Bcache
软件定义存储系统的性能加速

李勇
软件工程师
SUSE实验室

Coly Li
Software Engineer
SUSE Labs

SUSE
We adapt. You succeed.

什么是Bcache

- Bcache是Linux内核中基于块设备接口的高速cache
  - /dev/bcache0, /dev/bcache1, /dev/bcache2 …
- 当系统的IO负载具有明显的热数据特征时
  - 可以使用bcache将热数据缓存在高速cache设备
  - 当IO请求命中到cache设备中时，性能接近于直接访问高速存储设备，进而大大提高了热数据的IO访问性能
- Bcache主要特征
  - 对上层应用和后端设备透明
  - 高性能：当cache命中热数据时延迟接近于直接访问SSD（延迟~100us）
  - 在工业界被广泛使用（虚拟化、Ceph、超融合等场景的IO加速）
- 今天主要介绍bcache如何对（以Ceph为例的）软件定义存储系统进行IO性能加速

在什么位置部署Bcache

- 距离数据使用者越近，bcache的加速效果越好
  - 热数据命中时的延迟更低（没有后端处理开销）
  - 可以得到更多具体IO特征，更好的改进cache效率
- 将bcache直接部署在IO请求前端系统上的前提条件
  - 客户端机器的角色固定（配置固态硬盘和内存成本可控）
  - 不同客户端的IO请求彼此没有数据一致性要求

- 距离数据使用者越远，数据处理越抽象（简单）
  - 仍然具有明显的热数据特征
  - 数据服务器上看到的都是IO请求，不必适配各种客户端文件系统
  - 容易维护cache数据的一致性
- 多数Ceph服务商选择在对象存储节点（BlueStore）上部署

Bcache的不同cache模式

Bcache支持如下cache模式：

- Writeback

  回写模式，写请求先存入cache并由回写线程异步写入后端磁盘。

- Writethrough

  写透模式，写请求会同时写入cache设备和后端磁盘。

- Writearound

  绕写模式，写请求直接进入后端磁盘，读请求会被cache。

- None

  读写都不会被cache，同直接使用后端磁盘一样。

（不同cache模式有不同的侧重点）

Bcache可以识别出连续IO来，不将其存入cache而是直接写到后端磁盘，以节省cache空间。

Bcache的cache淘汰算法

当cache存储空间不足时，bcache会根据配置采用如下算法来淘汰干净数据，为新数据让出存储空间：

LRU：将访问频次最低的数据块淘汰

FIFO：将缓存时间最长的数据块淘汰

RANDOM：随机选择数据块淘汰

如下场景使用哪种淘汰算法？

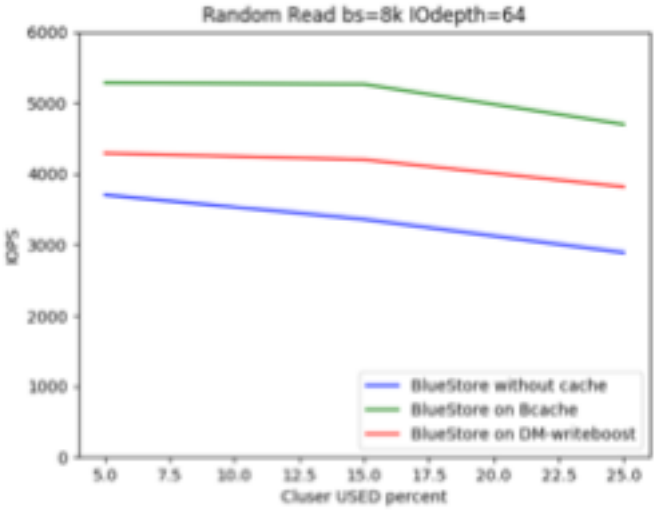
最近3天的数据库表访问最热，5天之前的数据库表几乎没有访问。需要显式的将最近5天的数据库表预热缓存在bcache中，并将5天前的数据表从cache中清理出去。

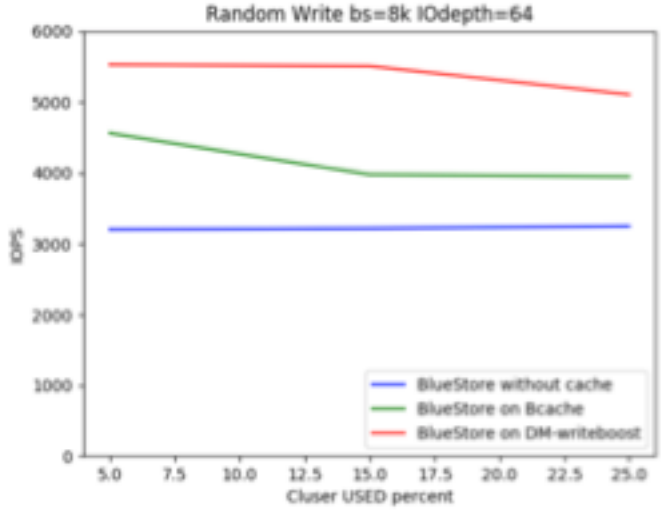
Bcache的特点

- 几乎是目前性能最好的块设备层cache实现。
- 配置灵活。同一个cache设备可以被多个磁盘设备公用，并且cache上的空间也可以动态划分出来独立使用。
- Bcache设备在没有cache设备的时候也可以继续独立访问。
- 有比较完善的设备故障处理机制。

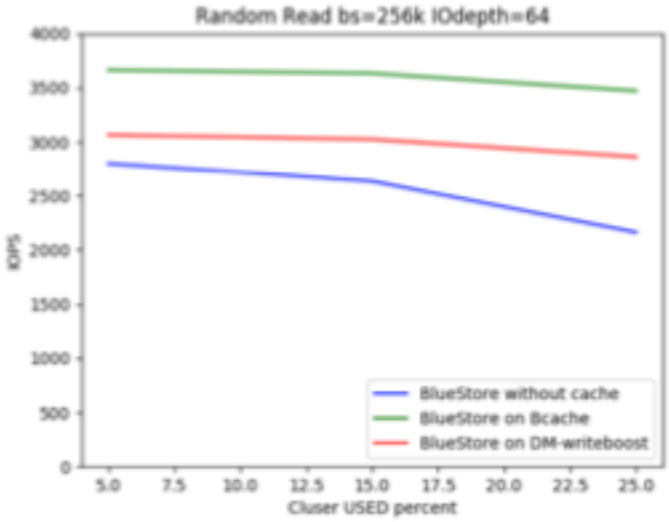
性能对比

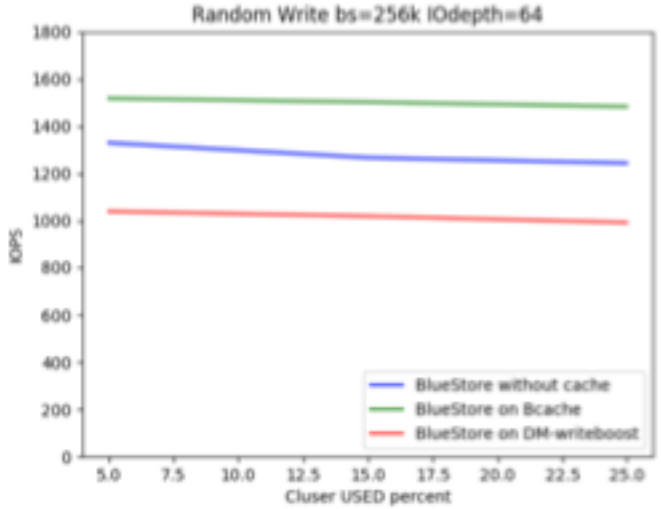
8KB块





256KB块





上图内容引用自张俊钦在Cephalocon APAC 2018的演讲”Linux Block Cache Practice on Ceph BlueStore”

SUSE抽奖活动及规则介绍

参与方式：

①扫描左侧二维码，关注 SUSE 官方微信；

②发送“抽奖”至SUSE官方微信；

③简单填写信息后，进入幸运大转盘抽取礼品；

④凭中奖页面，前往SUSE展台领取礼品。


SUSE
We adapt. You succeed.

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.