Sage Weil
@liewegas
Project Leader for Ceph
THANK YOU: PLATINUM SPONSORS

- Intel Optane DC Solid State Drive
- Red Hat
- SoftIron
- SUSE
THANK YOU: GOLD SPONSORS

SUPERMICRO

Western Digital

ZTE
THANK YOU: SILVER SPONSORS

DigitalOcean

Ubuntu
THANK YOU: STARTUP SPONSORS

- Ambedded Technology Co., Ltd.
- upbound
THANK YOU!
FOUNDING PREMIER MEMBERS
FOUNDING GENERAL MEMBERS
FOUNDING ASSOCIATE MEMBERS
NEW MEMBERS THIS YEAR
EVENT PLANNING

#Cephalocon
PROGRAM COMMITTEE

- Dan van der Ster (CERN)
- Florian Haas (City Networks)
- Greg Farnum (Red Hat)
- Josh Durgin (Red Hat)
- Joao Eduardo Luis (SUSE)
- Lenz Grimmer (SUSE)
- Lars Marowsky-Brée (SUSE)
- Neha Ojha (Red Hat)
- Patrick Donnelly (Red Hat)
- Sage Weil (Red Hat)
- Wido den Hollander (42on)
- Yehuda Sadeh (Red Hat)
WELCOME!
Group photo at the end of this presentation
QUESTIONS AND FEEDBACK


https://bit.ly/2LYSMgO

Or contact Ceph Community Manager Mike Perez

thingee@redhat.com

+1-951-572-2633
TOWN HALL Q&A

- Tomorrow morning
- Submit questions via the etherpad


https://bit.ly/2LYSMgO
BIRDS-OF-A-FEATHER SESSIONS

- Tonight, 6:30 - 8:00
- Sign-up whiteboard outside by the expo area
UPCOMING CEPH DAYS

- Ceph Day Netherlands - Utrecht or Amsterdam - Jul 2, 3
- Ceph Day CERN - Geneva - September 16
- Ceph Day Poland - Wrocław - October 28
WHAT IS CEPH?
CEPH IS A UNIFIED STORAGE SYSTEM

- **RGW**: S3 and Swift object storage
- **RBD**: Virtual block device with robust feature set
- **CEPHFS**: Distributed network file system

**LIBRADOS**
Low-level storage API

**RADOS**
Reliable, elastic, distributed storage layer with replication and erasure coding
Project Goal

- **Reliable, high-performance** distributed file system with excellent **scalability**
  - Petabytes to exabytes, multi-terabyte files, billions of files
  - Tens or hundreds of thousands of clients simultaneously accessing same files or directories
  - POSIX interface

- Storage systems have long promised scalability, but have failed to deliver
  - Continued reliance on traditional file systems principles
    - Inode tables
    - Block (or object) list allocation metadata
  - Passive storage devices
Ceph storage services

- Ceph distributed file system
  - POSIX distributed file system with snapshots
- RBD: rados block device
  - Thinely provisioned, snapshottable network block device
  - Linux kernel driver; Native support in Qemu/KVM
- radosgw: RESTful object storage proxy
  - S3 and Swift compatible interfaces
- librados: native object storage
  - Fast, direct access to storage cluster
  - Flexible: pluggable object classes
unified storage system

- objects
  - small or large
  - multi-protocol
- block devices
  - snapshots, cloning
- files
  - cache coherent
  - snapshots
  - usage accounting
LIBRADOS
A library allowing apps to directly access RADOS, with support for C, C++, Java, Python, Ruby, and PHP.

RADOSGW
A bucket-based REST gateway, compatible with S3 and Swift.

RBD
A reliable and fully-distributed block device, with a Linux kernel client and a QEMU/KVM driver.

CEPH FS
A POSIX-compliant distributed file system, with a Linux kernel client and support for FUSE.

RADOS
A reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes.
CEPH STORAGE CLUSTER
A reliable, easy to manage, next-generation distributed object store that provides storage of unstructured data for applications

CEPH OBJECT GATEWAY
A powerful S3- and Swift-compatible gateway that brings the power of the Ceph Object Store to modern applications

CEPH BLOCK DEVICE
A distributed virtual block device that delivers high-performance, cost-effective storage for virtual machines and legacy applications

CEPH FILE SYSTEM
A distributed, scale-out filesystem with POSIX semantics that provides storage for a legacy and modern applications
ARCHITECTURAL COMPONENTS

APP
- RGW: A web services gateway for object storage, compatible with S3 and Swift

HOST/VM
- RBD: A reliable, fully-distributed block device with cloud platform integration

CLIENT
- CEPHFS: A distributed file system with POSIX semantics and scale-out metadata management

LIBRADOs
- A library allowing apps to directly access RADOS (C, C++, Java, Python, Ruby, PHP)

RADOS
- A software-based, reliable, autonomous, distributed object store comprised of self-healing, self-managing, intelligent storage nodes and lightweight monitors
CEPH IS A UNIFIED STORAGE SYSTEM

**OBJECT**
- RGW
  - S3 and Swift object storage

**BLOCK**
- RBD
  - Virtual block device with robust feature set

**FILE**
- CEPHFS
  - Distributed network file system

**LIBRADOS**
  - Low-level storage API

**RADOS**
  - Reliable, elastic, distributed storage layer with replication and erasure coding
WHAT IS CEPH?

- A unified storage system (file, block, and object)
- A storage platform
- Software defined storage
- The future of storage
- The Linux of storage
WHY IS CEPH?
CEPH IS FREE AND OPEN SOURCE SOFTWARE

- Freedom to use
- Freedom from vendor lock-in
- Freedom to innovate
- Freedom to share
CEPH IS RELIABLE

- Reliable, durable storage service out of unreliable components
  - No single point of failure
  - Replication and erasure coding
- Favor consistency and correctness over performance
CEPH IS SCALABLE

- Ceph provides elastic infrastructure
  - allows storage to grow or shrink
  - add or remove hardware while cluster is online
  - online, rolling software upgrades
- Scale-up to leverage bigger, faster HW
- Scale-out within a single cluster, site
- Multi-cluster federation across sites
FOUR CEPH PRIORITIES

Usability and management  Performance

Container ecosystem  Multi-site and hybrid cloud
WHAT’S NEW IN CEPH
NAUTILUS
RELEASE SCHEDULE

- Stable, named release every 9 months
- Backports for 2 releases
- Upgrade up to 2 releases at a time
  - (e.g., Luminous → Nautilus, Mimic → Octopus)
EASE OF USE AND MANAGEMENT
CEPH DASHBOARD

- Community convergence in single built-in dashboard implementation
- Built-in and self-hosted
  - Trivial deployment, tightly integrated with ceph-mgr
  - Easily skinned, localization for many languages
- Metrics and monitoring
- Management functions
- Hardware/deployment management in progress...
ORCHESTRATOR API “SANDWICH”

cceph-mgr: orchestrator API

API call
- Rook
- ceph-ansible
- DeepSea
- ssh

Provision
- ceph-mon
- ceph-mds
- ceph-osd
- radosgw
- rbd-mirror
ORCHESTRATOR SANDWICH

- Abstract deployment functions
  - Fetch node inventory
  - Create or destroying daemon instances/deployments
  - Blink device LEDs
- **Unified CLI and GUI for managing the Ceph cluster deployment**
  - `ceph orchestrator device ls [node]`
  - `ceph orchestrator osd create [flags] node device [device]`
  - `ceph orchestrator mon rm [name]`
  - ...
- Nautilus includes framework and partial implementation
- Focus area for Octopus release
MANAGED CEPHFS NFS GATEWAYS

- Clustered nfs-ganesha gateways
  - NFSv4
  - active/active
  - Stateless: nfs-ganesha daemons use RADOS for configuration, grace period state
  - Correct clustered failover semantics (i.e., managed NFS grace period)
  - (See Jeff Layton’s devconf.cz talk recording)

- nfs-ganesha daemons fully managed via new orchestrator interface
  - Fully supported with Rook; others to follow
  - Full support from CLI to Dashboard

- See recent blog on ceph.com
PG AUTOSCALING

- Picking pg_num has historically been “black magic”
  - Limited/confusing guidance on what value(s) to choose
  - pg_num could be increased, but never decreased

- Nautilus: pg_num can be decreased

- Nautilus: pg_num can be automagically tuned in the background
  - Based on usage (how much data in each pool)
  - Administrator can optionally hint about future/expected usage
  - Ceph can either issue health warning or initiate changes itself

```
$ ceph osd pool autoscale-status
POOL      SIZE   TARGET SIZE  RATE  RAW CAPACITY   RATIO  TARGET RATIO  PG_NUM  NEW PG_NUM  AUTOSCALE
a    12900M                 3.0        82431M  0.4695                     8         128  warn
c         0                 3.0        82431M  0.0000        0.2000       1          64  warn
b         0        953.6M   3.0        82431M  0.0347                     8              warn
```
## DEVICE HEALTH METRICS

- OSD and mon report underlying storage devices, scrape SMART metrics

```
# ceph device ls
```

<table>
<thead>
<tr>
<th>DEVICE</th>
<th>HOST:DEV</th>
<th>DAEMONS</th>
<th>LIFE EXPECTANCY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crucial_CT1024M550SSD1_14160C164100</td>
<td>stud:sdd</td>
<td>osd.40</td>
<td>&gt;5w</td>
</tr>
<tr>
<td>Crucial_CT1024M550SSD1_14210C25EB65</td>
<td>cpach:sde</td>
<td>osd.18</td>
<td>&gt;5w</td>
</tr>
<tr>
<td>Crucial_CT1024M550SSD1_14210C25F936</td>
<td>stud:sde</td>
<td>osd.41</td>
<td>&gt;8d</td>
</tr>
<tr>
<td>INTEL_SSDPE2ME400G4_CVMDC542003M400FGN</td>
<td>cpach:nvme1n1</td>
<td>osd.10</td>
<td></td>
</tr>
<tr>
<td>INTEL_SSDPE2MX012T4_CVPD6185002R1P2QGN</td>
<td>eutow:nvme0n1</td>
<td>osd.1</td>
<td></td>
</tr>
<tr>
<td>ST2000NX0253_S4608PDF</td>
<td>cpach:sdo</td>
<td>osd.7</td>
<td></td>
</tr>
<tr>
<td>ST2000NX0253_S460971P</td>
<td>cpach:sdn</td>
<td>osd.8</td>
<td></td>
</tr>
<tr>
<td>Samsung_SSD_850_EVO_1TB_S2RENX0J500066T</td>
<td>cpach:sdb</td>
<td>mon.cpach</td>
<td>&gt;5w</td>
</tr>
</tbody>
</table>

- **Failure prediction**
  - Local mode: pretrained model in ceph-mgr predicts remaining life
  - Cloud mode: SaaS based service (free or paid) from ProphetStor

- **Optional automatic mitigation**
  - Raise health alerts (about specific failing devices, or looming failure storm)
  - Automatically mark soon-to-fail OSDs “out”
CONFIGURATION MANAGEMENT

● Cluster configs centrally managed by the monitor
  ○ Introduced in Mimic; improved in Nautilus
  ○ Now managed via both CLI and GUI
  ○ Built-in documentation for (almost) all options
  ○ Most options can be updated in realtime

● OSD memory utilization configured via a single osd_memory_target
  ○ Other daemons coming soon

● OSD NUMA management
  ○ Inspect NUMA state via ‘ceph osd numa-status’
  ○ Control NUMA pinning via ‘ceph config set osd.123 osd_numa_node <n>’ (+ mark down OSD)

● RBD image options
  ○ Per image options can also be set at pool level
  ○ rbd-mirror daemon now stateless--all configs are stored by the cluster
<table>
<thead>
<tr>
<th>WRITE OPS</th>
<th>READS OPS</th>
<th>WRITE BYTES</th>
<th>READ BYTES</th>
<th>WRITE LAT</th>
<th>READ LAT</th>
<th>IMAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.09K/s</td>
<td>125/s</td>
<td>4.3 MiB/s</td>
<td>503 KiB/s</td>
<td>10.57 ms</td>
<td>1.39 ms</td>
<td>rbd/image3</td>
</tr>
<tr>
<td>1.05K/s</td>
<td>470/s</td>
<td>4.1 MiB/s</td>
<td>1.8 MiB/s</td>
<td>10.63 ms</td>
<td>1.43 ms</td>
<td>rbd/image8</td>
</tr>
<tr>
<td>866/s</td>
<td>853/s</td>
<td>3.4 MiB/s</td>
<td>3.3 MiB/s</td>
<td>10.44 ms</td>
<td>1.41 ms</td>
<td>rbd/image7</td>
</tr>
<tr>
<td>677/s</td>
<td>1.57K/s</td>
<td>2.6 MiB/s</td>
<td>6.1 MiB/s</td>
<td>10.51 ms</td>
<td>1.08 ms</td>
<td>rbd/image0</td>
</tr>
<tr>
<td>609/s</td>
<td>1.79K/s</td>
<td>2.4 MiB/s</td>
<td>7.0 MiB/s</td>
<td>10.49 ms</td>
<td>1.10 ms</td>
<td>rbd/image2</td>
</tr>
<tr>
<td>535/s</td>
<td>2.14K/s</td>
<td>2.1 MiB/s</td>
<td>8.4 MiB/s</td>
<td>10.60 ms</td>
<td>973.15 us</td>
<td>rbd/image1</td>
</tr>
<tr>
<td>513/s</td>
<td>952/s</td>
<td>2.1 MiB/s</td>
<td>3.7 MiB/s</td>
<td>11.76 ms</td>
<td>4.11 ms</td>
<td>rbd/image4</td>
</tr>
<tr>
<td>444/s</td>
<td>2.44K/s</td>
<td>1.7 MiB/s</td>
<td>9.5 MiB/s</td>
<td>10.56 ms</td>
<td>948.41 us</td>
<td>rbd/image6</td>
</tr>
<tr>
<td>433/s</td>
<td>2.46K/s</td>
<td>1.7 MiB/s</td>
<td>9.6 MiB/s</td>
<td>10.61 ms</td>
<td>965.23 us</td>
<td>rbd/image5</td>
</tr>
<tr>
<td>233/s</td>
<td>4.32K/s</td>
<td>934 KiB/s</td>
<td>17 MiB/s</td>
<td>10.26 ms</td>
<td>783.60 us</td>
<td>rbd/image9</td>
</tr>
</tbody>
</table>
CONTAINER ECOSYSTEM
- Expose Ceph storage to Kubernetes
  - Any scale-out infrastructure platform needs scale-out storage
- Run Ceph clusters in Kubernetes
  - Use Kubernetes as bare metal infrastructure layer
  - Simplify Ceph deployment and management
ROOK

- All-in on Rook as a robust operator for Ceph in Kubernetes
  ○ Extremely easy to get Ceph up and running!
- Intelligent management of Ceph daemons
  ○ Add/remove monitors while maintaining quorum
  ○ Schedule stateless daemons (rgw, nfs, rbd-mirror) across nodes
- “Kubernetes native” provisioning of storage resources
  ○ Declarative CRDs for clusters, pools, filesystems, etc.
  ○ Managed ceph-csi for Persistent Volumes (RWO and RWX)
  ○ Coming: dynamic provisioning of RGW object storage buckets
- Enthusiastic user community, CNCF incubation project
- v1.0 just released
  ○ Focus on ability to support in production environments
BAREBONES CONTAINERS

- **Why?**
  - Enables granular, per-daemon upgrades
  - Eliminates variance in distro libraries (e.g., tcmalloc, smartmontools)

- **ceph-container project**
  - Standard upstream container image for Ceph
  - Single image with all daemons

- **ceph-ansible**
  - Supports running daemons inside docker containers via systemd units

- **We plan to teach other orchestrators too**
  - SSH orchestrator, as the moral successor to ceph-deploy
RGW can generate event stream
  ○ PUTs and DELETEs

Currently integrated with knative
  ○ Kubernetes serverless framework
  ○ First demoed at KubeCon Seattle
  ○ e.g., Trigger FaaS functions via RGW upload

Other modes coming
  ○ Kafka, AMQ
MULTI-SITE AND HYBRID CLOUD
NEW RGW MULTI-SITE CAPABILITIES

- **Archive zone**
  - Enable bucket versioning and retain all copies of all objects

- **Cloud sync**
  - Replicate RGW buckets to external cloud storage (e.g., S3)

- **Ongoing cleanup and refactoring to lay foundation for multi-site v3**

- **S3 lifecycle policy API**
  - Automated, schedule-based tiering and/or object expiration
PERFORMANCE
BLUESTORE PERFORMANCE

- New allocator implementation
  - Better fragmentation behavior
  - Faster (less CPU)
  - Fixed memory overhead
- RocksDB improvements
  - Readahead support, improving compaction and iteration performance
  - Likely to get backported to Mimic and Luminous
- Misc optimizations
RGW PERFORMANCE

- New Beast front-end
  - Based on boost::asio
  - Better performance and efficiency
  - Replaces civetweb web server

- One more step of ongoing RGW refactoring to a more async model
● 8 high-performance nodes
  ○ Dual socket Xeon
  ○ 8 HDD
  ○ 4 NVMe
● Donated by Intel in 2016
● Part of the upstream Sepia community test lab
SEPIA OFFICINALIS
NEW COMMUNITY CLUSTER FOR CEPH PERFORMANCE TESTING

- 10 Node Cluster of 1U Servers
- NUMA Balanced (NVMe / Networking)
- 100GbE (2x 25GbE per CPU Socket)
- Expandable with Intel® Optane™ DC Persistent Memory
# 10 Node Sepia Officinalis Ceph Community Performance Testing Cluster

Latest Generation Reference Cluster for Ceph Performance Testing

<table>
<thead>
<tr>
<th>Component</th>
<th>Cluster Count</th>
<th>Per Node Count</th>
<th>Model</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>20</td>
<td>2</td>
<td>Intel® Xeon® Platinum 8276 Processor</td>
<td>56 cores per node</td>
</tr>
<tr>
<td>Metadata Drive</td>
<td>20</td>
<td>2</td>
<td>Intel® Optane™ SSD DC P4800X Series (750 GB)</td>
<td>1.5 TB per node / 15 TB cluster</td>
</tr>
<tr>
<td>OSD Drive</td>
<td>80</td>
<td>8</td>
<td>Intel® SSD DC P4510 Series (8 TB)</td>
<td>64 TB per node / 640 TB cluster</td>
</tr>
<tr>
<td>NIC</td>
<td>20</td>
<td>2</td>
<td>Intel® Ethernet Network Adapter XXV710-DA2</td>
<td>2x 25 GbE per NIC</td>
</tr>
<tr>
<td>Boot Drive</td>
<td>10</td>
<td>1</td>
<td>Intel® SSD D3-S4510 Series 960 GB</td>
<td></td>
</tr>
<tr>
<td>Server</td>
<td>10</td>
<td>1</td>
<td>QCT™ QuantaGrid D52B-1U</td>
<td>1U all NVMe Flash SKU (NUMA Balanced)</td>
</tr>
<tr>
<td>RAM</td>
<td>120</td>
<td>12</td>
<td>32 GB DRAM DIMMS @ 2666 MHz</td>
<td>384 GB DRAM per node</td>
</tr>
</tbody>
</table>
Why
● Not just about how many IOPS we do…
● More about IOPS per CPU core
● Current Ceph is based on traditional multi-threaded programming model
● Context switching is too expensive when storage is almost as fast as memory

What
● Rewrite IO path in using Seastar
  ○ Preallocate cores
  ○ One thread per core
  ○ Explicitly shard all data structures and work over cores
  ○ No locks and no blocking
  ○ Message passing between cores
  ○ Polling for IO
● DPDK, SPDK
  ○ Kernel bypass for network and storage IO
FIVE THEMES

Usability

Quality

Performance

Multi-site

Ecosystem
Previously crashes would manifest as a splat in a daemon log file, usually unnoticed...

Now concise crash reports logged to `/var/lib/ceph/crash/`
- Daemon, timestamp, version
- Stack trace
- Failed assertion condition (if any)

Reports are regularly posted to the mon/mgr

‘ceph crash ls’, ‘ceph crash info <id>’, ...

Opt-in telemetry
- Phone home basic cluster stats to developers
- Including crash reports
QUESTIONS AND FEEDBACK


https://bit.ly/2LYSMgO
THANK YOU

Sage Weil
sage@redhat.com
@liewegas
http://ceph.io
Picture time!