Exporting Splunk Data at Scale with Scribl

scribl.py

Exporting Splunk Data at Scale with Scribl. This is a python script that can be run on each Splunk Indexer for the purpose of exporting historical bucket data (raw events + metadata) at scale by balancing the work across multiple CPUs then forwarding to Cribl.

Demo Videos

- HD - Exporting Splunk Data at Scale with Scribl
- 4K - Exporting Splunk Data at Scale with Scribl

Background

Splunk to Cribl = scribl (#thanksKam)

Exporting large amounts of previously indexed data from Splunk is challenging via the Splunk-supported approaches detailed here: https://docs.splunk.com/Documentation/Splunk/8.2.6/Search/Exportsearchresults.

The core Splunk binary in every install provides a switch (cmd exporttool) that allows you to export the data from the compressed buckets on the indexers back into their original raw events. You can dump them to very large local csv files or stream them to stdout so a script can redirect over the network to a receiver such as Cribl Stream. This swith has been used by others for quite a while but it isn't well documented.

Assuming that Splunk is installed in /opt/splunk/, the below commands can be applied to a particular bucket in an index called “bots” to export it.

Exporting to stdout:

```
/opt/splunk/bin/splunk cmd exporttool /opt/splunk/var/lib/splunk/bots/db/db_1564739504_1564732800_2394 /dev/stdout -csv
```

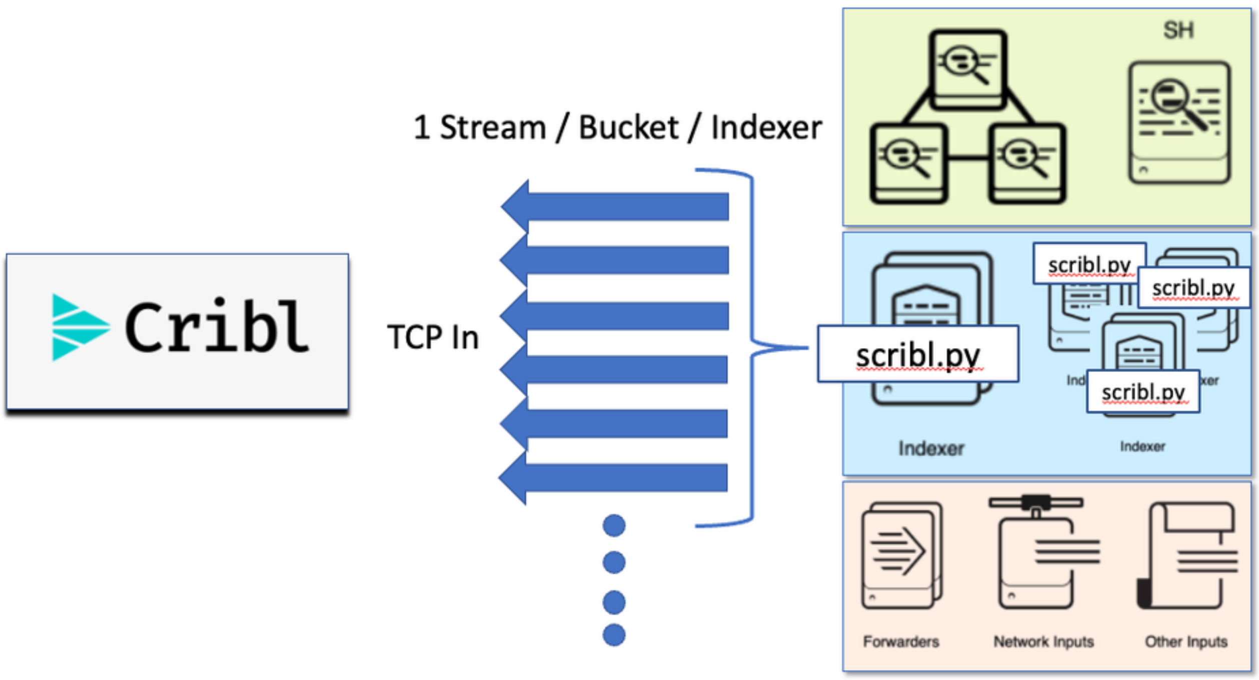
Exporting to a local csv file:

```
/opt/splunk/bin/splunk cmd exporttool /opt/splunk/var/lib/splunk/bots/db/db_1564739504_1564732800_2394
/exports/bots/db_1564739504_1564732800_2394.csv -csv
```

There will be many buckets so some poor soul will need to build a script to export all or some of the buckets and some sort of parallelization should be used to speed the process up. The exported data will be very large (uncompressed, 3-20x) compared to the size of the individual buckets that make up the index!

Requirements

Splunk stores its collected data on the indexers within the “Indexing Tier” as detailed below. The data is compressed and stored in a collection of time series buckets that reside on each indexer. Each bucket contains a rawdata journal, along with associated tsidx, and metadata files. The search heads access these buckets and it’s very rare for someone to access them directly from the indexer CLI unless there is a need to export data to retrieve the original raw events. We will use the indexer CLI to export the original raw events (per bucket and in parallel) as well as a few other pieces of important metadata as detailed below.



For a deeper dive into how Splunk indexes data, see this: How the indexer stores indexes - Splunk Documentation

You will need:

- CLI access to each Linux indexer with the index/buckets that need to be exported. This process only applies to on-prem or non-SplunkCloud deployments.
- To install nc (netcat) on each indexer to act as the transport mechanism until we have enough demand to build the transport into the script.
- To make sure outbound communication from each indexer to the Cribl Worker TCP port is open.

Frozen Data

The Splunk exporttool switch that scribl depends on requires a complete hot/warm/cold directory containing all of the metadata files in addition to the journal.gz file. When buckets are moved to a frozen archive, all of the metadata files are removed with only the journal.gz file remaining. Scribl can not extract raw events from frozen archives.

Buckets must first be “thawed” as described here. It’s a straightforward process of copying the frozen buckets somewhere and running a “splunk rebuild” for each bucket to recreate the metadata. Scribl can be run against this thawed data.

Technical

Scale

We achieve scale for large volumes of data by processing buckets in parallel across as many CPUs as you would like to dedicate AND by streaming the data directly from disk with a single read to Cribl without ever having to write extracted/uncompressed event data to disk. Extracting/uncompressing the event data to disk would result in enormous disk IO bottlenecks and disk space consumption.

Disk speed (IOPS) and the number of CPUs are generally your limiting factors on the indexers. While disk speed is a factor, it's usually not a factor in the overall scale picture because Splunk indexers will usually have high IOPs capabilities. You can monitor the linux processes to get a feel for whether scribl processes are in SLEEP or RUN mode. If they spend the majority of their time in SLEEP mode, they are being throttled by disk, network, Cribl workers, etc and adding more CPUs will probably not buy you more speed.

The scribl script running on your indexers and Cribl Stream workers are built to scale and will usually not be your bottleneck. Your bottlenecks will almost certainly be bandwidth constraints between your indexers and your final destination. Depending on where you deploy your Cribl Stream workers, that bandwidth bottleneck might exist between the indexers and Cribl workers or between your Cribl workers and the final destination. If you happen to have unlimited bandwidth, you will likely find your next bottleneck to be the ingest rate at your destination platform.

Exported Data Format

The exported data will be csv formatted with a header followed by the individual events. It's important to call out that these events are often multiline events with the most common example being windows logs. The below events are examples that are generated by Splunk and then passed via stdin to the scribl.py script.

The _raw field contains the original event and the other fields were captured/created during ingest. _time is the time extracted from the event which will be the primary time reference used by the destination analytics platform. The sourcetype field will likely be what is used by the destination to determine how to parse and where to route the event.

Example:

```
"_time",source,host,sourcetype,"_raw","_meta"
```

```
1564734905,"source::10.1.1.1","host::hogshead","sourcetype::fgt_utm","date=2019-08-02 time=08:35:05 devname=hogshead
devid=FGT60D4614044725 logid=1059028704 type=utm subtype=app-ctrl eventtype=app-ctrl-all level=information vd=root
appid=38131 user="""" srcip=10.1.1.103 srcport=51971 srcintf=""internal"" dstip=172.217.11.227 dstport=443 dstintf=""wan1""
profiletype=""applist"" proto=6 service=""HTTPS"" policyid=1 sessionid=594789 applist=""default"" appcat=""General.Interest""
app=""Google.Accounts"" action=pass hostname=""ssl.gstatic.com"" url=""/"" msg=""General.Interest: Google.Accounts,""
apprisk=elevated","_indextime::1564734907 _subsecond::.000 syslog-server::jupiter severity::notice facility::user punct::""=--
_=::_=_=_=_=_=-_=--_=_=_=_=\""\""_=..._=_=\""\""_=..._="""
```

```
1564734846,"source::WinEventLog:Microsoft-Windows-
PowerShell/Operational","host::titan","sourcetype::XmlWinEventLog:Microsoft-Windows-PowerShell/Operational","<Event
xmlns='http://schemas.microsoft.com/win/2004/08/events/event'><System><Provider Name='Microsoft-Windows-PowerShell'
Guid='{A0C1853B-5C40-4B15-8766-3CF1C58F985A}'/><EventID>4103</EventID><Version>1</Version><Level>4</Level>
<Task>106</Task><Opcode>20</Opcode><Keywords>0x0</Keywords><TimeCreated SystemTime='2019-08-
02T08:34:06.167139700Z'/><EventRecordID>5968761</EventRecordID><Correlation ActivityID='{135BC459-4718-0000-AAD1-
74131847D501}'/><Execution ProcessID='3016' ThreadID='3720'/><Channel>Microsoft-Windows-
PowerShell/Operational</Channel><Computer>titan.thirstyberner.com</Computer><Security UserID='S-1-5-18'/></System>
<EventData><Data Name='ContextInfo'>        Severity = Informational

        Host Name = Default Host

        Host Version = 4.0

        Host ID = 89297903-4c6b-4e9d-b0a4-49c76b2c36ae

        Engine Version = 4.0

        Runspace ID = 657f43dd-6fb5-42c9-8b93-154f3a1e53dd

        Pipeline ID = 1

        Command Name = Out-Null
        Command Type = Cmdlet

        Script Name =

        Command Path =

        Sequence Number = 1565734

        User = THIRSTYBERNER\SYSTEM

        Shell ID = Microsoft.PowerShell

</Data><Data Name='UserData'></Data><Data Name='Payload'>ParameterBinding(Out-Null): name=""InputObject""; 
value=""True""

</Data></EventData></Event>","_indextime::1564734908 punct::<_='://..//////'><><_='--'_='{----}'/><></><></><>"
```

Cribl Stream Routing

The routing of data as you need it into the destination you need it to be in is one of the most important use cases Cribl Stream brings to the table. Scribl is a great use case for that exact scenario. You will likely have indexes you wish to export which contain multiple sourcetypes. The Splunk sourcetype assignment is contained in every event that Cribl Stream processes. You can filter, optimize, route, etc each of those sourcetypes however you choose. We used Splunk's Boss of the SOC dataset for testing largely because it is real-world security data ingested during a live campaign and it contains a very diverse collection of data (souretypes) to best flush out unexpected bugs (multiline events, gigantic events, etc). The github repo details over 100 sourcetypes available in the BOTSv3 dataset.

scribl.py

The scribl.py script is available in this repo.

Example Usage:

```
./scribl.py -h
usage: scribl.py [-h] [-t] [-n NUMSTREAMS] [-l LOGFILE] [-et EARLIEST]
        [-lt LATEST] -d DIRECTORY -r REMOTEIP -p REMOTEPORT

This is to be run on a Splunk Indexer for the purpose of exporting buckets and
streaming their contents to Cribl Stream

optional arguments:
        -h, --help          show this help message and exit
        -t, --TLS           Send with TLS enabled
        -n NUMSTREAMS, --numstreams NUMSTREAMS
                the number of parallel stream to utilize
        -l LOGFILE, --logfile LOGFILE
                specify the location to write/append the logging
        -et EARLIEST, --earliest EARLIEST
                specify the earliest epoch time for bucket selection
        -lt LATEST, --latest LATEST
                specify the latest epoch time for bucket selection

required named arguments:
        -d DIRECTORY, --directory DIRECTORY
                Source directory containing the buckets
```

```
  -r REMOTEIP, --remoteIP REMOTEIP
        Remote address to send the exported data to
  -p REMOTEPORT, --remotePort REMOTEPORT
        Remote TCP port to be used
```

```
scribl.py -d /opt/splunk/var/lib/splunk/bots/db/ -r 14.2.39.121 -p 20000 -t -n4 -l /tmp/scribl.log -et 1564819155 -lt 1566429310
```

On No! My Splunk license expired!

Worry not, my friend. When the enterprise license expires, Splunk customers are free to use the 60-day trial or even the free version of Splunk to perform the export. Sanity check my claim here: https://docs.splunk.com/Documentation/Splunk/9.0.0/Admin/MoreaboutSplunkFree .

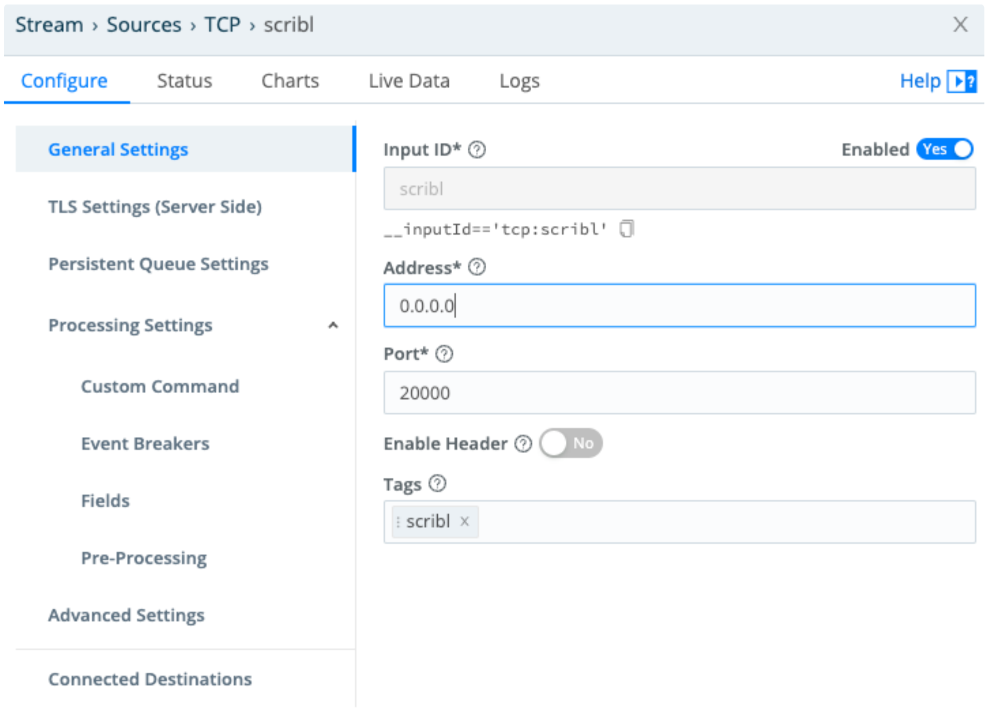
We don't care about indexing new data and we don't care about distributed search since we will use the trial/free Splunk binary in a standalone manner on each of the indexers that have data we need to migrate. Just install trial/free Splunk on top of or alongside the existing install and point scribl.py at your splunk binary and the directory containing the buckets you need to export.

Cribl Stream Config

You can get started instantly with Cribl Cloud or even using the Cribl Free license option but keep in mind daily ingest limits (very generous) and # of cores (also very generous at 10) that can be used may factor into a full scale data export. If you choose to install Cribl Stream on-prem on in your own cloud, the documentation is your friend and will get you going quickly.

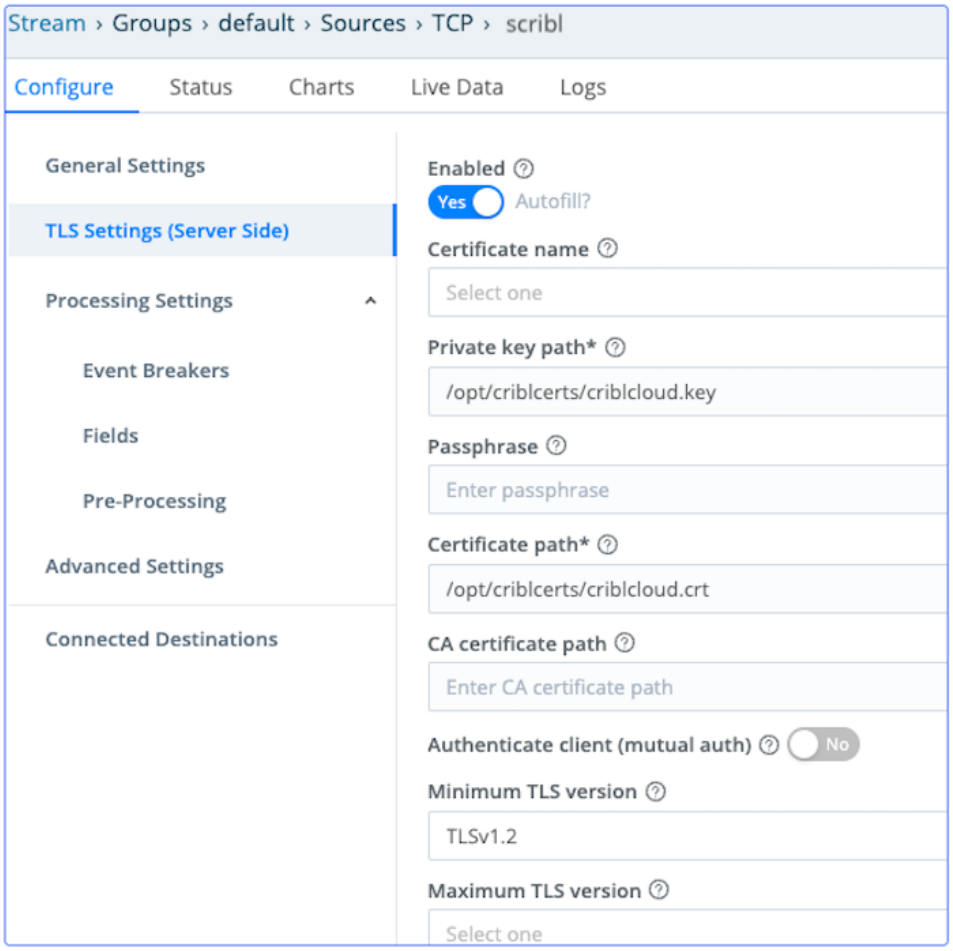
Once you have satisfied the above requirements (CLI, nc, and firewall) on your Splunk indexers, grab the scribl.py script from the github repo and copy it over to each indexer. The only thing in the script that is hard coded is the default install location of Splunk (/opt/splunk) which you can easily modify if you are running a non-default config. Keep in mind that we are running the script directly on the Splunk indexers and a python binary is kept under $SPLUNK_HOME/bin.

Add a new Source:



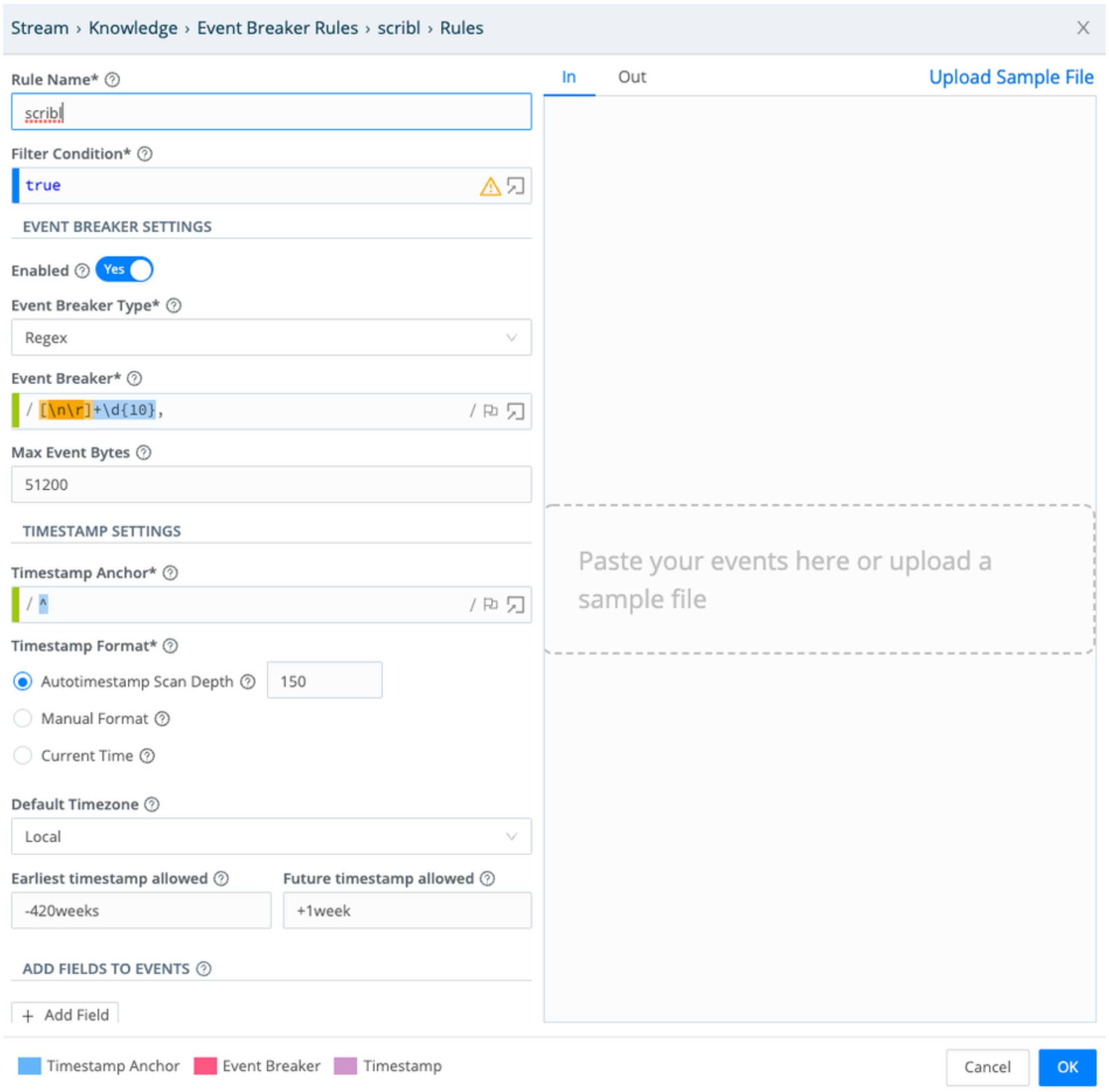
Input ID*: scribl
Enabled: Yes
__inputId=='tcp:scribl'
Address*: 0.0.0.0
Port*: 20000
Enable Header: No
Tags: scribl

Enable and Config TLS (recommended):



Enabled: Yes
Autofill?
Certificate name: Select one
Private key path*: /opt/criblcerts/criblcloud.key
Passphrase: Enter passphrase
Certificate path*: /opt/criblcerts/criblcloud.crt
CA certificate path: Enter CA certificate path
Authenticate client (mutual auth): No
Minimum TLS version: TLSv1.2
Maximum TLS version: Select one

Create a new Event Breaker and set it to “[\n\r]+\d{10},” as detailed below:



Stream › Knowledge › Event Breaker Rules › scribl › Rules

Rule Name*: scribl

Filter Condition*: `true`

EVENT BREAKER SETTINGS

Enabled: Yes

Event Breaker Type*: Regex

Event Breaker*: `/ [\n\r]+\d{10}, /`

Max Event Bytes: 51200

TIMESTAMP SETTINGS

Timestamp Anchor*: `/ ^ /`

Timestamp Format*:
- [✓ Autotimestamp Scan Depth] 150
- Manual Format
- Current Time

Default Timezone: Local

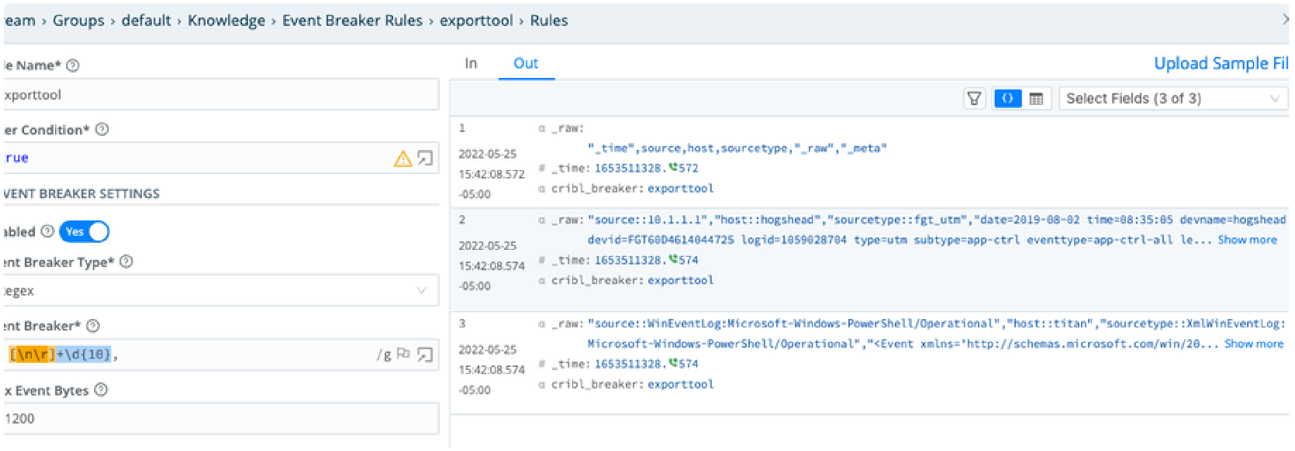
Earliest timestamp allowed: -420weeks

Future timestamp allowed: +1week

ADD FIELDS TO EVENTS

Paste your events here or upload a sample file

Validate:



eam › Groups › default › Knowledge › Event Breaker Rules › exporttool › Rules

le Name*: xporttool

er Condition*: `rue`

VENT BREAKER SETTINGS

abled: Yes

ent Breaker Type*: egex

ent Breaker*: `[\n\r]+\d{10}, /g`

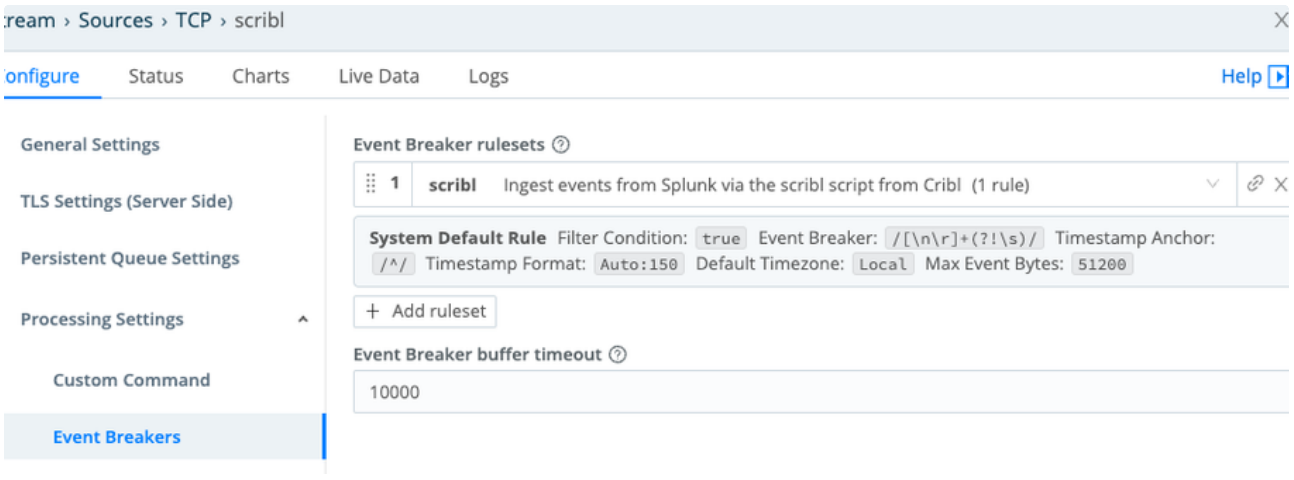
x Event Bytes: 1200

```
1                        _raw:
2022-05-25                  "_time",source,host,sourcetype,"_raw","_meta"
15:42:08.572             _time: 1653511328.572
-05:00                   cribl_breaker: exporttool

2                        _raw: "source::10.1.1.1","host::hogshead","sourcetype::fgt_utm","date=2019-08-02 time=08:35:05 devname=hogshead
2022-05-25                  devid=FGT60D4614044725 logid=1059028704 type=utm subtype=app-ctrl eventtype=app-ctrl-all le... Show more
15:42:08.574             _time: 1653511328.574
-05:00                   cribl_breaker: exporttool

3                        _raw: "source::WinEventLog:Microsoft-Windows-PowerShell/Operational","host::titan","sourcetype::XmlWinEventLog:
2022-05-25                  Microsoft-Windows-PowerShell/Operational","<Event xmlns='http://schemas.microsoft.com/win/20... Show more
15:42:08.574             _time: 1653511328.574
-05:00                   cribl_breaker: exporttool
```

Attach the new Event Breaker to your Source:



:ream › Sources › TCP › scribl

onfigure | Status | Charts | Live Data | Logs | Help

General Settings
TLS Settings (Server Side)
Persistent Queue Settings
Processing Settings
Custom Command
Event Breakers

Event Breaker rulesets

1 scribl Ingest events from Splunk via the scribl script from Cribl (1 rule)

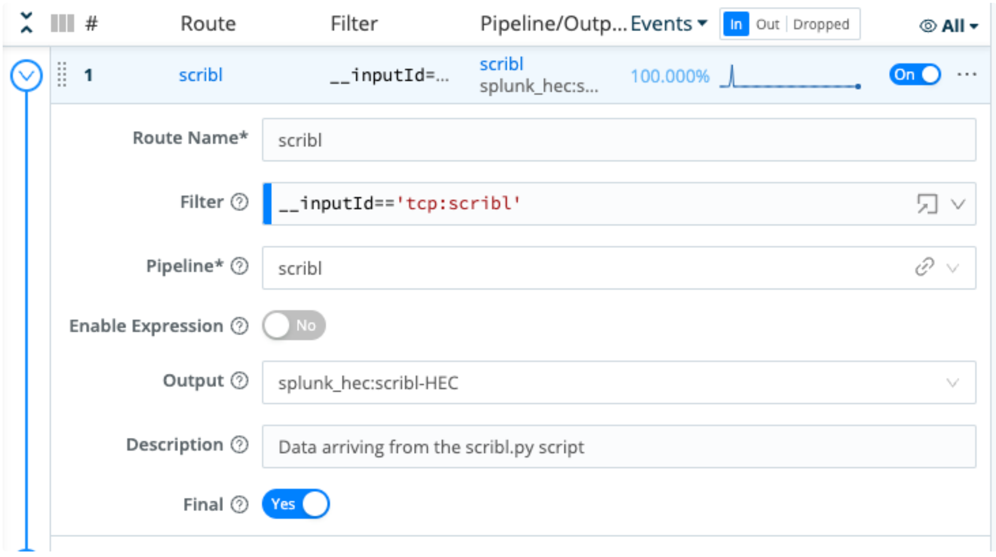
System Default Rule Filter Condition: `true` Event Breaker: `/[\n\r]+(?!\s)/` Timestamp Anchor: `/^/` Timestamp Format: `Auto:150` Default Timezone: `Local` Max Event Bytes: `51200`

+ Add ruleset

Event Breaker buffer timeout

10000

Add a new route and attach a pipeline:



#	Route	Filter	Pipeline/Outp...	Events
1	scribl	__inputId=...	scribl splunk_hec:s...	100.000%

Route Name*: scribl

Filter: `__inputId=='tcp:scribl'`

Pipeline*: scribl

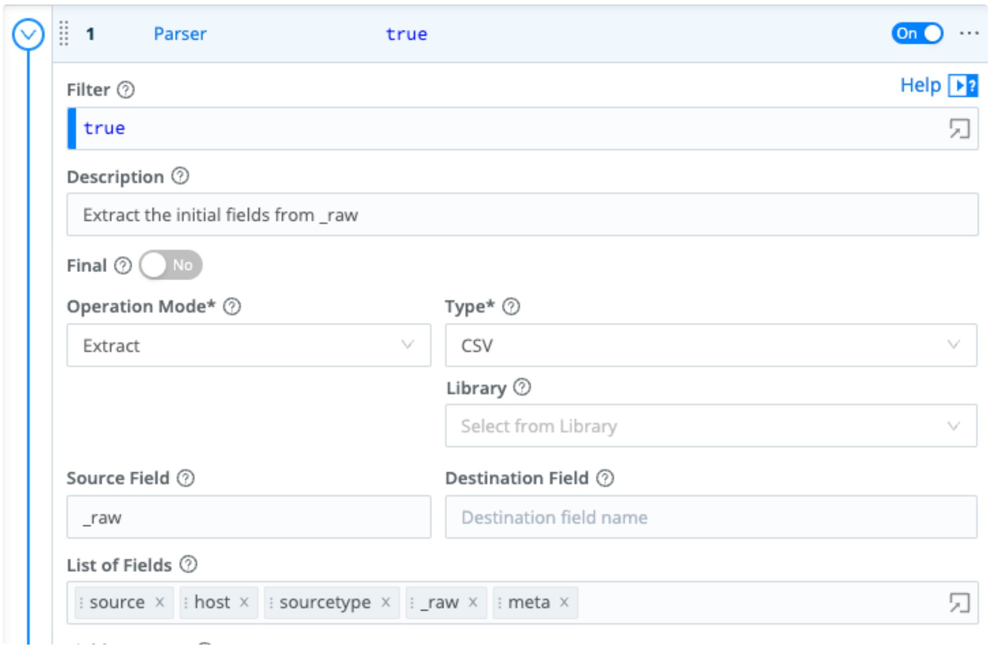
Enable Expression: No

Output: splunk_hec:scribl-HEC

Description: Data arriving from the scribl.py script

Final: Yes

Attach functions to your pipeline to transform the inbound data as detailed below:



1 Parser true On

Filter: true

Description: Extract the initial fields from _raw

Final: No

Operation Mode*: Extract

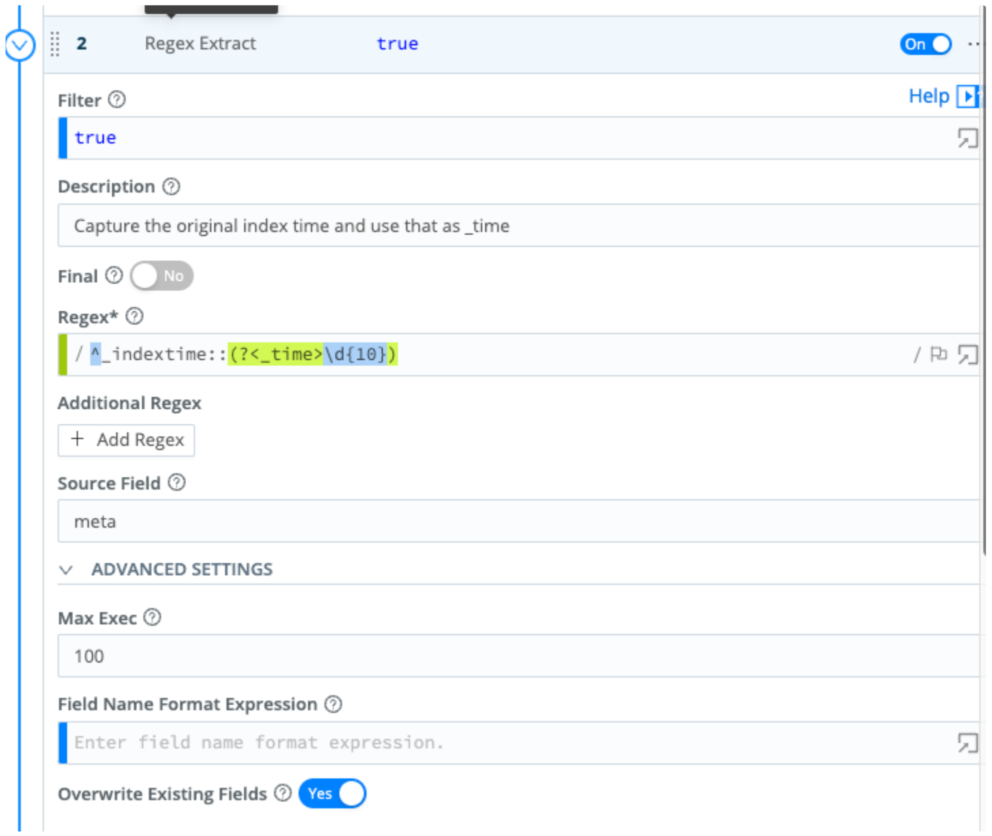
Type*: CSV

Library: Select from Library

Source Field: _raw

Destination Field: Destination field name

List of Fields: source, host, sourcetype, _raw, meta



2 Regex Extract true On

Filter: true

Description: Capture the original index time and use that as _time

Final: No

Regex*:

```
/^_indextime::(?<_time>\d{10})/
```

Additional Regex: + Add Regex

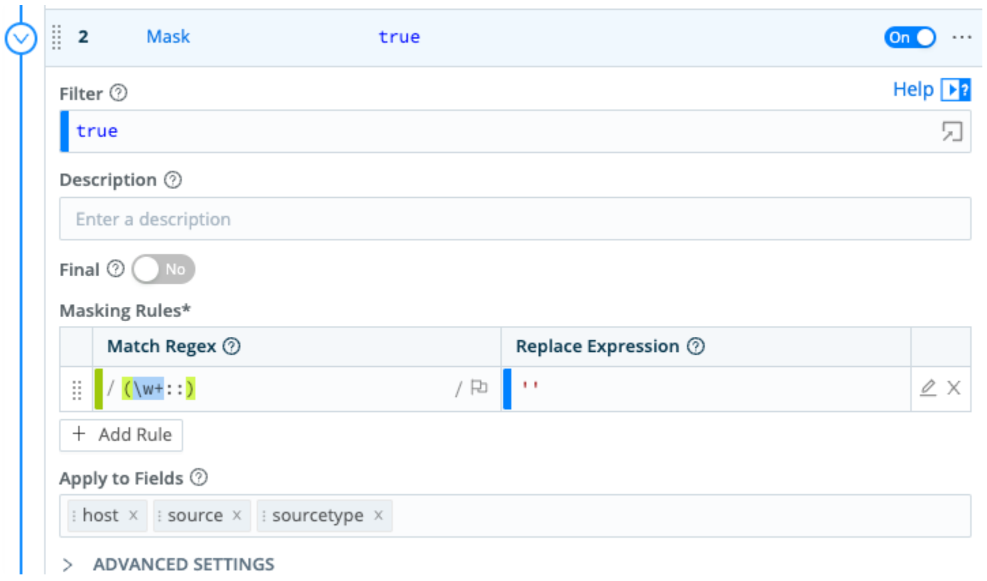
Source Field: meta

ADVANCED SETTINGS

Max Exec: 100

Field Name Format Expression: Enter field name format expression.

Overwrite Existing Fields: Yes


2 Mask true On

Filter: true

Description: Enter a description

Final: No

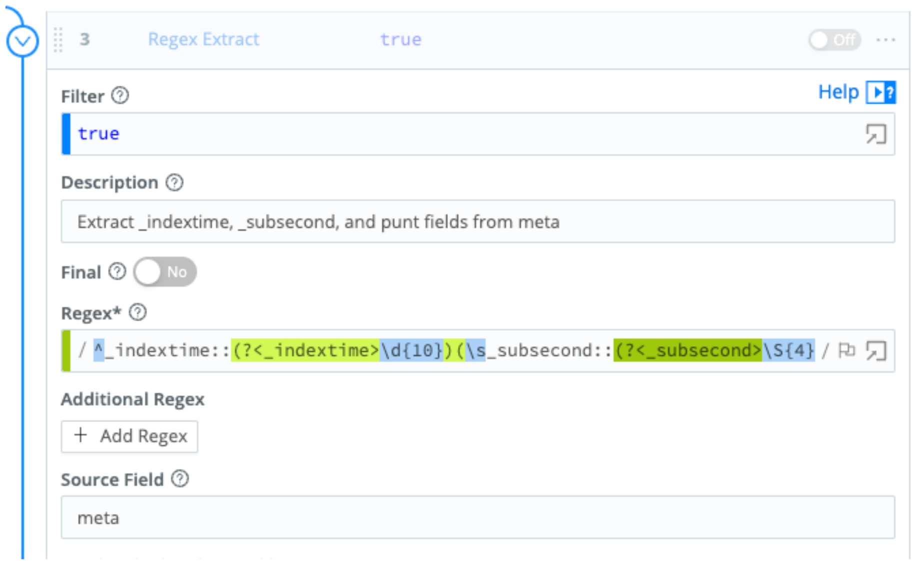
Masking Rules*

Match Regex	Replace Expression
`/(\w+::)/`	`''`

Add Rule

Apply to Fields: host, source, sourcetype

ADVANCED SETTINGS


3 Regex Extract true Off

Filter: true

Description: Extract _indextime, _subsecond, and punt fields from meta

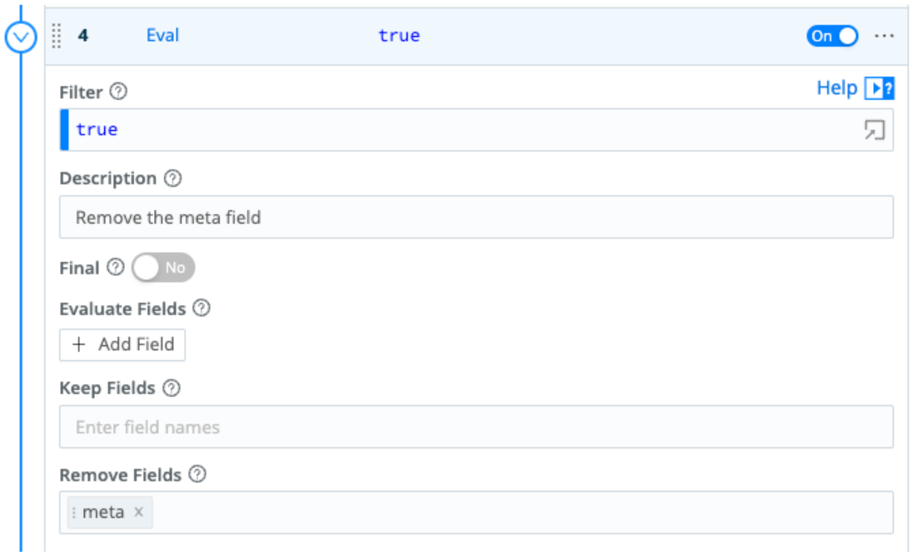
Final: No

Regex*: `/^_indextime::(?<_indextime>\d{10})(\s_subsecond::(?<_subsecond>\S{4}/`

Additional Regex

Add Regex

Source Field: meta


4 Eval true On

Filter: true

Description: Remove the meta field

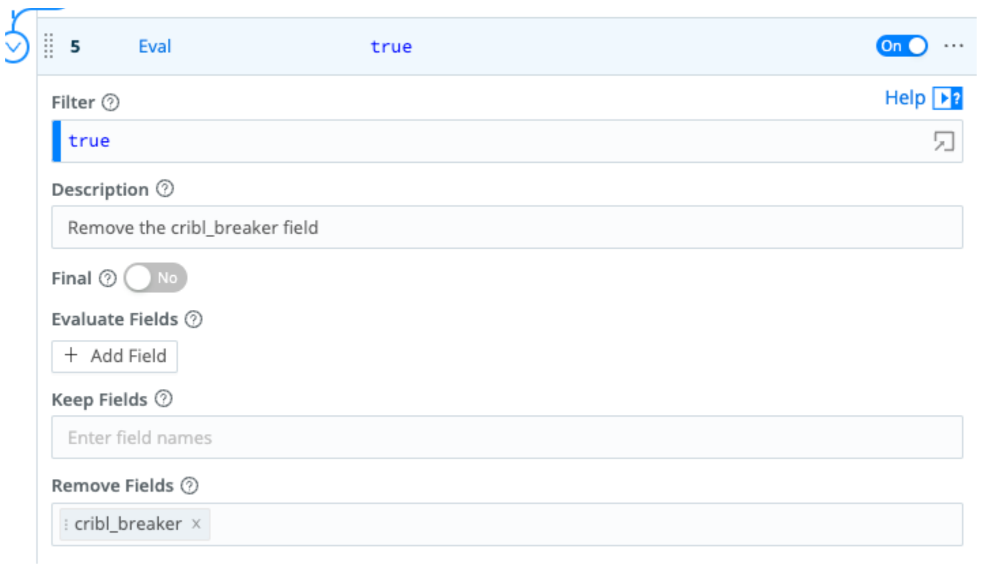
Final: No

Evaluate Fields

Add Field

Keep Fields: Enter field names

Remove Fields: meta



5 Eval true On

Filter: true

Description: Remove the cribl_breaker field

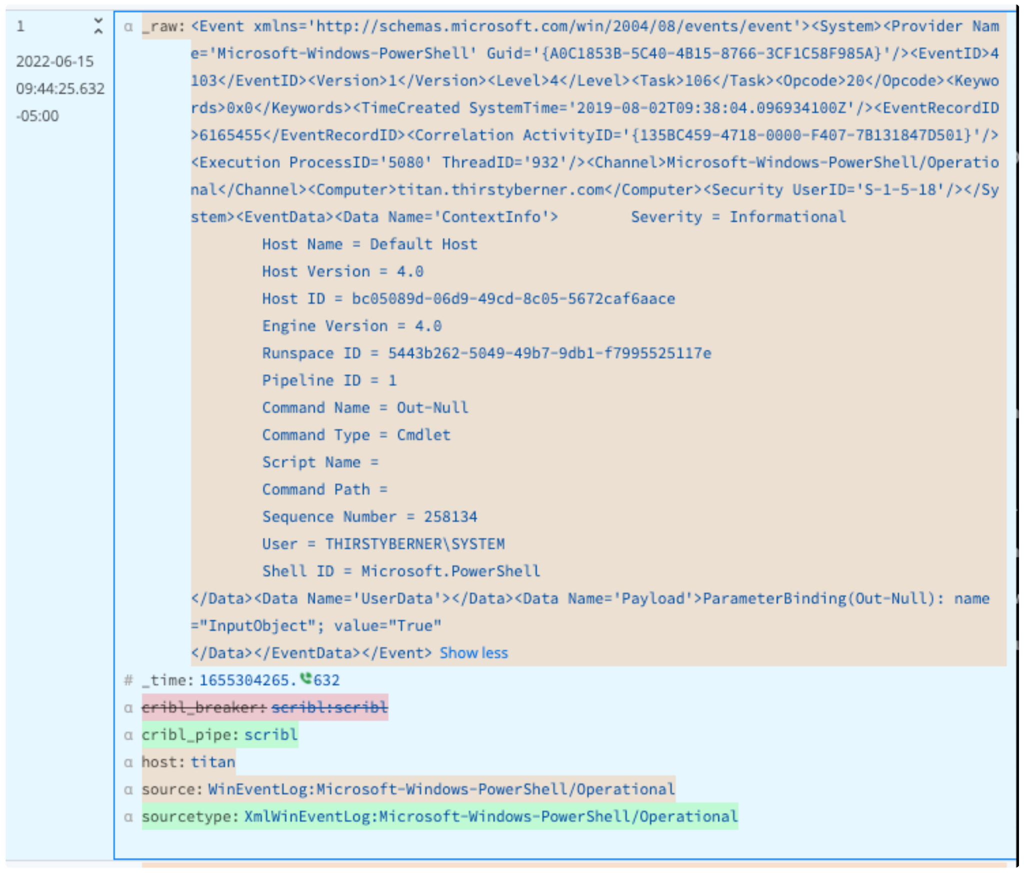
Final: No

Evaluate Fields: + Add Field

Keep Fields: Enter field names

Remove Fields: cribl_breaker

Validate the transformation then attach your destination(s) to the route:



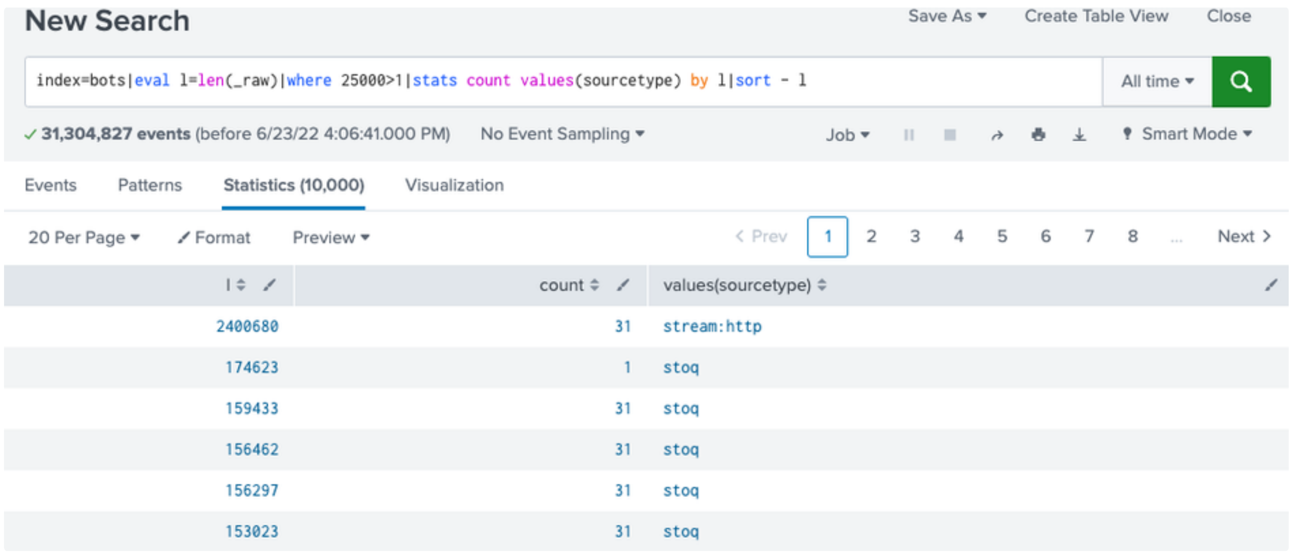
```
1                  _raw: <Event xmlns='http://schemas.microsoft.com/win/2004/08/events/event'><System><Provider Nam
2022-06-15               e='Microsoft-Windows-PowerShell' Guid='{A0C1853B-5C40-4B15-8766-3CF1C58F985A}'/><EventID>4
09:44:25.632             103</EventID><Version>1</Version><Level>4</Level><Task>106</Task><Opcode>20</Opcode><Keywo
-05:00                   rds>0x0</Keywords><TimeCreated SystemTime='2019-08-02T09:38:04.096934100Z'/><EventRecordID
                         >6165455</EventRecordID><Correlation ActivityID='{135BC459-4718-0000-F407-7B131847D501}'/>
                         <Execution ProcessID='5080' ThreadID='932'/><Channel>Microsoft-Windows-PowerShell/Operatio
                         nal</Channel><Computer>titan.thirstyberner.com</Computer><Security UserID='S-1-5-18'/></Sy
                         stem><EventData><Data Name='ContextInfo'>        Severity = Informational
                                 Host Name = Default Host
                                 Host Version = 4.0
                                 Host ID = bc05089d-06d9-49cd-8c05-5672caf6aace
                                 Engine Version = 4.0
                                 Runspace ID = 5443b262-5049-49b7-9db1-f7995525117e
                                 Pipeline ID = 1
                                 Command Name = Out-Null
                                 Command Type = Cmdlet
                                 Script Name =
                                 Command Path =
                                 Sequence Number = 258134
                                 User = THIRSTYBERNER\SYSTEM
                                 Shell ID = Microsoft.PowerShell
                         </Data><Data Name='UserData'></Data><Data Name='Payload'>ParameterBinding(Out-Null): name
                         ="InputObject"; value="True"
                         </Data></EventData></Event> Show less
                   _time: 1655304265.632
                   ~~cribl_breaker: scribl:scribl~~
                   cribl_pipe: scribl
                   host: titan
                   source: WinEventLog:Microsoft-Windows-PowerShell/Operational
                   sourcetype: XmlWinEventLog:Microsoft-Windows-PowerShell/Operational
```

Caveats:

Splunk Event Sizes

You need to pay attention to event sizes in Splunk as it pertains to the Event breaking in Cribl. As noted above in the Event Breaker screenshot, the max event size has a default setting of 51200 bytes. If you use scribl to send events into Cribl Stream larger than that, things break. Either increase your event breaking max event size, use the Cribl Stream Pipeline to drop the large events (example: by sourcetype), or do not use scribl to export the buckets containing the large events.

Here is a quick Splunk search highlighting the large events that need to be dealt with:



```
index=bots|eval l=len(_raw)|where 25000>l|stats count values(sourcetype) by l|sort - l
```

✓ 31,304,827 events (before 6/23/22 4:06:41.000 PM)

l	count	values(sourcetype)
2400680	31	stream:http
174623	1	stoq
159433	31	stoq
156462	31	stoq
156297	31	stoq
153023	31	stoq

Bottlenecks

As mentioned above, the bottleneck you will most likely run into will be bndwidth in your data path or ingest rate at the final destination. Anything you can do to parallelize that final write will pay dividends. For example, you may want to use Cribl Stream's Output Router to write to multiple S3 buckets based on the original Splunk Index or Sourcetype if bandwidth is not your bottleneck.

To Do:

- ~~Add min and max times to determine which buckets should be exported (complete 6/2022)~~
- Applies this to a smartstore config (auth, encryption, data format, etc)
- Sync with Spico and other partners who would be interested
- How about this as a collector?