general-search-examples.md

github.com/SumoLogic/sumologic-documentation/blob/main/docs/search/search-cheat-sheets/general-search-examples.md

jpipkin1
DOCS-606 - Wistia video links (#4905)
2b8d0cd · Dec 30, 2024

id	title	sidebar_label	description
general-search-examples	General Search Examples Cheat Sheet	General Search Examples	The General Search Examples cheat sheet provides examples of useful search queries for different use cases.

The examples use this sample Apache log message where applicable:

```
10.154.181.28 - - [24/Jan/2012:12:34:58 -0700] "GET /Courses/Topics/54.htm HTTP/1.1"
200 9951 "Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.7 (KHTML, like Gecko)
Chrome/16.0.912.75 Safari/535.7"
Host: raw_hosted_apps Name: /usr/sumo/collector-16.1-5/logs/reporter.log Category: apache
```

Keyword Expressions

Look for failed attempts to su or sudo to root.

```
(su OR sudo ) AND (fail* OR error)
```

Look for errors in sshd logs.

```
sshd AND (fail* OR error OR allowed OR identity)
```

Look for general authorization failures excluding router messages.

```
auth* AND (fail* OR error?) NOT _sourceCategory=routers
```

:::sumo More Info For more information, see Keyword Search Expressions. :::

Parse, Count, and Top operators

Extract `"from"` and `"to"` fields. For example, if a raw event contains "From: Jane To: John", then from=Jane and to=John.

```
* | parse "From: * To: *" as (from, to)
```

Extract the source IP addresses using a regular expression for the four octets of an IP address.

```
*| parse regex "(?<src_ip>\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3})"
```

Identify all URL addresses visited, extract them as the `url` field.

```
_sourceCategory=apache
| parse "GET * " as url
```

Identify traffic from Source Category `apache` and extract the source addresses, message sizes, and the URLs visited.

```
_sourceCategory=apache
| parse "* " as src_IP
| parse " 200 * " as size
| parse "GET * " as url
```

For the Source Category `apache`, calculate the total number of bytes transferred to each source IP address.

```
_sourceCategory=apache
| parse "* " as src_IP
| parse " 200 * " as size
| count, sum(size) by src_IP
```

For the Source Category `apache`, calculate the average size of all successful HTTP responses.

```
_sourceCategory=apache
| parse " 200 * " as size
| avg(size)
```

For the Source Category `apache`, extract src, size, and URL even if the size field is missing from the log message (`nodrop`).

```
_sourceCategory=apache
| parse "* " as src_IP
| parse " 200 * " as size nodrop
| parse "GET * " as url
```

Identify the number of times a URL has been visited.

```
_sourceCategory=apache
| parse "GET * " as url
| count by url
```

Identify the total number of pages by source IP address.

```
_sourceCategory=apache
| parse "* -" as src_ip
| count by src_ip
```

Identify the total number of pages by source IP address and re-order them by most frequently loaded pages.

```
_sourceCategory=apache
| parse "* " as src_ip
| parse "GET * " as url
| count by url
| sort by _count
```

Identify the top 10 requested pages.

```
* | parse "GET * " as url
| count by url
| top 10 url by _count
```

Identify the top 10 source IP addresses by bandwidth usage.

```
_sourceCategory=apache
| parse " 200 * " as size
| parse "* -" as src_ip
| sum(size) as total_bytes by src_ip
| top 10 src_ip by total_bytes
```

Identify the top 100 source IP addresses by number of hits.

```
_sourceCategory=apache
| parse "* -" as src_ip
| count by src_ip
| top 100 src_ip by _count
```

:::sumo More Info For more information, see Parsing, Count, and Top. :::

Timeslice and Transpose

For the Source Category `apache`, count by `status_code` and `timeslice` of 1 hour.

```
_sourceCategory=apache*
| parse "HTTP/1.1\" * * \"" as (status_code, size)
| timeslice 1h
| count by _timeslice, status_code
```

For the Source Category `apache`, count by `status_code` and `timeslice` of 1 hour, transpose status_code to column.

```
_sourceCategory=apache*
| parse "HTTP/1.1\" * * \"" as (status_code, size)
| timeslice 1h
| count by _timeslice, status_code
| transpose row _timeslice column status_code
```

For the Source Category `apache`, count by `status_code` and `timeslice` into 5 buckets over search result.

```
_sourceCategory=apache*
| parse "HTTP/1.1\" * * \"" as (status_code, size)
| timeslice 5 buckets
| count by _timeslice, status_code
```

For the Source Category `Apache/Access`, count messages by status code categories, grouping all 200s, 300s, 400s, and 500s together.

```
_sourceCategory=Apache/Access
| timeslice 15m
| if (status_code matches "20*",1,0) as resp_200
| if (status_code matches "30*",1,0) as resp_300
| if (status_code matches "40*",1,0) as resp_400
| if (status_code matches "50*",1,0) as resp_500
| if (!(status_code matches "20*" or status_code matches "30*" or status_code matches
"40*" or status_code matches "50*"),1,0) as resp_others
| count(*), sum(resp_200) as tot_200, sum(resp_300) as tot_300, sum(resp_400) as tot_400,
sum(resp_500) as tot_500, sum(resp_others) as tot_others by _timeslice
```

Or, alternately, you can use:

```
_sourceCategory=Apache/Access
| timeslice 15m
| if(status_code matches "20*","200s",
if(status_code matches "30*","300s",
if(status_code matches "40*","400s",
if(status_code matches "50*","500s","Other")))) as status_code_group
| count by _timeslice, status_code_group
| transpose row _timeslice column status_code_group
```

:::sumo More Info For more information, see `timeslice` operator and `transpose` operator. :::

Conditional operators

For the Source Category `apache`, find all messages with a client error status code (`40*`):

```
_sourceCategory=apache*
| parse "HTTP/1.1\" * * \"" as (status_code, size)
| where status_code matches "40*"
```

For the Source Category `apache`, count hits by browser:

```
_sourceCategory=Apache/Access
| extract "\"[A-Z]+ \S+ HTTP/[\d\.]+\" \S+ \S+ \S+ \"(?<agent>[^\"]+?)\""
| if (agent matches "*MSIE*",1,0) as ie
| if (agent matches "*Firefox*",1,0) as firefox
| if (agent matches "*Safari*",1,0) as safari
| if (agent matches "*Chrome*",1,0) as chrome
| sum(ie) as ie, sum(firefox) as firefox, sum(safari) as safari, sum(chrome) as chrome
```

Use the `where` operator to match only weekend days.

```
* | parse "day=*:" as day_of_week
| where day_of_week in ("Saturday","Sunday")
```

Identify all URLs that contain the subdirectory "Courses" in the path.

```
*| parse "GET * " as url
| where url matches "*Courses*"
```

Find version numbers that match numeric values 2, 3 or 1. Use the num operator to change the string into a number.

```
* | parse "Version=*." as number | num(number)
| where number in (2,3,6)
```

:::sumo More Info For more information, see `where` operator and `if` operator. :::

LogReduce operator

Use Sumo Logic's clustering algorithm to look for patterns in error/exception incidents in your deployment.

```
exception* or fail* or error* or fatal*
| logreduce
```

:::sumo More Info For more information, see LogReduce. :::

Add metadata fields

For any query, you can increase specificity by adding metadata fields to the keyword expression. Metadata fields include `_sourceCategory`, `_sourceHost` , and `_sourceName`.

Edit Source metadata in the **Collection** tab.

For details, see Search Metadata.