OpenStack Telemetry

Rethinking Ceilometer metric storage with Gnocchi: Time-series as a Service

Eoghan Glynn
eglynn@redhat.com
eglynn@Freenode

Julien Danjou
julien@danjou.info
jd__@Freenode

Dina Belova
dbelova@mirantis.com
DinaBelova@Freenode

Speakers

Eoghan Glynn - current Ceilometer PTL, is defining overall vector of project development

Julien Danjou - core Ceilometer contributor from the outset, is driving Gnocchi initiative

Dina Belova - Ceilometer contributor, joined core team during Juno cycle

Once upon a time, Ceilometer…

Let’s meter OpenStack!

What do we store?

How do we do store that?

Let’s store everything in some DB,

we’ll figure out later!

How Ceilometer could have been better

- Flexible and heavy Ceilometer samples model with ***free-form metadata***
- Scalability API issue (***queries ~ O(n)***, where n = number of samples)
- No API consistency
- API cannot solve all problems efficiently

Gnocchi to the rescue

- Let’s track **resources**
- Let’s ***separate events and metrics*** concepts!
  - Metric = ***Time-Series Data***
- Let’s link **resources** and **metrics**
- Let's do **aggregation** eagerly

Compare and contrast ...

“classic” Ceilometer	Gnocchi
Heavy-weight samples with embedded metadata	Light-weight time-series shorn of metadata
Global data expiry policy set across the board	Per time-series configurable retention policies

Compare and contrast ...

"classic" Ceilometer	Gnocchi
On-demand aggregation	Eager pre-aggregation
Intertwined storage of resources and samples	Separated storage and data models for resources & time-series data

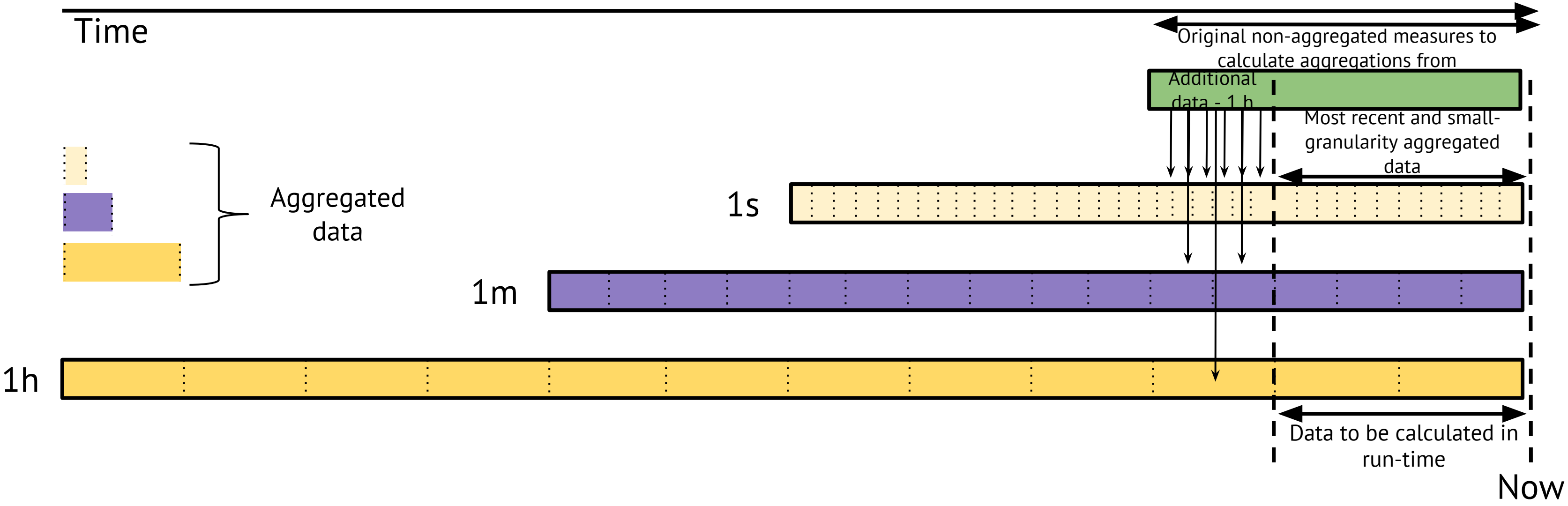
Gnocchi basics

- ***Resource*** = cloud resource (instance, volume, etc.)
- ***Entity*** = anything you'd like to collect data about
  - identified by UUID, or by name combined with resource ID
- ***Measure*** = *(timestamp, value)* time-series datapoint

Gnocchi basics

- ***Archive policy*** = data storage policy defined by admin
  - 1 second resolution over a day, 1 hour resolution over a year, or even both
  - Consists of ***granularity*** (in seconds) and ***retention time-span***
- ***Aggregation*** = function used to roll up data
- ***Retention*** = do not store fine grained data forever, instead store aggregated data according to the *per-entity* archive policies

Gnocchi aggregation mechanism


Time
Original non-aggregated measures to calculate aggregations from
Additional data - 1 h
Most recent and small-granularity aggregated data
Aggregated data
1s
1m
1h
Data to be calculated in run-time
Now

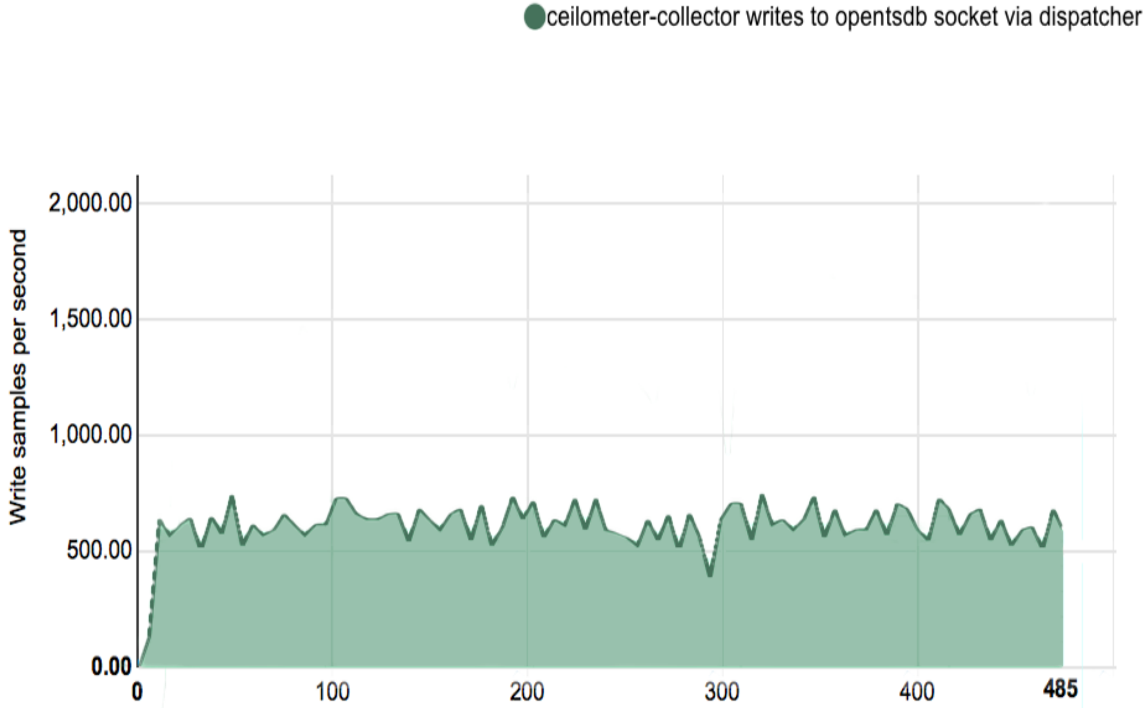
Gnocchi Indexer concept

- Measuring different entities is the main concept of Gnocchi
- Although, entities have no actual use without entity-resource association

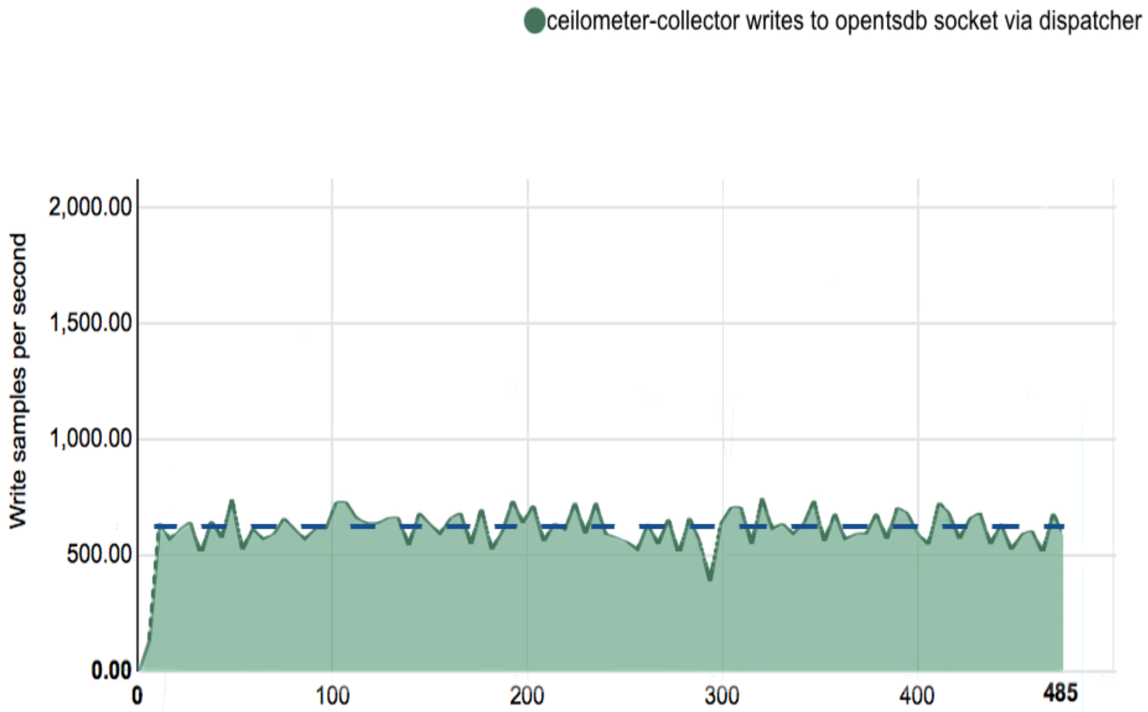
Gnocchi Indexer concept

- Gnocchi indexer is responsible for indexing entities, resources, and linking them together
- Resources and their attributes are well-defined, typed, and indexed
  - The generic type can be used if the resource type is unknown to Gnocchi

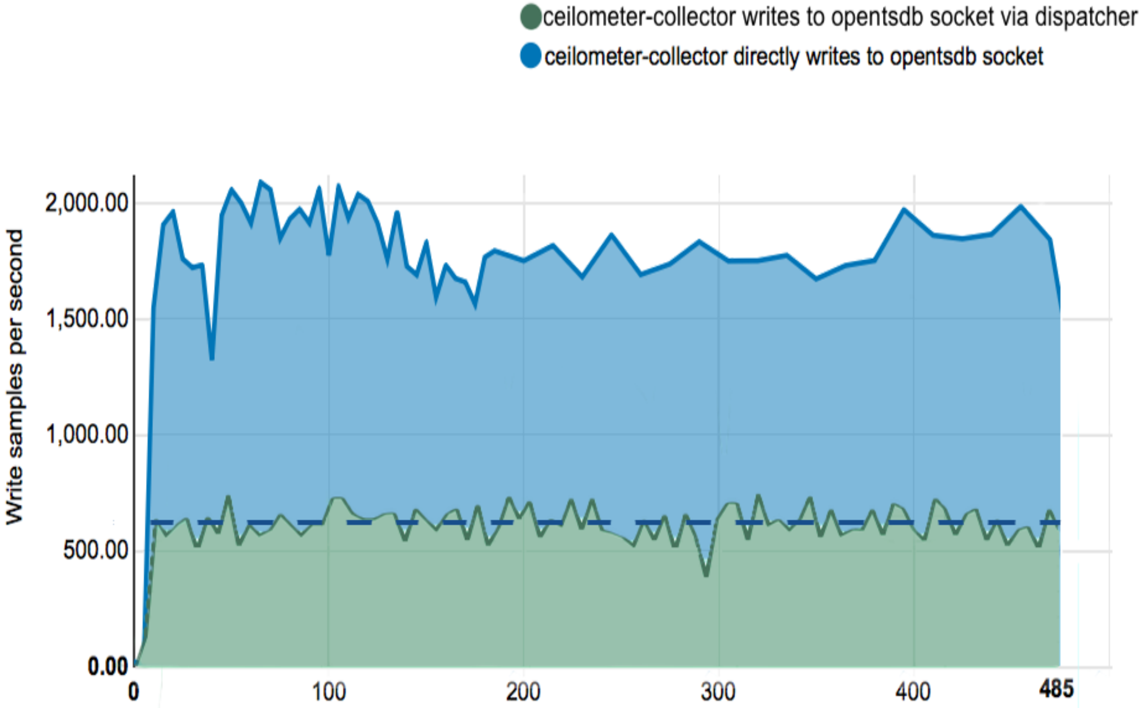
Gnocchi performance



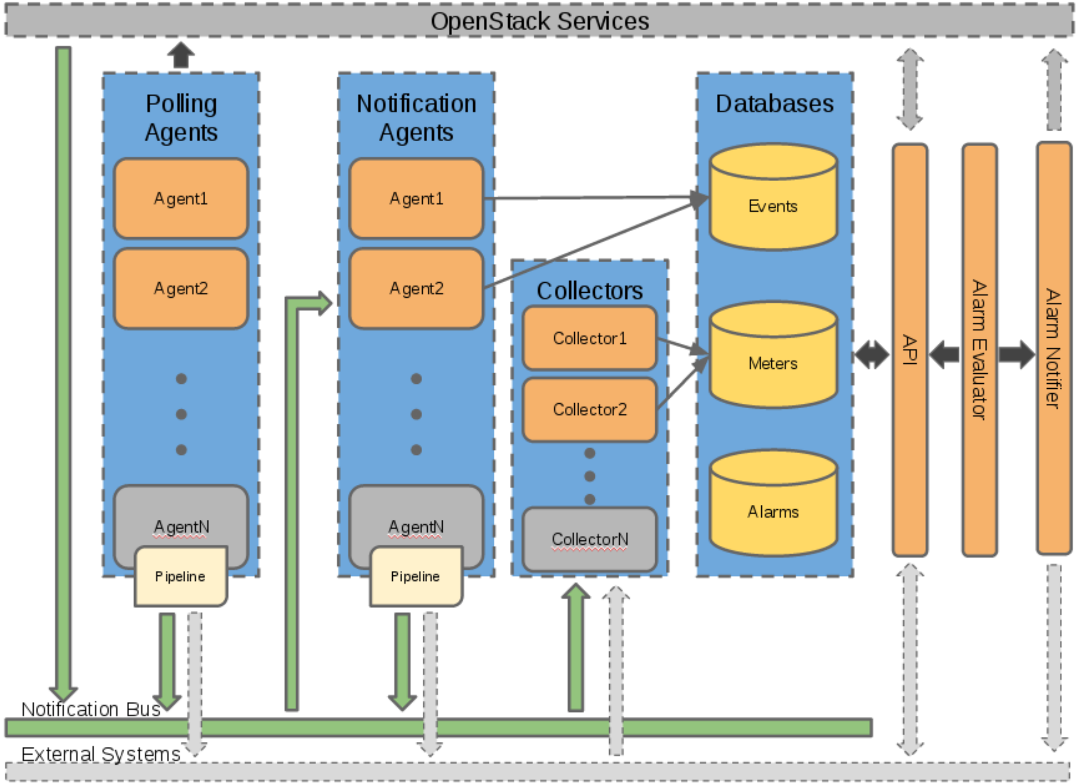
Gnocchi performance



Gnocchi performance



Quick refresher on existing infrastructure



Ceilometer + Gnocchi integration

- Future plans:
  - transition Gnocchi API to Ceilometer API v3
  - move TSDB interaction to Ceilometer collector
- Steps we’re doing right now:
  - integrate Gnocchi as separated piece of code inside Ceilometer workflow via Ceilometer Database Dispatcher mechanism

Covering existing ceilometer use-cases

- Alarming to drive Heat autoscaling, based on aggregating samples across all instances with matching metadata
  - ***use cross-entity aggregation based on strongly-typed resource attributes, as opposed to free-from metadata***
- Reconstructing the resource state timeline, from per-sample resource metadata
  - ***use queries over relatively infrequent events capturing state transitions***

No, we’re not re-inventing TSDB here

- Existing specialized metrics-oriented DBs can be leveraged by Gnocchi’s pluggable driver model
    - ***actively working on drivers for InfluxDB and OpenTSDB***
- Gnocchi itself provides a canonical storage driver based on Pandas and Swift
- In the specialized TSBD use-case, Gnocchi manages the resource-entity association & abstract archive policy concepts

Forward-looking questions to be resolved

- Consolidation of Gnocchi and Ceilometer codebases and core teams
- Migration of pre-existing datastores built up using classic Ceilometer
- Duration of deprecation path for the Ceilometer v2 API

Useful links

- https://wiki.openstack.org/wiki/Gnocchi
- http://bit.ly/gnocchi_blog
- http://bit.ly/gnocchi_docs
- http://bit.ly/gnocchi_changes

openstack™

CLOUD SOFTWARE