Metrics

The wiki pages are not used for documentation any more. Please visit http://bookkeeper.apache.org/docs/latest/admin/metrics/ for the documentation about metrics collection.

“You cannot manage what you do not measure” says the old adage. Which also applies to software we write and use. This is why BookKeeper instruments its operations and exposes various metrics for latency, throughput and capacity.

Exposed metrics

By default, in version 4.2.2, the metrics exposed are:

- latencies (time per op) / throughput (ops / s):
  - journal write
  - ledger creation / opening / deletion
  - add and read latency in the bookkeeper client
    - per quorum
    - per bookie
- counters
  - ensemble change and reconfiguration
  - fencing
- gauges
  - pending operations

It is easy to add metrics in the codebase or in the client application, by using the metrics service provider interface.

Metrics service providers

There are many libraries available in Java for reporting and aggregating metrics, therefore we provide a generic SPI (Service Provider Interface) and various pluggable implementations.

The generic SPI can be found in module “bookkeeper-stats”. It offers interfaces for manipulating the concepts of `Counter`, `Gauge`, and statistics (`OpStatsLogger`).

Metrics service providers can gather and aggregate metrics in various ways, depending on specific needs and ecosystem.

We provide 3 providers for the metrics service:

- codahale (http://metrics.codahale.com/) in module codahale-provider
- twitter ostrich (https://github.com/twitter/ostrich) in module twitter-ostrich
- twitter common stats (https://github.com/twitter/commons/tree/master/src/java/com/twitter/common/stats) in module twitter-science

Configuration

What is mandatory is the class of the stats provider (statsProviderClass parameter). We must specify it in:

- the bookie configuration
- the client application configuration

For instance, for using codahale metrics provider:

```
statsProviderClass=org.apache.bookkeeper.stats.CodahaleMetricsProvider
```

Metrics providers usually expose the collected metrics through either JMX, log files, or by appending to graphing systems such as graphite (http://graphite.wikidot.com/). How this is configured is specific to each provider.

Codahale metrics provider configuration

Here are the configuration parameters for the codahale metrics provider. To include in the bookie and / or bookkeeper client configuration files. We can report to graphite, csv files or slf4j loggers, non-exclusively.

```
# we must specify the metrics provider class
statsProviderClass=org.apache.bookkeeper.stats.CodahaleMetricsProvider
# output frequency, default is 60s
codahaleStatsOutputFrequencySeconds=60
# prefix for defining a scope for the bookie or bookkeeper client metrics, e.g. bookie
codahaleStatsPrefix=bookie
# graphite endpoint
codahaleStatsGraphiteEndpoint=serverA:port12
# directory for appending metrics in csv files
codahaleStatsCSVEndpoint=myDir
# slf4j logger for dumping logs to the console or some file
codahaleStatsSlf4jEndpoint=myLogger
```