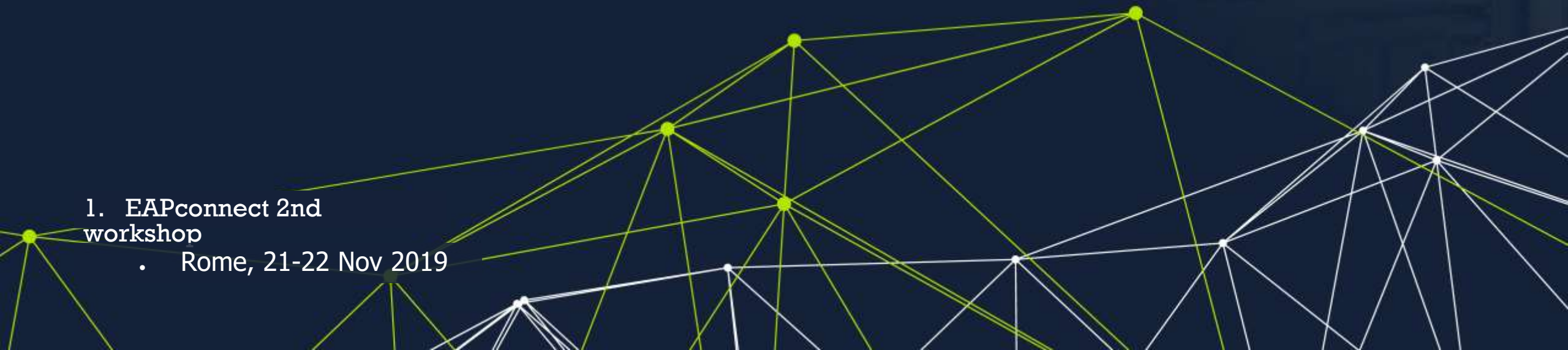
Consortium GARR | THE ITALIAN EDUCATION & RESEARCH NETWORK

CEPH usage & (near-future) developments

FULVIO GALEAZZI, GARR-CSD



1. EAPconnect 2nd workshop
   - Rome, 21-22 Nov 2019

Introduction

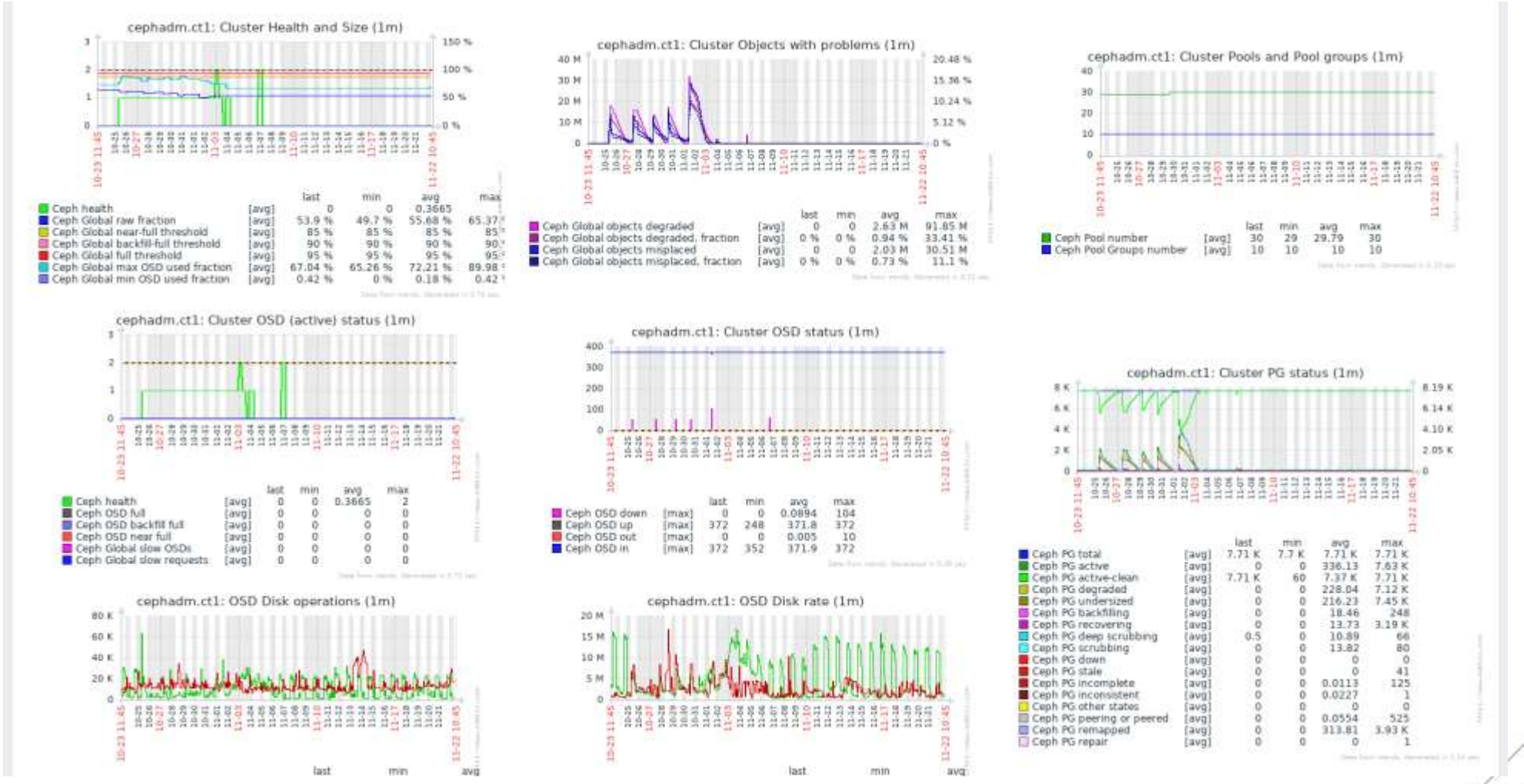
- Shifted the focus of this presentation from "advanced usage" to "things we are working on".
- Meant as a way to kick-start discussion and sharing of experiences(/tools/scripts/…)
- Quick recap on our infrastructure:
  - 6 OSD servers (20 Gbps private, 10 Gbps public networks): 3 MON, co-located on OSD
  - 300 * 3.7 TB disks for block storage
  - 60 * 14.5 TB disks for object storage (EC: 6+4)
  - essentially no SSD disk, so not much fancy configuration
- Luminous 12.2.X, Bluestore, journal and data on same disks
- managed by ceph-ansible: fork, to cope with disks from FC storage boxes

Ceph cluster monitoring

Originally, mostly done via Zabbix.

Some info also channeled to Grafana.

Looking forward to upgrade to Nautilus, which also has more functionality-rich console



Zabbix templates in: https://git.garr.it/CSD/public/zabbix-templates.git

Object Storage activity: migrate to EC pool

For historical reasons, pool *default.rgw.buckets.data* was born "replicated".

- Now it holds ~100 TB (net) data and would like to migrate to EC pool
- no official recipe around,
- nautilus apparently allows pool migration, but not for object storage
- Important points:
- Thorough test!
- Minimize downtime

Envisaged procedure:

- create EC pool data.ec
  - stop rgw access momentarily
  - rename: data -> data.orig, data.ec -> data
  - config data.orig as cache to data, cache-mode readproxy
  - direct writes to cache pool (the original one, replicated),
  - set-overlay data data.orig
- resume rgw access
- play with cache parameters (age, size) to ensure all objects are gradually flushed/evicted
  - once "few" objects are left:
  - stop rgw access
  - cache-flush-evict-all
  - remove overlay, remove cache
  - resume rgw access

Object Storage activity: migrate to multi-site

Have the two production sites as two *zones* of a multi-zone Ceph, to provide redundancy/high-availability

Issues:

- minimize downtime
- ensure migration happens safely

Not sure how to proceed here, but have two distinct clusters on which to run test.

Other from this, anyone using additional placement rules for rgw?

Object storage: manage quota

Quota management (within OpenStack) severely limited:

- specified for each container, rather than for each project
- would need to prevent (or admin-manage) container creation, pre-create default container for each project,...
- managed via *swift* API v1

Experiences to share, here?

Block storage: multiple Cinder backends

Create several Cinder backends, with different characteristics, for example: standard (size=3), reduced-redundancy (size=2) - to save on latency, fast (size=2, device_class=ssd) - for specific usage (and tenants),...

Already tested, works OK, quota hierarchy also works.

Easy to setup with charms:

- juju deploy cinder-ceph cinder-stage-rr
- juju config cinder-stage-rr ceph-osd-replication-count=2 restrict-ceph-pools=True
- juju add-relation ceph-proxy cinder-stage-rr
- juju add-relation nova-compute cinder-stage-rr
- juju add-relation cinder cinder-stage-rr
- openstack --os-username admin --os-tenant-name admin volume type create --description "Points to reduced-redundancy pool" --property "volume_backend_name=cinder-stage-rr" --public Ceph_ReducedRedundancy

Encryption

Often requested by users, especially in the domain of health.

Straightforward possibilities:

- user creates own LUKS volume (+ works, key is in user's hands, - cumbersome)
- administrator creates LUKS OSDs (+ works with ceph-ansible, - user still has to trust admin)

Alternative possibility: explore Barbican (https://docs.openstack.org/barbican/latest/) which would nicely integrate with OpenStack, and leave encryption key management to the user.

File storage

CephFS:

- Not explored yet, apart in extremely small test environment, due to lack of fast storage
- Experiences?

NFS-Ganesha with object-store backend:

- looks promising and would be ideal in many cases where full-POSIX is not needed.
- Experiences? Multi-tenant configuration?