CHINA
OpenStack Days

虚拟机的IO模型分析

徐凯
腾讯私有云架构师

01 虚拟机 = 黑盒子？

02 虚拟机读写流程简介

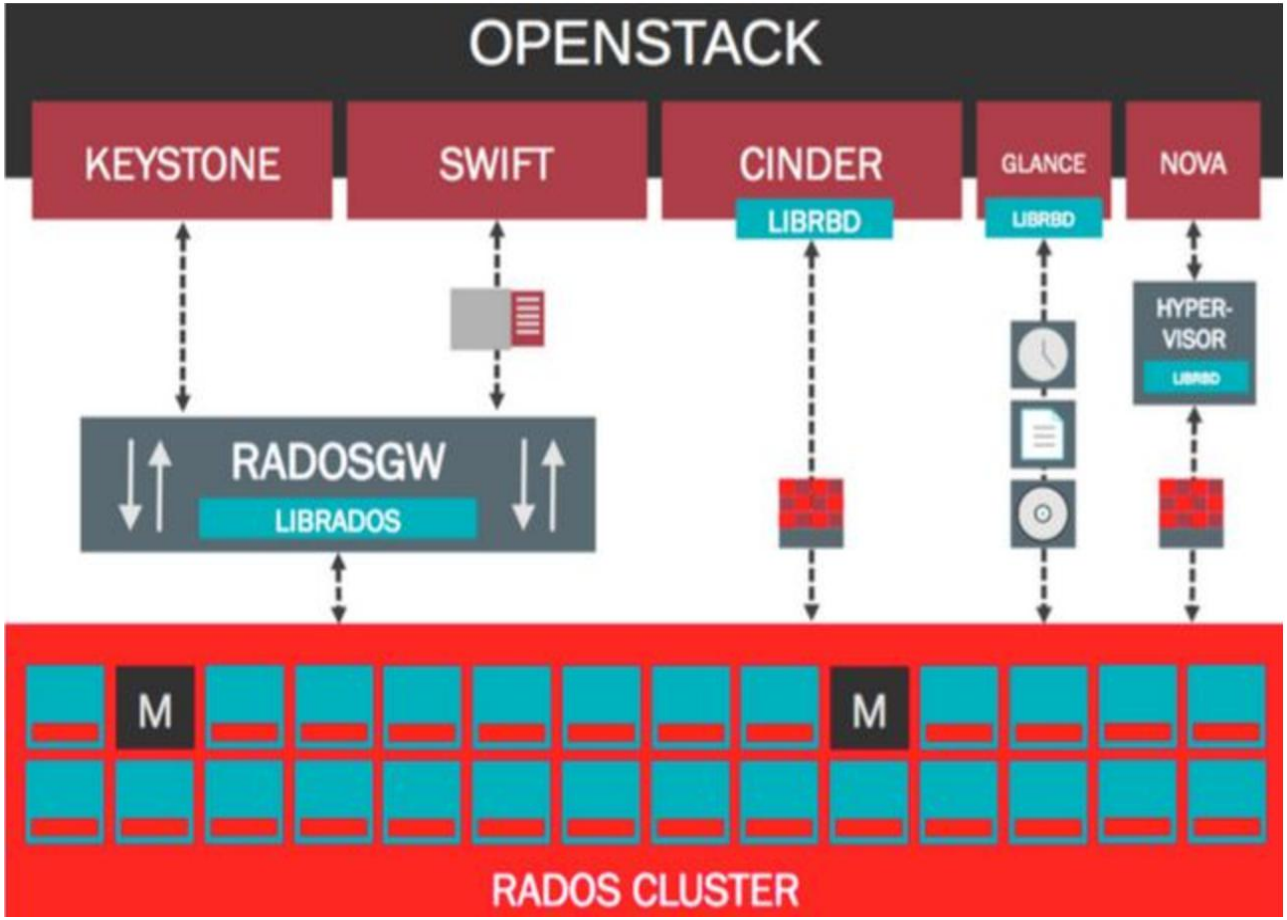
03 截取虚拟机的IO方法

04 几种情形下的虚拟机的IO分析

05 应用场景以及一些思考

腾讯私有云-TStack

01 虚拟机 = 黑盒子？



背景介绍：

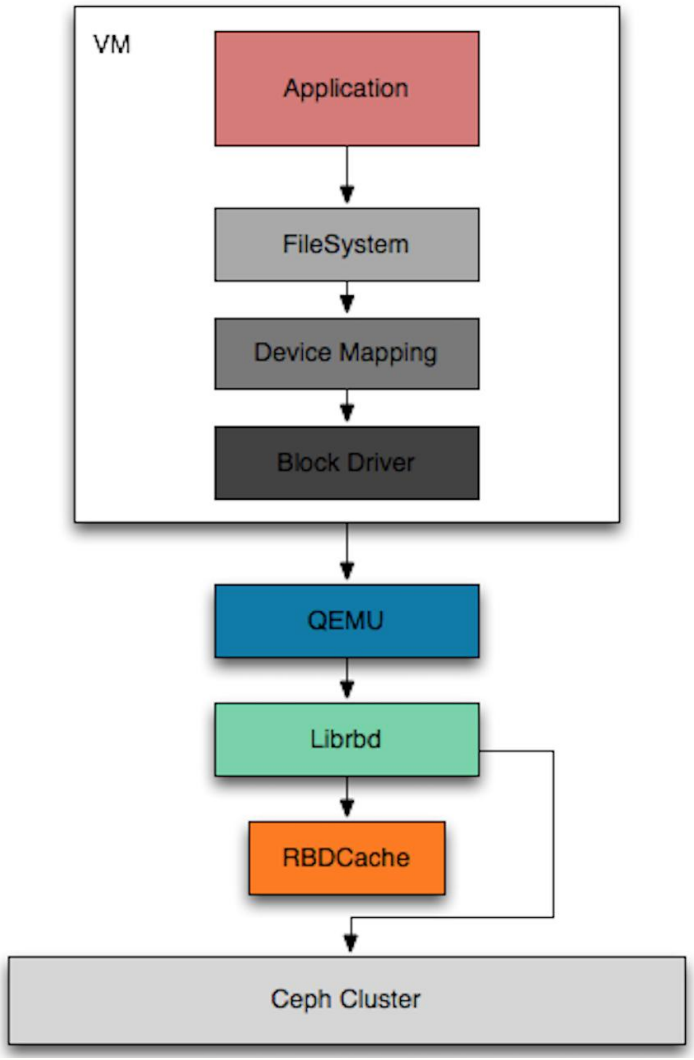
Openstack+Ceph可谓是当前最为火热的云计算组合。

本文的探讨内容即基于该组合下的虚拟机 IO 分析。

01 虚拟机 = 黑盒子？

- 虚拟机给Ceph发送了哪些IO？
  - 基于 Openstack + Ceph 架构。
  - 虚拟机运行于 qemu-kvm。
  - **能否知道虚拟机给Ceph发送的每个IO的用途？**
- 不论白天黑夜，集群的 ops 始终都有两三千?
  - 随着虚拟机的数量增加，ops 均值稳定增加。
  - 虚拟机闲置也会对集群产生 IO吗？
- 虚拟机启动风暴的来源？
  - 同时启动100台虚拟机，集群会产生OSD挂掉？
  - 谁是这场风暴的始作俑者?

虚拟机读写流程简介



- **qemu-kvm** 调用 **librbd** 来读写数据
- **librbd** 调用 **librados** 向Ceph集群读写数据
- librbd 的统一读写接口：
  - **aio_write**
  - **aio_read**
- aio_write/aio_read 会记录每一个IO在RBD内的 **offset, len, time**。

- 暂不关注 rbd_cache 和 rados 层的合并效应。

截取虚拟机IO的方法

- 打开 admin socket

- 默认(绝大多数)情况下，都是关闭的。
- 需要在客户端(计算节点)的ceph.conf 中添加：

```
admin socket=/var/run/qemu/$pid.asok
log file = /var/run/qemu/$pid.log
## debug_rbd=20
```

- 确保 /var/run/qemu 目录存在且权限为 qemu:qemu
- 对于已经启动的虚拟机(未开启socket)的，必须重启虚拟机才行。
- 对于新建的虚拟机，可以正确生成 socket的文件。
- 通过查找对应虚拟机的pid， 可以生成对应的 $pid.asok 文件。

- 打开 log 文件

- 默认情况下，生成的 $pid.log 是空的。
- 包含 aio_wrte/aio_read 的log 级别在 20 ，但是长时间生成log 会占用大量空间。
- 动态开启/关闭 rbd 的log：

```
ceph daemon /var/run/qemu/$pid.asok config set debug_osd 20/20 ( 0/5 关闭 )
```

截取虚拟机IO的方法

aio_write 的 log 形如：

2017-07-21 20:20:01.970549 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22709379072 len = 16384 buf = 0x7f5d5024fc00
2017-07-21 20:20:01.970614 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22732169216 len = 8192 buf = 0x7f5d50253e00
2017-07-21 20:20:01.970677 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22732185600 len = 20480 buf = 0x7f5d50256000
2017-07-21 20:20:01.970749 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22732283904 len = 24576 buf = 0x7f5d5025b200
2017-07-21 20:20:01.970812 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22771761152 len = 8192 buf = 0x7f5d50261400
2017-07-21 20:20:01.970875 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22771810304 len = 8192 buf = 0x7f5d50263600
2017-07-21 20:20:01.970930 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22771826688 len = 16384 buf = 0x7f5d50265800
2017-07-21 20:20:01.970997 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 22792372224 len = 16384 buf = 0x7f5d50269a00

aio_read的 log 形如:

2017-07-21 14:02:04.202375 7f5d2d6f9700 20 librbd: aio_read 0x7f5d4b103240 completion 0x7f5ca0010c60 [132185088,1024]
2017-07-21 14:02:04.203508 7f5d2cef8700 20 librbd: aio_read 0x7f5d4b0f2360 completion 0x7f5d4b1ef9c0 [266698752,26112]
2017-07-21 14:02:04.206529 7f5d2cef8700 20 librbd: aio_read 0x7f5d4b0f2360 completion 0x7f5d4b1ef9c0 [266724864,6656]
2017-07-21 14:02:04.207840 7f5d2cef8700 20 librbd: aio_read 0x7f5d4b0f2360 completion 0x7f5d4b1ef9c0 [26247168,9216]
2017-07-21 14:02:04.210791 7f5d2d6f9700 20 librbd: aio_read 0x7f5d4b103240 completion 0x7f5ca0014110 [26247168,9216]
2017-07-21 14:02:04.212390 7f5d2cef8700 20 librbd: aio_read 0x7f5d4b0f2360 completion 0x7f5d4b1ef9c0 [26256384,23552]
2017-07-21 14:02:04.213289 7f5d2d6f9700 20 librbd: aio_read 0x7f5d4b103240 completion 0x7f5ca0016310 [26256384,23552]
2017-07-21 14:02:04.215230 7f5d2cef8700 20 librbd: aio_read 0x7f5d4b0f2360 completion 0x7f5d4b1ef9c0 [59670528,3072]

03 截取虚拟机IO的方法

- 将虚拟机对应的 RBD 挂载出来
  - 找到虚拟机对应的 qemu-kvm 进程，找到进程中加载的 RBD：

    qemu-kvm -name... file=rbd:volumes/9b7eadca-db9d-4837-909b-835effbb1492_disk ...
  - 挂载这个RBD

    rbd map volumes/9b7eadca-db9d-4837-909b-835effbb1492_disk
    /dev/rbd0

- 如何得到每个IO的内容?

off 为 这个 IO 在 RBD 中的偏移量，**len** 为这个 IO的 大小，单位均为byte。

可以通过下面的指令得到这个IO的内容：

dd bs=1 if=/dev/rbd0 count=${len} skip=${off} of=/root/IO

03 截取虚拟机IO的方法

2017-07-21 14:02:34.271187 7f5d2cef8700 20 librbd: aio_write 0x7f5d4b0f2360 off = 15302053888 len = 12288 buf = 0x7f5d4b227000

dd bs=1 if=/dev/rbd0 count=12288 skip=15302053888 of=/root/IO && cat /root/IO

```
[root@centos ~]# dd bs=1 if=/dev/rbd0   count=12288   skip=15302053888  of=/root/IO
12288+0 records in
12288+0 records out
12288 bytes (12 kB) copied, 0.0342979 s, 358 kB/s
[root@centos ~]# cat IO
[  OK  ] Started Show Plymouth Boot Screen.
[  OK  ] Reached target Paths.
[  OK  ] Reached target Basic System.
[  OK  ] Found device /dev/mapper/centos-root.
         Starting File System Check on /dev/mapper/centos-root...
[  OK  ] Started dracut initqueue hook.
[  OK  ] Reached target Remote File Systems (Pre).
[  OK  ] Reached target Remote File Systems.
systemd-fsck[348]: /sbin/fsck.xfs: XFS file system.
[  OK  ] Started File System Check on /dev/mapper/centos-root.
         Mounting /sysroot...
[  OK  ] Mounted /sysroot.
[  OK  ] Reached target Initrd Root File System.
         Starting Reload Configuration from the Real Root...
[  OK  ] Started Reload Configuration from the Real Root.
[  OK  ] Reached target Initrd File Systems.
[  OK  ] Reached target Initrd Default Target.

Welcome to CentOS Linux 7 (Core)!
```

原来这个IO的内容就是 **CentOS启动界面！** 并且这个IO的**偏移量**始终不变。

几种情形下的虚拟机IO分析

新建的虚拟机静置一小时产生的IO

IO 类型	个数
XFS 相关	65
XFS INODE	56
XFS AG	20
XFS ABTB	14
/var/log/messages	7
/var/log/cron	7
/var/log/audit/audit.log	9
eth0	6
总数	184

04 几种情形下的虚拟机IO分析

新建的虚拟机 关闭crond 静置一小时产生的IO

IO 类型	关闭前个数	关闭后个数
XFS 相关	65	16
XFS INODE	56	12
XFS AG	20	10
XFS ABTB	14	0
/var/log/messages	7	1
/var/log/cron	7	0
/var/log/audit/audit.log	9	1
eth0	6	0
总数	184	40

关闭crond 20min后不再产生aio_write。
说明crond定时写日志的操作会带来xfs层的连带写操作，会产生放大效应。

04 几种情形下的虚拟机IO分析

新建的虚拟机执行 `echo 123456 > /root/testfile` 产生的 IO

IO 类型	个数
XFS 相关	4
XFS INODE	3
XFS AG	7
实际文件内容	1
总数	15

说明实际上，生成一个文件除去**1个**包含文件内容的IO外，还生成了许多XFS文件系统层的IO。

几种情形下的虚拟机IO分析

虚拟机启动（20s）产生的写 IO

IO 类型	IO 大小	IO 个数	IO分布	时长
OS相关	2MB	1		2.5s
OS相关	4KB	384	连续	2.5s
OS相关	4KB	128	连续	1s
日志相关(audit.log, service , dmesg, messages, interface)	4K ~ 8K	25	随机	10s
XFS 相关	4K ~ 8K	58	较为随机	0.01s
总数		596		

虚拟机启动（25s）产生的读 IO 共 9250个，绝大多数为加载库文件和可执行程序。

应用场景

1. 对于不同的类型的虚拟机(web, mysql)，进行 IO 的统计和分析，当样本数量足够多时，就可以估算出某种类型的虚拟机的平均 iops。

2. 通过对所有 IO 的偏移量的统计，可以得到每个 RBD 4M 对象的命中概率，概率异常高的对象可能引发 OSD 热点问题。

3. 可以通过 IO 模型估算出的 iops 值，来预测集群的整体承受能力，比如一个集群能承载多少台混合类型的虚拟机。

05 应用场景以及一些思考

- 限制条件

- 线上集群没有打开socket，无法输出log进行分析。重启虚拟机成本过高。
- 热迁移虚拟机一次可以重新加载相关配置。
- 需要大量的虚拟机的日志文件来分析出IO模型。

- RBD_CACHE

- 在虚拟机启动期间，开启 rbd_cache 可以合并OS相关的512个4K 顺序IO，并且可以为读操作提供缓存。提高了顺序读写的性能。
- 可以增大rbd_cache大小和下刷间隔，但是危险性也增加，可以尝试使用非易失性的SSD而不是内存作为cache。

- XFS

- 每有一个文件写 IO，XFS层会带来较大IO放大消耗，可以尝试从XFS层去降低此类消耗。

- 数据安全性

- **尽管有千万种方法获取数据，但一定要有基本的职业修养！**

06

还有疑惑怎么解？

THANK YOU