chronosphere

Three Steps to Using Roll-Up Rules in the Cost Control Cycle

TECHNICAL REPORT

TABLE OF CONTENTS

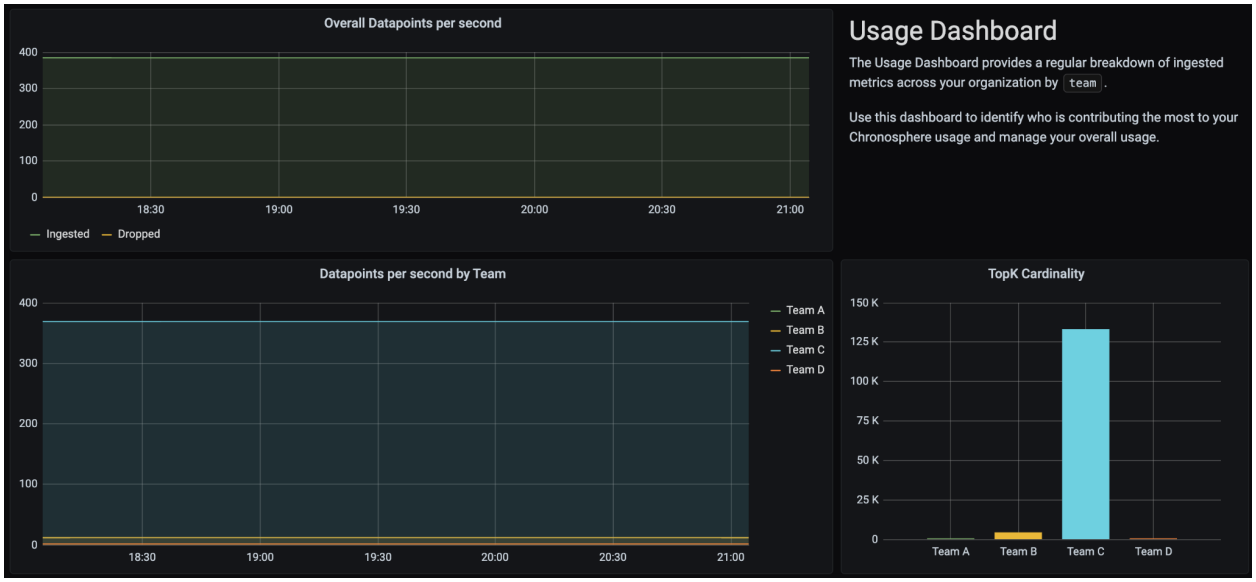
Introduction

Roll-Up rules are an effective way to dynamically control the granularity and cardinality of metric data which directly impacts storage costs. However, Roll-Up rules are only one step of an effective cost control plan, which also include usage measurement, limit enforcement and introspection & understanding.

The ultimate goal isn't for a monitoring system or a central Observability team to control costs for each team, but for each developer to understand the metric data they are emitted, understand the associated cost and then take action on the data by making trade-offs in retention, granularity and cardinality. After all, it's the developers that are both emitting this metric data as well as consuming it.

Follow these three steps to start getting your metrics data growth back under control using Roll-up rules.

Step one: measure usage and enforce limits

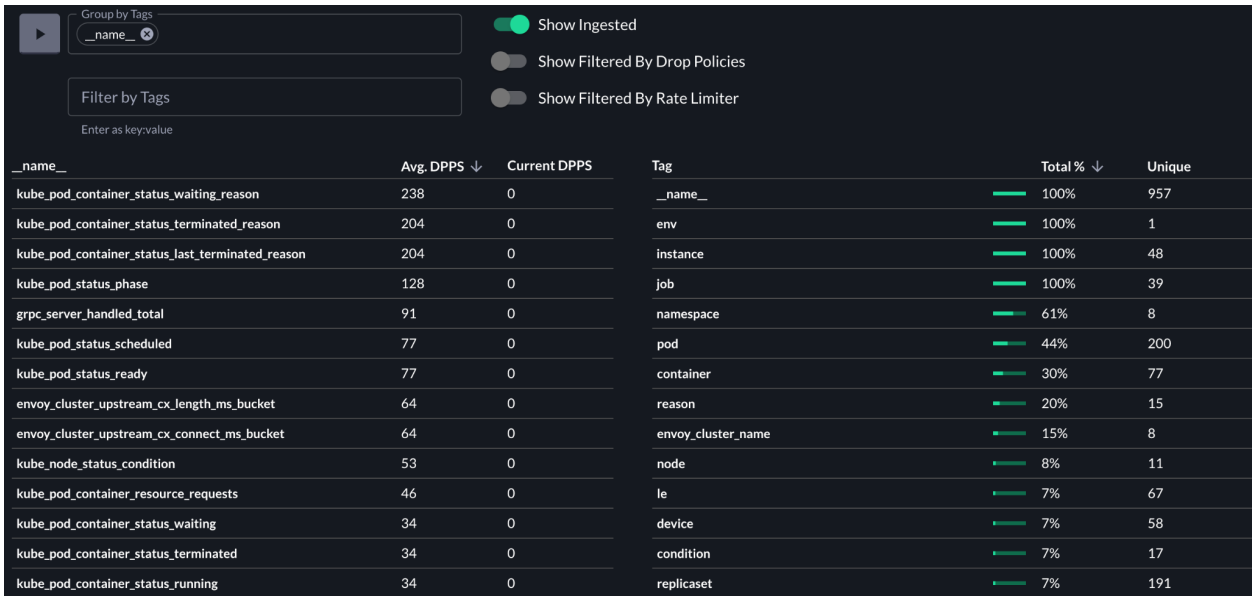


In the Chronosphere platform, we measure and cost-account every incoming datapoint to either a service or a team. We then make this available as metric data inside the platform so that everyone can see what the resources they are consuming on the backend which includes datapoints written per second as well as cardinality.

This metric data can be alerted upon just like any other series, so both the central Observability team and/or the individual team can get alerted if they pass a threshold. Rate limits are also available to ensure there are no overage charges.

Note - a similar dashboard is also available for query resources used which get mapped back to each user who issues a query and measures how much data they are requesting. There is also per user rate limiting such that a single bad user cannot overwhelm or hog resources from other users. This rate limiting simply throttles responses so that users still get their requested data, but just over a longer period of time.

Step two: introspect and understand



Once developers know they are using too much resources, the next step is to help them understand the data they are producing. It's often quite hard to comprehend that a single line of code adding a single label value can result in millions of additional metric time series and datapoint writes per second. There is a tool in the Chronosphere platform called the Profiler which introspects the live stream of data being ingested by the platform and allows the developer to inspect and understand the metrics that are being produced.

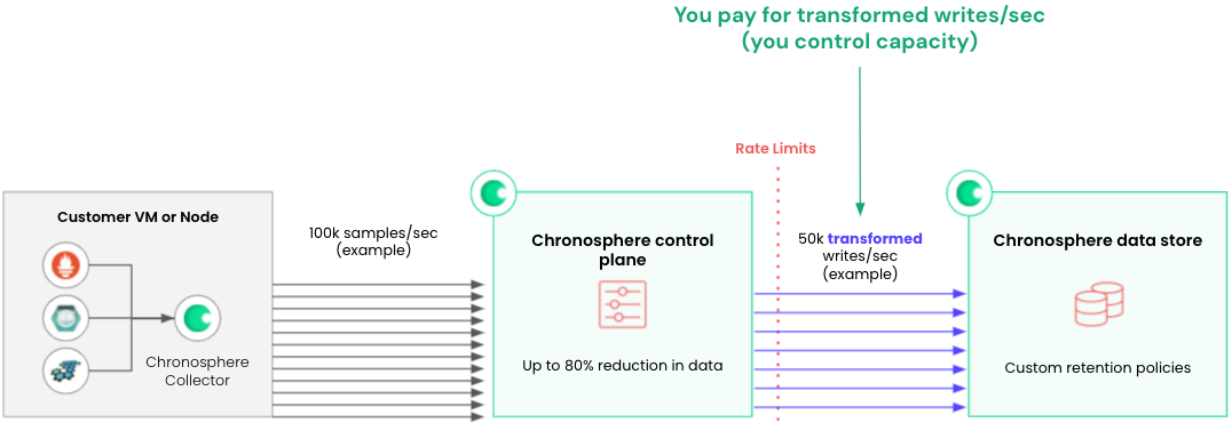
As seen in the above diagram, it's not just a view of all unique metric names, but also the breakdown of each label name, unique label values per label name as well as % of metrics that have the label attached. For example, from the diagram, we can see that there are 48 unique instance label values and 100% of metrics have the instance label. This tells the end user that the presence of the instance label is creating 48X more datapoint writes and cardinality for the data set.

Step three: dynamic aggregation using Roll-Up rules

```
- aggregation: SUM
  filter: __name__:*
  exclude_by:
  - instance
  metric_type: COUNTER
  name: sum counters without instance
  slug: sum_counters_without_instance
  new_metric: '{{ .MetricName }}'
  storage_policies:
  - resolution: 1m
    retention: 120h
  - resolution: 5m
    retention: 1440h
```

As mentioned previously in the document, Roll-Up rules can be created to dynamically aggregate metric data while also modifying the resolution and retention periods of the metric data. They can be scoped for a single metric name, a whole service or team or company wide. The above rule will roll up all metrics across the instance tag - effectively eliminating it and reducing the overall writes per second as well as cardinality by 48X in our example. This will directly reduce the cost of the Chronosphere platform by 48X as well since pricing is on **transformed writes per second** after aggregation, not ingested writes per second.

Note: This rule will also reduce query time for the aggregated data since 48X less data is being fetched. It's always an option to delete the underlying data so if a developer wants to keep all the raw data and only create the aggregate time series, that's supported as well.



Next steps

Ready to learn more about how Chronosphere can help you get back in control of your metrics data? Get in touch by visiting chronosphere.io/demo.