you logged in with (at the top of the screen) and then choose **Account Settings** from the drop-down menu.
4. In the **Full name** box, enter your first and last name.
5. Click **Save**.
6. Reload your browser to reflect the recent changes to the interface. (This area of the web interface will be referred to as ***user name***.)



After you complete step 6, you will see your name in the web interface.

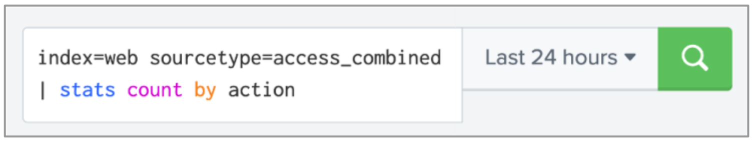
NOTE: Sometimes there can be delays in executing an action like saving in the UI or returning results of a search. If you are experiencing a delay, please allow the UI a few minutes to execute your action.

7. Navigate to ***user name*** **> Preferences.**
8. Choose your local time zone from the **Time zone** drop-down menu.
9. Click **Apply**.
10. (Optional) Navigate to ***user name*** **> Preferences > SPL Editor > Search auto-format** and click on the toggle to activate auto-formatting. Then click **Apply**. When the pipe character is used in search, the SPL Editor will automatically begin the pipe on a new line.



```
index=web sourcetype=access_combined | stats count by action
```

Search auto-format disabled (default)



```
index=web sourcetype=access_combined
| stats count by action
```

Search auto-format enabled

Scenario: Search the web server data where the source port is 1062.

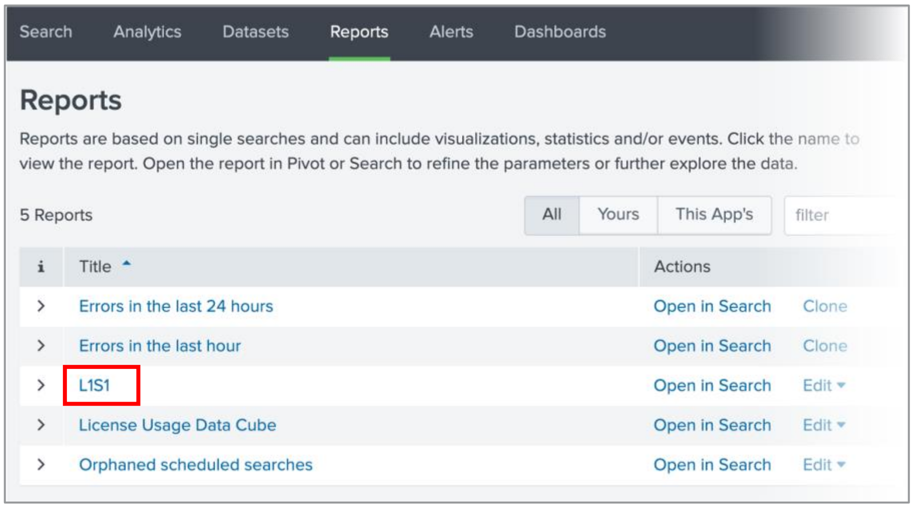
Task 2: Find the `lispy` expression for a search.

11. In the top left corner of Splunk Web, select **Apps** > **Search & Reporting**. This sets our app context to the search app.
12. Execute this search over the **Previous month.**

```
index=security sourcetype=linux_secure host=www1 src_port=1062
```

NOTE: The data generators for the **`linux_secure`** sourcetype are random. Therefore, if your search does not return results for **`src_port=1062`** then you should choose a different value. Run the search without **`src_port=1062`**. Then, find **`src_port`** in the **Interesting Fields** list and click on a port value. The search will re-execute with the new port value. You will still receive full credit for saving the L1S1 report even if you have a different value for **`src_port`**.

13. Click **Job > Inspect Job** to open the job inspector. The **Search job inspector** launches in a new window.
14. On the **Search job inspector** page, note the number of events returned vs. the number of events scanned. You should notice that more events are scanned than returned. Next, you will view the **`lispy`** used to retrieve these events.
15. In the **Search job inspector**, click the **`search.log`** link.
16. Using your browser's Find feature, find the **`lispy`** expression.
17. Close the **`search.log`** window.
18. Save your search as a report with the name **L1S1**.
    a. Click **Save As** > **Report**
    b. For **Title**, enter L1S1.
    c. **Save**.
    d. You can **View** your report or exit out of the **Your Report Has Been Created** window by clicking the **X** in the upper-right corner.
    e. You can access your saved reports using the **Reports** tab in the application bar.



Reports

Reports are based on single searches and can include visualizations, statistics and/or events. Click the name to view the report. Open the report in Pivot or Search to refine the parameters or further explore the data.

5 Reports

i	Title ▲	Actions	
>	Errors in the last 24 hours	Open in Search	Clone
>	Errors in the last hour	Open in Search	Clone
>	L1S1	Open in Search	Edit ▾
>	License Usage Data Cube	Open in Search	Edit ▾
>	Orphaned scheduled searches	Open in Search	Edit ▾

*Your recently saved **L1S1** report will be visible in the **Reports** tab.*

Scenario: SecOps suspects that some unsafe websites have been accessed by employees. Search for Cisco Web Security log events that occurred over the previous 30 days with a reputation score (`x_wbrs_score`) equal to -6.4.

Task 3: Use the `TERM` directive to optimize your search so that the number of events scanned is equal to the number of events returned.

19. From the menu, click **Search**. (The search box re-initializes and should now be empty.)
20. Execute this search over the **Last 30 days**:

```
index=network sourcetype=cisco_wsa_squid x_wbrs_score=-6.4
```

NOTE: The web-based reputation score (WBRS) assigned to URLs determines the likelihood that the webpage contains URL-based malware. The Cisco Web Security appliance uses this information to stop malware attacks before they occur. Scores range from 10.0 to -10.0 and anything under 6.0 is scanned or blocked.

21. Click **Job > Inspect Job** to open the **Search job inspector**. The job inspector launches in a new window.
22. On the **Search job inspector** main page, note the number of events returned vs. the number of events scanned. You should notice that more events are scanned than returned. Next, you will view the **`lispy`** used to retrieve these events.
23. In the **Job Inspector**, click the **`search.log`** link.
24. Using your browser's Find feature, find the **`lispy`** expression. The value for **`lispy`** should look like this:

```
[ AND 4 6 index::network sourcetype::cisco_wsa_squid ]
```

    Notice that the value on which you were searching, **`-6.4`** is broken into separate terms for the search and thus returns all events that contain a **`6`** or a **`4`**.

25. Close the **`search.log`** window.
26. The training videos demonstrated a technique to prevent the **`lispy`** from breaking up search terms. Rewrite and run the search using this technique so that Splunk generates a lispy expression that searches for the number **`-6.4`**, not two separate numbers **`6`** and **`4`**.
27. Click **Job > Inspect Job**. Note the improvement in the number of events scanned. Is there a difference between the number of events scanned and the number of events returned?
28. Click the **`search.log`** link.
29. Use the browser's Find feature to find the **`lispy`** expression. The value for **`lispy`** should look like this:

```
[ AND -6.4 index::network sourcetype::cisco_wsa_squid ]
```

    Notice that this expression is more specific than it was before you edited and re-executed the search.

30. Close the **`search.log`** window.
31. Save your search as a report with the name **L1S2**.

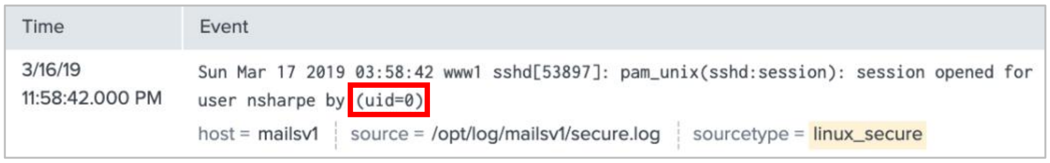
Scenario: Search for OS login data on Buttercup Games *nix servers (`linux_secure`) that occurred during the previous week with a user id (`uid`) equal to 0.

Task 4: Use the `TERM` directive to optimize your search so that the number of events scanned is equal to the number of events returned.

32. From the menu, click **Search**. (The search box re-initializes and should now be empty.)
33. Execute this search over the **Previous week**:

    ```
    index=security sourcetype=linux_secure uid=0
    ```

    You should see events that include raw data that looks similar to what is shown below. Notice that the string **`uid=0`** can be found in the event data.



Time	Event
3/16/19 11:58:42.000 PM	Sun Mar 17 2019 03:58:42 www1 sshd[53897]: pam_unix(sshd:session): session opened for user nsharpe by (uid=0) host = mailsv1 source = /opt/log/mailsv1/secure.log sourcetype = linux_secure

34. Click **Job > Inspect Job** to open the **Job Inspector**. What do you notice about the number of events scanned versus the number of events returned by the search?
35. In the **Search job inspector**, click the **`search.log`** link.
36. Use your browser's Find feature to find the **`lispy`** expression. The value for **`lispy`** should look like this:

    ```
    [ AND 0 index::security sourcetype::linux_secure ]
    ```

37. Close the **`search.log`** window.
38. Rewrite and run a search to produce a **`lispy`** that causes fewer events to be scanned from disk.
39. Execute the new search.
40. Click **Job > Inspect Job**. How has the relationship changed between the number of events scanned and number of events returned?
41. Click the **`search.log`** link.
42. Use the browser's Find feature to find **`lispy`**. The value for **`lispy`** should look like this:

    ```
    [ AND index::security sourcetype::linux_secure uid=0 ]
    ```

    Notice that this expression is more specific than it was before you edited and re-executed the search.

43. Close the **`search.log`** window.
44. Save your search as a report with the name **L1S3**.

Scenario: Search for Cisco Firewall log events that occurred during the previous month, where the `bcg_ip` field is equal to 10.1.10.107.

Task 5: Optimize a search that uses a field from a lookup.

45. From the menu, click **Search**. (The search box re-initializes and should now be empty.)

46. Execute this search over the **Previous month**:

```
index=network sourcetype=cisco_firewall bcg_ip=TERM(10.1.10.107)
```

NOTE: Please ignore any warnings you see during this step. Expand the time range to **All time** if your search does not return results.

47. Click **Job** > **Inspect Job** to open the **Search job inspector**.
48. In the **Search job inspector**, click the `search.log` link.
49. Use your browser's Find feature to find `lispy`. The value for `lispy` should look like what is shown below.

```
[ AND index::network sourcetype::cisco_firewall ]
```

NOTE: `10.1.10.107` does not appear in the `lispy` and therefore is not used to determine which events to read from disk.

This value does not appear because the `bcg_ip` field is obtained using a lookup; it is not populated directly from the raw event. In this situation, you can examine the data to see what else can be used to limit the events returned from disk.

50. Rewrite and run a search to produce a `lispy` that retrieves fewer events from disk and filters results from a field other than `bcg_ip`. (Hint: `bcg_ip` is an alias.)

51. Click **Job** > **Inspect Job** to open the **Search job inspector**.
52. In the **Search job inspector**, click the `search.log` link.
53. Use your browser's Find feature to find `lispy`. The value for `lispy` should look like what is shown below.

```
[ AND 10.1.10.107 index::network sourcetype::cisco_firewall ]
```

54. Why does this `lispy` contain `10.1.10.107`?

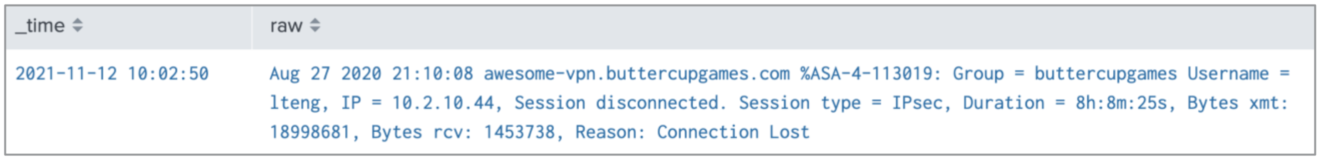
55. Save your search as a report with the name **L1S4**.

Scenario: Create firewall data to test a regex expression with the `rex` command.

Task 6: Use the `makeresults` command to create data for testing.

56. From the menu, click **Search**. (The search box re-initializes and should now be empty.)
57. Write a search that will use the following `eval` command to create firewall data and run this search over the **Last 24 hours**.

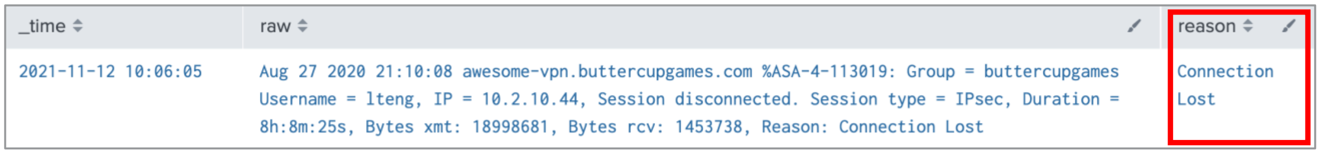
```
| eval raw = "Aug 27 2020 21:10:08 awesome-vpn.buttercupgames.com %ASA-4-113019:
Group = buttercupgames Username = lteng, IP = 10.2.10.44, Session disconnected.
Session type = IPsec, Duration = 8h:8m:25s, Bytes xmt: 18998681, Bytes rcv: 1453738,
Reason: Connection Lost"
```



_time	raw
2021-11-12 10:02:50	Aug 27 2020 21:10:08 awesome-vpn.buttercupgames.com %ASA-4-113019: Group = buttercupgames Username = lteng, IP = 10.2.10.44, Session disconnected. Session type = IPsec, Duration = 8h:8m:25s, Bytes xmt: 18998681, Bytes rcv: 1453738, Reason: Connection Lost

58. Now that you have data to work with, you can test the following **`rex`** command. This command should pull reason description information from the event and populate a column called **`reason`**.

```
| rex field=raw "^(?:[^:\n]*:){8}\s+(?P<reason>.+)"
```



_time	raw	reason
2021-11-12 10:06:05	Aug 27 2020 21:10:08 awesome-vpn.buttercupgames.com %ASA-4-113019: Group = buttercupgames Username = lteng, IP = 10.2.10.44, Session disconnected. Session type = IPsec, Duration = 8h:8m:25s, Bytes xmt: 18998681, Bytes rcv: 1453738, Reason: Connection Lost	Connection Lost

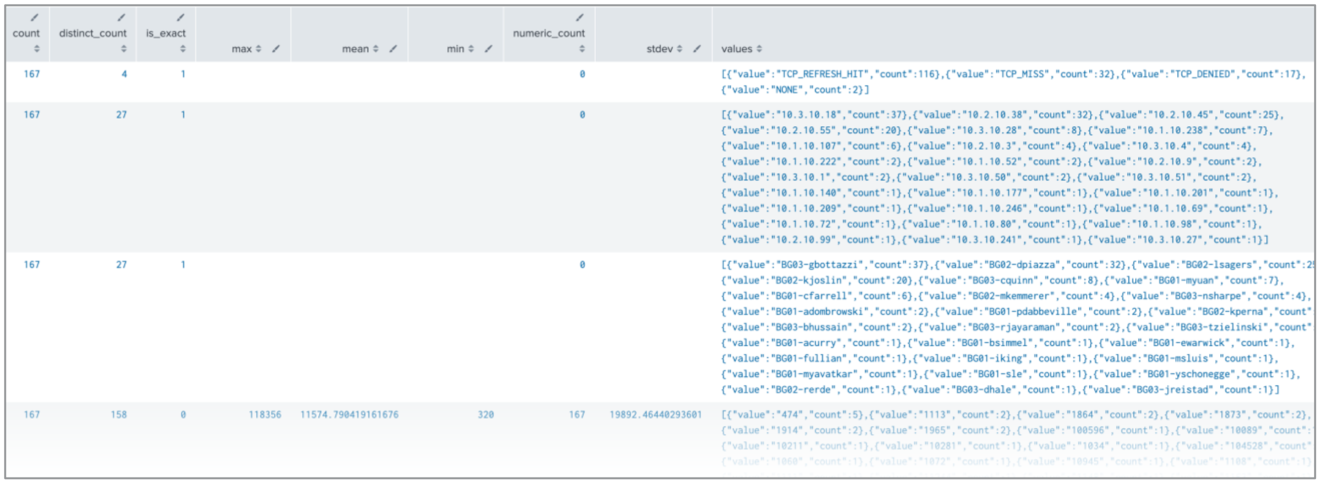
59. Save your search as a report with the name **L1S5**.

Scenario: A security operations manager wants to compare summary statistics for the size of requested objects (`sc_bytes`) and the threat score of the requested object (`x_wbrs_score`) from the web security appliance data over the past 24 hours.

Task 7: Use the `fieldsummary` command to generate summary statistics on 10 values from 2 fields.

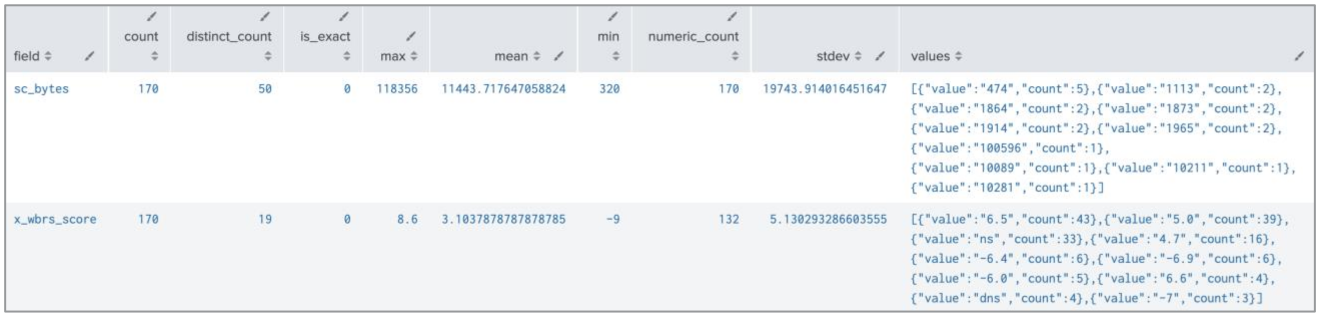
60. From the menu, click **Search**. (The search box re-initializes and should now be empty.)
61. Pipe the results of this search to the **`fieldsummary`** command and run over the **Last 24 hours.** The resulting table will display information for every field returned by the search.

```
index=network sourcetype=cisco_wsa_squid
```



count	distinct_count	is_exact	max	mean	min	numeric_count	stdev	values
167	4	1				0		[{"value":"TCP_REFRESH_HIT","count":116},{"value":"TCP_MISS","count":32},{"value":"TCP_DENIED","count":17}, {"value":"NONE","count":2}]
167	27	1				0		[{"value":"10.3.10.18","count":37},{"value":"10.2.10.38","count":32},{"value":"10.2.10.45","count":25}, {"value":"10.2.10.55","count":20},{"value":"10.3.10.28","count":8},{"value":"10.1.10.238","count":7}, {"value":"10.1.10.107","count":6},{"value":"10.2.10.3","count":4},{"value":"10.3.10.4","count":4}, {"value":"10.1.10.222","count":2},{"value":"10.1.10.52","count":2},{"value":"10.2.10.9","count":2}, {"value":"10.3.10.1","count":2},{"value":"10.3.10.50","count":2},{"value":"10.3.10.51","count":2}, {"value":"10.1.10.140","count":1},{"value":"10.1.10.177","count":1},{"value":"10.1.10.201","count":1}, {"value":"10.1.10.209","count":1},{"value":"10.1.10.246","count":1},{"value":"10.1.10.69","count":1}, {"value":"10.1.10.72","count":1},{"value":"10.1.10.80","count":1},{"value":"10.1.10.98","count":1}, {"value":"10.2.10.99","count":1},{"value":"10.3.10.241","count":1},{"value":"10.3.10.27","count":1}]
167	27	1				0		[{"value":"BG03-gbottazzi","count":37},{"value":"BG02-dpiazza","count":32},{"value":"BG02-lsagers","count":2 {"value":"BG02-kjoslin","count":20},{"value":"BG03-cquinn","count":8},{"value":"BG01-myuan","count":7}, {"value":"BG01-cfarrell","count":6},{"value":"BG02-mkemmerer","count":4},{"value":"BG03-nsharpe","count":4}, {"value":"BG01-adombrowski","count":2},{"value":"BG01-pdabbeville","count":2},{"value":"BG02-kperna","count" {"value":"BG03-bhussain","count":2},{"value":"BG03-rjayaraman","count":2},{"value":"BG03-tzielinski","count" {"value":"BG01-acurry","count":1},{"value":"BG01-bsimmel","count":1},{"value":"BG01-ewarwick","count":1}, {"value":"BG01-fullian","count":1},{"value":"BG01-iking","count":1},{"value":"BG01-msluis","count":1}, {"value":"BG01-myavatkar","count":1},{"value":"BG01-sle","count":1},{"value":"BG01-yschonegge","count":1}, {"value":"BG02-rerde","count":1},{"value":"BG03-dhale","count":1},{"value":"BG03-jreistad","count":1}]
167	158	0	118356	11574.790419161676	320	167	19892.46440293601	[{"value":"474","count":5},{"value":"1113","count":2},{"value":"1864","count":2},{"value":"1873","count":2}, {"value":"1914","count":2},{"value":"1965","count":2},{"value":"100596","count":1},{"value":"10089","count": {"value":"10211","count":1},{"value":"10281","count":1},{"value":"1034","count":1},{"value":"104528","count" {"value":"1060","count":1},{"value":"1072","count":1},{"value":"10945","count":1},{"value":"1108","count":1}

62. Modify the search to limit results to the top 10 values of **`sc_bytes`** and **`x_wbrs_score`**.



field	count	distinct_count	is_exact	max	mean	min	numeric_count	stdev	values
sc_bytes	170	50	0	118356	11443.717647058824	320	170	19743.914016451647	[{"value":"474","count":5},{"value":"1113","count":2}, {"value":"1864","count":2},{"value":"1873","count":2}, {"value":"1914","count":2},{"value":"1965","count":2}, {"value":"100596","count":1}, {"value":"10089","count":1},{"value":"10211","count":1}, {"value":"10281","count":1}]
x_wbrs_score	170	19	0	8.6	3.1037878787878785	-9	132	5.130293286603555	[{"value":"6.5","count":43},{"value":"5.0","count":39}, {"value":"ns","count":33},{"value":"4.7","count":16}, {"value":"-6.4","count":6},{"value":"-6.9","count":6}, {"value":"-6.0","count":5},{"value":"6.6","count":4}, {"value":"dns","count":4},{"value":"-7","count":3}]

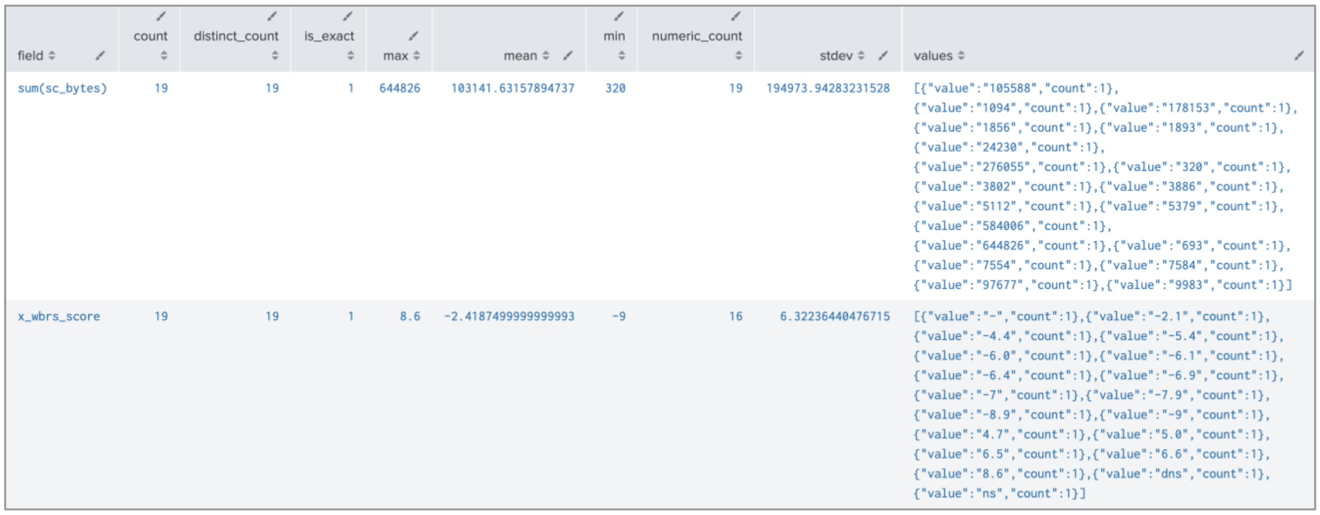
63. Save your search as a report with the name **L1S6**.

Challenge: A security analyst wants to study the variance between bytes consumption and web scores in the web security appliance data from yesterday.

Task 8: Use the `fieldsummary` command to calculate statistics on transformed data.

64. From the menu, click **Search**. (The search box re-initializes and should now be empty.)
65. Search web security appliance data (**`index=network sourcetype=cisco_wsa_squid`**) and calculate the sum of **`sc_bytes`** values by **`x_wbrs_score`**. Run the following search over **Yesterday.**

66. Make the resulting statistics table more useful for analysis by using the **`fieldsummary`** command to place the results in a comparison table.



field	count	distinct_count	is_exact	max	mean	min	numeric_count	stdev	values
sum(sc_bytes)	19	19	1	644826	103141.63157894737	320	19	194973.94283231528	[{"value":"105588","count":1}, {"value":"1094","count":1},{"value":"178153","count":1}, {"value":"1856","count":1},{"value":"1893","count":1}, {"value":"24230","count":1}, {"value":"276055","count":1},{"value":"320","count":1}, {"value":"3802","count":1},{"value":"3886","count":1}, {"value":"5112","count":1},{"value":"5379","count":1}, {"value":"584006","count":1}, {"value":"644826","count":1},{"value":"693","count":1}, {"value":"7554","count":1},{"value":"7584","count":1}, {"value":"97677","count":1},{"value":"9983","count":1}]
x_wbrs_score	19	19	1	8.6	-2.4187499999999993	-9	16	6.32236440476715	[{"value":"-","count":1},{"value":"-2.1","count":1}, {"value":"-4.4","count":1},{"value":"-5.4","count":1}, {"value":"-6.0","count":1},{"value":"-6.1","count":1}, {"value":"-6.4","count":1},{"value":"-6.9","count":1}, {"value":"-7","count":1},{"value":"-7.9","count":1}, {"value":"-8.9","count":1},{"value":"-9","count":1}, {"value":"4.7","count":1},{"value":"5.0","count":1}, {"value":"6.5","count":1},{"value":"6.6","count":1}, {"value":"8.6","count":1},{"value":"dns","count":1}, {"value":"ns","count":1}]

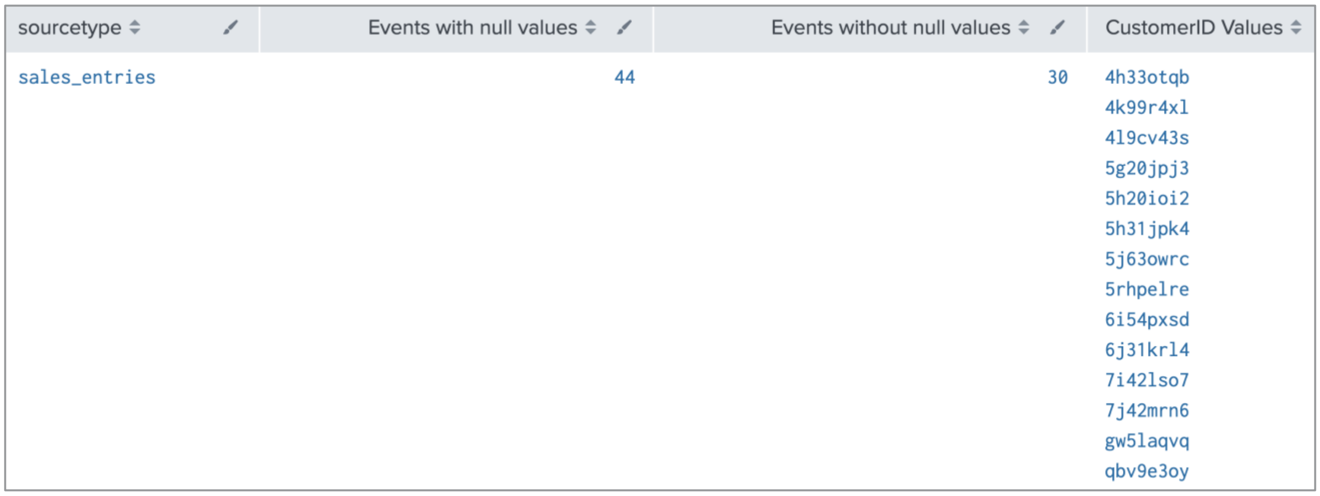
67. Save your search as a report with the name **LX1**.

Challenge: Count events from the BCG e-commerce system that have and do not have a value for `CustomerID` over the last 15 minutes. Count these events by sourcetype and list the unique `CustomerID` values present in the events.

Task 9: Complete the search to fulfill the scenario request.

68. From the menu, click **Search**. (The search box re-initializes and should now be empty.)
69. Complete the missing portions of the search so that the **`eval`** command assigns "yes" to events that do not contain a value for **`CustomerID`** and "no" to events that do contain a value for **`CustomerID`**. Execute the search over the **Last 15 minutes**.

```
index=sales sourcetype=sales_entries
| eval IsCustomerIDNull = if(???(???),"yes","no")
| stats count(eval(IsCustomerIDNull="yes")) as "Events with null values",
count(eval(IsCustomerIDNull="no")) as "Events without null values",
values(CustomerID) as "CustomerID Values" by sourcetype
```



sourcetype	Events with null values	Events without null values	CustomerID Values
sales_entries	44	30	4h33otqb
			4k99r4xl
			4l9cv43s
			5g20jpj3
			5h20ioi2
			5h31jpk4
			5j63owrc
			5rhpelre
			6i54pxsd
			6j31krl4
			7i42lso7
			7j42mrn6
			gw5laqvq
			qbv9e3oy

70. Save your search as a report with the name **LX2**.