splunk>

Miscellaneous

The iplocation command in this case will never be run on remote peers. All events from remote peers from the initial search for the terms FOO and BAR will be forwarded to the search head where the iplocation command will be run.	`FOO BAR | localop | iplocation`

Administrative

View information in the "audit" index.	`index=audit | audit`
Crawl root and home directories and add all possible inputs found (adds configuration information to "inputs.conf").	`| crawl root="/;/Users/" | input add`
Display a chart with the span size of 1 day.	`| dbinspect index=_internal span=1d`
Return the values of "host" for events in the "_internal" index.	`| metadata type=hosts index=_internal`
Return typeahead information for sources in the "_internal" index.	`| typeahead prefix=source count=10 index=_internal`

Alerting

Send search results to the specified email.	`... | sendemail to="elvis@splunk.com"`

Fields

add

Save the running total of "count" in a field called "total_count".	`... | accum count AS total_count`
Add information about the search to each event.	`... |addinfo`
Search for "404" events and append the fields in each event to the previous search results.	`... | appendcols [search 404]`
For each event where 'count' exists, compute the difference between count and its previous value and store the result in 'countdiff'.	`... | delta count AS countdiff`
Extracts out values like "7/01", putting them into the "monthday" attribute.	`... | erex monthday examples="7/01"`
Set velocity to distance / time.	`... | eval velocity=distance/time`
Extract field/value pairs and reload field extraction settings from disk.	`... | extract reload=true`

Extract field/value pairs that are delimited by "|;", and values of fields that are delimited by "=:".	`... | extract pairdelim="|;", kvdelim="=:", auto=f`
Add location information (based on IP address).	`... | iplocation`
Extract values from "eventtype.form" if the file exists.	`... | kvform field=eventtype`
Extract the "COMMAND" field when it occurs in rows that contain "splunkd".	`... | multikv fields COMMAND filter splunkd`
Set RANGE to "green" if the date_second is between 1-30; "blue", if between 31-39; "red", if between 40-59; and "gray", if no range matches (e.g. "0").	`... | rangemap field=date_second green=1-30 blue=31-39 red=40-59 default=gray`
Calculate the relevancy of the search and sort the results in descending order.	`disk error | relevancy | sort -relevancy`
Extract "from" and "to" fields using regular expressions. If a raw event contains "From: Susan To: Bob", then from=Susan and to=Bob.	`... | rex field=_raw "From: (?<from>.*) To: (?<to>.*)"`
Add the field: "comboIP". Values of "comboIP" = ""sourceIP" + "/" + "destIP"".	`... | strcat sourceIP "/" destIP comboIP`
Extract field/value pairs from XML formatted data. "xmlkv" automatically extracts values between XML tags.	`... | xmlkv`

convert

Convert every field value to a number value except for values in the field "foo" (use the "none" argument to specify fields to ignore).	`... | convert auto(*) none(foo)`
Change all memory values in the "virt" field to Kilobytes.	`... | convert memk(virt)`
Change the sendmail syslog duration format (D+HH:MM:SS) to seconds. For example, if "delay="00:10:15"", the resulting value will be "delay="615"".	`... | convert dur2sec(delay)`
Convert values of the "duration" field into number value by removing string values in the field value. For example, if "duration="212 sec"", the resulting value will be "duration="212"".	`... | convert rmunit(duration)`
Separate the value of "foo" into multiple values.	`... | makemv delim=":" allowempty=t foo`
For sendmail events, combine the values of the senders field into a single value; then, display the top 10 values.	`eventtype="sendmail" | nomv senders | top senders`

filter

Keep only the "host" and "ip" fields, and display them in the order: "host", "ip".	`... | fields host, ip`
Remove the "host" and "ip" fields.	`... | fields - host, ip`

modify

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`
Rename the "_ip" field as "IPAddress".	`... | rename _ip as IPAddress`
Change any host value that ends with "localhost" to "localhost".	`... | replace *localhost with localhost in host`

read

There is a lookup table specified in a stanza name 'usertogroup' in transform.conf. This lookup table contains (at least) two fields, 'user' and 'group'. For each event, we look up the value of the field 'local_user' in the table and for any entries that matches, the value of the 'group' field in the lookup table will be written to the field 'user_group' in the event.	`... | lookup usertogroup user as local_user OUTPUT group as user_group`

Formatting

Show a summary of up to 5 lines for each search result.	`... |abstract maxlines=5`
Compare the "ip" values of the first and third search results.	`... | diff pos1=1 pos2=3 attribute=ip`
Highlight the terms "login" and "logout".	`... | highlight login,logout`
Displays an different icon for each eventtype.	`... | iconify eventtype`
Output the "_raw" field of your current search into "_xml".	`... | outputtext`
Anonymize the current search results.	`... | scrub`
Un-escape all XML characters.	`... | xmlunescape`

Index

add

Add each source found by crawl in the default index with automatic source classification (sourcetyping)	`| crawl | input add`

delete

Delete events from the "imap" index that contain the word "invalid"	`index=imap invalid | delete`

summary

Put "download" events into an index named "downloadcount".	`eventtypetag="download" | collect index=downloadcount`
Find overlapping events in "summary".	`index=summary | overlap`

Compute the necessary information to later do 'chart avg(foo) by bar' on summary indexed results.	`... | sichart avg(foo) by bar`
Compute the necessary information to later do 'rare foo bar' on summary indexed results.	`... | sirare foo bar`
Compute the necessary information to later do 'stats avg(foo) by bar' on summary indexed results	`... | sistats avg(foo) by bar`
Compute the necessary information to later do 'timechart avg(foo) by bar' on summary indexed results.	`... | sitimechart avg(foo) by bar`
Compute the necessary information to later do 'top foo bar' on summary indexed results.	`... | sitop foo bar`

Reporting

Calculate the sums of the numeric fields of each result, and put the sums in the field "sum".	`... | addtotals fieldname=sum`
Analyze the numerical fields to predict the value of "is_activated".	`... | af classfield=is_activated`
Return events with uncommon values.	`... | anomalousvalue action=filter pthresh=0.02`
Return results associated with each other (that have at least 3 references to each other).	`... | associate supcnt=3`
For each event, copy the 2nd, 3rd, 4th, and 5th previous values of the 'count' field into the respective fields 'count_p2', 'count_p3', 'count_p4', and 'count_p5'.	`... | autoregress count p=2-5`
Bucket search results into 10 bins, and return the count of raw events for each bucket.	`... | bucket size bins=10 | stats count(_raw) by size`
Return the average "thruput" of each "host" for each 5 minute time span.	`... | bucket _time span=5m | stats avg(thruput) by _time host`
Return the average (mean) "size" for each distinct "host".	`... | chart avg(size) by host`
Return the the maximum "delay" by "size", where "size" is broken down into a maximum of 10 equal sized buckets.	`... | chart max(delay) by size bins=10`
Return the ratio of the average (mean) "size" to the maximum "delay" for each distinct "host" and "user" pair.	`... | chart eval(avg(size)/max(delay)) by host user`
Return max(delay) for each value of foo split by the value of bar.	`... | chart max(delay) over foo by bar`
Return max(delay) for each value of foo.	`... | chart max(delay) over foo`
Build a contingency table of "datafields" from all events.	`... | contingency datafield1 datafield2 maxrows=5 maxcols=5 usetotal=F`
Calculate the co-occurrence correlation between all fields.	`... | correlate type=cocur`
Return the number of events in the '_internal' index.	`| eventcount index=_internal`

Compute the overall average duration and add 'avgdur' as a new field to each event where the 'duration' field exists	`... | eventstats avg(duration) as avgdur`
Make "_time" continuous with a span of 10 minutes.	`... | makecontinuous _time span=10m`
Remove all outlying numerical values.	`... | outlier`
Return the least common values of the "url" field.	`... | rare url`
Remove duplicates of results with the same "host" value and return the total count of the remaining results.	`... | stats dc(host)`
Return the average for each hour, of any unique field that ends with the string "lay" (for example, delay, xdelay, relay, etc).	`... | stats avg(*lay) BY date_hour`
Search the access logs, and return the number of hits from the top 100 values of "referer_domain".	`sourcetype=access_combined | top limit=100 referer_domain | stats sum(count)`
For each event, add a count field that represent the number of event seen so far (including that event). i.e., 1 for the first event, 2 for the second, 3, 4 ... and so on	`... | streamstats count`
Graph the average "thruput" of hosts over time.	`... | timechart span=5m avg(thruput) by host`
Create a timechart of average "cpu_seconds" by "host", and remove data (outlying values) that may distort the timechart's axis.	`... | timechart avg(cpu_seconds) by host | outlier action=tf`
Calculate the average value of "CPU" each minute for each "host".	`... | timechart span=1m avg(CPU) by host`
Create a timechart of the count of from "web" sources by "host"	`... | timechart count by host`
Compute the product of the average "CPU" and average "MEM" each minute for each "host"	`... | timechart span=1m eval(avg(CPU) * avg(MEM)) by host`
Return the 20 most common values of the "url" field.	`... | top limit=20 url`
Computes a 5 event simple moving average for field 'foo' and write to new field 'smoothed_foo' also computes N=10 exponential moving average for field 'bar' and write to field 'ema10(bar)'.	`... | trendline sma5(foo) as smoothed_foo ema10(bar)`
Reformat the search results.	`... | timechart avg(delay) by host | untable _time host avg_delay`
Reformat the search results.	`... | xyseries delay host_type host`

Results

append

Append the current results with the tabular results of "fubar".	`... | chart count by bar | append [search fubar | chart count by baz]`
Joins previous result set with results from 'search foo', on the id field.	`... | join id [search foo]`

filter

Return only anomalous events.	`... | anomalies`
Remove duplicates of results with the same host value.	`... | dedup host`
Combine the values of "foo" with ":" delimiter.	`... | mvcombine delim=":" foo`
Keep only search results whose "_raw" field contains IP addresses in the non-routable class A (10.0.0.0/8).	`... | regex _raw="(?<!\d)10.\d{1,3}\.\d{1,3}\.\d{1,3}(?!\d)"`
Join results with itself on 'id' field.	`... | selfjoin id`
For the current search, keep only unique results.	`... | uniq`
Return "physicjobs" events with a speed is greater than 100.	`sourcetype=physicsobjs | where distance/time > 100`

generate

All daily time ranges from oct 25 till today	`| gentimes start=10/25/07`
Loads the events that were generated by the search job with id=1233886270.2	`| loadjob 1233886270.2 events=t`
Create new events for each value of multi-value field, "foo".	`... | mvexpand foo`
Run the "mysecurityquery" saved search.	`| savedsearch mysecurityquery`

group

Cluster events together, sort them by their "cluster_count" values, and then return the 20 largest clusters (in data size).	`... | cluster t=0.9 showcount=true | sort - cluster_count | head 20`
Group search results into 4 clusters based on the values of the "date_hour" and "date_minute" fields.	`... | kmeans k=4 date_hour date_minute`
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`
Have Splunk automatically discover and apply event types to search results	`... | typelearner`
Force Splunk to apply event types that you have configured (Splunk Web automatically does this when you view the "eventtype" field).	`... | typer`

order

Return the first 20 results.	`... | head 20`
Reverse the order of a result set.	`... | reverse`
Sort results by "ip" value in ascending order and then by "url" value in descending order.	`... | sort ip, -url`
Return the last 20 results (in reverse order).	`... | tail 20`

read

Display events from the file "messages.1" as if the events were indexed in Splunk.	`| file /var/log/messages.1`
Read in results from the CSV file: "$SPLUNK_HOME/var/run/splunk/all.csv", keep any that contain the string "error", and save the results to the file: "$SPLUNK_HOME/var/run/splunk/error.csv"	`| inputcsv all.csv | search error | outputcsv errors.csv`
Read in "users.csv" lookup file (under $SPLUNK_HOME/etc/system/lookups or $SPLUNK_HOME/etc/apps/*/lookups).	`| inputlookup users.csv`

write

Output search results to the CSV file 'mysearch.csv'.	`... | outputcsv mysearch`
Write to "users.csv" lookup file (under $SPLUNK_HOME/etc/system/lookups or $SPLUNK_HOME/etc/apps/*/lookups).	`| outputlookup users.csv`

Search

external

Run the Python script "myscript" with arguments, myarg1 and myarg2; then, email the results.	`... | script python myscript myarg1 myarg2 | sendemail to=david@splunk.com`

search

Keep only search results that have the specified "src" or "dst" values.	`src="10.9.165.*" OR dst="10.9.165.8"`

subsearch

Get top 2 results and create a search from their host, source and sourcetype, resulting in a single search result with a _query field: _query=(("host::mylaptop" AND "source::syslog.log" AND "sourcetype::syslog") OR ("host::bobslaptop" AND	`... | head 2 | fields source, sourcetype, host | format`

"source::bob-syslog.log" AND "sourcetype::syslog"))	
Search the time range of each previous result for "failure".	`... | localize maxpause=5m | map search="search failure starttimeu=$starttime$ endtimeu=$endtime$"`
Return values of "URL" that contain the string "404" or "303" but not both.	`| set diff [search 404 | fields url] [search 303 | fields url]`