Ceph Deployment and Monitoring at Lancaster

GridP47, 23 March 2022

Gerard Hand, Steven Simpson, Matt Doidge

Motivation

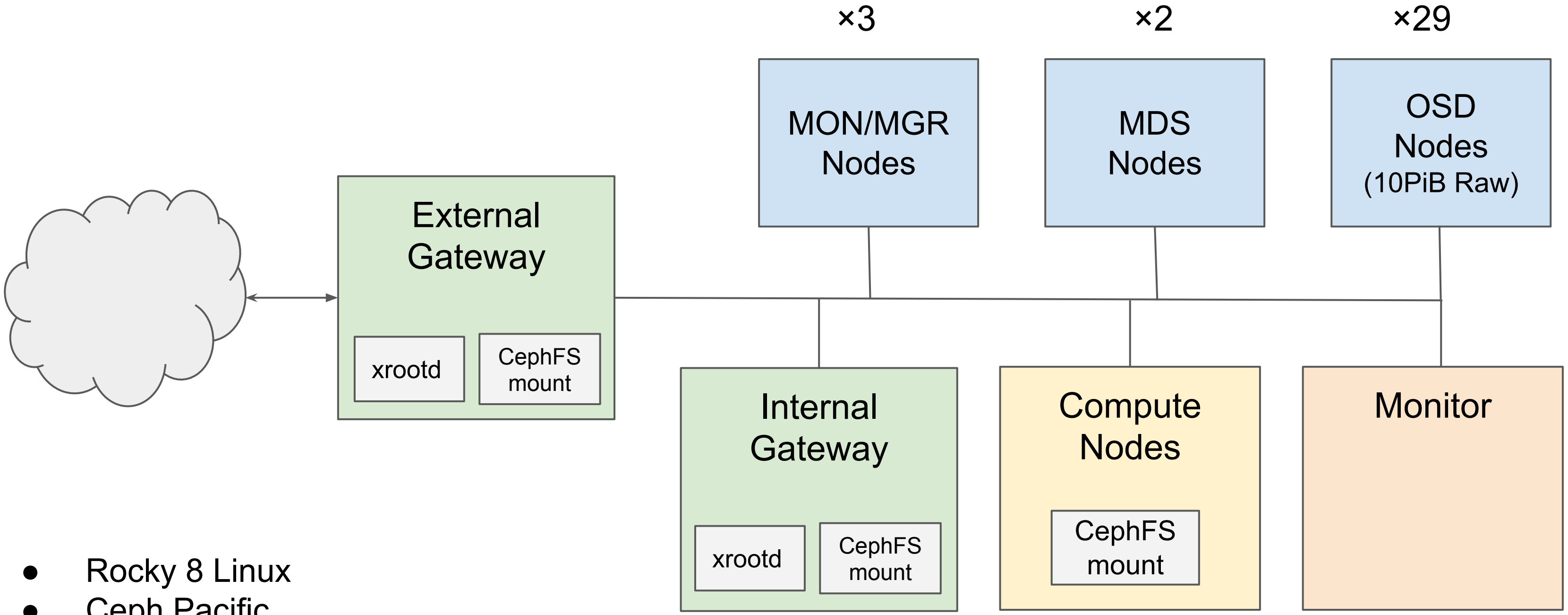
Lancaster found ourselves needing a new Storage solution

- DPM data placement limitations were causing load spikes on disk servers.
  - Firefighting DPM can take up a lot of admin time.
- Active DPM development has stopped and EOL is in sight.
- Large data volumes on modern disks led to a desire for a solution that has server-level redundancy.
  - There’s only so many times you can declare a million files lost and not rethink all your life choices.

Chosen solution was CEPHFS + XRootD

- Strong CEPH knowledge base has been built in the UK (RAL, Glasgow).
- Many advantages to splitting what stores the data and what serves the data.

Ceph architecture



- Rocky 8 Linux
- Ceph Pacific
- Installed using cephadm

Hardware

Admin Nodes (x 7):

- 3 MON/MGR, 2 MDS, 2 XRoot Gateway
- 2×12 core CPU, 128GB RAM
- 2×25Gb networking
- Mirrored 480GB SSD OS disk
- 2×1.8TB SSD disk (Data+Logs)

Monitor Node:

- 2 CPU, 4GB RAM, 200GB disk
- VM

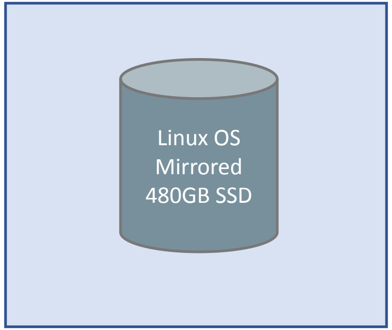
Storage Nodes (x 29):

- 2×10 core CPU, 256GB RAM
- 2×25Gb networking
- Mirrored 480GB SSD OS disk
- 24×16TB SAS disks for data
- 2×1.6TB NVMe drive (BlueStore)

INFRASTRUCTURE:

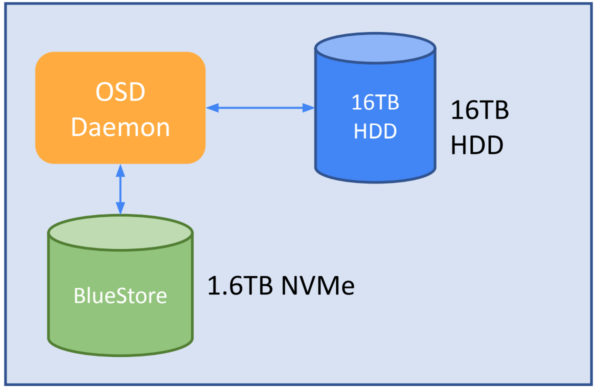
- 25Gb ports on all switches,
- 100Gb between rack. 4-8 disk nodes per rack.

OSD Configuration

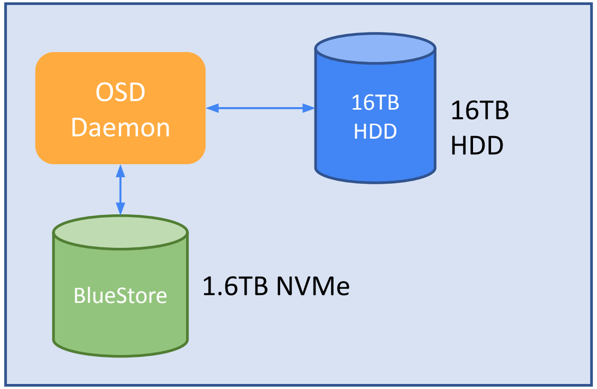


- Only had software option to mirror NVMe.
- Non-mirrored means increased capacity and reduced wear.
- Downside is recovery will happen when NVMe fails.
- Decided against failure domain on group of 12 HDDs as it will potentially remove two “buckets” from the 8+3 pool when the host fails.

12 × OSD Daemons

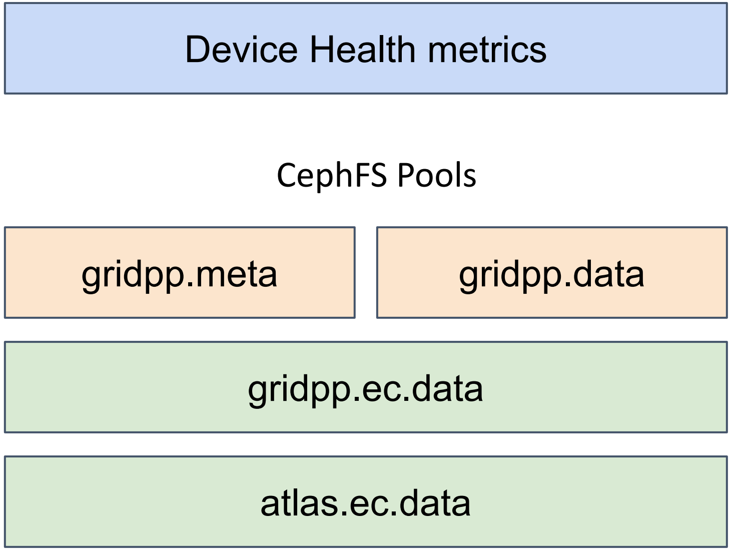


12 × OSD Daemons



Storage

- Created two EC pools. One for Atlas and the other for all other VOs
- Considered 8+2 EC but decided 8+3 EC was safer
- Using Linux ACL on CephFS mount to control writing to VOs directories
- Storage limits set using Ceph Quotas (ceph.quota.max_bytes)
- Determining space used using ceph.dir.rbytes
- Created python scripts to generate SRR JSON for using with XRootD



Device Health metrics

CephFS Pools

gridpp.meta

gridpp.data

gridpp.ec.data

atlas.ec.data

Lessons learned/Gotcha’s Experienced (part 1)

- Don’t expect creating a new cluster in a pandemic to go quickly.
- Learning and setting up CEPH can be a full time job.
- CentOS7 is holding us back (have to use the Octopus repos on the compute).
- Some hardware just doesn’t work anymore if you try to bios boot (specifically the 25GB Broadcom Mezzanine NICs in the storage).
- On a positive note, we have done a couple of OS updates that have required rebooting the nodes. The process was straightforward and went without problems. The cluster just kept on running.
- File transfers to/from the XRootD gateway have been limited by the speed of the 25Gb NIC on gateway.

XRootD

Original Plan - A simple standalone xrootd server running root and https endpoints.

- WAN traffic over https/root
- LAN access via direct mount on the compute

“No plan survives contact with the users (or code bugs)”. Also, make sure your functional tests for a new service cover all functions (or you might be bitten by a problem in the macaroon code that means you have to reinstall your xroot box with CentOS7).

- Always planned to have some form of internal xroot access point “just in case”
  - Found that cephfs traffic easily fills the internal NIC of the xroot gateway so can’t just rely on that.
- Situation complicated by “plain” posix access not being implemented in rucio yet
  - Internal xroot gateway needs to be more than just a back up or internal mirror of the external gateway.
- So we need a robust, internal xroot access point that can serve all jobs
  - Hope is a Xcache set to use Direct Cache Access will do the trick.
    - This would have xroot return a file:// URL rather shovelling data to the client
    - Not got this up and running yet, partly because the “cache” aspect of this functionality could lead to peril.
  - Interim solution is just have a internal mirror of the external gateway.
- Eventually we will likely need a byzantine array of xroot redirectors, which seems to be ground state for xroot systems.

XRoot Access Control (1)

The “cleverest” thing we needed to get right with xroot on ceph was the access control.

- We wanted to authenticate using VOMS, but the VOMS to user mapping libraries in XrootD are not up to snuff.
- So we used the authorisation database (authdb) method.
  - Authentication done via voms roles/groups/organisations
  - But all files and directories written, read or deleted by the xrootd service would be done so by the xroot user.
  - Acces
- Conversely files written by jobs via posix would be by the mapped user.
  - By mounting the ceph volume with the “acl” option, and using file access control lists (setfacl) to set group level acls and default masks were able to get the behaviour we wanted.

XRoot Access Control (2)

Example Authdb Rules:

```
= xatlasprd o: atlas r: production
x xatlasprd /cephfs/grid/atlas a /cephfs/grid/srr lr
```

Example ACLs… well they’re on the right,

and as you can see they’re not particularly pretty.

```
# getfacl /cephfs/grid/atlas/atlasdatadisk/
# file: cephfs/grid/atlas/atlasdatadisk/
# owner: xrootd
# group: xrootd
user::rwx
group::r-x
group:atlas:r-x
group:pltatlas:rwx
group:prdatlas:rwx
mask::rwx
other::r-x
default:user::rwx
default:user:xrootd:rwx
default:group::r-x
default:group:atlas:r-x
default:group:pltatlas:rwx
default:group:prdatlas:rwx
default:mask::rwx
default:other::r-x
```

Lessons Learned/Gotchas Experienced (part 2)

- Just because something seems simple does not mean that you’re not the first one to try it out.
  - Example 1 - xroot macaroon library problems from running on a RHEL8 clone.
  - Example 2 - finding out that “simple” POSIX access in Rucio isn’t in there yet.
- When testing make no assumptions that you’ve covered all contingencies.
  - We’d have had a few weeks headstart on the aforementioned macaroon library problem if Matt had spotted it in the initial tests.
  - Paul Millar’s smoke-test script is a really good tool.
- XrootD doesn’t want to be simple.
  - It was naive beyond belief for Matt to think we’d only need one XRoot Server.

Endpoint Checklist:

- ATLAS HC jobs are running fine against the current setup.
    - There’s so much work on the ATLAS side to get things working - thanks James for all the effort.
- (ATLAS) FTS transfers working.
- DIRAC data management access tested.
    - Will ask hyperk.org to copy their 20TB of data in, with the carrot of extra space.
- Ops tests pass.
- SRR works.
- Tim set up a dteam RSE which “just worked”.
- WLCG Token Authentication setup and passes initial tests.

Endpoint To Do:

- Join WLCG JWT compliance tests.
- Start adding more user groups, and migrating more data in.
    - speed up the retirement of DPM.
- Investigate alternative endpoints - e.g. S3 + on an Objectstore pool

Monitoring architecture

Prometheus stores metrics

- Pulls (scrapes) from exporters
- Multiple per host
- Distinct groups of metrics (node_*, ceph_*)
- Resolves PromQL queries
- Generates alerts

Loki stores logs

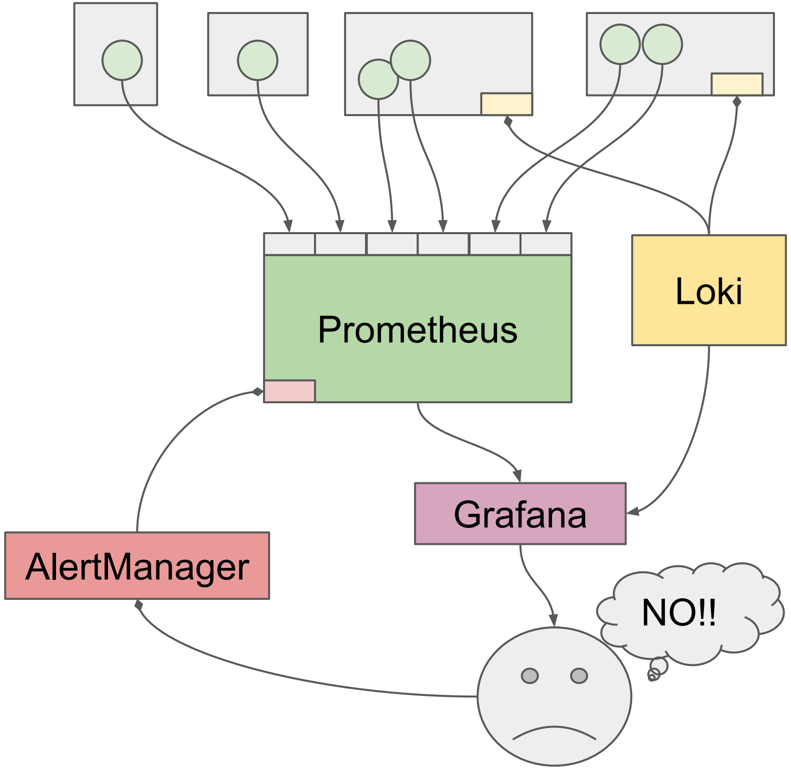
- Exporters push
- Resolves LogQL queries

Grafana visualizes

- Invokes PromQL/LogQL
- Produces graphs, tables, …
- Web interface

AlertManager

- Delivers (via email, SMS, Slack, …)
- Groups similar alerts in single message
- Inhibits implied alerts



Ceph-specific monitoring

Physical:

- 3 monitors/managers
- 2 metadata servers
- Umpteen storage nodes
  - 24 discs/OSDs each

Node exporters

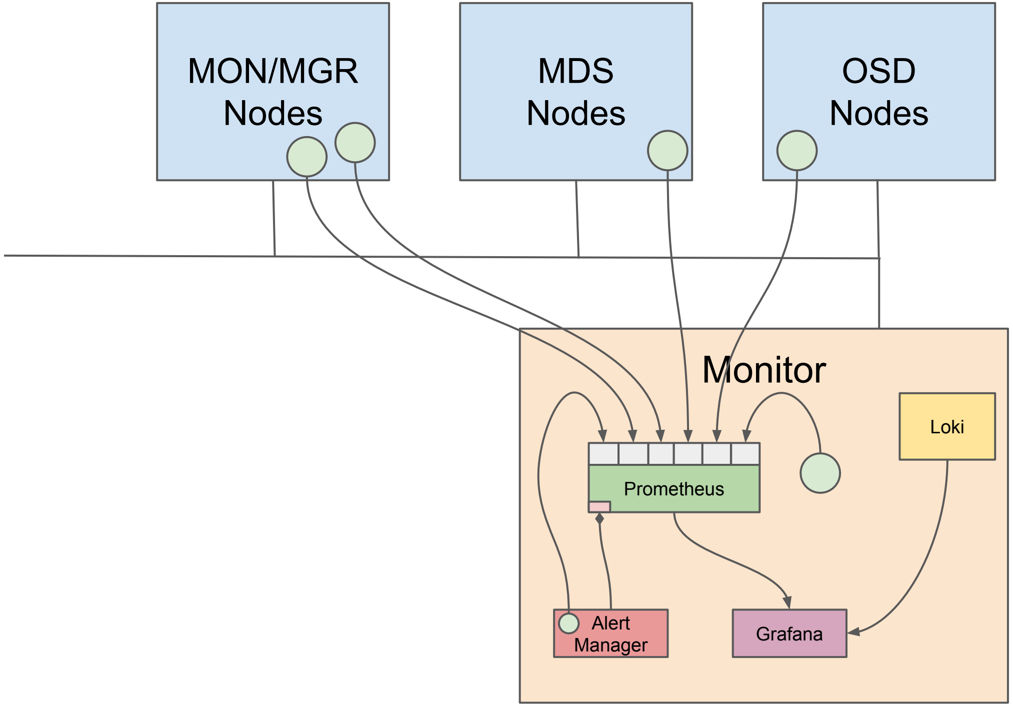
- On every node
  - As part of Ceph installation
- CPU, memory, disc, …

Ceph exporters

- On monitors only
- Pools, PGs, objects, OSDs

Static configuration and reachability

- Local text file of ‘expectations’
- Ping RTTs



Monitoring nomination

Ceph monitors nominate a speaker

- Only speaker yields metrics when asked
- Other monitors yield empty content
- Can switch at any time
  - `instance` label changes
  - Disrupts time series
    - Label names and values must match exactly
- As stable as connectivity

Techniques:

- Scrape from all monitors
- Pare down labels
  - Minimum: ignore `instance` with `avg()`
  - Use `without (instance)` or `by (`*`interesting-labels`*`)`

Metadata

Special metrics with value 1:

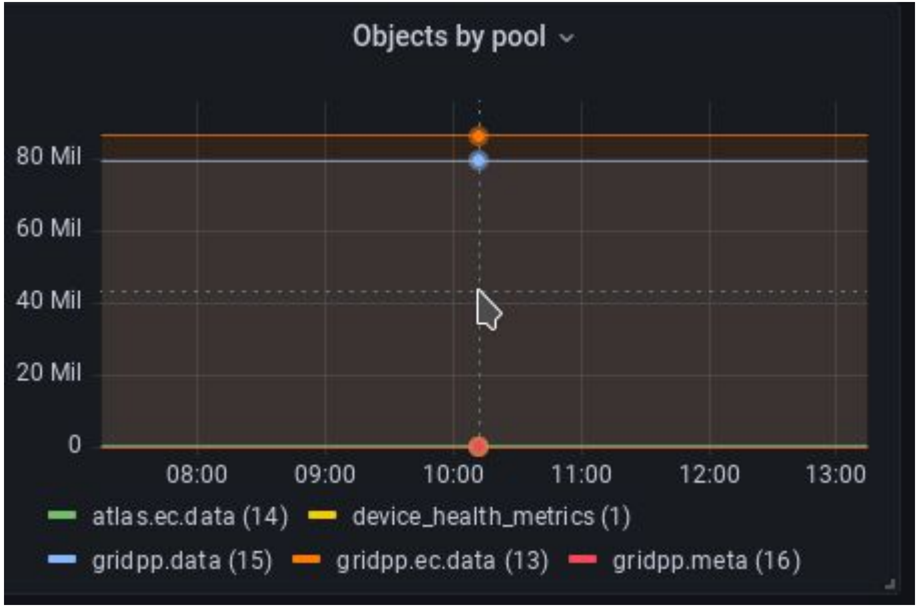
- `ceph_{fs,pool,mds,mgr,mon,osd}_metadata`
- `ceph_disk_occupation`
- `node_{disk,uname,dmi,network}_info`
- `node_hwmon_{chip_names,sensor_label}`

Useful for labelling graphs:

- OSD↔⟨host, disk⟩
- pool id↔⟨name, type, description⟩
- ⟨chip, sensor⟩↔⟨type, label⟩

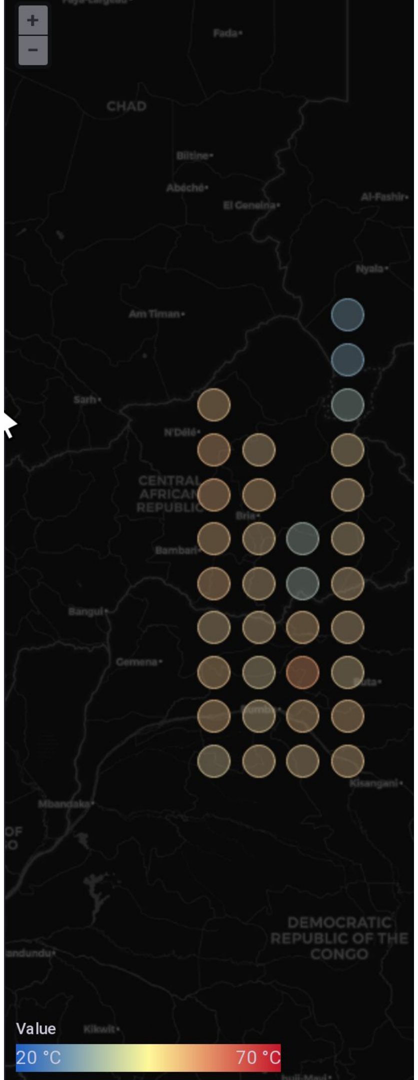
How to use:

- `*useful-metric* * on (pool_id) group_right() ceph_pool_metadata`
- Multiplicative identity ⇒ no change in value
- Labels appended from metadata
- `avg(ceph_pool_objects{}) by (pool_id) * on(pool_id) group_right() avg(ceph_pool_metadata) by (pool_id,name)`



Static expectations

- Some data absent when you need to know it's absent
  - `ceph_osd_up{ceph_daemon="osd.34",...} 0` vs no such metric with that label
- Some just not provided
  - physical location, room, rack, position
- Could just embed constants into queries
  - `sum(ceph_osd_up) by (exported_instance) < 24`
  - Tedious to change
  - Can't vary by node
  - Historical data is compared against current expectations
- Bung static metrics in a file
  - `ip_osd_drives{exported_instance="stor046"} 24`
  - `ip_metadata{exported_instance="stor058", building="physics", room="sauna",rack="11",level="4", roles="/storage/ceph_data/"} 1`
  - Include some missing dynamic ones too: `ip_up{exported_instance="stor052"} 1`
  - Serve through localhost httpd
  - Historical data is compared against contemporary expectations

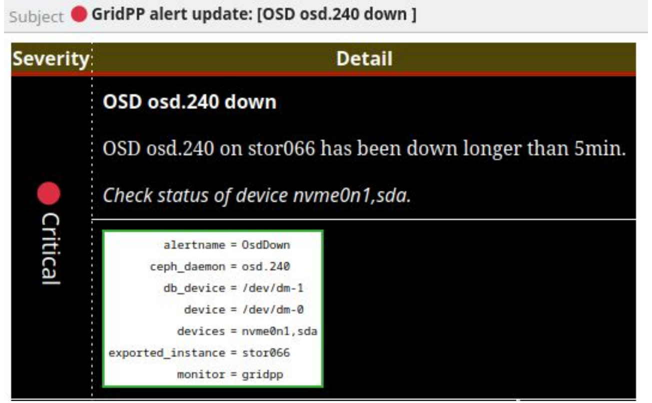


Alerting

Roles:

- Prometheus takes rules
  - Alert name
  - Expression
  - Additional labels (e.g., severity)
    - Others from expression
    - Alert name + labels = alert
  - Annotations (human-readable templates)
- AlertManager
  - Groups related alerts into messages
  - Delivers messages by email, SMS, Slack, …
  - Inhibits implied alerts
    - Manual implication

```
- alert: OsdDown
  expr: >
    avg(ceph_osd_up) by (ceph_daemon) * on(ceph_daemon)
    group_right() avg(ceph_disk_occupation) by (ceph_daemon, device,
    devices, db_device, exported_instance) < 1
  for: 5m
  labels:
    severity: critical
  annotations:
    description: >
      OSD {{ $labels.ceph_daemon }} on {{ $labels.exported_instance
      }} has been down longer than 5min.
    action: >
      Check status of device {{ $labels.devices }}.
    summary: >
      OSD {{ $labels.ceph_daemon }} down
```



Subject ● **GridPP alert update: [OSD osd.240 down]**

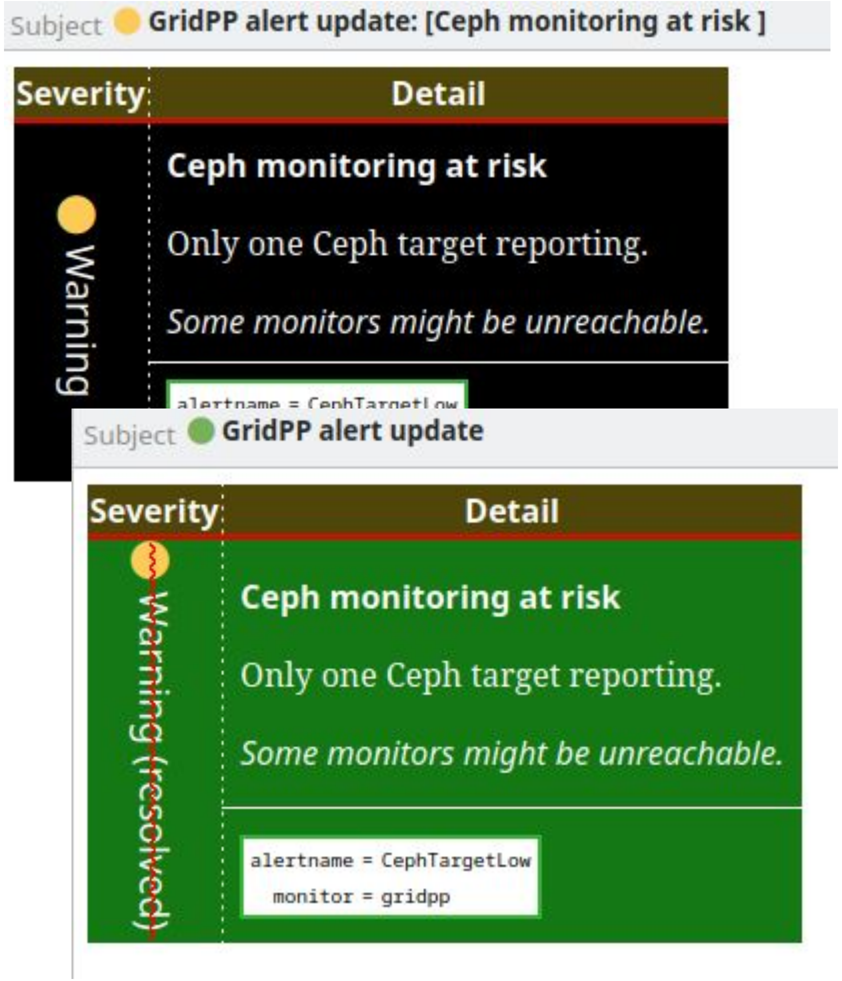
Severity	Detail

● Critical	**OSD osd.240 down**
	OSD osd.240 on stor066 has been down longer than 5min.
	Check status of device nvme0n1,sda.
	alertname = OsdDown
	ceph_daemon = osd.240
	db_device = /dev/dm-1
	device = /dev/dm-0
	devices = nvme0n1,sda
	exported_instance = stor066
	monitor = gridpp

```
inhibit_rules:
- target_matchers:
    - alertname = OsdOut
  source_matchers:
    - alertname = OsdDown
  equal:
    - ceph_daemon
 - target_matchers:
    - alertname = CephWarnState
  source_matchers:
    - alertname = CephErrorState
```

Email notification

```
{{ define "email.notify.subject" -}}
  {{ if gt (len .Alerts.Firing) 0 -}}
    {{ $crit := 0 -}}
    {{ $panic := 0 -}}
    {{ $warn := 0 -}}
    {{ range .Alerts.Firing -}}
      {{ range .Labels.SortedPairs -}}
        {{ if (and (eq .Name "severity") (eq .Value "critical")) -}}
          {{ $crit = 1 }}
        {{- else if (and (eq .Name "severity") (eq .Value "panic")) -}}
          {{ $panic = 1 }}
        {{- else if (and (eq .Name "severity") (eq .Value "warning")) -}}
          {{ $warn = 1 }}
        {{- end }}
      {{- end }}
    {{- end }}
    {{- if gt $panic 0 -}}
      💣
    {{- else if gt $crit 0 -}}
      🔴
    {{- else if gt $warn 0 -}}
      🟡
    {{- else -}}
      🔵
    {{- end }}
  {{- else -}}
    🟢
```



Subject 🟡 GridPP alert update: [Ceph monitoring at risk]

Severity	Detail
🟡 Warning	**Ceph monitoring at risk** Only one Ceph target reporting. *Some monitors might be unreachable.* alertname = CephTargetLow

Subject 🟢 GridPP alert update

Severity	Detail
🟡 ~~Warning (resolved)~~	**Ceph monitoring at risk** Only one Ceph target reporting. *Some monitors might be unreachable.* alertname = CephTargetLow monitor = gridpp

> is not a Boolean relational operator!

It's a filter!

- If true:
  - yield the tuple operand, or
  - LHS if both are tuples

Important for alerting

- Relational operator needed to determine when to report
- Value detected as exceeding threshold retains labels
- Labels fill in alert messages

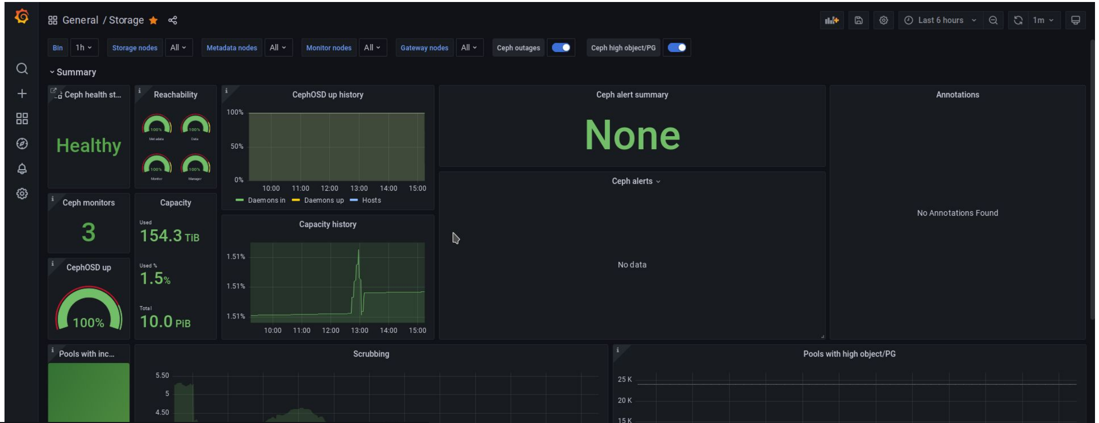
Monitoring TODO/Lessons learned

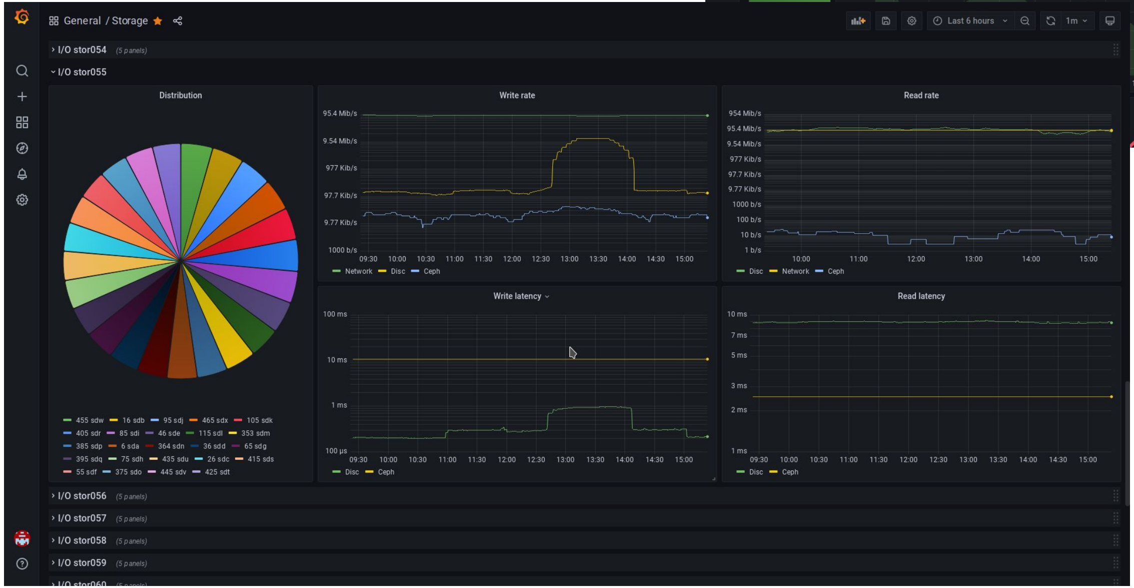
- Log capture to Loki
- Alerting framework in later Grafana
- Compress alerts with embedded queries
- Monitor XRootD…

- Speaker nomination
- Include static expectations
- What > does!
- (explain off-line)
    - Custom metrics need to be live
    - Some Ceph alerts/conditions can be translated into Prometheus alerts
        - Details from Ceph dashboard

Status

- Ready for production





FIN!