



# Understanding BlueStore

Ceph's New Storage Backend

Tim Serong  
Senior Clustering Engineer  
SUSE  
[tserong@suse.com](mailto:tserong@suse.com)

# Introduction to Ceph (the short, short version)

# Ceph Provides

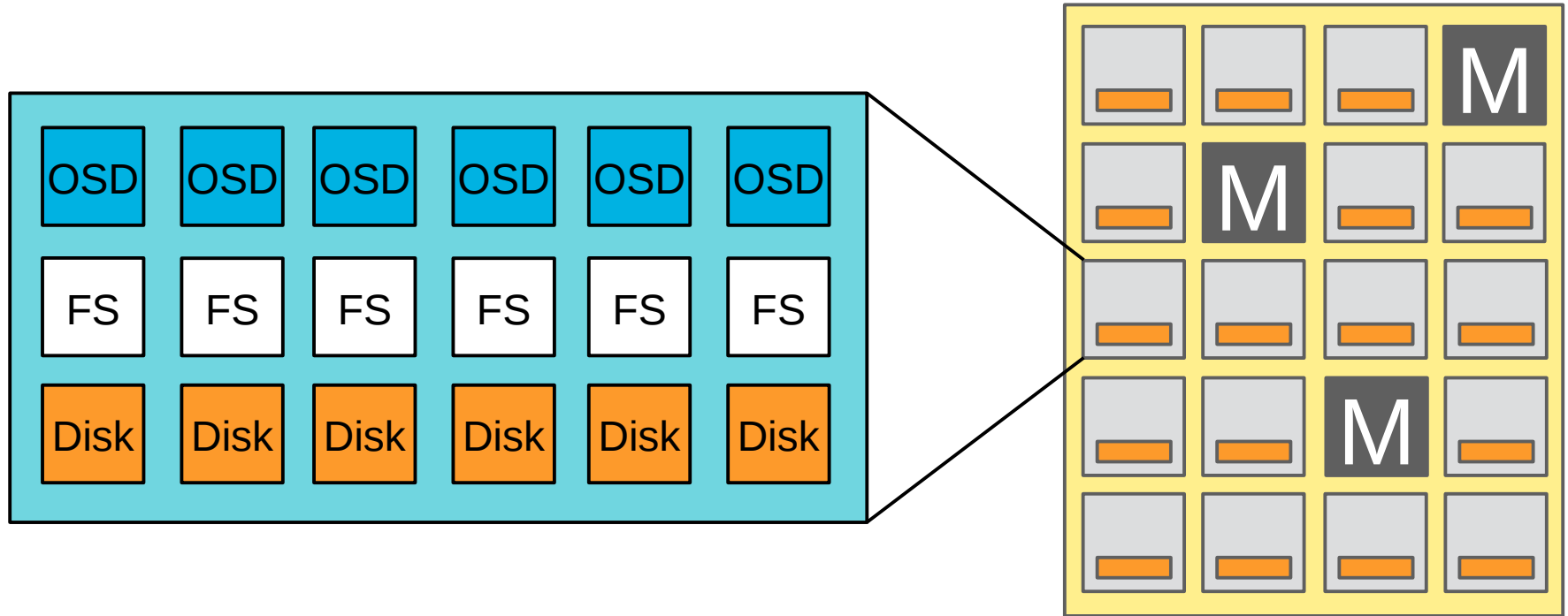
- **A resilient, scale-out storage cluster**
  - On commodity hardware
  - No bottlenecks
  - No single points of failure
- **Three interfaces**
  - Object (radosgw)
  - Block (rbd)
  - Distributed Filesystem (cephfs)

# Internally, Regardless of Interface

- **All data stored as “objects”**
  - Aggregated into Placement Groups
- **Objects have:**
  - Data (byte stream)
  - Attributes (key/value pairs)
- **Objects live on Object Storage Devices**
  - Managed by Ceph Object Storage Daemons
- **An OSD is a disk, in a server**



# More Internally



# Even *More* Internally

- **FileStore**

- XFS (previously ext4, btrfs)
- Placement Group → directory
- Object → file
- LevelDB for key/value data

```
/var/lib/ceph/osd/ceph-0/  
  current/  
    0.1_head/  
      object1  
      object12  
    0.7_head/  
      object3  
      object5  
    0.a_head/  
      object4  
      object6  
  meta/  
    # OSD maps  
  omap/  
    # leveldb files
```

# Problems with FileStore

# Double Writes

- **No atomic write/update on POSIX filesystems**
- **Solution: Ceph Journal**
  - Write → OSD journal (O\_DIRECT)
  - Write → OSD filesystem (buffered)
  - Sync filesystem, trim journal
- **Throughput halved**



# Enumeration Sucks

- **Objects distributed by 32-bit hash**
- **Enumerated for scrub, rebalance etc.**
- **POSIX readdir() isn't ordered**
- **Directory tree with hash-value prefix**
  - Split when > ~100 files
  - Merged when < ~50 files
- **Complicated, odd performance issues**

```
...  
/DIR_1  
/DIR_1/DIR_0/obj-xxxxxx01  
/DIR_1/DIR_4/obj-xxxxxx41  
/DIR_1/DIR_8/obj-xxxxxx81  
/DIR_1/DIR_C/obj-xxxxxxC1  
...  
/DIR_A/DIR_1/obj-xxxxx1A  
/DIR_A/DIR_5/obj-xxxxx5A  
/DIR_A/DIR_9/obj-xxxxx9A  
/DIR_A/DIR_D/obj-xxxxxDA  
...
```

# Other Problems

- CPU utilization
- Unpredictable filesystem flushing
- Still finding bugs/issues with FileStore...

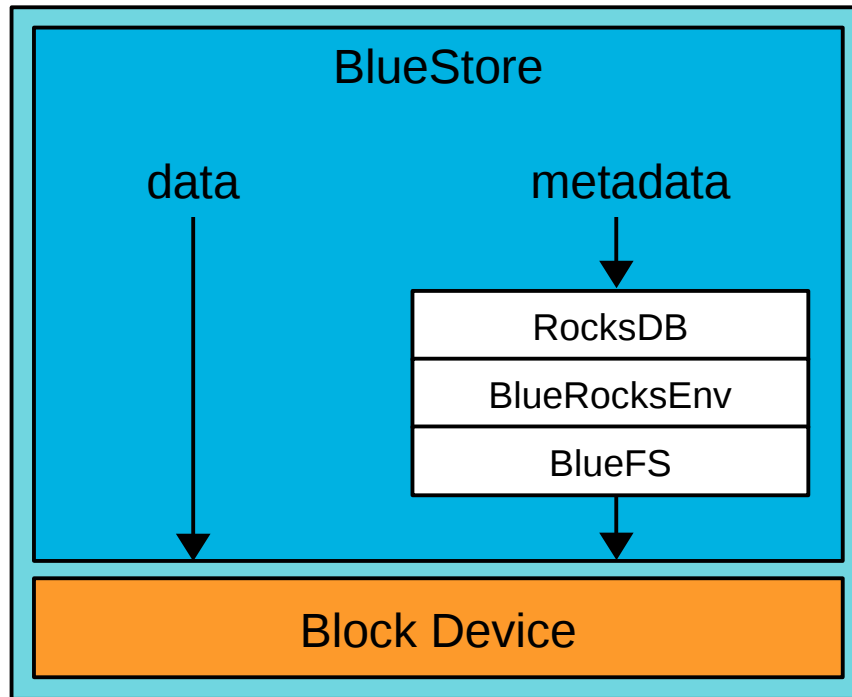
# Solutions with BlueStore

# BlueStore = Block + NewStore

- Raw block devices
- RocksDB key/value store for metadata
- Data written direct to block device
- 2-3x performance boost over FileStore

# Internally

- All data and metadata on same block device by default
- **Optionally can be split, e.g.:**
  - Data on HDD
  - DB on SSD
  - WAL on NVRAM





# Bonus!

- **Pluggable block allocator**
- **Pluggable compression**
- **Checksums on reads**
- **RBD and CephFS usable with EC pools**

# Availability

- **Early prototype in Ceph Jewel (April 2016)**
- **Stable on-disk format in Ceph Kraken (January 2017)**
- **Recommended default in Ceph Luminous (RSN)**
- **Can co-exist with FileStore**

# Migration Strategies

# Several Approaches

- **Fail in place**
  - Fail FileStore OSD
  - Create new BlueStore OSD on same device
- **Disk-wise replacement**
  - Create new BlueStore OSD on spare disk
  - Stop FileStore OSD on same host
- **Host-wise replacement**
  - Provision entire new host with BlueStore OSDs
  - Swap into old host's CRUSH position
- **Evacuate and rebuild in place**
  - Evacuate FileStore OSD, fail when empty
  - Create new BlueStore OSD on same device

# Several Approaches

- **Fail in place**

- Fail FileStore OSD
- Create new BlueStore OSD on same device

→ **period of reduced redundancy**

- **Disk-wise replacement**

- Create new BlueStore OSD on spare disk
- Stop FileStore OSD on same host

→ **reduced *online* redundancy, requires extra drive slot**

- **Host-wise replacement**

- Provision entire new host with BlueStore OSDs
- Swap into old host's CRUSH position

→ **no reduced redundancy, requires spare host**

- **Evacuate and rebuild in place**

- Evacuate FileStore OSD, fail when empty
- Create new BlueStore OSD on same device

→ **no reduced redundancy, requires free disk space**



Questions?





## **Unpublished Work of SUSE LLC. All Rights Reserved.**

This work is an unpublished work and contains confidential, proprietary and trade secret information of SUSE LLC. Access to this work is restricted to SUSE employees who have a need to know to perform tasks within the scope of their assignments. No part of this work may be practiced, performed, copied, distributed, revised, modified, translated, abridged, condensed, expanded, collected, or adapted without the prior written consent of SUSE. Any use or exploitation of this work without authorization could subject the perpetrator to criminal and civil liability.

## **General Disclaimer**

This document is not to be construed as a promise by any participating company to develop, deliver, or market a product. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. SUSE makes no representations or warranties with respect to the contents of this document, and specifically disclaims any express or implied warranties of merchantability or fitness for any particular purpose. The development, release, and timing of features or functionality described for SUSE products remains at the sole discretion of SUSE. Further, SUSE reserves the right to revise this document and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes. All SUSE marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States and other countries. All third-party trademarks are the property of their respective owners.