http://thenewstack.io/monitoring-101-collecting-right-data/

What is a metric?

Metrics capture a value pertaining to your systems at a specific point in time.

Metrics are usually collected once per second, one per minute, or at another regular interval to monitor a system over time.

> For different Metric Data Types and the metric names they produce see https://docs.datadoghq.com/developers/metrics/types/?tab=histogram#metric-types

Custom Metrics and High-Cardinality Tags

Datadog's pricing structure is to bill custom metrics at $5/month per 100 custom metrics.

So what constitutes a 'custom metric'?

> A custom metric is uniquely identified by a combination of a metric name and tag values (including the host tag). — [Datadog](https://docs.datadoghq.com/account_management/billing/custom_metrics/)

This means a metric like `request.latency` with no tags would be charged as a single 'custom metric'.

Where as if there were three tags (host, endpoint, status), then with the following combination of unique tag values, we'd end up with four custom metrics:

![custom-metric](https://user-images.githubusercontent.com/180050/80495836-54281480-8960-11ea-9c3c-c9b9498be248.png)

This demonstrates how we must be careful with the cardinality of any tags we apply to our metrics.

The situation is made worse when using a `HISTOGRAM` metric type, as it ultimately produces five separate metrics. Meaning the `request.latency` example, if reported as a `HISTOGRAM`, would result in twenty custom metrics (`5 metrics * 4 tag combinations`).

> Note: the problem of the `HISTOGRAM` metric type can be mitigated using a `DISTRIBUTION` metric instead (see following section).

Metric Types

Overall there are five distinct metric 'types' to be aware of ([docs](https://docs.datadoghq.com/developers/metrics/types/?tab=histogram#metric-types)).

Two of those metric types require additional clarification with regards to the cost implications associated with instrumenting your code to report metric data.

- `HISTOGRAM`
- `DISTRIBUTION`

> **Note**: there is also a `TIMER` metric type, which is a subset of `HISTOGRAM` ([docs](https://docs.datadoghq.com/developers/metrics/dogstatsd_metrics_submission/?tab=python#timer)).

The key differences between `HISTOGRAM` and `DISTRIBUTION` are...

| | `HISTOGRAM` | `DISTRIBUTION` |

| - | - | - |
Multiple Metrics	YES	YES
Percentile Aggregations	PARTIAL	YES
Tag Filtering	NO	YES

Multiple Metrics

The `HISTOGRAM` metric type will generate five unique custom metrics to represent different aggregations:

For example, if you report a histogram metric `foo.bar`, then this will result in the following metrics being created (representing different aggregation types):

- `foo.bar.avg`
- `foo.bar.count`
- `foo.bar.median`
- `foo.bar.max`
- `foo.bar.95percentile`

The `DISTRIBUTION` metric type will generate one metric, but provide multiple aggregations via the Datadog UI.

The aggregations for a `DISTRIBUTION` metric type are:

- `max`
- `min`
- `avg`
- `sum`
- `count`

Now although the `DISTRIBUTION` aggregations may well be applied to a 'single' metric, _internally_ Datadog considers each aggregation a _unique_ metric. This means at face value a `DISTRIBUTION` metric type is no more cost effective than a `HISTOGRAM` with regards to the calculation of ['custom metrics'](#custom-metrics-and-high-cardinality-tags) (as they both ultimately yield five individual metrics). But this isn't necessarily the case, as a `DISTRIBUTION` metric type has the added ability to [filter tags](#tag-filtering) thus reducing the number of calculated custom metrics.

Percentile Aggregations

The `HISTOGRAM` metric type provides a `95percentile`, while the `DISTRIBUTION` metric type _can_ provide a `p95` along with `p50`, `p75`, `p90` and `p99` but these aggregations need to be manually generated via the [Datadog UI](https://buzzfeed.datadoghq.com/metric/distribution_metrics).

Each percentile aggregation for the `DISTRIBUTION` metric type is internally considered a _unique_ metric and thus is subject to Datadog's custom metric cost implications.

Tag Filtering

The `DISTRIBUTION` metric type allows tags to be filtered, thus reducing the potential number of custom metrics Datadog will charge us for. This is not possible with any other metric type.

Choosing `HISTOGRAM` or `DISTRIBUTION` ?

The fact that the `DISTRIBUTION` metric type enables tag filtering is an important consideration when choosing between it and a

`HISTOGRAM`. This is why the `bf_metrics` timer abstraction (which is used to time your functions and/or code) will use the `DISTRIBUTION` metric type rather than Datadog's `TIMER` metric type (which is a subset of a `HISTOGRAM`).

It means that we're able to reduce the number of custom metrics while also allowing consumers to opt-in to percentile aggregations if they require them, and to again utilize tag filtering to help constrain the number of custom metrics.

The `DISTRIBUTION` and `HISTOGRAM` have overlapping aggregations (`count`, `avg`, `max`) which means if you do not require an aggregation outside of those specific ones, then choosing a `DISTRIBUTION` metric type would be better as you can utilize tag filtering to help reduce the number of custom metrics.

If you do require a percentile aggregation then the trade-off you need to make is between whether a `HISTOGRAM` (with `95percentile` available by default) is more cost effective than a `DISTRIBUTION` which with percentiles added will add up to more individual metrics but with tag filtering available might still end up being more cost effective overall as you can't filter your high-cardinality tags with a `HISTOGRAM`/`TIMER`.

The `DISTRIBUTION` percentile 'custom metric' conspiracy

The way Datadog calculates the number of 'custom metrics' is slightly different (and more costly) for percentile aggregations of a `DISTRIBUTION` metric type.

The `DISTRIBUTION` percentiles take into account every _potentially_ queryable varation of a metric.

Imagine your metric has three tags A, B and C. Datadog calculates the number of custom metrics like so:

- each tag value of {A}
- each tag value of {B}
- each tag value of {C}
- each tag value of {A,B}
- each tag value of {A,C}
- each tag value of {B,C}
- each tag value of {A,B,C}
- {*}

Datadog's rationale for this difference is...

> The reason we have to store percentiles for each potentially queryable tag value combination is to preserve mathematical accuracy of your values; unlike the non-percentile aggregations which can be mathematically accurate when reaggregated (i.e the global max is the maximum of the maxes), you can't calculate the globally accurate p95 by recombining p95s.

---

https://docs.datadoghq.com/developers/dogstatsd/data_types/

- Counter: track how many times something happens per N (N: seconds or minutes)
- Gauge: "last write wins"
- Histogram: when you want to understand the value over time.
- Distribution: server-side aggregation (+ allows percentile generation for only specific tags).

> **Note**: When thinking about 'guages': imagine many processes sending a gauge for a moment in time. Which one do you record? A histogram can correctly record the distribution of those values.

Warning about Custom Metrics

> Distribution metrics with percentile aggregations (p50, p75, p90, p95, p99) generate custom metrics or timeseries differently than gauges, counts, histograms, and distributions with nonpercentile aggregations (sum, count, min, max, avg). Because percentiles aren’t reaggregatable, Datadog preserves five timeseries for every potentially queryable tag combination. -- Datadog Docs

Ultimately this means that if you used a normal timing metric (which is a sub-set of a histogram) then you'd get five custom metrics created (and for each one you'd have to be mindful of any tags as that will increase the custom metric numbers as we mentioned earlier), but with a distribution metric you're able to have it ignore all tags except for specific tags you specify which can help reduce the number of metrics.

- **Timer metric** (a sub-set of Histogram) generates five custom metrics: `count`, `avg`, `median`, `max`, `95percentile` (each will then be multiplied by the number of unique tag combinations).
- **Distribution metric** generates five custom metrics (in its default mode, i.e. non-percentile mode): `max`, `min`, `avg`, `sum`, `count` (each will then be multiplied by the number of unique tag combinations). If we calculate percentiles for the distribution metric this results in an additional five custom metrics for the percentiles: `p50`, `p75`, `p90`, `p95`, `p99` (again, each will then be multiplied by the number of unique tag combinations).

My understanding is that if we recorded a metric called `foo.bar`, then...

- `foo.bar` is itself considered a **_single_** custom metric (multiplied by any available 'tags')
- while the percentile aggregation will generate five **_additional_** custom metrics (again, multiplied by any available 'tags')

What should we check?

For a particular 'work metric' you would check the corresponding 'resource metric'.

Work Metrics	Resource Metrics	Events
Throughput	Utilization	Code Changes
Success	Saturation	Alerts
Error	Error	Scaling Events
Performance	Availability	...

Work metrics indicate the top-level health of your system by measuring its useful output

Resource metrics (CPU, memory, disks, and network interfaces) are especially valuable for investigation and diagnosis of resource problems

There are a few other types of metrics that are neither work nor resource metrics, but that nonetheless may come in handy in diagnosing causes of problems. Common examples include counts of cache hits or database locks

In addition to metrics, which are collected more or less continuously, some monitoring systems can also capture events: discrete, infrequent occurrences that can provide crucial context for understanding what changed in your system’s behavior