# Ceph存储集群

## 配置

### 硬盘和文件系统

#### 准备硬盘

Ceph注重数据安全，就是说，Ceph客户端收到数据已写入存储器的通知时，数据确实已写入硬盘。使用较老的内核（版本小于2.6.33）时，如果日志在原始硬盘上，就要禁用写缓存；较新的内核没问题。**（版本小于2.6.33则要禁用硬盘的写缓冲功能，可以忽略这个问题）**

Sudo hdparm -W 0 /dev/hda 0

**在生产环境，我们建议操作系统和Ceph OSD守护进程数据分别放到不同的硬盘。如果必须把数据和系统放在同一硬盘里，最好给数据分配一个单独的分区。**

#### 文件系统

Ceph的OSD依赖于底层文件系统的稳定性和性能。

**Note：**当前，我们推荐部署生产系统时使用xfs文件系统；推荐使用btrfs做测试、开发和其他不太要紧的部署。我们相信，长远来看btrfs适合Ceph的功能需求和发展方向，但是xfs和ext4能够提供当前部署所必须的稳定性。btrfs开发在迅速推进，所以他的用户应该有能力经常更新到最新内核发布，而且能跟踪严重缺陷的修正进度。

OSD守护进程有赖于底层文件系统的扩展属性（XATTR）存储各种内部对象状态和元数据。底层文件系统必须能为XATTR提供足够容量，btrfs没有限制随文件的xattr元数据量；xfs的限制相对大（64KB），多数部署都不会有瓶颈；ext4的则太小而不可用。**（推荐文件系统使用xfs）**

使用ext4文件系统时，一定要把下面的配置放与ceph.conf配置文件的[osd]段下；用btrfs和xfs时可以选填。

#### 文件系统背景知识

XFS、btrfs和ext4相比较ext3而言，在高伸缩性数据存储方面有几个又是。

XFS、btrfs和ext4都是日志文件系统，这使得在崩溃、断电后恢复时更健壮，因为这些文件系统在写入数据前会先记录所有变更。

Xfs由Silicon Graphics开发，是一个成熟、稳定的文件系统。相反，btrfs是相对年轻的文件系统，他致力于实现系统管理员梦寐以求的大规模数据存储基础，和其他Linux文件系统相比他有独一无二的功能和优势。

btrfs是写时复制（copy-on-write，cow）文件系统，他支持文件创建时间戳和校验和（可验证元数据完整性）功能，所以他能探测到数据副本，并且用好副本修复。写时复制功能是说btrfs支持科协文件系统快照。Btrfs也支持透明压缩和其他功能。

btrfs也集成了多设备管理功能，据此可以在底层支持异质硬盘存储，和数据分配策略。未来开发社区还会提供fsck、拆分、数据加密功能，这些诱人的功能正式Ceph集群的理想选择。

### 配置Ceph

你启动Ceph服务时，初始化进程会把一系列守护进程放到后台运行。Ceph存储集群运行两种守护进程：

Ceph监视器（Ceph-mon）

Ceph OSD守护进程（ceph-osd）

要支持Ceph文件系统功能，他还需要运行至少一个Ceph元数据服务器（ceph-mds）；支持Ceph对象存储功能的集群还要运行网关守护进程（radosgw）。为方便起见，各类守护进程都有一系列默认值（很多由ceph/src/common/config\_opts.h）配置，你可以用Ceph配置文件覆盖这些默认值。

#### 配置文件

启动Ceph存储集群时，各守护进程都从同一个配置文件（即默认的ceph.conf）里查找他自己的配置。手动部署时，你需要创建此配置文件；用部署工具（如ceph-deploy、chef等）创建配置文件时可以参考下面的信息。配置文件定义了：

* 集群身份
* 认证配置
* 集群成员
* 主机名
* 主机IP地址
* 秘钥环路径
* 日志路径
* 数据路径
* 其他运行时选项

默认的Ceph配置文件位置相继排列如下：

1. $CEPH\_CONF（就是$CEPH\_CONF环境变量所指示的路径）
2. -c path/path（就是-c命令行参数）
3. /etc/ceph/ceph.conf
4. ~/.ceph/config
5. ./ceph.conf（就是当前工作所在的工作路径）

Ceph配置文件使用ini风格的语法，以分好（；）和井号（#）开始的行时注释，如下：

*# <--A number (#) sign precedes a comment.; A comment may be anything.# Comments always follow a semi-colon (;) or a pound (#) on each line.# The end of the line terminates a comment.# We recommend that you provide comments in your configuration file(s).*

#### 配置段落

Ceph配置文件可用于配置存储集群内的所有守护进程、或者某一类型的所有守护进程。要配置一系列守护进程，这些配置必须位于能收到配置的段落之下，比如：

[global]

**描述：**[global]下的配置影响Ceph集群里的所有守护进程

**实例：**auth supported = cephx

[osd]

**描述：**[osd]下的配置影响存储集群里的所有ceph-osd进程，并且会覆盖[global]下的同一选项。

**实例：**osd journal size = 1000

[mon]

**描述：**[mon]下的配置影响集群里的所有ceph-mon进程，并且会覆盖[global]下的同一选项。

**实例：**mon addr = 10.0.0.101:6789

[mds]

**描述：**[mds]下的配置影响集群里的所有ceph-mds进程，并且会覆盖[global]下的同一选项。

**实例：**host = myserver01

[client]

**描述：**[client]下的配置影响所有客户端（如挂载的Ceph文件系统、挂载的块设备等等）。

**实例：**log file = /var/log/ceph/radosgw.log

全局设置影响集群内所有守护进程，所以[global]可用于设置适用所有守护进程的选项。但可以用这些覆盖[global]设置：

1. [osd]、[mon]、[mds]下更改某一类进程的配置。
2. 更改特定进程的设置，如[osd.1]。

覆盖全局设置会影响所有子进程，明确提出的例外。

典型的全局设置包括激活认证，例如：

**[global]**

*#Enable authentication between hosts within the cluster.*

*#v 0.54 and earlier*

auth supported = cephx

*#v 0.55 and after*

auth cluster required = cephx

auth service required = cephx

auth client required = cephx

你可以统一配置一类守护进程。配置写到[osd]、[mon]、[mds]下时，无须再指定某个特定例程，即可分别影响所有OSD、监视器、元数据进程。

典型的类范畴配置包括日志尺寸、filestore选项等，如：

**[osd]**osd journal size = 1000

你也可以配置某个特定例程。一个例程由类型及其例程ID确定，OSD的例程ID只能是数字，但监视器和元数据服务器的ID可包含字母和数字。

**[osd.1]***# settings affect osd.1 only.*

**[mon.a]***# settings affect mon.a only.*

**[mds.b]***# settings affect mds.b only.*

如果你想配置某个Ceph网关客户端，可以用点（.）分隔的守护进程和例程来指定，例如：

**[client.radosgw.instance-name]***# settings affect client.radosgw.instance-name only.*

#### 元变量

元变量大大简化了集群配置。Ceph会把配置的元变量展开为具体值；元变量功能很强大，可以用在配置文件的[global]、[osd] 、 [mon] 、 [mds]段里，类似于Bash的shell扩展。

Ceph支持下列元变量。

**$cluster**

**描述：**展开为存储集群名字，在同一套硬件上运行多个集群时有用

**实例：**/etc/ceph/$cluster.keyring

**默认值：**ceph

**$type**

**描述：**可展开为mds，osd，mon中的一个，依赖于当前守护进程的类型

**实例：**/var/lib/ceph/$type

**$id**

**描述：**展开为守护进程标识符；osd.0应为0，mds.a是a。

**实例：**/var/lib/ceph/$type/$cluster-$id

**$host**

**描述：**展开为当前守护进程的主机名

**$name**

**描述：**展开为$type.$id

实例：/var/run/ceph/$cluster-$name.asok

#### 共有选项

硬件推荐段提供了一些配置Ceph存储集群的硬件指导。一个Ceph节点可以运行多个进程，例如一个节点有多个硬盘，可以为每个硬盘配置一个ceph-osd守护进程。理想情况下一台主句应该只运行一类进程，例如：一台主机运行着ceph-osd进程，另一台主机运行着ceph-mds进程，ceph-mon进程又在另外一台主机上。

各节点都用host选项指定主机名字，监视器还需要用addr选项指定网络地址和端口（即域名或IP地址）。基本配置文件可以智能指定最小配置，例如：

[global]

mon\_initial\_members = ceph1

mon\_host = 10.0.0.1

Important：host选项是此节点的短名字，不是全资域名（FADN），也不是IP地址；执行hostname -s即可得到短名字。不要给初始监视器之外的例程设置host，除非你想手动部署；一定不能用与chef或ceph-deploy，这些工具会自动获取正确结果。

#### 监视器

典型的Ceph生产集群至少部署3个监视器来确保高可靠性，他允许一个监视器例程崩溃。奇数个监视器（3个）确保PAXOS算法能确定一批监视器里那个版本的集群运行图是最新的。

Note：一个Ceph集群可以只有一个监视器，但是如果他失败了，因没有监视器数据服务就会中断。

Ceph监视器默认监听6789端口，例如：

**[mon.a]**host = hostNamemon addr = 150.140.130.120:6789

默认情况下，Ceph会在下面的路径存储监视器数据：

/var/lib/ceph/mon/$cluster-$id

你必须手动或通过部署工具（如ceph-deploy）创建对应目录。前述元变量必须先全部展开，名为“ceph”的集群将展开为：

/var/lib/ceph/mon/ceph-a

#### 认证

对于v0.56及后来版本，要在配置文件的[global]中明确启用或禁用认证。

auth cluster required = cephx

auth service required = cephx

auth client required = cephx

另外，你应该启动消息签名，详情见Cephx配置参考和Cephx认证。

Important：我们建议，升级时先明确的关闭认证，再进行升级。等升级完成后再重新启用认证。

#### OSDS

通常，Ceph生产集群在一个节点上只运行一个Ceph OSD守护进程，此守护进程在一个存储驱动器上只运行一个filestore；典型部署需要指定日志尺寸。例如：

**[osd]**osd journal size = 10000

**[osd.0]**host = {hostname} #manual deployments only.

默认情况下，Ceph认为你把OSD数据存储到了一下路径：

/var/lib/ceph/osd/$cluster-$id

你必须手动或通过部署工具（如ceph-deploy）创建对应目录，名为“ceph”的集群其元变量完全展开后，前述的目录结构将是：

/var/lib/ceph/osd/ceph-0

你可以用osd data 选项更改默认值，但我们不建议修改。用下面的命令在新OSD主机上创建默认目录：

ssh {osd-host}

sudo mkdir /var/lib/ceph/osd/ceph-{osd-number}

osd data路径应该纸箱一个独立硬盘的挂载点，这个硬盘应该独立于操作系统和守护进程所在的硬盘。按下列步骤准备好挂载：

ssh {new-osd-host}

sudo mkfs -t {fstype} /dev/{disk}

sudo mount -o user\_xattr /dev/{hdd} /var/lib/ceph/osd/ceph-{osd-number}

#### 心跳

在运行时，OSD守护进程会相互检查邻居OSD、并把结果报告给Ceph监视器，一般不需要更改默认配置。但如果你的网络延时比较大，也许需要更改某些选项。其他细节部分见监视器与OSD交互的配置。

#### 日志、调试

有时候你可能遇到一些麻烦，需要修改日志或调试选项，请参考调试和日志记录。

#### CEPF.CONF实例

**[global]**

fsid = {cluster-id}

mon initial members = {hostname}[, {hostname}]

mon host = {ip-address}[, {ip-address}]

*#All clusters have a front-side public network.#If you have two NICs, you can configure a back side cluster #network for OSD object replication, heart beats, backfilling,#recovery, etc.*

public network = {network}[, {network}]

*#cluster network = {network}[, {network}]*

*#Clusters require authentication by default.*

auth cluster required = cephx

auth service required = cephx

auth client required = cephx

*#Choose reasonable numbers for your journals, number of replicas#and placement groups.*

osd journal size = {n}

osd pool default size = {n}

# Write an object n times.

osd pool default min size = {n}

# Allow writing n copy in a degraded state.

osd pool default pg num = {n}

osd pool default pgp num = {n}

*#Choose a reasonable crush leaf type.*

*#0 for a 1-node cluster.*

*#1 for a multi node cluster in a single rack*

*#2 for a multi node, multi chassis cluster with multiple hosts in a chassis*

*#3 for a multi node cluster with hosts across racks, etc.*

osd crush chooseleaf type = {n}

#### 运行时更改

Ceph可以在运行时更改ceph-osd、ceph-mon、ceph-mds守护进程的配置，此功能在增加/降低日志输出、启用/禁用调试设置、甚至是运行时优化的时候非常有用，下面是运行时配置方法：

ceph tell {daemon-type}.{id or \*} injectargs --{name} {value} [--{name} {value}]

用osd、mon、mds中的一个代替{daemon-type}，你可以用星号（\*）更改一类进程的所有例程配置、或者更改某一具体进程ID（即数字或字母）的配置。例如提高名为osd.0的ceph-osd进程之调试级别的命令如下：

ceph tell osd.0 injectargs --debug-osd 20 --debug-ms 1

在ceph.conf文件里配置时用空格分隔关键字；但在命令行使用的时候要用下划线或连字符（\_或-）分隔，例如debug\_osd变成debug-osd。

#### 查看运行时配置

如果你的Ceph存储集群在运行，而你想看一个在运行进程的配置，用下面的命令：

ceph daemon {daemon-type}.{id} config show | less

如果你现在位于osd.0所在的主机，命令将是：

ceph daemon osd.0 config show | less

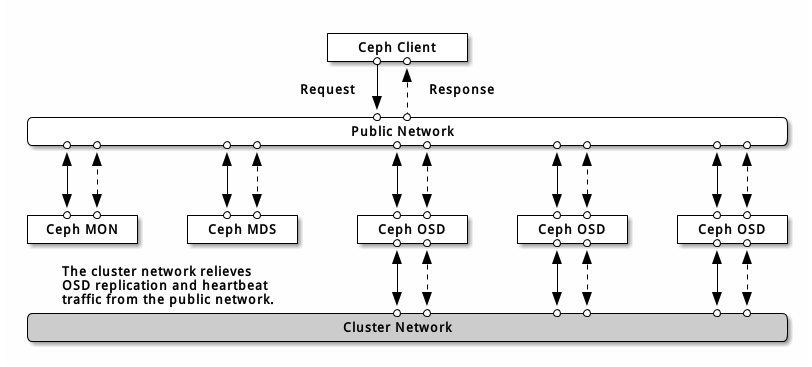
### 网络选项

#### 网络配置参考

网路配置对构建高性能Ceph存储集群来说相当重要。Ceph存储集群不会代表Ceph客户端执行请求路由或调度，相反，Ceph客户端（如设备、CephFS、REST网关）直接向OSD请求，然后向OSD为客户端执行数据复制，也就是说复制和其他因素会额外增加集群网的负载。

我们快速入门配置提供了一个简陋的Ceph配置文件，其中只设置了监视器IP地址和守护进程所在的主机名。如果没有配置集群网，那么Ceph假设你只有一个“公共网”。只用一个网可以运行Ceph，但是在大型集群里面单独的“集群”网可显著的提升性能。

我们建议用两个网络运营Ceph存储集群：一个公共网络（前端）和一个集群网（后端）。为此，各节点得配置多个网卡，见硬件推荐—网络。



运营两个独立网络的考量主要有：

1. 性能：OSD为客户端处理数据复制，复制多份时OSD间的网络负载势必会影响到客户端和Ceph集群的通讯，包括延时增加、产生性能问题；恢复和重均衡也会显著增加公共网络延时。关于Ceph如果复制参见伸缩性和高可用性；关于心跳流量参见监视器与OSD交互。
2. 安全：大多数人都是良民，很少的一撮人喜欢折腾拒绝服务攻击（DoS）。当OSD间的流量失控时，归置组再也不能达到active-clean状态，这样用户就不能读写数据了。挫败此类攻击的一种好方法是维护一个完全独立的集群网，使之不能直连互联网；另外，请考虑使用消息签名防止欺骗攻击。

#### 防火墙

守护进程默认会绑定到6800:7300间的端口，你可以更改此范围。更改防火墙配置前先检查下iptables配置

Sudo iptables -L

一些Linux发行版的规则拒绝出SSH之外的所有网卡的所有入栈连接，例如：

REJECT all -- anywhere anywhere reject-with icmp-host-prohibited

你得先删除掉公共网络和集群网对应的这些规则，然后再增加安全保护规则。

**监视器防火墙：**

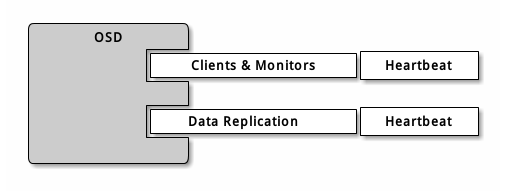
监视器默监听6789端口，而且监视器总是运行在公共网。按下例增加规则时，要把{iface}替换为公共网接口（如eth0，eth1等等）、{ip-address}替换为公共网IP，{netmask}替换为公共网掩码。

sudo iptables -A INPUT -i {iface} -p tcp -s {ip-address}/{netmask} --dport 6789 -j ACCEPT

**OSD防火墙**

OSD守护进程默认绑定从6800起的第一个可用端口，需要注意的是，这种行为是不确定的，所以如果你在同一主机运行多个OSD或MDS、或者再很短的时间内重启了多个守护进程，他们会绑定更高的端口号。一主机的各个OSD最多会用到4个端口：

1. 一个用于和客户端、监视器通讯；
2. 一个用于发送数据到其他OSD；
3. 两个用于各个网卡上的心跳；



当某个守护进程失败并重启时没释放端口，重启后的进程就会监听新端口。你应该打开整个6800-7300端口区间，以应对这种可能性。

如果你分开了公共网和集群网，必须分别位置设置防火墙，因为客户端会通过公共网连接、而其他OSD会通过集群网连接。按下例增加规则时，要把{iface}替换为网口（如 eth0 、 eth1 等等）、 {ip-address} 替换为公共网或集群网 IP 、 {netmask} 替换为公共网或集群网掩码。例如：

sudo iptables -A INPUT -i {iface} -m multiport -p tcp -s {ip-address}/{netmask} --dports 6800:7300 -j ACCEPT

Tip：如果你的元数据服务器和OSD在同一节点上，可以合并公共网络配置。

#### CEPH网络

Ceph的网络配置要放到[global]段下。前述的5分钟快速入门提供了一个简陋的Ceph配置文件，他假设服务器和客户端都位于同一网段，Ceph可很好的适应这种情形。然而Ceph允许配置更精细的公共网，包括多IP和掩码；也能用单独的集群网处理OSD心跳、对象复制、和恢复流量。不要混淆你配置的IP地址和客户端用来访问存储服务的公共网地址。典型的内网常常是192.168.0.0或10.0.0.0。

Tip：如果你给公共网或集群网配置了多个IP地址及子网掩码，这些子网必须能互通。另外要确保在防火墙上为各IP和子网开放了必要的端口。

Note：Ceph用CIDR法表示子网，如10.0.0.0/24、

配置完几个网络后，可以重启集群或挨个重启守护进程。Ceph守护进程动态的绑定端口，所以更改网络陪之后无需重启整个集群。

公共网

要配置一个公共网，把下列选项加到配置文件的[global]段下。

[global]

...

public network = {public-network/netmask}

集群网

如果你生命了集群网，OSD将把心跳、对象复制和恢复流量路由到集群网，把下列选项加进配置文件的[global]段

[global]

...

cluster network = {cluster-network/netmask}

为安全起见，从公共网或互联网到集群网应该是不可达的。

#### CEPH守护进程

有一个网络配置是所有守护进程都要配置的：各个守护进程都必须指定host，Ceph也要求指定监视器IP地址及端口。

Important：一些部署工具（如ceph-deploy、Chef）会给你创建配置文件，如果他能胜任那就别设置这些值。

Tip：host选项是主机的短名，不是全资域名FQDN，也不是IP地址。在命令行下输入hostname -s获取主机名。

[mon.a]

host = {hostname} mon addr = {ip-address}:6789

[osd.0]

host = {hostname}

并非一定要给守护进程设置IP地址。如果你有一个静态配置，且分离了公共网和集群网，Ceph允许你在配置文件里指定主机的IP地址。要给守护进程设置静态IP，可把下列选项加到ceph.conf。

[osd.0]

public addr = {host-public-ip-address} cluster addr = {host-cluster-ip-address}

单网卡ODS、双网络集群

一般来说，我们不建议用单网卡OSD主机部署两个网络。然而这事可以实现，把public addr选项配在[osd.n]段下即可强制OSD主机运行在公共网，其中n是其OSD号。另外，公共网络和集群网必须互通，考虑到安全因素我们不建议这样做。

#### 网络配置选项

公共网

公共网配置用于明确的为公共网定义IP地址和子网。你可以分配静态IP或用public addr覆盖public network选项

**public network**

|  |  |
| --- | --- |
| 描述: | 公共网（前端）的 IP 地址和掩码（如 192.168.0.0/24 ），置于 [global] 下。多个子网用逗号分隔。 |
| 类型: | {ip-address}/{netmask} [, {ip-address}/{netmask}] |
| 是否必需: | No |
| 默认值: | N/A |

**public addr**

|  |  |
| --- | --- |
| 描述: | 用于公共网（前端）的 IP 地址。适用于各守护进程。 |
| 类型: | IP 地址 |
| 是否必需: | No |
| 默认值: | N/A |

**集群网**

集群网配置用来声明一个集群网，并明确地定义其 IP 地址和子网。你可以配置静态 IP 或为某 OSD 守护进程配置 cluster addr 以覆盖 cluster network 选项。

cluster network

|  |  |
| --- | --- |
| 描述: | 集群网（后端）的 IP 地址及掩码（如 10.0.0.0/24 ），置于 [global] 下。多个子网用逗号分隔。 |
| 类型: | {ip-address}/{netmask} [, {ip-address}/{netmask}] |
| 是否必需: | No |
| 默认值: | N/A |

**cluster addr**

|  |  |
| --- | --- |
| 描述: | 集群网（后端） IP 地址。置于各守护进程下。 |
| 类型: | Address |
| 是否必需: | No |
| 默认值: | N/A |

**绑定**

绑定选项用于设置 OSD 和 MDS 默认使用的端口范围，默认范围是 6800:7300 。确保[防火墙](http://docs.ceph.org.cn/rados/configuration/network-config-ref/" \l "id2)开放了对应端口范围。

你也可以让 Ceph 守护进程绑定到 IPv6 地址。

**ms bind port min**

|  |  |
| --- | --- |
| 描述: | OSD 或 MDS 可绑定的最小端口号。 |
| 类型: | 32-bit Integer |
| 默认值: | 6800 |
| 是否必需: | No |

**ms bind port max**

|  |  |
| --- | --- |
| 描述: | OSD 或 MDS 可绑定的最大端口号。 |
| 类型: | 32-bit Integer |
| 默认值: | 7300 |
| 是否必需: | No. |

**ms bind ipv6**

|  |  |
| --- | --- |
| 描述: | 允许 Ceph 守护进程绑定 IPv6 地址。 |
| 类型: | Boolean |
| 默认值: | false |
| 是否必需: | No |

**主机**

Ceph 配置文件里至少要写一个监视器、且每个监视器下都要配置 mon addr 选项；每个监视器、元数据服务器和 OSD 下都要配 host 选项。

**mon addr**

|  |  |
| --- | --- |
| 描述: | {hostname}:{port} 条目列表，用以让客户端连接 Ceph 监视器。如果未设置， Ceph 查找 [mon.\*] 段。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | N/A |

**host**

|  |  |
| --- | --- |
| 描述: | 主机名。此选项用于特定守护进程，如 [osd.0] 。 |
| 类型: | String |
| 是否必需: | Yes, for daemon instances. |
| 默认值: | localhost |

Tip：不要用localhost。在命令行下执行hostname -s获取主机名（到第一个点，不是全资域名），并用于配置文件。

Important：用第三方部署工具时不要指定host选项，他会自行获取。

#### TCP

Ceph默认禁用TCP缓冲。

ms tcp nodelay

**描述：**Ceph用ms tcp nodelay使系统尽快（不缓冲）发送每个请求。禁用Nagle算法可增加吞吐量，但会引进延时。如果你遇到大量小包，可以禁用ms tcp nodelay试试

**类型：**Boolean

**是否必须：**No

**默认值：**true

ms tcp rcvbuf

描述：网络套接字接收缓冲池尺寸，默认禁用。

类型：32-bit Integer

是否必须：No

默认值：0

ms tcp timeout

描述：如果一客户端或守护进程发送请求到另一个Ceph守护进程，且没有断开不再使用的连接，在ms tcp read timeout指定的秒数之后他将被标记为空闲。

类型：Unsigned 64-bit Integer

是否必须：No

默认值：900 15 minutes

### 认证选项：

#### CEPHX配置参考

（认证配置）

Cephx协议已默认开启。加密认证要耗费一定计算资源，但通常很低。如果你的客户端和服务器网络环境相当安全，而且认证的负面效应更大，你可以关闭它，通常不推荐这么做。

Note：如果禁用了认证，就会有篡改客户端/服务器消息这样的中间人攻击风险，这回导致灾难性后果。

关于创建用户请参考用户管理；关于Cephx的体系结构请参考体系结构——高可用性认证。

#### 部署场景

有两种主要的部署Ceph集群的方案，这回影响你最初配置Cephx的方式。大多数Ceph用户首次使用ceph-deploy创建一个集群（最简单）。对于使用其他部署工具（例如Chef，Juju，Pupper等）的集群，您将需要使用手动过程或配置部署工具来引导您的Monitor。

**CEPH-DEPLOY**

使用ceph-deploy部署集群时，你不必手动引导Monitor，也不必创建client.admin用户或秘钥环。您在存储集群入门中执行的步骤将调用ceph-deploy来为你执行此操作。

当你执行ceph-deploy new {initial-monitor(s)}时，Ceph将为你创建一个监视器秘钥环（只用于启动监视器），并且会为你生成一个初始的Ceph配置文件，他包含以下认证设置，表明Ceph默认启用认证：

auth\_cluster\_required = cephx

auth\_service\_required = cephx

auth\_client\_required = cephx

当你执行ceph-deploy mon create-initial时，Ceph将引导初始Monitors，检索包含client.admin用户秘钥的ceph.client.admin.keyring。此外，他还将检索是ceph-deploy和ceph-disk使用程序能够准备和激活OSD和元数据服务器的秘钥环。

当你执行ceph-deploy admin {node-name}时（注意：必须先安装Ceph），你需要将Ceph配置文件和ceph.client.admin.keyring推送到节点的/etc/ceph目录下。你将能够在该节点的命令行以上root身份执行Ceph管理功能。

**手动部署**

在手动部署集群时，必须手动引导Monitors并创建client.admin用户和密钥匙。要自举Monitors,请按照Monitors的自举引导中的步骤操作。监视器引导的步骤是使用第三方部署工具是必须执行的逻辑步骤。

#### 启用和禁用CEPHX

启用Cephx需要你为Monitors，OSD和元数据服务器部署秘钥。如果你只是打开/关闭Cephx，则不必重复自举程序。

启用CEPHX

启用cephx后，Ceph将在默认的搜索路径（包括/etc/ceph/ceph.$name.keyring）里查找秘钥环。你可以在Ceph配置文件的[global]段里添加keyring选项来修改，但不推荐。

在禁用了cephx的集群上执行下面的步骤来启用它，如果你（或者部署工具）已经生成了秘钥，你可以跳过相关步骤。

1. 创建client.admin秘钥，并为客户端保存此秘钥的副本

ceph auth get-or-create client.admin mon 'allow \*' mds 'allow \*' osd 'allow \*' -o /etc/ceph/ceph.client.admin.keyring

警告：此命令会覆盖任何存在的/etc/ceph/client.admin.kerying文件，如果部署工具已经完成此步骤，千万别再执行此命令。多加小心！

1. 创建监视器集群所需的秘钥环、并给他们生成秘钥

创建监视器集群所需的秘钥环、并给他们生成秘钥。

ceph-authtool --create-keyring /tmp/ceph.mon.keyring --gen-key -n mon. --cap mon 'allow \*'

1. 把监视器秘钥复制到ceph.mon.keyring文件，再把此文件复制到各监视器的mon data目录下。比如要把他复制给名为ceph集群的mon.a，用此命令：

cp /tmp/ceph.mon.keyring /var/lib/ceph/mon/ceph-a/keyring

1. 为每个OSD生成秘钥，{$id}是OSD编号：

ceph auth get-or-create osd.{$id} mon 'allow rwx' osd 'allow \*' -o /var/lib/ceph/osd/ceph-{$id}/keyring

1. 为每个MDS生成秘钥，{$id}是OSD编号：

ceph auth get-or-create mds.{$id} mon 'allow rwx' osd 'allow \*' mds 'allow \*' -o /var/lib/ceph/mds/ceph-{$id}/keyring

1. 把以下配置加入Ceph配置文件的[global]段以下启用cephx认证

auth cluster required = cephx

auth service required = cephx

auth client required = cephx

1. 启动或重启Ceph集群，详情见操纵集群。
2. 要手动自启监视器，请参考手动部署。

**禁用CEPHX**

下述步骤描述了如何禁用Cephx。如果你的集群环境相对安全，你可以见面认证耗费的计算资源，然而我们不推荐。但是临时禁用认证会是安装、和/或排障更简单。

1. 把下列配置加入Ceph配置文件的[global]段下即可禁用cephx认证：

auth cluster required = none

auth service required = none

auth client required = none

1. 启动或重启Ceph集群，具体参考操纵集群。

#### 配置选项

**启用事项**

**auth cluster required**

|  |  |
| --- | --- |
| 描述: | 如果启用了，集群守护进程（如 ceph-mon 、 ceph-osd 和 ceph-mds ）间必须相互认证。可用选项有 cephx 或 none 。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | cephx. |

**auth service required**

|  |  |
| --- | --- |
| 描述: | 如果启用，客户端要访问 Ceph 服务的话，集群守护进程会要求它和集群认证。可用选项为 cephx 或 none 。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | cephx. |

**auth client required**

|  |  |
| --- | --- |
| 描述: | 如果启用，客户端会要求 Ceph 集群和它认证。可用选项为 cephx 或 none 。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | cephx. |

**秘钥：**

如果你的集群启用了认证，ceph管理命令和客户端得有秘钥才能访问集群。

给ceph管理命令和客户端提供秘钥的最常用方法就是把秘钥环放到/etc/ceph，通过ceph-deploy部署的Cuttlefish即更高版本，其文件名称通常是ceph.client.admin.keyring（或$cluster.client.admin.keyriing）。如果你的秘钥环位于/etc/ceph下，就不需要在Ceph配置文件里指定keyring选项了。

我们建议把集群的秘钥环复制到你执行管理命令的节点，他包含client.admin秘钥。

你可以用ceph-deploy admin命令做此事，详情见“部署管理主机”，手动复制可执行此

命令：

sudo scp {user}@{ceph-cluster-host}:/etc/ceph/ceph.client.admin.keyring /etc/ceph/ceph.client.admin.keyring

Tip：确保给客户端上的ceph.keyring设置合理的权限位（如chmod 644）

你可以用key选项把秘钥卸载配置文件里（别这样），或者用keyfile选项指定个路径

keyring

|  |  |
| --- | --- |
| 描述: | 密钥环文件的路径。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | /etc/ceph/$cluster.$name.keyring,/etc/ceph/$cluster.keyring,/etc/ceph/keyring,/etc/ceph/keyring.bin |

keyfile

|  |  |
| --- | --- |
| 描述: | 到密钥文件的路径，如一个只包含密钥的文件。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | None |

key

|  |  |
| --- | --- |
| 描述: | 密钥（密钥文本），最好别这样做。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | None |

**守护进程秘钥环：**

管理员用户们或部署工具（如ceph-deploy）生成守护进程秘钥环与生成用户秘钥环的方法一样。默认情况下，守护进程把秘钥环保存在各自的数据目录下，默认秘钥环位置、和守护进程发挥作用必须的能力展示如下：

ceph-mon

|  |  |
| --- | --- |
| 位置: | $mon\_data/keyring |
| 能力: | mon 'allow \*' |

ceph-osd

|  |  |
| --- | --- |
| 位置: | $osd\_data/keyring |
| 能力: | mon 'allow profile osd' osd 'allow \*' |

ceph-mds

|  |  |
| --- | --- |
| 位置: | $mds\_data/keyring |
| 能力: | mds 'allow' mon 'allow profile mds' osd 'allow rwx' |

radosgw

|  |  |
| --- | --- |
| 位置: | $rgw\_data/keyring |
| 能力: | mon 'allow rwx' osd 'allow rwx' |

Note：监视器秘钥环（即mon.）包含一个秘钥，但没有能力，且不是集群auth数据库的一部分。

守护进程数据目录位置默认格式如下：

/var/lib/ceph/$type/$cluster-$id

例如，osd.12的目录是：

/var/lib/ceph/osd/ceph-12

你可以覆盖这些位置，但不推荐。

**签名**

在Bobtail及后续版本中，Ceph会用开始认证时生成的会话秘钥认证所有在线实体。然后Argonaut及之前版本不知道如何认证在线消息，为保持向后兼容性（如在同一个集群里运行Bobtail和Argonaut），消息签名默认是关闭你的。如果你只运行Bobtail和后续版本，可以让Ceph要求签名。

像Ceph认证的其他部分一样，客户端和集群间的消息签名也能做到细粒度控制；而且能启用或禁用Ceph守护进程间的签名。

cephx require signatures

|  |  |
| --- | --- |
| 描述: | 若设置为 true ， Ceph 集群会要求客户端签名所有消息，包括集群内其他守护进程间的。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

cephx cluster require signatures

|  |  |
| --- | --- |
| 描述: | 若设置为 true ， Ceph 要求集群内所有守护进程签名相互之间的消息。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

cephx service require signatures

|  |  |
| --- | --- |
| 描述: | 若设置为 true ， Ceph 要求签名所有客户端和集群间的消息。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

cephx sign messages

|  |  |
| --- | --- |
| 描述: | 如果 Ceph 版本支持消息签名， Ceph 会签名所有消息以防欺骗。 |
| 类型: | Boolean |
| 默认值: | true |

**生存期**

auth service ticket ttl

|  |  |
| --- | --- |
| 描述: | Ceph 存储集群发给客户端一个用于认证的票据时分配给这个票据的生存期。 |
| 类型: | Double |
| 默认值: | 60\*60 |

#### 后兼容性

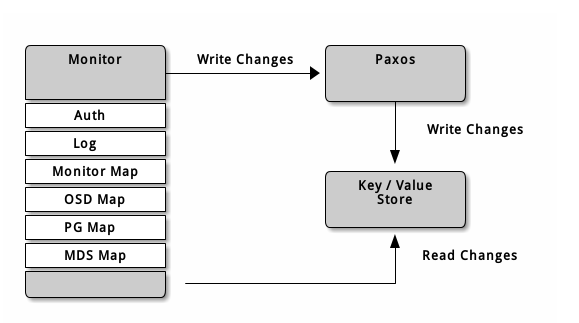
### 监视器选项

理解如何配置Ceph监视器是构建可靠的Ceph存储集群的重要方面，任何Ceph集群都需要至少一个监视器。一个监视器通常相当一致，但是你可以增加，删除，或替换集群中的监视器，详情见**增加/删除监视器和增加/删除监视器（ceph-deploy）。**

#### 背景

监视器们维护着集群运行图的“主副本”，就是说客户端连到一个监视器并获取当前运行图就能确定所有监视器、OSD和元数据服务器的位置。Ceph客户端读写OSD或元数据服务器前，必须先连到一个监视器，靠当前集群运行图的副本和CRUSH算法，客户端能计算出任何对象的位置，故此客户端有能力直接连到OSD，这对Ceph的高伸缩性、高性能来说非常重要。更多信息间**伸缩性和高可用性**。

监视器的主要角色是维护集群运行图的主副本，他也提供认证和日志记录服务。Ceph监视器们把监视器服务的所有更改写入一个单独的Paxos例程，然后Paxos以键值对方式存储所有变更以实现高度一致性。同步期间，Ceph监视器能查询集群运行图的近期版本，他们通过操作键值存储快照和迭代器（用leveldb）来进行存储级同步。



**集群运行图**

集群运行图是多个图的组合，包括监视器图、OSD图、归置组图和元数据服务器图。集群运行图追踪几个重要事件：哪些进程在集群里（in）；哪些进程在集群里（in）是up且在运行、或down；归置组状态是active或inactive、clean或其他状态；和其他反应当前集群状态的信息，像总存储容量、和使用量。

当集群状态有明显变更时，如一OSD挂了、一归置组降级了等等，集群运行图会被更新以反应集群当前状态。另外，监视器也维护者集群的主要状态历史。监视器图、OSD图、归置组图和元数据服务器图各自维护着他们的运行图版本。我们把各图的版本成为一个epoch。

运营集群时，跟踪这些状态是系统管理任务的重要部分。详情见监控集群和监控OSD和归置组。

**监视器法定人数**

本文入门部分提供了一个简陋的Ceph配置文件，他提供了一个监视器用于测试。只用一个监视器集群可以良好的运行，然而单监视器是一个单故障点，生产集群要实现高可用性的话得配置多个监视器，这样单个监视器的失效才不会影响整个集群。

集群用多个监视器实现高可用性时，多个监视器用Paxos算法对主集群运行图达成一致，这里的一致要求大多数监视器都在运行且构成法定人数（如1个，3之2在运行、5之3,、6之4等等）。

**一致性**

你把监视器加进Ceph配置文件时，得注意一些架构问题，Ceph发现集群内的其他监视器对其有着严格的一致性要求。尽管如此，Ceph客户端和其他Ceph守护进程用配置文件发现监视器，监视器却用监视器图（monmap）相互发现而非配置文件。

一个监视器发现集群内的其他监视器时总是参考monmap的本地副本，用monmap而非变更都是通过成为Paxos的分布式一致性算法传递的。监视器们必须就monmap的每次更新达成一致，以确保法定人数里的每个监视器monmap版本相同，如增加、删除一个监视器。Monmap的更新是增量的，所以监视器们都有最新的移植版本，以及一系列之前版本。历史版本的存在允许一个落后的监视器跟上集群当前状态。

如果监视器通过配置文件而非monmap相互发现，这会引进其他风险，因为Ceph配置文件不是自动更新并分发的，监视器有可能不小心用了较老的配置文件，以至于不认识某监视器、放弃法定人数、或者生产一种Paxos不能确定当前系统状态的情形。

**初始化监视器**

在大多数配置和部署案例中，部署Ceph的工具可以帮你生成一个监视器图来初始化监视器（如ceph-deploy等），一个监视器需要4个选项：

**文件系统标识符：**fsid是对象存储的唯一标识符。因为你可以在一套硬件上运行多个集群，所以在初始化监视器时必须指定对象存储的唯一标识符。部署工具通常可替你完成（如ceph-deploy会调用类似uuidgen的程序），但是你也可以手动指定fsid。

监视器标识符：监视器标识符是分配给集群内各监视器的唯一ID，他是一个字母数字组合，为方便起见，标识符通常以字母顺序结尾（如a、b等等），可以设置于Ceph文件（如[mon.a]、[mon.b]等等）、部署工具、或ceph命令行工具。

**秘钥：**监视器必须有秘钥。像ceph-deploy这样的部署工具通常会自动生成，也可以手动完成。见监视器秘钥环。

#### 监视器的配置

要把配置应用到整个集群，把他们放到[global]下；要勇于所有监视器，至于[mon]下；要用于某监视器，指定监视器例程，如[mon.a]。按惯例，监视器例程用字母命名。

**[global]**

**[mon]**

**[mon.a]**

**[mon.b]**

**[mon.c]**

最小配置

Ceph监视器的最简单配置必须包括一主机名及其监视器地址，这些配置可至于[mon]下或某个监视器下。

**[mon]**

mon host = hostname1,hostname2,hostname3

mon addr = 10.0.0.10:6789,10.0.0.11:6789,10.0.0.12:6789

**[mon.a]**

Host = hostname1 mon addr = 10.0.0.10:6789

详见网络配置参考。

Note：这里的监视器最简单配置加入部署工具会自动给你生成fsid和mon.秘钥。

一旦部署了Ceph集群，监视器IP地址不应该更改，然而，如果你决议要修改，必须严格按照更改监视器IP来改。

集群ID

每个Ceph存储集群都有一个唯一标识符（fsid）。如果指定了，他应该出现在配置文件的[global]段下。部署工具通常会产生fsid并存于监视器图，所以不一定会写如配置文件，fsid使得在一套硬件上运行多个集群成为可能。

fsid

**描述：**集群ID，一个集群一个。

**类型：**UUID

**是否必须：**Yes

**默认值：**无。若未指定，部署工具会生成。

Note：如果你用部署工具就不能设置。

初始成员

我们建议在生产环境下最少部署3个监视器，以确保高可用性。运行多个监视器时，你可以指定为形成法定人数成员所需的初始监视器，这能减少集群上线时间。

**[mon]**

mon initial members = a,b,c

mon initial members

描述：集群启动时初始监视器的ID，若指定，Ceph需要奇数个监视器来确定最初法定人数（如3）

类型：String

默认值：None

Note：集群内的大多数监视器必须能互通建立法定人数，你可以用此选项减少初始检视器数量来形成。

数据

Ceph监视器有存储数据的默认路径，生产集群为实现更高性能可把监视器部署到非OSD节点的独立主机上。因为监视器会频繁fsync()，这可能影响OSD。

在Ceph 0.58及更早版本，监视器数据以文件保存，这样人们可以用ls和cat这些普通工具检查监视器数据，然而他不能提供健壮的一致性。

在Ceph 0.59及后续版本中，监视器以键/值对存储数据。监视器需要ACID事务，数据存储的使用防止监视器用损坏的版本进行恢复，除此之外，他允许在一个原子批量操作中进行多个修改操作。

一般来说我们不建议更改默认数据位置，如果要改，我们建议所有监视器统一配置，加到配置文件的[mon]下。

mon data

**描述：**监视器的数据位置。

**类型：**String

**默认值：**/var/lib/ceph/mon/$cluster-$id

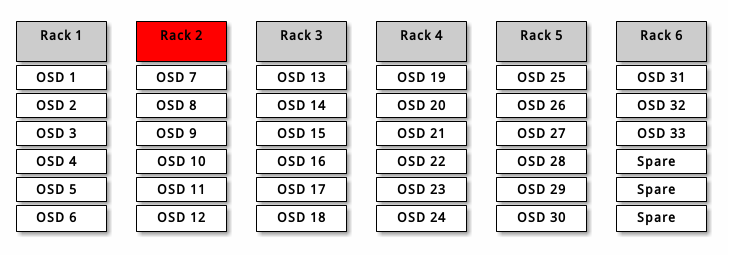
存储容量

Ceph存储集群利用率接近最大容量时（即mon osd full ratio），作为放置数据丢失的安全措施，他会阻止你读写OSD。因此，让生产集群用满可不是好事，因为牺牲了高可用性。full ratio默认值是.95或容量的95%。对小型测试集群来说这是非常激进的设置。

Tip：监控集群时，要警惕和nearfull相关的警告。这意味着一些OSD的失败会导致临时服务中断，应该增加一些OSD来扩展存储容量。

在测试集群时，一个常见场景是：系统管理员从集群删除一个OSD、接着观察均衡；然后继续删除其他OSD，知道集群达到占满率并锁死。我们建议，及时在测试集群李也要规划一点空闲容量用于保证高可用性。理想情况下，要做好这样的预案：一系列OSD失败后，短时间内不更换他们仍能恢复到actie+clean状态。你也可以在active+degraded状态运行集群，但对正常使用来说并不好。

下图描述了一个简化的Ceph集群，他包含33个节点，每个主机一个OSD、每OSD 3TB容量，所以这个小白鼠集群又99TB的实际容量，其mon osd full ratio为.95.如果他只剩余5TB容量，集群就不允许客户端再读写数据，所以他的运行容量是95TB，而非99TB。



在这样的集群里，坏一或两个OSD很平常；一种罕见但可能发生的情形是一个机架的路由器或电源挂了，这会导致多个OSD同时离线（如OSD-7-12），在这种情况下，你仍要力争保持集群可运行并达到avtive+clean状态，及时这意味着你得在短期内额外增加一些OSD及主机。如果集群利用率太高，在解决故障域期间也许不会丢数据，但很可能牺牲数据可用性，因为利用率超过了full ratio。故此，我们建议至少要粗略的规划一下容量。

找出你集群的两个数字：

1. OSD数量
2. 集群总容量

用集群里OSD总数除以集群总容量，就能得到OSD平均容量；如果按预计的OSD数乘以这个值所得的结果计算（偏小），实际应用时将出错；最后再用集群容量乘以占满率能够得到最大运行容量，然后扣除预估的OSD失败率；用较高的失败率（如整机架的OSD）重复前述过程看是否接近满率。

**[global]**

mon osd full ratio = .80

mon osd nearfull ratio = .70

mon osd full ratio

**描述：**OSD硬盘使用率达到多少就认为他full。

**类型：**Float

**默认值：**.95

Mon osd nearfull ratio

**描述：**OSD硬盘使用率达到多少就认为他nearfull

**类型：**Float

**默认值：**.85

**Tip：**如果一些OSD快满了，但其他的仍有足够空间，你可能配置错CRUSH权重。

**心跳**

Ceph监视器要求各OSD向他报告、并接受OSD们的邻居状态报告，以此来掌握集群。Ceph提供了监视器与OSD交互的合理默认值，然而你可以按需修改，详情见监视器与OSD的交互。

监视器存储同步

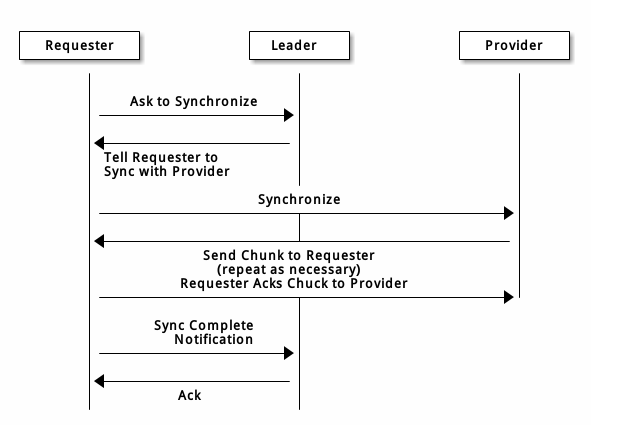
当你用多个监视器支撑一个生产集群时，各监视器都要检查邻居是否有集群运行图的最新版本（如，邻居监视器的图有一或多个epoch版本高于当前监视器的最高版本epoch），过一段时间，集群里的某个监视器可能落后于其他监视器太多而不得不离开法定人数，然后同步到集群当前状态，并重回法定人数。为了同步，监视器可能承担三种中的一种角色：

Leader：Leader是实现最新Paxos版本的第一个监视器

Provider：Provider有最新集群运行图的监视器，但不是第一个实现最新版。

Requester：Requester落后于leader，重回法定人数之前，必须同步以获取关于集群的最新信息。

有了这些角色分区，leader就可以给provider委派同步任务，这会避免同步请求压垮leader、影响性能。在下面的图示中，requester已经知道它落后于其他监视器，然后向leader请求同步，leader让他去和provider同步。



新监视器加入集群时有必要进行同步。在运行中，监视器不定时收到集群运行图的更新，这就意味着leader和prodider角色可能在监视器间变换。如果这事发生在同步期间（如provider落后于leader），provider能中介和requester间的同步。

一旦同步完成，Ceph需要修复整个集群，使归置组回到active+clean状态。

mon sync trim timeout

|  |  |
| --- | --- |
| 描述: |  |
| 类型: | Double |
| 默认值: | 30.0 |

mon sync heartbeat timeout

|  |  |
| --- | --- |
| 描述: |  |
| 类型: | Double |
| 默认值: | 30.0 |

mon sync heartbeat interval

|  |  |
| --- | --- |
| 描述: |  |
| 类型: | Double |
| 默认值: | 5.0 |

mon sync backoff timeout

|  |  |
| --- | --- |
| 描述: |  |
| 类型: | Double |
| 默认值: | 30.0 |

mon sync timeout

|  |  |
| --- | --- |
| 描述: |  |
| 类型: | Double |
| 默认值: | 30.0 |

mon sync max retries

|  |  |
| --- | --- |
| 描述: |  |
| 类型: | Integer |
| 默认值: | 5 |

mon sync max payload size

|  |  |
| --- | --- |
| 描述: | 同步载荷的最大尺寸。 |
| 类型: | 32-bit Integer |
| 默认值: | 1045676 |

mon accept timeout

|  |  |
| --- | --- |
| 描述: | leader 等待 requester(s) 接受 PAXOS 更新的时间，出于同样的目的此值也用于 PAXOS 恢复阶段。 |
| 类型: | Float |
| 默认值: | 10.0 |

paxos propose interval

|  |  |
| --- | --- |
| 描述: | 提议更新之前收集本时间段的更新。 |
| 类型: | Double |
| 默认值: | 1.0 |

paxos min wait

|  |  |
| --- | --- |
| 描述: | 经过一段不活跃时间后，收集更新的最小等待时间。 |
| 类型: | Double |
| 默认值: | 0.05 |

paxos trim tolerance

|  |  |
| --- | --- |
| 描述: | 修复前容忍的其他提议数量。 |
| 类型: | Integer |
| 默认值: | 30 |

paxos trim disabled max versions

|  |  |
| --- | --- |
| 描述: | 允许不修复就通过的最大版本数。 |
| 类型: | Integer |
| 默认值: | 100 |

mon lease

|  |  |
| --- | --- |
| 描述: | 监视器版本租期（秒）。 |
| 类型: | Float |
| 默认值: | 5 |

mon lease renew interval

|  |  |
| --- | --- |
| 描述: | 监视器 leader （头领）刷新其他监视器租期的间隔。 |
| 类型: | Float |
| 默认值: | 3 |

mon lease ack timeout

|  |  |
| --- | --- |
| 描述: | leader 在等到 providers （随从）确认延长租期前等待的时间。 |
| 类型: | Float |
| 默认值: | 10.0 |

mon min osdmap epochs

|  |  |
| --- | --- |
| 描述: | 一直保存的 OSD 图元素最小数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 500 |

mon max pgmap epochs

|  |  |
| --- | --- |
| 描述: | 监视器应该一直保存的 PG 图元素最大数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 500 |

mon max log epochs

|  |  |
| --- | --- |
| 描述: | 监视器应该保留的最大日志数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 500 |

时钟

Ceph的守护进程会互相传递关键消息，这些消息必须在达到超时阈值前处理掉。如果Ceph监视器时钟不同步，就可能出现多种异常情况。例如：

守护进程忽略了收到的消息（如果时间戳过时了）

消息未及时收到时，超时触发的太快或太晚。

Tip：你应该在所有监视器主机上安装NTP以确保监视器集群的时钟同步。

时钟飘逸及时尚未造成损坏也能被NTP感知，Ceph的时钟飘逸或时钟偏差警告及时在NTP同步水平合理时也会被触发。提高时钟漂移值有时候尚可容忍，然而很多因素（像载荷、网络延迟、覆盖默认超时值和监视器存储同步选项）都能在不降低Paxos保证级别的情况下影响可接受的时钟飘逸水平。

Ceph提供了下列这些可调选项，让你自己琢磨可接受的值。

clock offset

|  |  |
| --- | --- |
| 描述: | 时钟可以漂移多少，详情见 Clock.cc 。 |
| 类型: | Double |
| 默认值: | 0 |

Deprecated since version 0.58.

mon tick interval

|  |  |
| --- | --- |
| 描述: | 监视器的心跳间隔，单位为秒。 |
| 类型: | 32-bit Integer |
| 默认值: | 5 |

mon clock drift allowed

|  |  |
| --- | --- |
| 描述: | 监视器间允许的时钟漂移量 |
| 类型: | Float |
| 默认值: | .050 |

mon clock drift warn backoff

|  |  |
| --- | --- |
| 描述: | 时钟偏移警告的退避指数。 |
| 类型: | Float |
| 默认值: | 5 |

mon timecheck interval

|  |  |
| --- | --- |
| 描述: | 和 leader 的时间偏移检查（时钟漂移检查）。单位为秒。 |
| 类型: | Float |
| 默认值: | 300.0 |

客户端

mon client hung interval

|  |  |
| --- | --- |
| 描述: | 客户端每 N 秒尝试一个新监视器，直到它建立连接。 |
| 类型: | Double |
| 默认值: | 3.0 |

mon client ping interval

|  |  |
| --- | --- |
| 描述: | 客户端每 N 秒 ping 一次监视器。 |
| 类型: | Double |
| 默认值: | 10.0 |

mon client max log entries per message

|  |  |
| --- | --- |
| 描述: | 某监视器为每客户端生成的最大日志条数。 |
| 类型: | Integer |
| 默认值: | 1000 |

mon client bytes

|  |  |
| --- | --- |
| 描述: | 内存中允许存留的客户端消息数量（字节数）。 |
| 类型: | 64-bit Integer Unsigned |
| 默认值: | 100ul << 20 |

#### 杂项

mon max osd

|  |  |
| --- | --- |
| 描述: | 集群允许的最大 OSD 数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 10000 |

mon globalid prealloc

|  |  |
| --- | --- |
| 描述: | 为集群预分配的全局 ID 数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 100 |

mon sync fs threshold

|  |  |
| --- | --- |
| 描述: | 数量达到设定值时和文件系统同步， 0 为禁用。 |
| 类型: | 32-bit Integer |
| 默认值: | 5 |

mon subscribe interval

|  |  |
| --- | --- |
| 描述: | 同步的刷新间隔（秒），同步机制允许获取集群运行图和日志信息。 |
| 类型: | Double |
| 默认值: | 300 |

mon stat smooth intervals

|  |  |
| --- | --- |
| 描述: | Ceph 将平滑最后 N 个归置组图的统计信息。 |
| 类型: | Integer |
| 默认值: | 2 |

mon probe timeout

|  |  |
| --- | --- |
| 描述: | 监视器自举无效，搜寻节点前等待的时间。 |
| 类型: | Double |
| 默认值: | 2.0 |

mon daemon bytes

|  |  |
| --- | --- |
| 描述: | 给元数据服务器和 OSD 的消息使用的内存空间（字节）。 |
| 类型: | 64-bit Integer Unsigned |
| 默认值: | 400ul << 20 |

mon max log entries per event

|  |  |
| --- | --- |
| 描述: | 每个事件允许的最大日志条数。 |
| 类型: | Integer |
| 默认值: | 4096 |

### 心跳选项

#### 监视器与OSD交互的配置

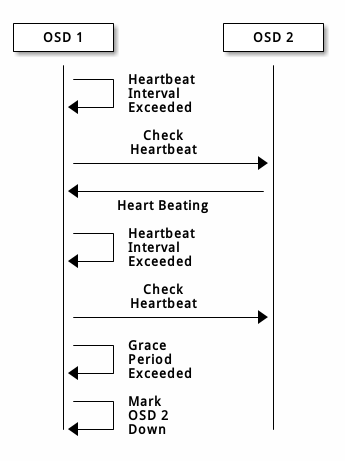
（心跳配置）

完成基本的配置后就可以部署、运行Ceph了。执行ceph health或ceph -s命令时，监视器会报告Ceph存储集群的当前状态。监视器通过让各OSD自己报告、并接收OSD关于邻居状态的报告来掌握集群动态。如果监视器没收到报告，或者他只收到集群的变更报告，那他就要更新集群运行图。

关于监视器与OSD的交互Ceph提供了合理的默认值，然而你可以覆盖他们。下面几段从集群监控角度描述了Ceph监视器与OSD如何交互。

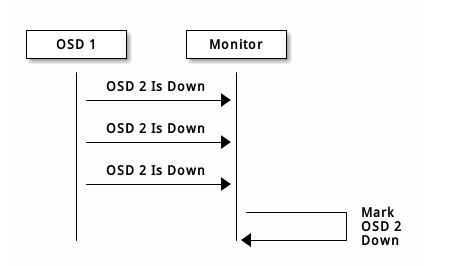
#### 验证心跳

各OSD每6秒会与其他OSD进行心跳检查，用[osd]下的osd heartbeat interval可更改此间隔、或运行时更改。如果一个OSD 20秒都没有心跳，集群就认为down了，用[osd]下的osd heartbeat grace可更改宽限期、或者运行时更改。



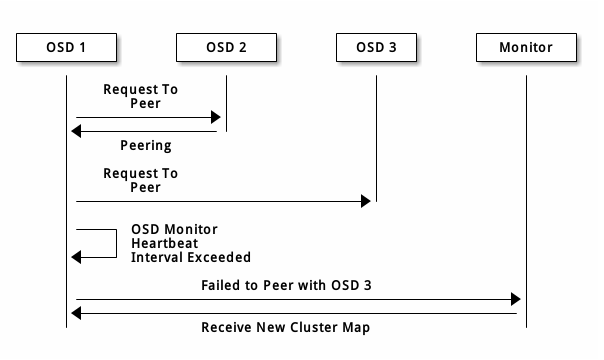
#### OSD报告死亡OSD

默认情况下，一个OSD必须向监视器报告三次另一个OSD down的消息，监视器才会认为哪个被报告的OSD down了；配置文件里[mon]段下的mon osd min down reports选项可更改这个最少osd down消息次数，或者运行时设置。默认情况下，只要有一个OSD报告另一个OSD挂的消息即可，配置文件里[mon]段下的mon osd min down reporters可用来更改必需OSD数，或者运行时更改。



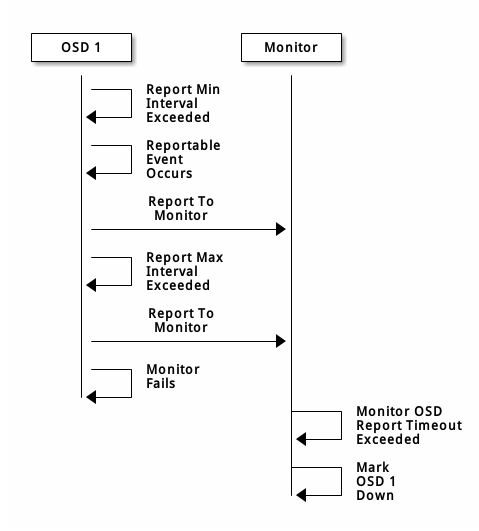
#### OSD报告互联失败

如果一OSD守护进程不能和配置文件中定义的任何OSD建立连接，它会每30向监视器索要一次最新集群运行图，你可以在[osd]下设置osd mon heartbeat interval来更改这个心跳间隔，或者运行时更改。



#### OSD报告自己的状态

如果一OSD在mon osd report timeout时间内没想监视器报告过，监视器就认为他down了。在OSD守护进程会向监视器报告某些事情，如某次操作失败、归置组状态变更、up\_thru变更、或他将在5秒内启动。你可以设置[osd]下的osd mon report interval min来更改最小报告间隔，或在运行时更改。



#### 配置选项

心跳选项应该置于配置文件的[global]段下

**监视器选项**

mon osd min up ratio

|  |  |
| --- | --- |
| 描述: | 在把 OSD 标记为 down 前，保持处于 up 状态的 OSD 最小比例。 |
| 类型: | Double |
| 默认值: | .3 |

mon osd min in ratio

|  |  |
| --- | --- |
| 描述: | 在把 OSD 标记为 out 前，保持处于 in 状态的 OSD 最小比例。 |
| 类型: | Double |
| 默认值: | .3 |

mon osd laggy halflife

|  |  |
| --- | --- |
| 描述: | 滞后量消退时间，秒。 |
| 类型: | Integer |
| 默认值: | 60\*60 |

mon osd laggy weight

|  |  |
| --- | --- |
| 描述: | 滞后量消退时新样本的权重。 |
| 类型: | Double |
| 默认值: | 0.3 |

mon osd adjust heartbeat grace

|  |  |
| --- | --- |
| 描述: | 设置为 true 时， Ceph 将根据滞后量伸缩。 |
| 类型: | Boolean |
| 默认值: | true |

mon osd adjust down out interval

|  |  |
| --- | --- |
| 描述: | 设置为 true 时， Ceph 将根据滞后量伸缩。 |
| 类型: | Boolean |
| 默认值: | true |

mon osd auto mark in

|  |  |
| --- | --- |
| 描述: | Ceph 将把任何启动中的 OSD 标记为在集群中（ in ）。 |
| 类型: | Boolean |
| 默认值: | false |

mon osd auto mark auto out in

|  |  |
| --- | --- |
| 描述: | 把正在启动、且被自动标记为 out 状态的 OSD 标记为 in 。 |
| 类型: | Boolean |
| 默认值: | true |

mon osd auto mark new in

|  |  |
| --- | --- |
| 描述: | 把正在启动的新 OSD 标记为 in 。 |
| 类型: | Boolean |
| 默认值: | true |

mon osd down out interval

|  |  |
| --- | --- |
| 描述: | 在 OSD 停止响应多少秒后把它标记为 down 且 out 。 |
| 类型: | 32-bit Integer |
| 默认值: | 300 |

mon osd downout subtree limit

|  |  |
| --- | --- |
| 描述: | Ceph 可以把最大为多少的 [CRUSH](http://docs.ceph.org.cn/glossary/" \l "term-crush) 单元自动标记为 out 。 |
| 类型: | String |
| 默认值: | rack |

mon osd report timeout

|  |  |
| --- | --- |
| 描述: | 宣布无响应 OSD down 前的宽限期，秒。 |
| 类型: | 32-bit Integer |
| 默认值: | 900 |

mon osd min down reporters

|  |  |
| --- | --- |
| 描述: | 确定一 OSD 状态为 down 的最少报告来源 OSD 数。 |
| 类型: | 32-bit Integer |
| 默认值: | 1 |

mon osd min down reports

|  |  |
| --- | --- |
| 描述: | 一 OSD 必须重复报告另一个 down 的次数。 |
| 类型: | 32-bit Integer |
| 默认值: | 3 |

OSD 选项

osd heartbeat address

|  |  |
| --- | --- |
| 描述: | OSD 用于心跳的网络地址。 |
| 类型: | Address |
| 默认值: | 此主机的地址。 |

osd heartbeat interval

|  |  |
| --- | --- |
| 描述: | 一 OSD 探测邻居的频率，秒。 |
| 类型: | 32-bit Integer |
| 默认值: | 6 |

osd heartbeat grace

|  |  |
| --- | --- |
| 描述: | OSD 多久没心跳就会被集群认为它挂（ down ）了。 |
| 类型: | 32-bit Integer |
| 默认值: | 20 |

osd mon heartbeat interval

|  |  |
| --- | --- |
| 描述: | OSD 没有邻居时多久探测一次监视器。 |
| 类型: | 32-bit Integer |
| 默认值: | 30 |

osd mon report interval max

|  |  |
| --- | --- |
| 描述: | 监视器允许 OSD 报告的最大间隔，超时将认为 OSD 挂了（ down ）。 |
| 类型: | 32-bit Integer |
| 默认值: | 120 |

osd mon report interval min

|  |  |
| --- | --- |
| 描述: | 从一 OSD 启动或其它可报告事件发生以来，多长时间内必须向监视器报告一次。 |
| 类型: | 32-bit Integer |
| 默认值: | 5 |
| 有效范围: | 要小于 osd mon report interval max 。 |

osd mon ack timeout

|  |  |
| --- | --- |
| 描述: | OSD 等待监视器提供统计信息的时间，秒。 |
| 类型: | 32-bit Integer |
| 默认值: | 30 |

### OSD选项

**OSD配置参考**

你可以通过配置文件调整OSD，但靠默认值和极少的配置OSD守护进程就能运行。最简OSD配置需设置osd journal size和host，其他几乎都能用默认值。

Ceph的OSD守护进程用递增的数字作标识，按惯例以0开始，如下：

osd.0

osd.1

osd.2

在配置文件里，[osd]段下的配置适用于所有OSD；要添加针对特定OSD的选项（如host），把他放到那个OSD段下即可，如：

**[osd]**

osd journal size = 1024

**[osd.0]**

host = osd-host-a

**[osd.1]**

host = osd-host-b

#### 常规配置

下列选项可配置一OSD的唯一标识符、以及数据和日志的路径。Ceph部署脚本通常会自动生成UUID。我们不建议更改数据和日志的默认路径，因为这样会增加后续的排障难度。

日志尺寸应该大于期望的驱动器速度和filestore max sync interval之积的二倍；最常见的方法是为日志驱动器（通常是SSD）分区并挂载好，这样Ceph就可以用整个分区做日志。

osd uuid

描述：OSD的全局唯一标识符（UUID）

类型：UUID

默认值：The UUID

备注：osd uuid 适用于单个OSD，fsid适用于整个集群。

osd data

描述：OSD数据存储位置，你得创建并把数据盘挂载到其下。我们不推荐更改默认值

类型：String

默认值：/var/lib/ceph/osd/$cluster-$id

osd max write size

描述：一次写入的最大尺寸，MB

类型：32-bit Integer

默认值：90

osd client message size cap

描述：内存里允许的最大客户端数据消息。

类型：63-bit Integer Unsigned

默认值：默认为500MB。500\*1024L\*1024L

osd class dir

描述：RADOS类插件的路径

类型：String

默认值：$libdir/rados-classes

#### 文件系统选项

Ceph可自动创建并挂载所需的文件系统

osd mkfs options {fs-type}

描述：为OSD新建{fs-type}类型的文件系统时使用的选项

类型：String

xfs：默认值

其余文件系统默认值：{empty string}

实例：osd mkfs option xfs = -f -d agcount=24

osd mount options {fs-type}

描述：挂载{fs-type}类型的文件系统作为OSD数据目录时所有的选项

类型：String

xfs默认值：rw,noatime,inode64

其余文件系统默认值：rw，noatime

实例：osd mount option xfs = rw，noatime，inode64，logbufs=8

#### 日志选项

默认情况下，Ceph觉得你会把OSD日志存储于下列路径：

/var/lib/ceph/osd/$cluster-$id/journal

未做性能优化时，Ceph会把日志存储在与OSD数据相同的硬盘上。追求高性能的OSD可用单独的硬盘存储日志数据，如固态硬盘能提供高性能日志。

osd journal size默认值是0，所以你得在ceph.conf里设置。日志尺寸应该是filestore max sync interval与期望吞吐量的乘积再乘以2。

osd journal size = {2 \* (expected throughput \* filestore max sync interval)}

期望吞吐量应该考虑期望的硬盘的吞吐量（即持续数据传输速率）、网络吞吐量，例如一个7200转硬盘的速度大致是100MB/S。硬盘和网络吞吐量中较小的（min（））一个是相对合理的吞吐量，有的用户则以10GB日志尺寸起步，例如：

osd journal size = 10000

osd journal

描述：OSD日志路径，可以是一个文件或块设备（SSD的一个分区）的路径。如果是文件，要先创建相应目录。我们建议用osd data以外的独立驱动器。

类型：String

默认值：/var/lib/ceph/osd/$cluster-$id/journal

osd journal size

描述：日志尺寸（MB）。如果是0且日志文件是块设备，他会使用整个块设备。从v0.54起，如果日志文件是块设备，这个选项会被忽略，且使用整个块设备。

类型：32-bit Integer

默认值：5120

推荐值:最少1G，应该是期望的驱动器速度和filestore max sync interval的乘积

#### 监视器和OSD的交互

OSD周期性的相互检查心跳并报告给监视器。Ceph默认配置可满足多数情况，但是如果你的网络延迟时大，就得用较长间隔。关于心跳的讨论参见监视器与OSD交互的配置。

#### 数据归置

详情见存储池和归置组配置参考

#### 洗刷

除了对象复制多个副本外，Ceph还要洗刷归置组以确保数据完整性。这类洗刷类似对象存储层的fsck，对每个归置组，Ceph生成一个所有对象的目录，并比对每个对象及其副本一确保没有对象丢失或搭配错。轻微洗刷（每天）检查对象尺寸和属性，深层洗刷（每周）会读出数据并用校验和方法确认数据完整性。

洗刷对维护数据完整性很重要，但会影响性能；你可以用下列选项来增加或减少洗刷操作。

osd max scrubs

|  |  |
| --- | --- |
| 描述: | 一 OSD 的最大并发洗刷操作数。 |
| 类型: | 32-bit Int |
| 默认值: | 1 |

osd scrub begin hour

|  |  |
| --- | --- |
| 描述: | 被调度的洗刷操作在一天中可以运行的时间下限。 |
| 类型: | 0 到 24 之间的整数 |
| 默认值: | 0 |

osd scrub end hour

|  |  |
| --- | --- |
| 描述: | 被调度的洗刷操作在一天中可以运行的时间上限。本选项与 osd scrub begin hour 一起定义了一个时间窗口，在此期间可以进行洗刷操作。但是，在某个归置组的洗刷间隔超过 osd scrub max interval 时，不管这个时间窗口是否合适都会执行。 |
| 类型: | 0 到 24 之间的整数 |
| 默认值: | 24 |

osd scrub thread timeout

|  |  |
| --- | --- |
| 描述: | 洗刷线程最大死亡时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 60 |

osd scrub finalize thread timeout

|  |  |
| --- | --- |
| 描述: | 洗刷终结线程最大超时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 60\*10 |

osd scrub load threshold

|  |  |
| --- | --- |
| 描述: | 最大负载，当前系统负载（ getloadavg() 所定义的）高于此值时 Ceph 不会洗刷。默认 0.5 。 |
| 类型: | Float |
| 默认值: | 0.5 |

osd scrub min interval

|  |  |
| --- | --- |
| 描述: | 集群负载低的时候，洗刷的最大间隔时间，秒。 |
| 类型: | Float |
| 默认值: | 每天一次。 60\*60\*24 |

osd scrub max interval

|  |  |
| --- | --- |
| 描述: | 不论集群负载如何，都要进行洗刷的时间间隔。 |
| 类型: | Float |
| 默认值: | 每周一次。 7\*60\*60\*24 |

osd deep scrub interval

|  |  |
| --- | --- |
| 描述: | 深层洗刷的间隔（完整地读所有数据）。 osd scrub load threshold 不会影响此选项。 |
| 类型: | Float |
| 默认值: | 每周一次。 60\*60\*24\*7 |

osd scrub interval randomize ratio

|  |  |
| --- | --- |
| 描述: | 在给某一归置组调度下一个洗刷作业时，给 osd scrub min interval 增加个随机延时，这个延时是个小于 osd scrub min interval \* osd scrub interval randomized ratio 的随机值。所以在实践中，这个默认设置会把洗刷操作随机地散布到允许的时间窗口内，即 [1, 1.5] \* osd scrub min interval 。 |
| 类型: | Float |
| 默认值: | 0.5 |

osd deep scrub stride

|  |  |
| --- | --- |
| 描述: | 深层洗刷时的读取尺寸。 |
| 类型: | 32-bit Integer |
| 默认值: | 512 KB. 524288 |

#### 操作数

操作数选项允许你设置用于服务的线程数，如果把osd op threads设置为0就禁用了多线程。默认情况下，ceph用30秒超时和30秒抗议时间来把握2个线程的运行情况。你可以调整客户端操作和恢复操作的优先程度来优化恢复期间的性能。

osd op threads

|  |  |
| --- | --- |
| 描述: | OSD 操作线程数， 0 为禁用。增大数量可以增加请求处理速度。 |
| 类型: | 32-bit Integer |
| 默认值: | 2 |

osd client op priority

|  |  |
| --- | --- |
| 描述: | 设置客户端操作优先级，它相对于 osd recovery op priority 。 |
| 类型: | 32-bit Integer |
| 默认值: | 63 |
| 有效范围: | 1-63 |

osd recovery op priority

|  |  |
| --- | --- |
| 描述: | 设置恢复优先级，其值相对于 osd client op priority 。 |
| 类型: | 32-bit Integer |
| 默认值: | 10 |
| 有效范围: | 1-63 |

osd op thread timeout

|  |  |
| --- | --- |
| 描述: | OSD 线程超时秒数。 |
| 类型: | 32-bit Integer |
| 默认值: | 30 |

osd op complaint time

|  |  |
| --- | --- |
| 描述: | 一个操作进行多久后开始抱怨。 |
| 类型: | Float |
| 默认值: | 30 |

osd disk threads

|  |  |
| --- | --- |
| 描述: | 硬盘线程数，用于在后台执行磁盘密集型操作，像数据洗刷和快照修复。 |
| 类型: | 32-bit Integer |
| 默认值: | 1 |

osd disk thread ioprio class

|  |  |
| --- | --- |
| 描述: | 警告：只有 osd disk thread ioprio class 和 osd disk thread ioprio priority 同时改为非默认值时此配置才生效。 OSD 用 ioprio\_set(2) 为磁盘线程设置 I/O 调度分类（ class ），当前支持 idle 、 be 或 rt 。 idle 类意味着磁盘线程的优先级在 OSD 中是最低的，适合需延缓洗刷操作的情形，如 OSD 正忙于处理客户端操作。 be 是默认值，将设置与其它 OSD 线程相同的优先级。 rt 意为磁盘线程的优先级将高于其它任何 OSD 线程；适用于急需洗刷、并且即使牺牲客户端操作也要进行时。注：只能与 Linux 内核的 CFQ 调度器配合使用。 |
| 类型: | String |
| 默认值: | 空字符串 |

osd disk thread ioprio priority

|  |  |
| --- | --- |
| 描述: | 警告：只有 osd disk thread ioprio class 和 osd disk thread ioprio priority 同时改为非默认值时此配置才生效。它通过 ioprio\_set(2) 设置磁盘线程的 I/O 调度优先级（ priority ），优先级从最高的 0 到最低的 7 。如果某主机上的所有 OSD 都在 idle 类中竞争 I/O 资源（即控制器拥塞了），那么你就可以用此选项把某 OSD 的磁盘线程优先级调低为 7 ，其它优先级为 0 的 OSD 就有可能洗刷得快一点。注：只能与 Linux 内核的 CFQ 调度器配合使用。 |
| 类型: | 0 到 7 间的整数， -1 禁用此功能。 |
| 默认值: | -1 |

osd op history size

|  |  |
| --- | --- |
| 描述: | 要跟踪的最大已完成操作数量。 |
| 类型: | 32-bit Unsigned Integer |
| 默认值: | 20 |

osd op history duration

|  |  |
| --- | --- |
| 描述: | 要跟踪的最老已完成操作。 |
| 类型: | 32-bit Unsigned Integer |
| 默认值: | 600 |

osd op log threshold

|  |  |
| --- | --- |
| 描述: | 一次显示多少操作日志。 |
| 类型: | 32-bit Integer |
| 默认值: | 5 |

#### 回填

当集群新增或移除OSD时，按照CRUSH算法应该重新均衡集群，他会把一些归置组移除或移入多个OSD以回到均衡状态。归置组和对象的迁移会导致集群运营性能显著降低，Ceph用backfilling来执行此迁移，他可以使得Ceph的回填操作优先级低于用户读写请求。

osd max backfills

|  |  |
| --- | --- |
| 描述: | 单个 OSD 允许的最大回填操作数。 |
| 类型: | 64-bit Unsigned Integer |
| 默认值: | 10 |

osd backfill scan min

|  |  |
| --- | --- |
| 描述: | 集群负载低时，回填操作时扫描间隔。 |
| 类型: | 32-bit Integer |
| 默认值: | 64 |

osd backfill scan max

|  |  |
| --- | --- |
| 描述: | 回填操作时最大扫描间隔。 |
| 类型: | 32-bit Integer |
| 默认值: | 512 |

osd backfill full ratio

|  |  |
| --- | --- |
| 描述: | OSD 的占满率达到多少时拒绝接受回填请求。 |
| 类型: | Float |
| 默认值: | 0.85 |

osd backfill retry interval

|  |  |
| --- | --- |
| 描述: | 重试回填请求前等待秒数。 |
| 类型: | Double |
| 默认值: | 10.0 |

#### OSD运行图

OSD运行图反映集群中运行的OSD守护进程，斗转星移，图元增加。Ceph用一些选项来确保OSD运行图增大时仍运行良好。

osd map dedup

|  |  |
| --- | --- |
| 描述: | 允许删除 OSD 图里的重复项。 |
| 类型: | Boolean |
| 默认值: | true |

osd map cache size

|  |  |
| --- | --- |
| 描述: | 缓存的 OSD 图个数。 |
| 类型: | 32-bit Integer |
| 默认值: | 500 |

osd map cache bl size

|  |  |
| --- | --- |
| 描述: | OSD 进程中，驻留内存的 OSD 图缓存尺寸。 |
| 类型: | 32-bit Integer |
| 默认值: | 50 |

osd map cache bl inc size

|  |  |
| --- | --- |
| 描述: | OSD 进程中，驻留内存的 OSD 图缓存增量尺寸。 |
| 类型: | 32-bit Integer |
| 默认值: | 100 |

osd map message max

|  |  |
| --- | --- |
| 描述: | 每个 MOSDMap 图消息允许的最大条目数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 100 |

#### 恢复

当集群启动、或某OSD守护进程崩溃后重启时，此OSD开始与其他OSD们建立连接，这样才能正常工作。详情见监控OSD和归置组。

如果某OSD崩溃并重生，同行会落后于其他OSD，也就是没有同归置组内最新版本的对象。这时，OSD守护进程进入恢复模式并检索最新数据副本，并更新运行图。根据OSD挂的时间长短，OSD的对象和归置组可能落后的很厉害，另外，如果挂的是一个实效域（如一个机柜），多个OSD会同时重生，这样回复时间更长、更耗资源。

为保持运营性能，Ceph进行恢复时会限制恢复请求数、线程数、对象块尺寸，这样在降级状态下也能保持良好的性能。

osd recovery delay start

|  |  |
| --- | --- |
| 描述: | 对等关系建立完毕后， Ceph 开始对象恢复前等待的时间（秒）。 |
| 类型: | Float |
| 默认值: | 0 |

osd recovery max active

|  |  |
| --- | --- |
| 描述: | 每个 OSD 一次处理的活跃恢复请求数量，增大此值能加速恢复，但它们会增加集群负载。 |
| 类型: | 32-bit Integer |
| 默认值: | 15 |

osd recovery max chunk

|  |  |
| --- | --- |
| 描述: | 一次推送的数据块的最大尺寸。 |
| 类型: | 64-bit Integer Unsigned |
| 默认值: | 8 << 20 |

osd recovery threads

|  |  |
| --- | --- |
| 描述: | 数据恢复时的线程数。 |
| 类型: | 32-bit Integer |
| 默认值: | 1 |

osd recovery thread timeout

|  |  |
| --- | --- |
| 描述: | 恢复线程最大死亡时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 30 |

osd recover clone overlap

|  |  |
| --- | --- |
| 描述: | 在数据恢复期间保留重叠副本。应该总是 true 。 |
| 类型: | Boolean |
| 默认值: | true |

#### 杂项

osd snap trim thread timeout

|  |  |
| --- | --- |
| 描述: | 快照修复线程最大死亡时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 60\*60\*1 |

osd backlog thread timeout

|  |  |
| --- | --- |
| 描述: | 积压线程最大死亡时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 60\*60\*1 |

osd default notify timeout

|  |  |
| --- | --- |
| 描述: | OSD 默认通告超时，秒。 |
| 类型: | 32-bit Integer Unsigned |
| 默认值: | 30 |

osd check for log corruption

|  |  |
| --- | --- |
| 描述: | 根据日志文件查找数据损坏，会耗费大量计算时间。 |
| 类型: | Boolean |
| 默认值: | false |

osd remove thread timeout

|  |  |
| --- | --- |
| 描述: | OSD 删除线程的最大死亡时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 60\*60 |

osd command thread timeout

|  |  |
| --- | --- |
| 描述: | 命令线程最大超时值。 |
| 类型: | 32-bit Integer |
| 默认值: | 10\*60 |

osd command max records

|  |  |
| --- | --- |
| 描述: | 限制返回的丢失对象数量。 |
| 类型: | 32-bit Integer |
| 默认值: | 256 |

osd auto upgrade tmap

|  |  |
| --- | --- |
| 描述: | 在旧对象上给 omap 使用 tmap 。 |
| 类型: | Boolean |
| 默认值: | true |

osd tmapput sets users tmap

|  |  |
| --- | --- |
| 描述: | 只在调试时使用 tmap 。 |
| 类型: | Boolean |
| 默认值: | false |

osd preserve trimmed log

|  |  |
| --- | --- |
| 描述: | 保留本该修剪掉的日志文件，但是会占用更多磁盘空间。 |
| 类型: | Boolean |
| 默认值: | false |

### FiLESTORE配置参考

filestore debug omap check

描述：打开对同步检查过程的调试。代价很高，仅用于调试。

类型：Boolean

是否必须：No

默认值：0

#### 扩展属性

扩展属性（XATTR）是配置里的重要一部分。一些文件系统对XATTR字节数有限制，另外在某些情况下文件系统存储XATTR的速度不如其他方法。下面的选项让你用独立文件系统的存储方法，或许能提升性能。

Ceph扩展属性用底层文件系统的XATTR（如果没有尺寸限制）存储为inline xattr。如果有限制，如ext4限制为4KB，达到filestore max inline xattr size或filestore max inline xattrs阈值时一些XATTR将存储为键/值数据库（也叫omap）

filestore xattr use omap

描述：用XATTR存储对象图，采用ext4文件系统时要设置为true

类型：Boolean

是否必须：No

默认值：false

filestore max inline xattr size

描述：每个对象在文件系统（如XFS， btrfs，ext4等）里存储的XATTR最大尺寸，应该小于文件系统支持的尺寸。

类型：Unsigned 32-bit Integer

是否必须：No

默认值：512

filestore max inline xattrs

描述：每个对象存储在文件系统里的XATTR数量。

类型：32-bit Integer

是否必须：No

默认值：2

#### 同步间隔

filestore需要周期性的静默写入、同步文件系统，这创建了一个提交点，然后就能释放相应的日志条目了。较大的同步频率可减少执行同步的时间及保存在日志里的数据量；较小的频率使得后端的文件系统能优化归并较小的数据和元数据写入，因此可能使同步更有效。

filestore max sync interval

|  |  |
| --- | --- |
| 描述: | 同步 filestore 的最大间隔秒数。 |
| 类型: | Double |
| 是否必需: | No |
| 默认值: | 5 |

filestore min sync interval

|  |  |
| --- | --- |
| 描述: | 同步 filestore 的最小间隔秒数。 |
| 类型: | Double |
| 是否必需: | No |
| 默认值: | .01 |

#### 回写器

filestore 回写器强制使用sync file range来写出大块数据，这样处理有望减少最终同步的代价。实践中，禁用“filestore”回写器有时候能提升性能。

filestore flusher

|  |  |
| --- | --- |
| 描述: | 启用 filestore 回写器。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

Deprecated since version v.65.

filestore flusher max fds

|  |  |
| --- | --- |
| 描述: | 设置回写器的最大文件描述符数量。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 512 |

Deprecated since version v.65.

filestore sync flush

|  |  |
| --- | --- |
| 描述: | 启用同步回写器。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

Deprecated since version v.65.

filestore fsync flushes journal data

|  |  |
| --- | --- |
| 描述: | 文件系统同步时也回写日志数据。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

#### 队列

下面的选项能限制filestore队列的尺寸。

filestore queue max ops

|  |  |
| --- | --- |
| 描述: | 文件存储操作接受的最大并发数，超过此设置的请求会被阻塞。 |
| 类型: | Integer |
| 是否必需: | 无。对性能影响最小。 |
| 默认值: | 500 |

filestore queue max bytes

|  |  |
| --- | --- |
| 描述: | 一个操作的最大字节数。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 100 << 20 |

filestore queue committing max ops

|  |  |
| --- | --- |
| 描述: | filestore 能提交的最大操作数。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 500 |

filestore queue committing max bytes

|  |  |
| --- | --- |
| 描述: | filestore 器能提交的最大字节数。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 100 << 20 |

#### 超时选项

filestore op threads

|  |  |
| --- | --- |
| 描述: | 允许并行操作文件系统的最大线程数。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 2 |

filestore op thread timeout

|  |  |
| --- | --- |
| 描述: | 文件系统操作线程超时值，单位为秒。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 60 |

filestore op thread suicide timeout

|  |  |
| --- | --- |
| 描述: | 提交操作超时值（秒），超时后会取消。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 180 |

#### TREE文件系统

filestore btrfs snap

|  |  |
| --- | --- |
| 描述: | 对 btrfs filestore 启用快照功能。 |
| 类型: | Boolean |
| 是否必需: | 不。仅适用于 btrfs 。 |
| 默认值: | true |

filestore btrfs clone range

|  |  |
| --- | --- |
| 描述: | 允许 btrfs filestore 克隆操作排队。 |
| 类型: | Boolean |
| 是否必需: | 不。仅适用于 btrfs 。 |
| 默认值: | true |

#### 日志

filestore journal parallel

|  |  |
| --- | --- |
| 描述: | 允许并行记日志，对 btrfs 默认开。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

filestore journal writeahead

|  |  |
| --- | --- |
| 描述: | 允许预写日志，对 xfs 默认开。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

filestore journal trailing

|  |  |
| --- | --- |
| 描述: | 过时了，从没用过。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

#### 杂项

filestore merge threshold

|  |  |
| --- | --- |
| 描述: | 并入父目录前，子目录内的最小文件数。注：负值表示禁用子目录合并功能。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 10 |

filestore split multiple

|  |  |
| --- | --- |
| 描述: | filestore\_split\_multiple \* abs(filestore\_merge\_threshold) \* 16 是分割为子目录前某目录内的最大文件数。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 2 |

filestore update to

|  |  |
| --- | --- |
| 描述: | 限制 filestore 自动更新到某个指定版本。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 1000 |

filestore blackhole

|  |  |
| --- | --- |
| 描述: | 丢弃任何讨论中的事务。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

filestore dump file

|  |  |
| --- | --- |
| 描述: | 存储事务转储目的文件。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

filestore kill at

|  |  |
| --- | --- |
| 描述: | 在第 N 次机会后注入一个失效。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | false |

filestore fail eio

|  |  |
| --- | --- |
| 描述: | 在 IO 错误的时候失败或崩溃。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | true |

### 日志配置参考

Ceph的OSD使用日志的原因有二：速度和一致性。

速度：日志使得OSD可以快速的提交小块数据的写入，Ceph把小片，随机IO一次写入日志，这样，后端文件系统就有可能归并写入动作，并最终提升并发承载力。因此，使用OSD日志能展现出优秀的突发写性能，实际上数据还没有写入OSD，因为文件系统把她们捕捉到了日志。

一致性：Ceph的OSD守护进程需要一个能保证原子操作的文件系统接口。OSD把一个操作的描述写入日志，然后把操作应用到文件系统，这需要原子更新一个对象（例如归置组元数据）。每隔一段filestore max sync iinterval和filestore min sync interval之间的时间，OSD停止写入、把日志同步到文件系统，这样允许OSD修整日志里的操作并重用空间。若失败，OSD从上个同步点开始重放日志。

OSD守护进程支持下面的日志选项：

journal dio

|  |  |
| --- | --- |
| 描述: | 对日志启用径直 IO ，需要 journal block align 设置为 true 。 |
| 类型: | Boolean |
| 是否必需: | 用 aio 时自动启用。 |
| 默认值: | true |

journal aio

Changed in version 0.61: Cuttlefish

|  |  |
| --- | --- |
| 描述: | 异步写入日志时用 libaio 库，需要 journal dio 设为 true 。 |
| 类型: | Boolean |
| 是否必需: | No. |
| 默认值: | 0.61 版之后为 true ， 0.60 及之前为 false 。 |

journal block align

|  |  |
| --- | --- |
| 描述: | 块对齐写， dio 和 aio 需要。 |
| 类型: | Boolean |
| 是否必需: | 用 dio 和 aio 时自动启用。 |
| 默认值: | true |

journal max write bytes

|  |  |
| --- | --- |
| 描述: | 一次写入日志的最大尺寸。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 10 << 20 |

journal max write entries

|  |  |
| --- | --- |
| 描述: | 一次写入日志的最大数量。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 100 |

journal queue max ops

|  |  |
| --- | --- |
| 描述: | 队列里一次允许的最大操作数量。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 500 |

journal queue max bytes

|  |  |
| --- | --- |
| 描述: | 队列里一次允许的最大字节数。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 10 << 20 |

journal align min size

|  |  |
| --- | --- |
| 描述: | 对齐大于指定最小值的数据有效载荷。 |
| 类型: | Integer |
| 是否必需: | No |
| 默认值: | 64 << 10 |

journal zero on create

|  |  |
| --- | --- |
| 描述: | 在创建文件系统（ mkfs ）期间用 0 填充整个日志。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

### 日志存储池、归置组和CRUSH配置参考

当你创建存储池并给他设置归置组数量时，如果你没指定Ceph就用默认值。我们建议更改某些默认值，特别是存储池的副本数和默认归置组数量，可以在运行pool命令的时候设置这些值。你也可以把配置写入Ceph配置文件的[global段来覆盖默认值。

[global]

# By default, Ceph makes 3 replicas of objects. If you want to make four

# copies of an object the default value--a primary copy and three replica

# copies--reset the default values as shown in 'osd pool default size'.

# If you want to allow Ceph to write a lesser number of copies in a degraded

# state, set 'osd pool default min size' to a number less than the

# 'osd pool default size' value.

osd pool default size = 4 # Write an object 4 times.

osd pool default min size = 1 # Allow writing one copy in a degraded state.

# Ensure you have a realistic number of placement groups. We recommend

# approximately 100 per OSD. E.g., total number of OSDs multiplied by 100

# divided by the number of replicas (i.e., osd pool default size). So for

# 10 OSDs and osd pool default size = 4, we'd recommend approximately

# (100 \* 10) / 4 = 250.

osd pool default pg num = 250

osd pool default pgp num = 250

mon max pool pg num

|  |  |
| --- | --- |
| 描述: | 每个存储的最大归置组数量。 |
| 类型: | Integer |
| 默认值: | 65536 |

mon pg create interval

|  |  |
| --- | --- |
| 描述: | 在同一个 OSD 里创建 PG 的间隔秒数。 |
| 类型: | Float |
| 默认值: | 30.0 |

mon pg stuck threshold

|  |  |
| --- | --- |
| 描述: | 多长时间无响应的 PG 才认为它卡住了。 |
| 类型: | 32-bit Integer |
| 默认值: | 300 |

osd pg bits

|  |  |
| --- | --- |
| 描述: | 每个 OSD 的归置组位数。 |
| 类型: | 32-bit Integer |
| 默认值: | 6 |

osd pgp bits

|  |  |
| --- | --- |
| 描述: | 每个 OSD 为 PGP 留的位数。 |
| 类型: | 32-bit Integer |
| 默认值: | 6 |

osd crush chooseleaf type

|  |  |
| --- | --- |
| 描述: | 在一个 CRUSH 规则内用于 chooseleaf 的桶类型。用序列号而不是名字。 |
| 类型: | 32-bit Integer |
| 默认值: | 1 ，通常一台主机包含一或多个 OSD 。 |

osd pool default crush replicated ruleset

|  |  |
| --- | --- |
| 描述: | 创建多副本存储池时用哪个默认 CRUSH 规则集。 |
| 类型: | 8-bit Integer |
| 默认值: | CEPH\_DEFAULT\_CRUSH\_REPLICATED\_RULESET ，也就是说，“挑选数字 ID 最小的规则集”。这样，没有规则集 0 时也能成功创建存储池。 |

osd pool erasure code stripe width

|  |  |
| --- | --- |
| 描述: | 设置每个已编码池内的对象条带尺寸（单位为字节）。尺寸为 S 的各对象将存储为 N 个条带，且各条带将分别编码/解码。 |
| 类型: | Unsigned 32-bit Integer |
| 默认值: | 4096 |

osd pool default size

|  |  |
| --- | --- |
| 描述: | 设置一存储池的对象副本数，默认值等同于 ceph osd pool set {pool-name} size {size} 。 |
| 类型: | 32-bit Integer |
| 默认值: | 3 |

osd pool default min size

|  |  |
| --- | --- |
| 描述: | 设置存储池中已写副本的最小数量，以向客户端确认写操作。如果未达到最小值， Ceph 就不会向客户端回复已写确认。此选项可确保降级（ degraded ）模式下的最小副本数。 |
| 类型: | 32-bit Integer |
| 默认值: | 0 ，意思是没有最小值。如果为 0 ，最小值是 size - (size / 2) 。 |

osd pool default pg num

|  |  |
| --- | --- |
| 描述: | 一个存储池的默认归置组数量，默认值即是 mkpool 的 pg\_num 参数。 |
| 类型: | 32-bit Integer |
| 默认值: | 8 |

osd pool default pgp num

|  |  |
| --- | --- |
| 描述: | 一个存储池里，为归置使用的归置组数量，默认值等同于 mkpool 的 pgp\_num 参数。当前 PG 和 PGP 应该相同。 |
| 类型: | 32-bit Integer |
| 默认值: | 8 |

osd pool default flags

|  |  |
| --- | --- |
| 描述: | 新存储池的默认标志。 |
| 类型: | 32-bit Integer |
| 默认值: | 0 |

osd max pgls

|  |  |
| --- | --- |
| 描述: | 将列出的最大归置组数量，一客户端请求量大时会影响 OSD 。 |
| 类型: | Unsigned 64-bit Integer |
| 默认值: | 1024 |
| Note: | 默认值应该没问题。 |

osd min pg log entries

|  |  |
| --- | --- |
| 描述: | 清理日志文件的时候保留的归置组日志量。 |
| 类型: | 32-bit Int Unsigned |
| 默认值: | 1000 |

osd default data pool replay window

|  |  |
| --- | --- |
| 描述: | 一 OSD 等待客户端重播请求的时间，秒。 |
| 类型: | 32-bit Integer |
| 默认值: | 45 |

### 消息传递选项

ms tcp nodelay

|  |  |
| --- | --- |
| 描述: | 在信差的 TCP 会话上禁用 nagle 算法。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | true |

ms initial backoff

|  |  |
| --- | --- |
| 描述: | 出错时重连的初始等待时间。 |
| 类型: | Double |
| 是否必需: | No |
| 默认值: | .2 |

ms max backoff

|  |  |
| --- | --- |
| 描述: | 出错重连时等待的最大时间。 |
| 类型: | Double |
| 是否必需: | No |
| 默认值: | 15.0 |

ms nocrc

|  |  |
| --- | --- |
| 描述: | 禁用网络消息的 crc 校验， CPU 不足时可提升性能。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

ms die on bad msg

|  |  |
| --- | --- |
| 描述: | 调试选项，不要配置。 |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

ms dispatch throttle bytes

|  |  |
| --- | --- |
| 描述: | 等着传送的消息尺寸阀值。 |
| 类型: | 64-bit Unsigned Integer |
| 是否必需: | No |
| 默认值: | 100 << 20 |

ms bind ipv6

|  |  |
| --- | --- |
| 描述: | 如果想让守护进程绑定到 IPv6 地址而非 IPv4 就得启用（如果你指定了守护进程或集群 IP 就不必要了） |
| 类型: | Boolean |
| 是否必需: | No |
| 默认值: | false |

ms rwthread stack bytes

|  |  |
| --- | --- |
| 描述: | 堆栈尺寸调试选项，不要配置。 |
| 类型: | 64-bit Unsigned Integer |
| 是否必需: | No |
| 默认值: | 1024 << 10 |

ms tcp read timeout

|  |  |
| --- | --- |
| 描述: | 控制信差关闭空闲连接前的等待秒数。 |
| 类型: | 64-bit Unsigned Integer |
| 是否必需: | No |
| 默认值: | 900 |

ms inject socket failures

|  |  |
| --- | --- |
| 描述: | 调试选项，别配置。 |
| 类型: | 64-bit Unsigned Integer |
| 是否必需: | No |
| 默认值: | 0 |

### 常规选项

fsid

|  |  |
| --- | --- |
| 描述: | 文件系统 ID ，每集群一个。 |
| 类型: | UUID |
| 是否必需: | No. |
| 默认值: | 无。通常由部署工具生成。 |

admin socket

|  |  |
| --- | --- |
| 描述: | 在某个守护进程上执行管理命令的套接字，不管 Ceph 监视器团体是否已建立。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | /var/run/ceph/$cluster-$name.asok |

pid file

|  |  |
| --- | --- |
| 描述: | mon 、 osd 、 mds 将把它们的 PID 写入此文件，其名为 /var/run/$cluster/$type.$id.pid 。如名为 Ceph 的集群，其 ID 为 a 的 mon 进程将创建 /var/run/ceph/mon.a.pid 。如果是正常停止的， pid file 就会被守护进程删除；如果进程未进入后台运行（即启动时加了 -f 或 -d 选项），它就不会创建 pid file 。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | No |

chdir

|  |  |
| --- | --- |
| 描述: | Ceph 进程一旦启动、运行就进入这个目录，默认推荐 / 。 |
| 类型: | String |
| 是否必需: | No |
| 默认值: | / |

max open files

|  |  |
| --- | --- |
| 描述: | 如果设置了， [Ceph 存储集群](http://docs.ceph.org.cn/glossary/" \l "term-21)启动时会设置操作系统级的 max open fds （即文件描述符最大允许值），这有助于防止耗尽文件描述符。 |
| 类型: | 64-bit Integer |
| 是否必需: | No |
| 默认值: | 0 |

fatal signal handlers

|  |  |
| --- | --- |
| 描述: | 如果设置了，将安装 SEGV 、 ABRT 、 BUS 、 ILL 、 FPE 、 XCPU 、 XFSZ 、 SYS 信号处理器，用于产生有用的日志信息。 |
| 类型: | Boolean |
| 默认值: | true |