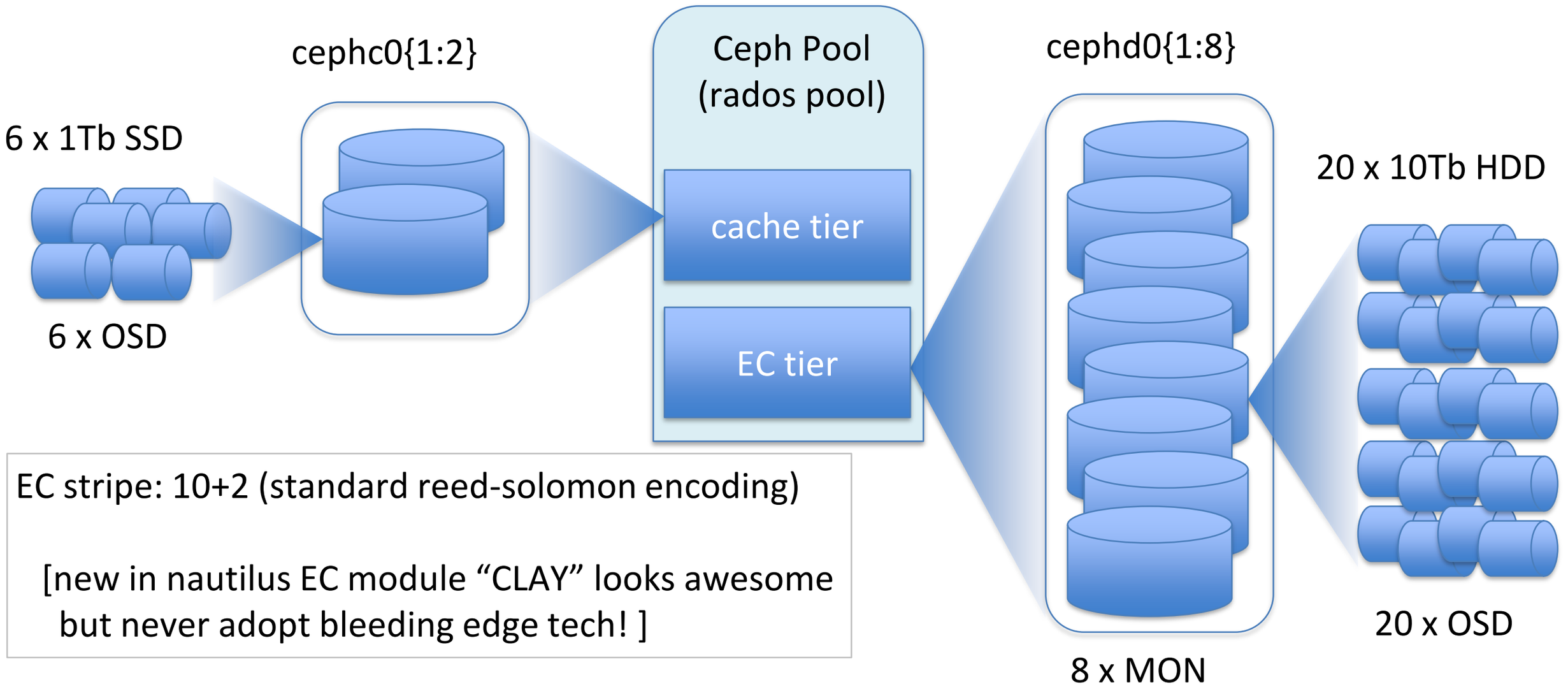
Ceph @ Glasgow

Post-DPM Transition: CEPH



EC stripe: 10+2 (standard reed-solomon encoding)

[new in nautilus EC module “CLAY” looks awesome but never adopt bleeding edge tech!]

- Basic config for ceph pool at GLA
- Config via ceph-ansible [which has some quirks if you use the Centos 7 Storage SIG Repo, as there’s one or two odd packaging dependencies]
- Plus: xrootd, gridftp daemons using libradosstriper for data placement and access.
  - libradosstriper splits incoming files into stripes of uniform chunks (chunk size is invariant) – chunks are then EC coded into stripes in the EC tier.

Config via ceph-ansible

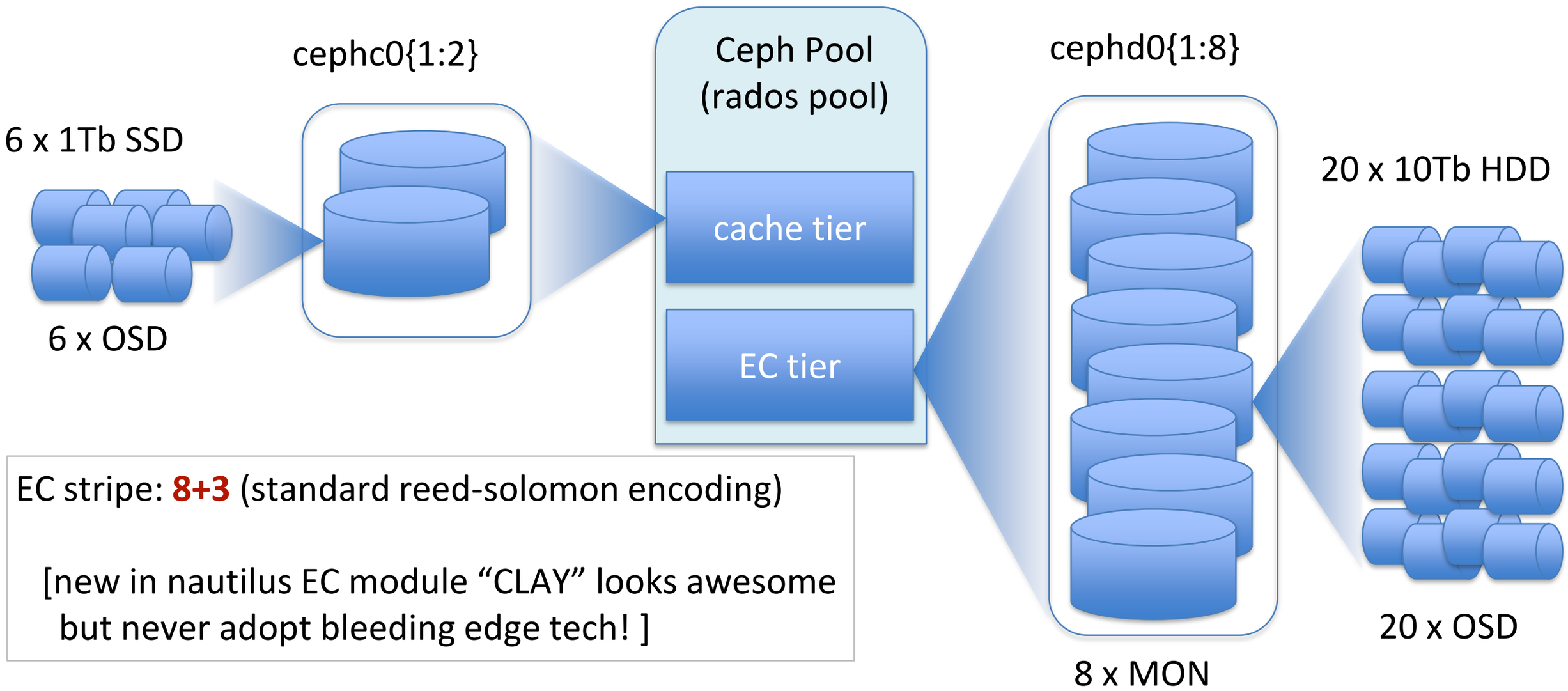
- ceph-ansible via centos-release-ceph-nautilus:

sudo yum -y install centos-release-ceph-nautilus
sudo yum -y install ceph-ansible

Error: Package: ceph-ansible-4.0.0-0.rc5.1.el7.noarch (centos-ceph-nautilus) Requires: python2-netaddr

- This is just the start of a series of weird packaging issues with the "official" Centos ceph-ansible, and ceph-nautilus builds.
- Things "just work" with the Ceph community build, from Ceph directly.

Post-DPM Transition: CEPH

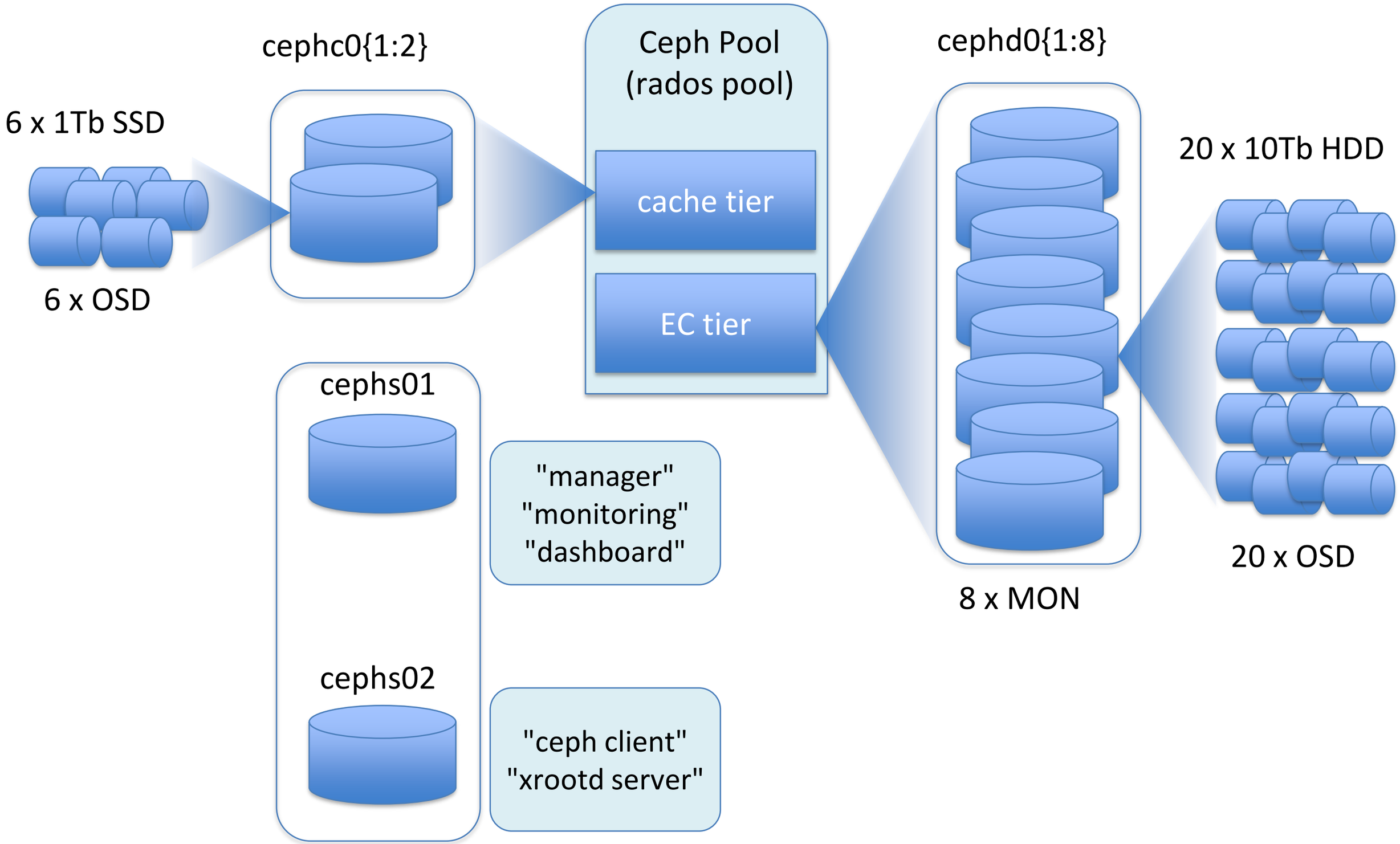


EC stripe: **8+3** (standard reed-solomon encoding)

[new in nautilus EC module “CLAY” looks awesome but never adopt bleeding edge tech!]

- Config via ceph-ansible [from **ceph-ansible git repository** ; ceph **from ceph repositories**]
- Plus: xrootd, gridftp daemons using libradosstriper for data placement and access.
  - libradosstriper splits incoming files into stripes of uniform chunks (chunk size is invariant) – chunks are then EC coded into stripes in the EC tier.

Post-DPM Transition: CEPH



cephs01 - mgr

- Ceph is administered via "ceph" (and "rados") multimodal commands:
  - ie "ceph osd x y" are commands to configure osd properties etc
  - [Admin needs keys in /etc/ceph/ for permissions]

- to make a dashboard service:
  - ceph mgr module enable dashboard
  - ceph dashboard create-self-signed-cert
  - ceph dashboard set-login-credentials username password

cephs01 - mgr

- Pool configuration:
  - EC pools are a bit finicky
  - need an "erasure code profile" [which can't be changed post-pool creation] to exist first.

- ceph osd erasure-code-profile set glasgow-eci-osd \ k=10 \
  m=2 \
  crush-failure-domain=osd

- ceph osd pool create ecpool 2048 2048 erasure glasgow-eci-osd

cephs01 - mgr

- Pool configuration:
  - cache tier easier: ansible script picks up that our cache nodes have ssds (and tags their OSDs appropriately)
  - Make a rule for our cache pool to only use ssds
  - ceph osd crush rule create-replicated hot-storage-rule default host ssd
  - ceph osd pool create hot-pool 2048 2048 replicated hot-storage-rule

cephs01 - dashboard

ceph

English

Dashboard | Cluster | Pools | Block | NFS | Filesystems | Object Gateway

Refresh 5 s

Status

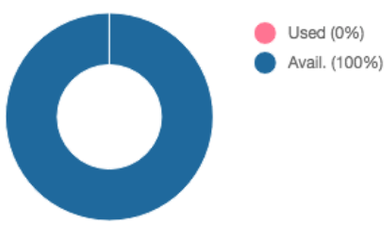
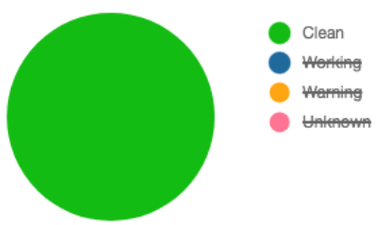
Cluster Status	Monitors	OSDs	Manager Daemons
HEALTH_WARN	8 (quorum 0, 1, 2, 3, 4, 5, 6, 7)	166 total 166 up, 166 in	1 active 0 standby

Hosts	Object Gateways	Metadata Servers	iSCSI Gateways
11 total	0 total	no filesystems	0 total

Performance

Client IOPS	Client Throughput	Client Read/Write	Recovery Throughput	Scrub
0	0 B/s	N/A	0 B/s	Inactive

Capacity

Pools	Raw Capacity	Objects	PGs per OSD	PG Status
2		3721	160.4	

cephs02 - access / client

- xrootd (and gridftp) access in "ECHO mode" needs:
  - librados, libradosstriper (provided by ceph)
  - xrootd-ceph plugin ("provided" by xrootd)
    - as source to compile yourself
    - as (custom) precompiled binaries against specific librados releases
      - **NOT** in the standard xrootd release build!

cephs02 - access / client

- xrootd-ceph
  - as (custom) precompiled binaries against specific librados releases
  - precompiled version generated for RAL build against *Ceph Luminous (v12.x)*
  - **does not run-time link against Ceph Nautilus (v14.x)**

cephs02 - access / client

- xrootd-ceph
  - as source to compile yourself
    - needs devtoolset-7 packages (Centos7 tools too old to build rados templates)
    - building standalone (against xrootd dev headers) could not get repo xrootd binaries to load plugin
    - **had to build entire custom xrootd from source**

cephs02 - access / client

- xrootd-ceph
  - as source to compile yourself
    - **had to build entire custom xrootd from source**

- **Is this a reasonable support position for a production service?**

CEPH Config

- Differences to RAL
  - RAL caching is in xrootd / per WN
    - Reconstructs entire libradosstriper stripe + caches file
    - Caches are local to WN (so less shared cache efficiency)
  - GLA caching is in ceph cache tier / per pool
    - Reconstructs EC stripe + caches libradosstriper chunks
    - (There's still a reconstruction overhead per libradosstriper chunk for accessing entire file)
    - Caches are shared for entire cluster (better shared cache efficiency)

- Plan A – as in these slides, ATLAS space entirely in ceph (migrate our other Nx10TB disk nodes also into the ceph pool)
- Plan B – add cephfs "posix" layer on top, export as "neoclassical SE"
- Plan C – try EOS (without EC as EOS EC performance is ?poor?, so efficiency loss)

cephs02 - client

- Testing cache-tier behaviour:
  - documentation at http://docs.ceph.com/docs/master/dev/cache-pool/#
  - Does not reflect current status of ceph modal commands!
    - some "ceph osd" commands are now "ceph tier" commands in Nautilus.
    - Should we be concerned by out of date documentation?

cephs02 - client

- Testing cache-tier behaviour:
  - benchmarking via rados bench modal commands (low-level performance)
  - write tests (and leave files for read tests later)
    - rados bench -p ecpool 10 write —no-cleanup
  - read tests (**rand**om or **seq**uential)
    - rados bench -p ecpool 10 rand

cephs02 - client

- Testing cache-tier behaviour:
  - Cache-tiering is hard to understand (not least because documentation out of sync with command options)
  - "readonly" cache tier doesn't seem to do anything

cephs02 - client

- "writeback" tier, configured to always cache all objects
- Some results:
  - 4MB chunks are optimal for performance with or without cache tier. [replicates RAL]
  - rados benchmark reads each file *once*, so cache tier *harms* performance

Complex Xrootd ECHO config

- ECHO access nodes appear to run a lot of xrootd services:

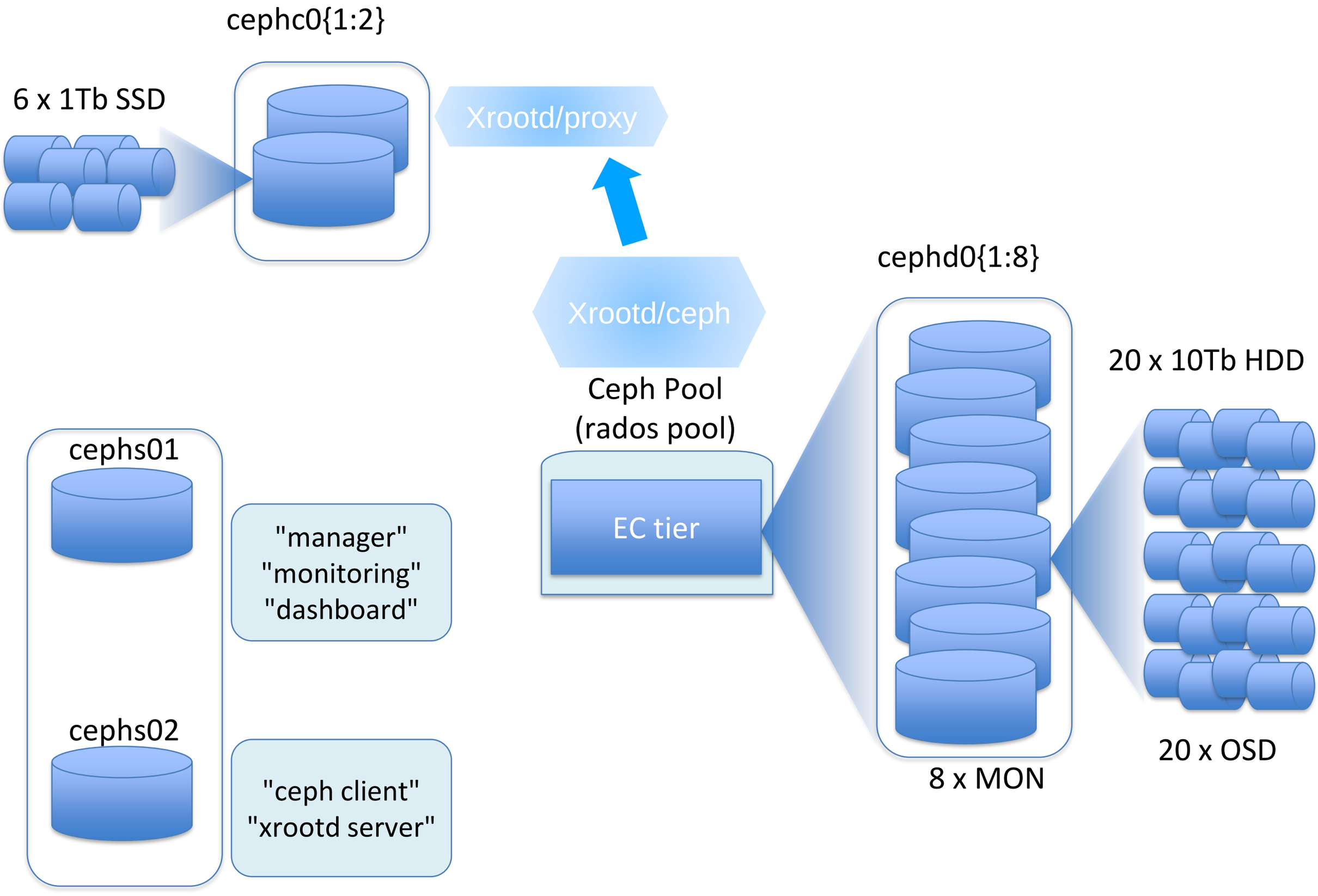
/etc/xrootd/xrootd-ceph.cfg
/etc/xrootd/xrootd-proxy.cfg
/etc/xrootd/xrdadler32-tpc.sh
/etc/xrootd/xrdcp-tpc.sh
/etc/xrootd/xrootd-clustered.cfg
/etc/grid-security/authdb

ceph plugin / access
gsi/grid auth frontend
TPC - checksum support
TPC support
caching, I think
ceph pool <-> user mapping db

New model (Plan A’)

- Ceph cache tiering not reliable
- Xrootd Proxy Caches (branded as “Xcache” for marketing) actually do work.
- Throw away caching in Ceph itself…

Post-DPM Transition: CEPH



A note on xrootd/ceph auth

- The xrootd ceph plugin is rados-level, so uses internal CEPH security model:
  - “user” = shared secret => bundle of capabilities
  - **ceph auth caps client.xrootd mon 'allow rwx' osd 'allow rwx pool=ecpool'**
  - Test pool has 1 user with rwx on mon (for metadata + striping) and on osd (for writing actual data chunks)
  - Prod pools will have 1 user per VO / VO-Role.
  - [Strictly, no limit to number of users]

Xrootd/proxies

- Next level of security:
- xrootd/ceph needs to trust xrootd/proxies

- Use sss security model in xrootd. (Shared secret between client (proxies) and server (xrootd/ceph).

(Generate with xrootdssadmin & ansible out to hosts)

Xroot/proxies -> grid

- Third and final onion layer of security.

- Proxies / Xcache servers as *servers* authenticate their own clients via xrootd x509 security.

- Currently use a simple gridmap file similar to RAL ECHO. (Don’t have RAL central banning yet).
- Would prefer ARGUS integration (is this possible?)

Current steps

- Performance testing (xroot/ceph -> rados; proxies to rados).
  - Currently have a weird “size-dependent” limit on files written to the object store. (in rados itself, even with striper)
    - Seems to break at 1 “chunk” per disk server
    - Changing osd settings (max write size etc) doesn’t fix this.
- Improving security layer on proxies.

Summary

- Ceph is quite nice (if you don’t install it from the Centos repos)
- Erasure coding is tricky.
- Cache tiers are awful.
- Xrootd is too much of a swiss army knife.
- Lots of configuration options to turn.