



ROUTING DATA AROUND THE WORLD WITH STREAM!

Steve Koelpin
Sr. Architect, TransUnion

Don Reilly
Sr. Manager, TransUnion



STEVE KOELPIN
Sr. Architect, TransUnion



DON REILLY
Sr. Architect, TransUnion
CCOE—Admin

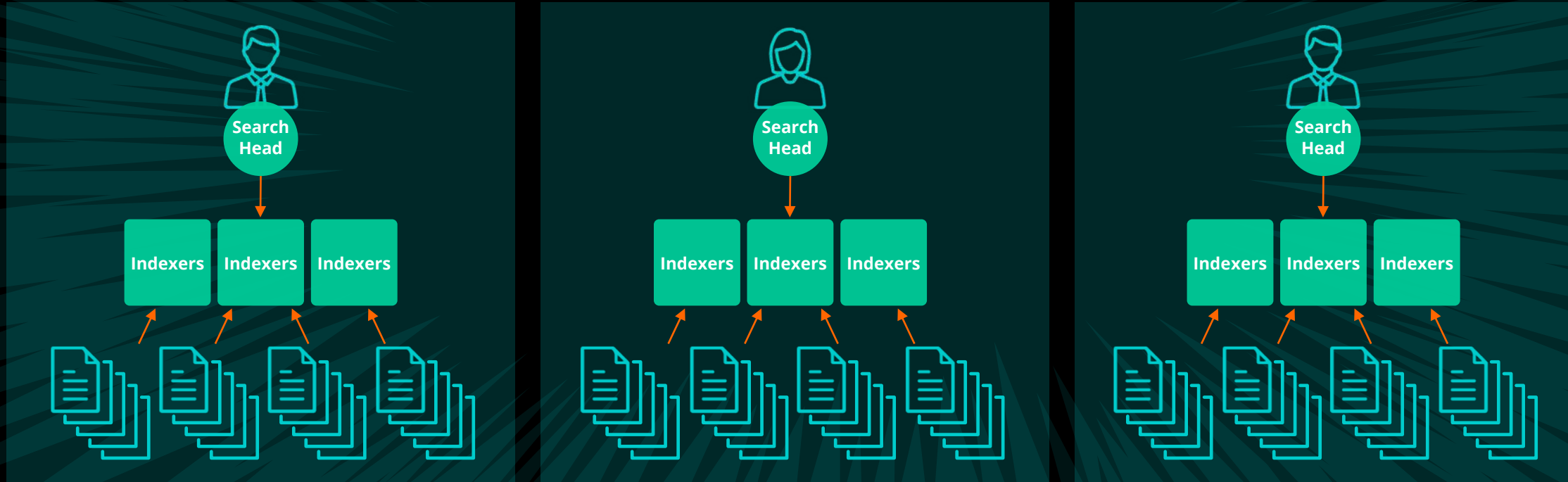




The Content of this presentation is intended for general informational purposes only. TransUnion make no representation or warranty, express or implied, regarding the content of this presentation, including but not limited to warranties as to its accuracy, completeness, ownership, or fitness for a particular purpose. Your reliance on the contents of this presentation is solely at your own risk. TransUnion does not warrant, endorse, or assume liability for any of the content contained in links to third party sites included in this presentation.

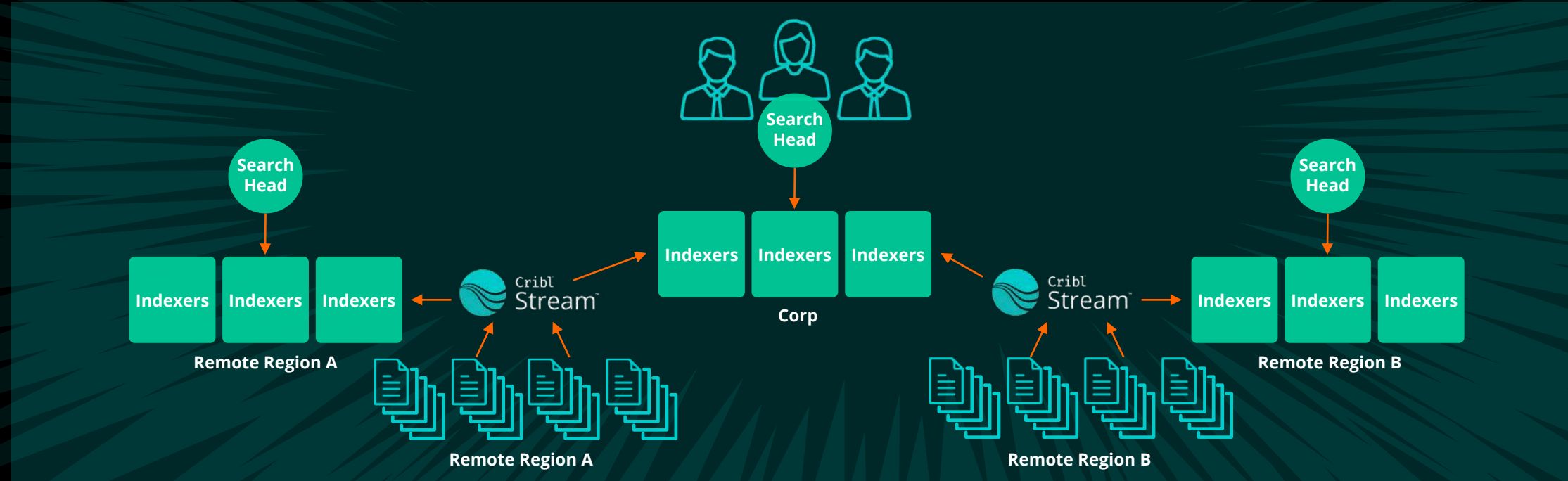
PROBLEM STATEMENT

We operate in **over 30 countries** around the world and lack visibility in the form of a single centralized view for our stakeholders and operators.



SOLUTION

Use Cribl Stream to aggregate and route data from remote regions into a centralized location.







Instead of sending this...



SIZE REDUCTION BY AGGREGATING



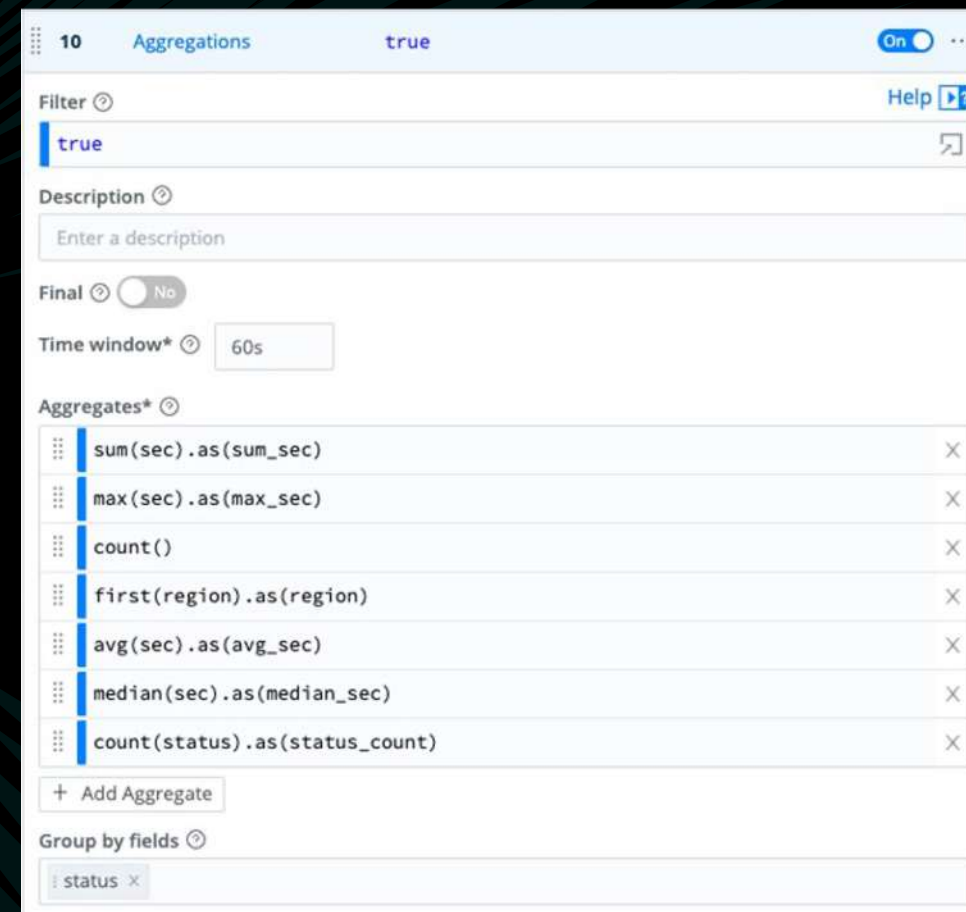
	_raw Length ?	Full Event Length ?	Number of Fields ?	Number of Events ?
IN	73.82KB	120.58KB	19	100
OUT	8.00B	1.38KB	16	4
DIFF	↓ -99.99%	↓ -98.85%	↓ -15.79%	↓ -96.00%

HOW AGGREGATION WORKS

Aggregating in Splunk

```
| bin _time span=60s  
| stats max(resp_time) by _time, status
```

Aggregating in Stream



The screenshot shows the 'Aggregations' configuration page in Splunk. The 'Filter' is set to 'true'. The 'Description' field is empty. The 'Final' toggle is set to 'No'. The 'Time window*' is set to '60s'. The 'Aggregates*' section contains a list of aggregation functions: 'sum(sec).as(sum_sec)', 'max(sec).as(max_sec)', 'count()', 'first(region).as(region)', 'avg(sec).as(avg_sec)', 'median(sec).as(median_sec)', and 'count(status).as(status_count)'. Each function has a small 'x' icon to its right. Below the list is a '+ Add Aggregate' button. The 'Group by fields' section shows 'status' as the selected field.

10 Aggregations true On ...

Filter ⓘ Help ⓘ

true

Description ⓘ

Enter a description

Final ⓘ No

Time window* ⓘ 60s

Aggregates* ⓘ

- sum(sec).as(sum_sec) x
- max(sec).as(max_sec) x
- count() x
- first(region).as(region) x
- avg(sec).as(avg_sec) x
- median(sec).as(median_sec) x
- count(status).as(status_count) x

+ Add Aggregate

Group by fields ⓘ

status x

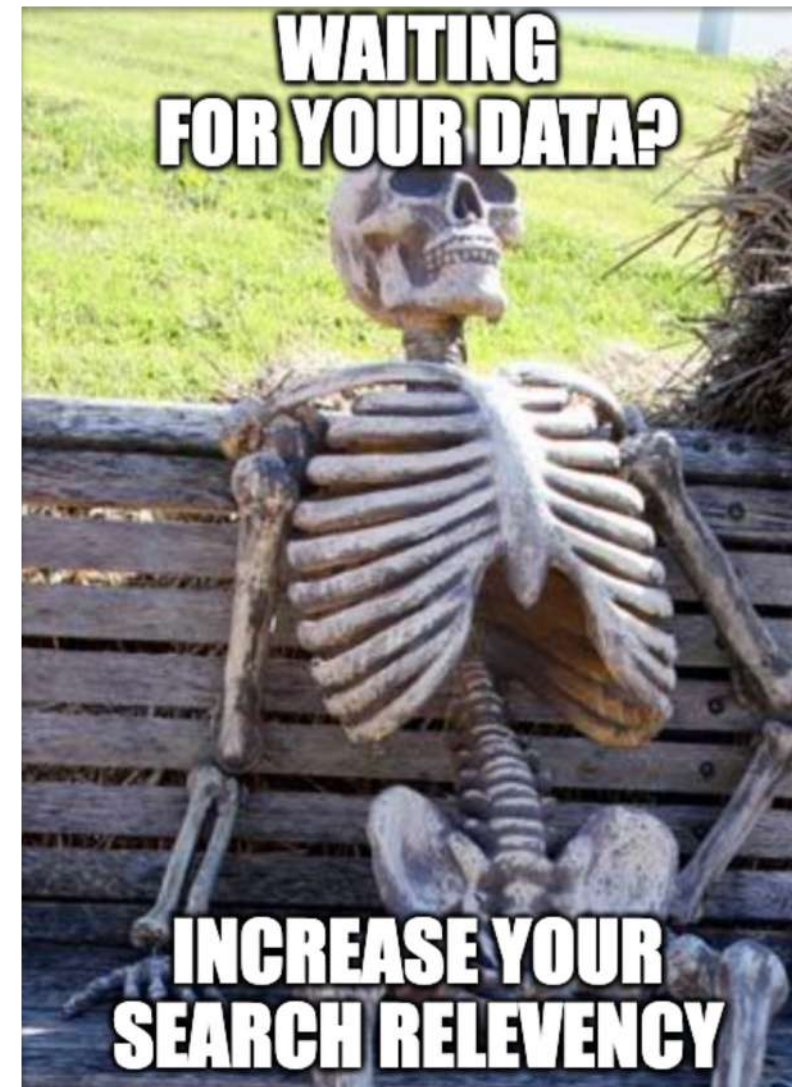
SEARCH RELEVANCE



**Search
Raw
Events?**



**Index
Relevant Data
Points.**



TO PUSH OR TO PULL



Distributed Search Against Data at Rest

Aggregating and Routing Data



PULL



PUSH



PULLING DATA



- Relatively easy to implement
- Don't need to understand format of data upfront
- Better suited for adhoc searching

PROS



- Increased network load
- Impossible to achieve 100% Search Relevance
- Unpredictable search run times

CONS



PUSHING DATA



- Better suited for recurrent searching
- Uniform data structures
- 100% search relevancy
- Predictable search run times
- Reduced network load

PROS



- Requires understanding of data usage
- More planning required for implementation
- Changes will not work retroactively
- Slight increase of Ingest & Storage

CONS



WORKER NODES HAVE THEIR INDEPENDENCE



Problem

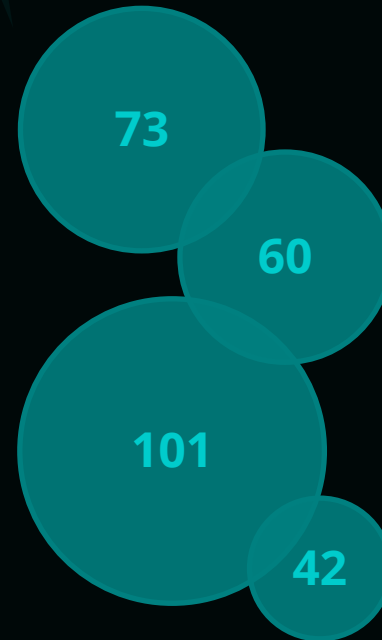
We're aggregating on one of many nodes per region



Solution

Use a second layer of aggregation (i.e., scheduled search) to further aggregate and push to a summary index

Worker Nodes



$$73 + 60 + 101 + 42 =$$

276



Splunk
(Corp)

Average
Formula

=

$$\frac{\text{Total Sum of All Numbers}}{\text{Number of Items in the Set}}$$

DecisionEdge Global Health

Decision Edge Global View

Select App

United States

USA India South Africa UK Hong Kong Philippines Columbia Canada DR

100 100 100 100 100 100 100 100 100

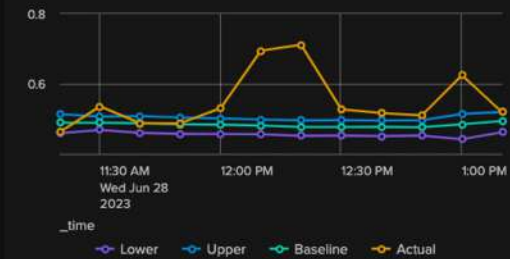
Volume
Response Time
Error Rate

Time
13:29:11

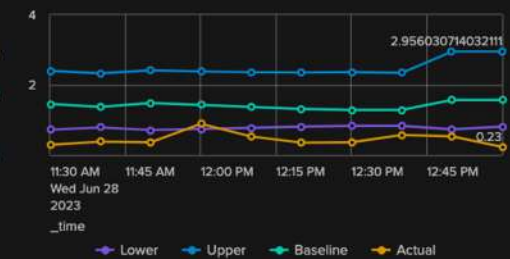
Past Day
Changes
0

Past Day
Incidents
55

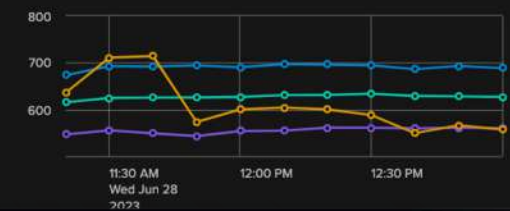
Response Time usa



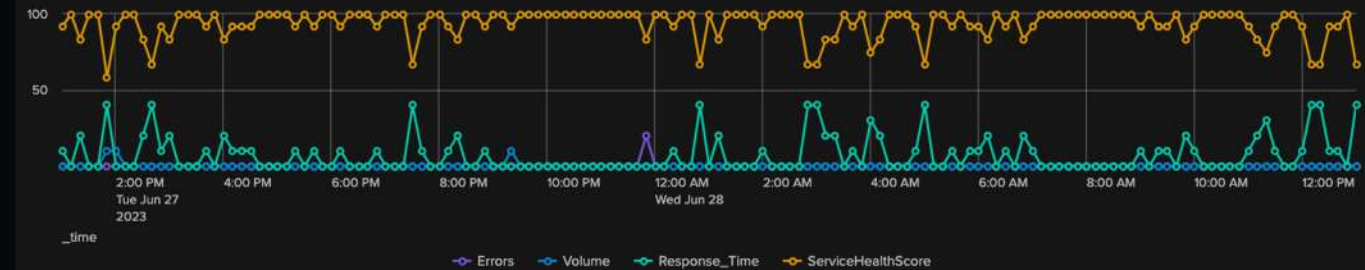
Error Count usa



Volume usa



Trending HealthScores



Notifications

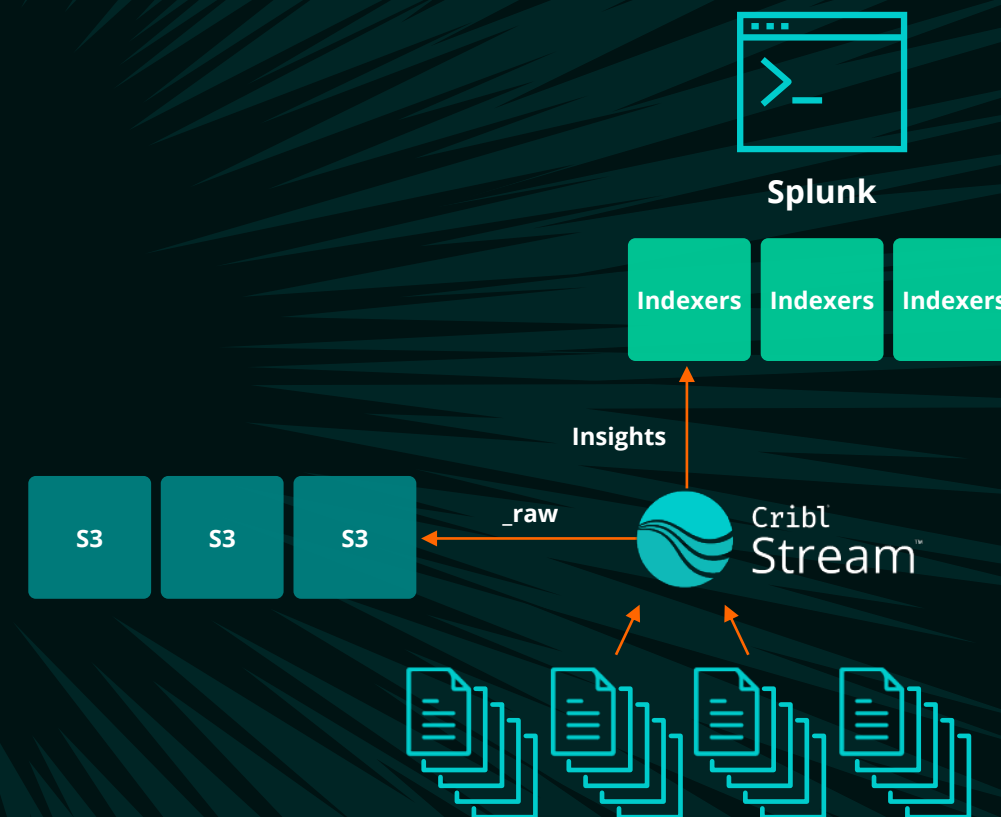
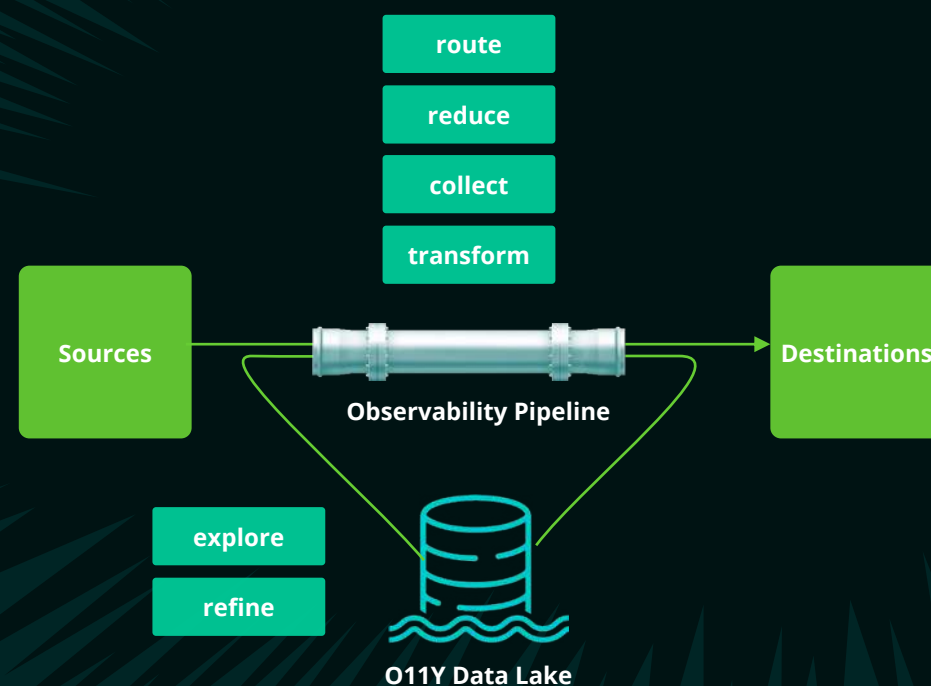
Incident Number	Description	AG	Urgency	TimeOfEvent
1000000001	ServiceHealthScore is low	1000000001	High	10:00 PM
1000000002	ServiceHealthScore is low	1000000002	High	10:00 PM
1000000003	ServiceHealthScore is low	1000000003	High	10:00 PM
1000000004	ServiceHealthScore is low	1000000004	High	10:00 PM

FUTURE STATE

Ingest the insights and store the _raw

- Significantly reduced license costs
- Fast search-time query speeds
- Lower storage costs

Why an Observability Lake





PUSHING DATA



- Better suited for recurrent searching
- Uniform data structures
- 100% search relevancy
- Predictable search run times
- Reduced network load
- Massive decrease of Storage & License!

PROS



- Requires understanding of data usage
- More planning required for implementation
- Changes will not work retroactively
- ~~Slight increase of Ingest & Storage~~

CONS

RESOURCES

Aggregating Data



<https://cribl.io/blog/search-observability-data/>

Cribl Search



<https://cribl.io/blog/search-observability-data/>

O11Y Data Lake



<https://cribl.io/blog/observability-lake/>



**THANK
YOU.**