PROMETHEUS PLUGIN

Provides a Prometheus exporter to pass on Ceph performance counters from the collection point in ceph-mgr. Ceph-mgr receives MMgrReport messages from all MgrClient processes (mons and OSDs, for instance) with performance counter schema data and actual counter data, and keeps a circular buffer of the last N samples. This plugin creates an HTTP endpoint (like all Prometheus exporters) and retrieves the latest sample of every counter when polled (or "scraped" in Prometheus terminology). The HTTP path and query parameters are ignored; all extant counters for all reporting entities are returned in text exposition format. (See the Prometheus documentation.)

ENABLING PROMETHEUS OUTPUT

The *prometheus* module is enabled with:

```
ceph mgr module enable prometheus
```

CONFIGURATION

By default the module will accept HTTP requests on port 9283 on all IPv4 and IPv6 addresses on the host. The port and listen address are both configurable with `ceph config-key set`, with keys `mgr/prometheus/server_addr` and `mgr/prometheus/server_port`. This port is registered with Prometheus's registry.

STATISTIC NAMES AND LABELS

The names of the stats are exactly as Ceph names them, with illegal characters `.`, `-` and `::` translated to `_`, and `ceph_` prefixed to all names.

All *daemon* statistics have a `ceph_daemon` label such as "osd.123" that identifies the type and ID of the daemon they come from. Some statistics can come from different types of daemon, so when querying e.g. an OSD's RocksDB stats, you would probably want to filter on `ceph_daemon` starting with "osd" to avoid mixing in the monitor rocksdb stats.

The *cluster* statistics (i.e. those global to the Ceph cluster) have labels appropriate to what they report on. For example, metrics relating to pools have a `pool_id` label.

POOL AND OSD METADATA SERIES

Special series are output to enable displaying and querying on certain metadata fields.

Pools have a `ceph_pool_metadata` field like this:

```
ceph_pool_metadata{pool_id="2",name="cephfs_metadata_a"} 0.0
```

OSDs have a `ceph_osd_metadata` field like this:

```
ceph_osd_metadata{cluster_addr="172.21.9.34:6802/19096",device_class="ssd",id="0",public_addr
```

CORRELATING DRIVE STATISTICS WITH NODE_EXPORTER

The prometheus output from Ceph is designed to be used in conjunction with the generic host monitoring from the Prometheus node_exporter.

To enable correlation of Ceph OSD statistics with node_exporter's drive statistics, special series are output like this:

```
ceph_disk_occupation{ceph_daemon="osd.0",device="sdd",instance="myhost",job="ceph"}
```

To use this to get disk statistics by OSD ID, use the and on syntax in your prometheus query like this:

```
rate(node_disk_bytes_written[30s]) and on (device,instance) ceph_disk_occupation{ceph_daemon=
```

See the prometheus documentation for more information about constructing queries.

Note that for this mechanism to work, Ceph and node_exporter must agree about the values of the `instance` label. See the following section for guidance about to to set up Prometheus in a way that sets `instance` properly.

CONFIGURING PROMETHEUS SERVER

See the prometheus documentation for full details of how to add scrape endpoints: the notes in this section are tips on how to configure Prometheus to capture the Ceph statistics in the most usefully-labelled form.

This configuration is necessary because Ceph is reporting metrics from many hosts and services via a single endpoint, and some metrics that relate to no physical host (such as pool statistics).

HONOR_LABELS

To enable Ceph to output properly-labelled data relating to any host, use the `honor_labels` setting when adding the ceph-mgr endpoints to your prometheus configuration.

Without this setting, any `instance` labels that Ceph outputs, such as those in `ceph_disk_occupation` series, will be overridden by Prometheus.

CEPH INSTANCE LABEL

By default, Prometheus applies an `instance` label that includes the hostname and port of the endpoint that the series game from. Because Ceph clusters have multiple manager daemons, this results in an `instance` label that changes spuriously when the active manager daemon changes.

Set a custom `instance` label in your Prometheus target configuration: you might wish to set it to the hostname of your first monitor, or something completely arbitrary like “ceph_cluster”.

NODE_EXPORTER INSTANCE LABELS

Set your `instance` labels to match what appears in Ceph’s OSD metadata in the `hostname` field. This is generally the short hostname of the node.

This is only necessary if you want to correlate Ceph stats with host stats, but you may find it useful to do it in all cases in case you want to do the correlation in the future.

EXAMPLE CONFIGURATION

This example shows a single node configuration running ceph-mgr and node_exporter on a server called `senta04`.

This is just an example: there are other ways to configure prometheus scrape targets and label rewrite rules.

PROMETHEUS.YML

```
global:
  scrape_interval:     15s
  evaluation_interval: 15s

scrape_configs:
  - job_name: 'node'
    file_sd_configs:
      - files:
        - node_targets.yml
  - job_name: 'ceph'
    honor_labels: true
```

```
    file_sd_configs:
      - files:
        - ceph_targets.yml
```

CEPH_TARGETS.YML

```
[
    {
        "targets": [ "senta04.mydomain.com:9283" ],
        "labels": {
            "instance": "ceph_cluster"
        }
    }
]
```

NODE_TARGETS.YML

```
[
    {
        "targets": [ "senta04.mydomain.com:9100" ],
        "labels": {
            "instance": "senta04"
        }
    }
]
```

NOTES

Counters and gauges are exported; currently histograms and long-running averages are not. It’s possible that Ceph’s 2-D histograms could be reduced to two separate 1-D histograms, and that long-running averages could be exported as Prometheus’ Summary type.

Timestamps, as with many Prometheus exporters, are established by the server’s scrape time (Prometheus expects that it is polling the actual counter process synchronously). It is possible to supply a timestamp along with the stat report, but the Prometheus team strongly advises against this. This means that timestamps will be delayed by an unpredictable amount; it’s not clear if this will be problematic, but it’s worth knowing about.