

智能预取

用户指南

文档版本

02

发布日期

2021-03-30



版权所有 © 华为技术有限公司 2021。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目录

1 简介	1
2 环境准备	3
2.1 服务器环境	3
2.2 安装 bcache-tools	3
3 软件编译和安装	5
3.1 编译 bcache	5
3.2 安装和配置 bcache	6
3.3 安装并配置 aclient 和 hcache	7
4 软件使用	10
4.1 配置开机自动启动智能预取功能	10
4.2 启动分布式存储智能预取	11
4.2.1 配置 bcache	11
4.2.2 启动预取引擎	11
4.3 停止分布式存储智能预取	11
4.3.1 停止预取引擎	12
4.3.2 停止 bcache	12
4.4 启动大数据智能预取	12
4.4.1 启动前须知	12
4.4.2 配置 bcache	12
4.4.3 启动预取引擎	15
4.5 停止大数据智能预取	16
4.5.1 停止预取引擎	16
4.5.2 停止 bcache	16
4.6 获取版本信息	17
4.7 关于 aclient 和 hcache 日志	18
4.8 获取 IO 信息功能	19
4.9 故障处理	20
4.9.1 磁盘故障	20
4.9.2 缓存盘故障	20
4.9.3 sequential_cutoff 功能缺失	20
4.9.4 服务器重启后出现 bcache 设备异常时的处理	21
4.9.5 Ramdisk 做缓存盘时服务器重启后无法自动注册 bcache	21

A 修订记录..... 23

1 简介

针对分布式存储、大数据的Spark/HBase等解决方案中的存储IO密集型场景，访问IO存储器的性能对业务整体性能影响明显。同时，在这些场景里面，用户对存储器的单位容量成本也很敏感。在现在以及未来的很长一段时间里面，存储器容量大小与IO性能不可能兼具。同时，利用小容量的高速存储介质作为缓存盘，把预测可能被访问到的IO数据提前放入缓存盘中，下次直接从高速缓存中获取数据，可以显著的改善系统整体的存储IO性能。

分布式存储场景下的智能预取软件框架和大数据场景下智能预取软件框架分别如图1-1和图1-2所示。

图 1-1 分布式存储场景下的智能预取软件框架图

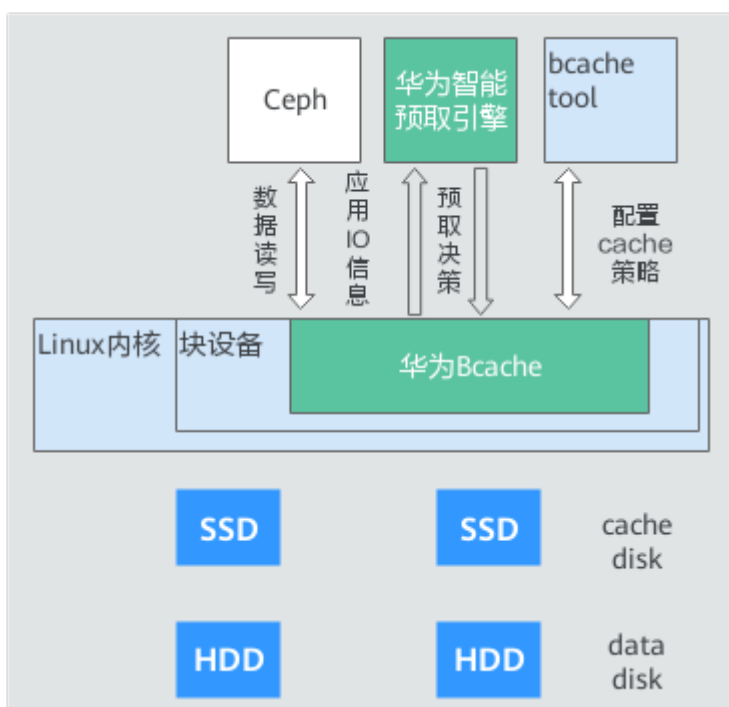
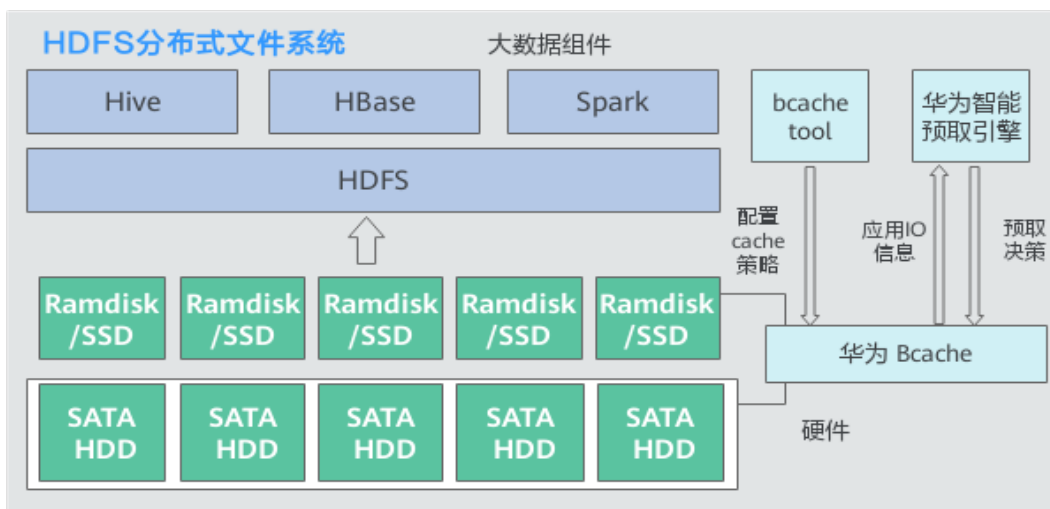


图 1-2 大数据场景下智能预取软件框架图



说明

- IO存储器，指硬盘、SSD。
- 性能即访问IO存储器的性能，指访问IO存储器的带宽、延迟、单位时间操作数。
- 小容量高速存储介质可以是基于RAM的Ramdisk，也可以是NVMe SSD。

智能预取加速功能，创新性的采用高速缓存盘配合高效的预取算法，提升系统存储IO性能，进而提升上述解决方案中存储IO密集型场景的整体性能。

智能预取加速功能，包含以下模块：

1. 内核态华为智能预取驱动bcache。
2. 用户态华为智能预取引擎框架acache_client。
3. 用户态华为智能预取引擎算法hcache。
4. bcache配置工具bcache-tools。

2 环境准备

- 2.1 服务器环境
- 2.2 安装bcache-tools

2.1 服务器环境

表 2-1 操作系统要求

操作系统	版本
CentOS	Linux release 7.6.1810 (AltArch)

表 2-2 软件要求

软件名称	版本	备注
GCC	7.3.0	acache_client在GCC7.3.0环境下编译，需要运行环境的GCC版本升级到7.3.0。
GNU Make	-	CentOS 7.6系统默认配套安装版本。
bcache-tools	-	请参考2.2 安装bcache-tools自行安装。

2.2 安装 bcache-tools

步骤1 安装bcache-tools的依赖包libblkid-devel。

```
yum install libblkid-devel
```

步骤2 获取bcache-tools。

下载链接：<https://github.com/g2p/bcache-tools/tree/v1.0.8>

步骤3 编译安装bcache-tools。

```
make && make install
```

----结束

3 软件编译和安装

3.1 编译bcache

3.2 安装和配置bcache

3.3 安装并配置acache_client和hcache

3.1 编译 bcache

编译环境准备

步骤1 确保环境使用CentOS-7-aarch64-Everything-1810.iso镜像安装。

步骤2 确认操作系统版本。

```
uname -a
```

操作系统显示版本如下所示。

```
Linux localhost.localdomain 4.14.0-115.10.1.el7a.aarch64 #1 SMP Thu Nov 26 14:15:20 CST 2020 aarch64  
aarch64 aarch64 GNU/Linux
```

步骤3 确认GCC版本。

```
gcc -v
```

说明

建议使用GCC 4.8.5（CentOS 7.6默认配套版本）进行编译。

GCC版本信息显示如下。

```
Using built-in specs.  
COLLECT_GCC=gcc  
COLLECT_LTO_WRAPPER=/usr/libexec/gcc/aarch64-redhat-linux/4.8.5/lto-wrapper  
Target: aarch64-redhat-linux  
Configured with: ../configure --prefix=/usr --mandir=/usr/share/man --infodir=/usr/share/info --with-  
bugurl=http://bugzilla.redhat.com/bugzilla --enable-bootstrap --enable-shared --enable-threads=posix --  
enable-checking=release --with-system-zlib --enable-__cxa_atexit --disable-libunwind-exceptions --enable-  
gnu-unique-object --enable-linker-build-id --with-linker-hash-style=gnu --enable-languages=c,c++,objc,obj-c  
++,java,fortran,ada,lto --enable-plugin --enable-initfini-array --disable-libgcj --with-isl=/build/builddir/build/  
BUILD/gcc-4.8.5-20150702/obj-aarch64-redhat-linux/isl-install --with-cloog=/build/builddir/build/BUILD/  
gcc-4.8.5-20150702/obj-aarch64-redhat-linux/cloog-install --enable-gnu-indirect-function --build=aarch64-  
redhat-linux  
Thread model: posix  
gcc version 4.8.5 20150623 (Red Hat 4.8.5-36) (GCC)
```

步骤4 确认内核头文件已安装。

```
ls /usr/src/kernels/$(uname -r)
```

安装情况如下所示。

```
arch block certs crypto drivers firmware fs include init ipc Kconfig kernel lib Makefile mm  
Module.symvers net samples scripts security sound System.map tools usr virt
```

----结束

源码编译

步骤1 获取智能预取bcache源码。

下载链接：<https://github.com/kunpengcompute/bcache>

步骤2 安装内核开发软件包。

```
yum install kernel-devel-4.14.0-115.el7a.0.1
```

步骤3 在“drivers/md/bcache”源码目录下，执行编译。

```
make -C /lib/modules/4.14.0-115.el7a.0.1.aarch64/build M=$(pwd)
```

步骤4 获取编译出来的目标文件：bcache.ko。

----结束

3.2 安装和配置 bcache

安装 bcache

以下就章节[3.1 编译bcache](#)编译好的bcache.ko文件进行安装说明。

步骤1 获取bcache.ko文件。**步骤2** 查看bcache模块的引用计数。

```
lsmod | grep -w bcache
```

确保bcache模块的引用计数为0。

```
bcache          458752  0
```

步骤3 确保当前环境没有加载bcache.ko。

```
modprobe -r bcache
```

或

```
rmmmod bcache
```

步骤4 删除安装目录。

```
rm -rf /lib/modules/$(uname -r)/kernel/drivers/md/bcache
```

步骤5 创建安装目录。

```
mkdir -p /lib/modules/$(uname -r)/kernel/drivers/md/bcache
```

步骤6 拷贝bcache.ko目标文件到安装目录。

```
cp bcache.ko /lib/modules/$(uname -r)/kernel/drivers/md/bcache
```

步骤7 压缩目标文件。

```
xz -z /lib/modules/$(uname -r)/kernel/drivers/md/bcache/bcache.ko
```

步骤8 更新bcache.ko模块依赖性。

```
depmod -a
```

步骤9 更新initramfs中的bcache.ko模块。

```
dracut --add-drivers bcache -f /boot/initramfs-$(uname -r).img
```

步骤10 确认bcache.ko是否安装成功。

以下两条命令执行后，显示的文件信息完全一致则表示安装成功。

```
lsinitrd /boot/initramfs-$(uname -r).img | grep bcache.ko.xz
-rw-r--r-- 1 root root 997136 Nov 30 11:45 usr/lib/modules/4.14.0-115.el7a.0.1.aarch64/kernel/drivers/md/
bcache/bcache.ko.xz
ls -l /lib/modules/$(uname -r)/kernel/drivers/md/bcache/bcache.ko.xz
-rw-r--r-- 1 root root 997136 Nov 30 11:45 /lib/modules/4.14.0-115.el7a.0.1.aarch64/kernel/drivers/md/
bcache/bcache.ko.xz
```

步骤11 加载bcache.ko。

```
modprobe bcache
```

----结束

配置 bcache

步骤1 在bcache.conf文件中配置bcache.ko模块加载参数。

acache_size值为20MB，会给bcache分配一个20MB大小的物理内存，作为存放IO信息的缓冲区。

```
echo "options bcache acache_size=20975616" > /etc/modprobe.d/bcache.conf
```

----结束

3.3 安装并配置 acache_client 和 hcache

RPM 方式安装 acache_client 和 hcache

步骤1 从官方渠道获取acache_client的RPM安装包和软件数字证书（安装包中包含hcache）。

- 方式一：华为企业网网站下载
[acache_client-1.0.0_0.10.0-1.el7.aarch64.rpm](#)
- 方式二：华为运营商网站下载
[acache_client-1.0.0_0.10.0-1.el7.aarch64.rpm](#)

RPM默认安装的文件及路径如表3-1所示。

表 3-1 RPM 方式安装文件与路径

文件	文件说明	安装位置
acache_client	智能预取框架。	/usr/local/bin/
bcache.conf	bcache的配置文件。	/etc/modprobe.d/
acache_client.service	systemctl启动acache_client所需要的配置文件。	/usr/lib/systemd/system/
hcache_config.cfg	hcache的配置文件。	/usr/local/bin/


文件	文件说明	安装位置
libhcache.so.[版本号]	hcache库文件。	/usr/lib64/

步骤2 获取软件包后，需要校验软件包，确保与网站上的原始软件包一致。

校验方法：


1. 获取软件数字证书和软件安装包，获取方式参见**步骤1**。
2. 在如下链接中获取校验工具和校验方法：
<https://support.huawei.com/enterprise/zh/tool/pgp-verify-TL1000000054>
3. 参见**2**中下载的《OpenPGP签名验证指南》进行软件包完整性检查。

步骤3 安装。

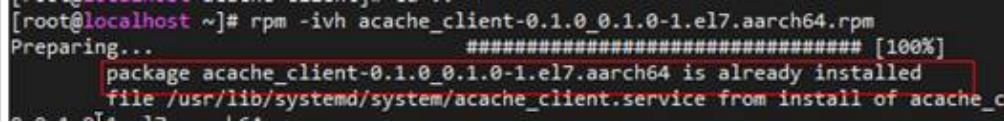
 **注意**

请使用root权限执行该步骤。

- 第一次安装
rpm -ivh acache_client-*x.x.x.x.x.x*-1.el7.aarch64.rpm
- 升级软件包安装
rpm -U acache_client-*x.x.x.x.x.x*-1.el7.aarch64.rpm

 **说明**

安装过程中若遇到“package acache_client... is already installed”的异常，可选择以下两种方法中的一种进行安装。



- 使用rpm -e acache_client卸载已经安装的rpm包再使用rpm -ivh安装。
- 使用rpm -ivh acache_client-*x.x.x.x.x.x*-1.el7.aarch64.rpm -force进行安装。

结束

配置 hcache

智能预取算法hcache会提供初始config文件hcache_config.cfg，存放目录默认与acache_client同目录，即“/usr/local/bin”下，算法在初始化加载时会读取文件，作为算法的初始参数。

hcache_config.cfg文件配置项说明如**表3-2**所示。

修改完后请按照**4.5.1 停止预取引擎**、**4.4.3 启动预取引擎**重启acache_client。

表 3-2 hcache_config.cfg 文件配置项说明

参数名	默认初始值	修改范围	含义
g_pagecache ReadaheadControl	1	[0,1]	预取算法接管pagecache功能开关，由客户授权。 <ul style="list-style-type: none">• 值为1，则打开智能预取接管pagecache功能，智能预取将动态修改pagecache的预取参数，协同Bcache侧进行预取。• 值为0时，关闭在智能预取接管pagecache功能，此时智能预取将只进行Bcache侧的预取。

4 软件使用

[4.1 配置开机自动启动智能预取功能](#)

[4.2 启动分布式存储智能预取](#)

[4.3 停止分布式存储智能预取](#)

[4.4 启动大数据智能预取](#)

[4.5 停止大数据智能预取](#)

[4.6 获取版本信息](#)

[4.7 关于acache_client和hcache日志](#)

[4.8 获取IO信息功能](#)

[4.9 故障处理](#)

4.1 配置开机自动启动智能预取功能

推荐将bcache和acache_client配置成开机启动。

配置 bcache 开机启动

步骤1 配置bcache.ko模块开机自动加载。

```
echo "bcache" > /etc/modules-load.d/bcache.conf
```

----结束

配置 acache_client 开机启动

步骤1 如果不想在root权限下执行，请先修改/dev/acache设备的所属用户和组。

查看/dev/acache设备所在用户和组。

```
ll /dev/acache
```

```
[root@agent1 system]# ll /dev/acache
crw----- 1 user1 group1 245, 0 Dec 15 11:58 /dev/acache
```

步骤2 再修改acache_client.service文件，增加user和group配置为当前/dev/acache设备的所属用户和组。

```
[Unit]
Description=acache client

[Service]
Type=simple
WorkingDirectory=/usr/local/bin/
ExecStart=/usr/local/bin/acache_client
Restart=on-failure
RestartSec=1s
User=user1
Group=group1

[Install]
WantedBy=multi-user.target
```

步骤3 设置acache_client开机启动。

```
sudo systemctl enable acache_client
```

----结束

说明

若不需要acache_client开机启动，则可以关闭该功能。

```
sudo systemctl disable acache_client
```

4.2 启动分布式存储智能预取

4.2.1 配置 bcache

为数据盘创建 bcache 盘

步骤1 擦除数据盘上的文件系统。

```
wipefs -a /dev/sdb --force
```

步骤2 创建数据盘的bcache盘。

```
make-bcache -B /dev/sdb --wipe-bcache
```

----结束

为 SSD 缓存盘创建 bcache 盘

参考[为SSD缓存盘创建bcache盘](#)章节的步骤。

4.2.2 启动预取引擎

参考[4.4.3 启动预取引擎](#)章节的步骤。

4.3 停止分布式存储智能预取

4.3.1 停止预取引擎

参考[4.5.1 停止预取引擎](#)章节的步骤。

4.3.2 停止 bcache

停用 ssd 缓存盘

参考[停用bcache ssd缓存盘](#)章节的步骤。

停用 bcache 盘

参考[停用bcache盘](#)章节的步骤。

卸载 bcache 驱动

参考[卸载bcache驱动](#)章节的步骤。

4.4 启动大数据智能预取

4.4.1 启动前须知

针对大数据场景部署智能预取，部署bcache后，原有的数据将会丢失，请谨慎处理。

4.4.2 配置 bcache

⚠ 注意

启动智能预取前，请停止当前集群服务程序。

📖 说明

以下以sdb盘为例说明操作步骤，读者请根据实际盘号操作。

为数据盘创建 bcache 盘

步骤1 删除磁盘上原有的数据，创建bcache盘，格式化bcache盘为ext4文件系统格式。

```
wipefs -a /dev/sdb --force  
make-bcache -B /dev/sdb --wipe-bcache  
mkfs.ext4 /dev/[bcache]
```

步骤2 修改bcache盘参数。

1. 获取max_sectors_kb（设备允许的最大请求大小）实际参数。

```
cat /sys/block/sdb/queue/max_sectors_kb
```

回显结果为：

[数据盘最大扇区数量]（例如1024）

2. 获取read_ahead_kb（预读数据大小）实际参数。

```
cat /sys/block/sdb/queue/read_ahead_kb
```

回显结果为：

[数据盘预读数量]（例如128）

3. 根据获取到的实际参数配置max_sectors_kb和read_ahead_kb。

```
echo 数据盘最大扇区数量 > /sys/block/[bcache]/queue/max_sectors_kb
echo 数据盘预读数量 > /sys/block/[bcache]/queue/read_ahead_kb
```

📖 说明

针对以上步骤命令中涉及的[数据盘最大扇区数量]、[数据盘预读数量]和[bcache]有如下两条说明。

- [数据盘最大扇区数量]和[数据盘预读数量]以具体的数据盘的值为准，不同的数据盘，其参数可能不一样。
- 具体[bcache]值需根据“/sys/block/”目录下内容进行设置，该目录下所有[bcache]设备都需要按此处理。

----结束

为 Ramdisk 缓存盘创建 bcache 盘

- 步骤1 创建Ramdisk盘。

```
modprobe brd rd_nr=12 rd_size=$((1048576))
```

📖 说明

命令中，12表示缓存盘数量；1048576表示缓存盘大小为1G，具体大小请根据业务自己配置，一般大数据场景配置1G大小即可。

- 步骤2 将缓存盘格式化为bcache盘。

```
make-bcache --wipe-bcache -b 262144 -C /dev/ram0
```

📖 说明

命令中，262144表示缓存盘block大小为256K。

- 步骤3 挂载缓存盘到对应的bcache盘。

```
echo $uuid > /sys/block/[bcache]/bcache/attach
```

📖 说明

命令中的\$uuid值取其结果的cset.uuid对应的uuid值，查看\$uuid值。
bcache-super-show /dev/ram0

- 步骤4 修改bcache策略。

```
echo writearound > /sys/block/[bcache]/bcache/cache_mode
echo 1 > /sys/block/[bcache]/bcache/clear_stats
echo 0 > /sys/block/[bcache]/bcache/readahead
echo 10 > /sys/block/[bcache]/bcache/writeback_percent
echo 0 > /sys/block/[bcache]/bcache/cache/congested_read_threshold_us
echo 0 > /sys/block/[bcache]/bcache/cache/congested_write_threshold_us
echo 1 > /sys/block/[bcache]/bcache/read_bypass
echo lru > /sys/block/[bcache]/bcache/cache/cache0/cache_replacement_policy
```

📖 说明

命令中涉及的具体[bcache]值需根据“/sys/block/”目录下内容进行设置，该目录下所有[bcache]设备都需要按此处理。

----结束

注意

创建完Ramdisk缓存盘后，系统会划分对应的内存给bcache使用，如果运行的业务需要使用大量内存，建议调整业务内存资源配置，以免业务运行因为内存不足而失败。比如Spark Kmeans，应用会消耗大量内存，此时需要调整Yarn的内存资源配置，确保业务不会因为内存不足而失败。

为 SSD 缓存盘创建 bcache 盘

步骤1 确保有空闲的SSD缓存盘。

举例说明：

```
nvme0n1      259:0      0  2.9T  0 disk
├─nvme0n1p5  259:5      0 147.2G  0 part
├─nvme0n1p11 259:11     0 147.2G  0 part
├─nvme0n1p3   259:3      0 147.2G  0 part
├─nvme0n1p1   259:1      0 147.2G  0 part
├─nvme0n1p8   259:8      0 147.2G  0 part
├─nvme0n1p6   259:6      0 147.2G  0 part
├─nvme0n1p12  259:12     0 147.2G  0 part
├─nvme0n1p4   259:4      0 147.2G  0 part
├─nvme0n1p10  259:10     0 147.2G  0 part
├─nvme0n1p2   259:2      0 147.2G  0 part
├─nvme0n1p9   259:9      0 147.2G  0 part
└─nvme0n1p7   259:7      0 147.2G  0 part
```

步骤2 格式化缓存盘为bcache盘。

```
make-bcache --wipe-bcache -b 262144 -C /dev/nvme0n1p1
echo /dev/nvme0n1p1 > /sys/fs/bcache/register
```

说明

262144表示缓存盘block大小为256K。

步骤3 挂载缓存盘到对应的bcache盘。

```
echo $uuid > /sys/block/[bcache]/bcache/attach
```

说明

命令中的\$uuid值取其结果的cset.uuid对应的uuid值，查看\$uuid值。
bcache-super-show /dev/nvme0n1p1

步骤4 修改bcache策略。

```
echo writeback > /sys/block/[bcache]/bcache/cache_mode
echo 1 > /sys/block/[bcache]/bcache/clear_stats
echo 0 > /sys/block/[bcache]/bcache/readahead
echo 10 > /sys/block/[bcache]/bcache/writeback_percent
echo 0 > /sys/block/[bcache]/bcache/cache/congested_read_threshold_us
echo 0 > /sys/block/[bcache]/bcache/cache/congested_write_threshold_us
echo 0 > /sys/block/[bcache]/bcache/cache/read_bypass
echo 0 > /sys/block/[bcache]/bcache/sequential_cutoff
echo lru > /sys/block/[bcache]/bcache/cache/cache0/cache_replacement_policy
```

步骤5 修改bcache的queue目录参数。

在以下命令中，根据后端盘修改max_sectors_kb的参数为适合值。
echo 1024 > /sys/block/[**bcache**]/queue/max_sectors_kb

说明

命令中涉及的具体 **[bcache]** 值需根据 “/sys/block/” 目录下内容进行设置，该目录下所有 [bcache] 设备都需要按此处理。

----结束

挂载数据盘对应的 bcache 盘

步骤1 手动装载 bcache 盘对应的数据盘。

```
mount /dev/[bcache] data/[data]
```

说明

请根据实际数据盘目录配置 **[bcache]** 和 **[data]**。

----结束

注意

创建完成 bcache 磁盘后，原先 /etc/fstab 中的操作系统默认启动挂载数据盘的配置需要删除，以免重启服务器后数据盘被重新格式化成非 bcache 盘。

4.4.3 启动预取引擎

注意

请确保 bcache 盘的文件系统格式为 ext4，其他文件系统暂不支持。
通过 **lsblk -fs** 命令，查看 bcache 盘的文件系统格式为 ext4：

```
[root@agent1 ~]# lsblk -fs
NAME        FSTYPE LABEL UUID                                MOUNTPOINT
bcache0     ext4      91e09f8a-2619-47e8-afbc-bd4699382f2d /data/data10
├─ram0      bcache    86cbc8dd-893e-40bc-b988-cb7daa4d33e7
├─sdb1      bcache    5d33496f-07ef-45da-a601-ca00a2e57a27
└─sdb
```

步骤1 启动 acache_client 服务。

```
sudo systemctl start acache_client
```

步骤2 确认 acache_client 已经启动成功。

```
sudo systemctl status acache_client
```

```
[root@localhost home]# sudo systemctl status acache_client
● acache_client.service - acache client
   Loaded: loaded (/usr/lib/systemd/system/acache_client.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2020-12-04 15:45:00 CST; 1s ago
     Main PID: 82176 (acache_client)
    CGroup: /system.slice/acache_client.service
            └─82176 /usr/local/bin/acache_client

Dec 04 15:45:00 localhost.localdomain systemd[1]: Started acache client.
```

若回显信息中显示 “**active (running)**”，则 acache_client 启动成功。

📖 说明

启动acache_client服务时若遇到“Warning: acache_client.service changed on disk. Run 'systemctl daemon-reload' to reload units.”的告警信息，请参考以下步骤解决。

```
[root@localhost ~]# systemctl start acache_client
Warning: acache_client.service changed on disk. Run 'systemctl daemon-reload' to reload units.
```

1. 重新加载服务配置文件。
systemctl daemon-reload
2. 启动acache_client服务。
systemctl start acache_client

----结束

4.5 停止大数据智能预取

4.5.1 停止预取引擎

步骤1 停止acache_client服务。

```
sudo systemctl stop acache_client
```

----结束

4.5.2 停止 bcache

⚠ 警告

停止智能预取前，请停止当前集群服务程序。

停用 bcache ramdisk 缓存盘

步骤1 卸载缓存盘。

```
echo 1 > /sys/block/[bcache]/bcache/detach
```

步骤2 注销缓存盘。

```
echo 1 > /sys/fs/bcache/[uuid]/unregister
```

步骤3 删除ramdisk盘。

```
modprobe -r brd
```

📖 说明

针对上述步骤命令中的**[bcache]**和**[uuid]**有如下两点说明。

1. 具体**[bcache]**值需根据“/sys/block/”目录下内容进行置，该目录下所有[bcache]设备都需要按此处理。
2. 具体**[uuid]**值需根据“/sys/fs/bcache”目录下内容进行设置，该目录下所有[uuid]设备都需要按此处理。

----结束

停用 bcache ssd 缓存盘

步骤1 卸载缓存盘。

```
echo 1 > /sys/block/[bcache]/bcache/detach
```

步骤2 注销缓存盘。

```
echo 1 > /sys/fs/bcache/[uuid]/unregister
```

说明

针对上述步骤命令中的 **[bcache]** 和 **[uuid]** 有如下两点说明。

1. 具体 **[bcache]** 值需根据 “/sys/block/” 目录下内容进行设置，该目录下所有 [bcache] 设备都需要按此处理。
2. 具体 **[uuid]** 值需根据 “/sys/fs/bcache” 目录下内容进行设置，该目录下所有 [uuid] 设备都需要按此处理。

----结束

卸载需要格式化 bcache 盘的数据盘

步骤1 卸载需格式化为 bcache 盘的数据盘。

```
umount /data/[data]
```

说明

卸载目录 **[data]** 根据实际情况配置。

----结束

停用 bcache 盘

步骤1 停用 bcache 盘。

```
echo 1 > /sys/block/[bcache]/bcache/stop
```

说明

上述命令中涉及的具体 **[bcache]** 值需根据 “/sys/block/” 目录下内容进行设置，该目录下所有 [bcache] 设备都需要按此处理。

----结束

卸载 bcache 驱动

步骤1 卸载 bcache 驱动。

```
modprobe -r bcache
```

----结束

4.6 获取版本信息

获取 bcache 版本信息

步骤1 获取 bcache 版本信息。

```
modinfo bcache
```

例如：

```
[root@localhost ~]# modinfo bcache
filename:      /lib/modules/4.14.0-115.el7a.0.1.aarch64/kernel/drivers/md/bcache/bcache.ko.xz
license:       GPL
author:        Kent Overstreet <koverstreet@google.com>
version:       0.2.0
author:        Kent Overstreet <kent.overstreet@gmail.com>
license:       GPL
rhelversion:   7.6
srcversion:    BFF3AE30F3480BA68DD817C
depends:
name:          bcache
vermagic:      4.14.0-115.el7a.0.1.aarch64 SMP mod_unload modversions aarch64
parm:          acache_size:acache ringbuf size in byte (int)
parm:          prefetch_workers:num of workers which process samples (int)
```

----结束

获取 acache_client 和 hcache 版本信息

步骤1 获取acache_client和hcache的版本信息。

```
acache_client --version
```

或

```
acache_client -v
```

----结束

4.7 关于 acache_client 和 hcache 日志

acache_client和hcache的日志都保存在“/var/log/prefetch/”目录下的acache_client.log和acache_client_[序号].log文件中，时间较久的在acache_client_[序号].log中。一共可以有20个acache_client_[序号].log文件，[序号]中的序号越大时间越久。

默认单个日志文件大小为10M，默认打印info级别日志，可以在启动命令后面添加选项对日志文件大小和日志打印等级进行修改。

举例说明如下：

- 使用--logsize来修改单个日志文件的大小，--loglevel来修改日志级别。

- 将单个日志文件的大小设置成20M（20480KB）。

```
acache_client --logsize=20480
```

- 将日志打印等级设置成ERROR级别。

ERROR级别主要有以下4个选项：

- 1：打印error等级的信息日志。
- 2：打印warning等级的信息日志。
- 3：打印info等级的信息日志。
- 4：打印debug等级的信息日志。

```
acache_client --loglevel=4
```

- 将日志文件大小和日志打印等级配置修改进acache_client.service文件中。

- 按照[4.5.1 停止预取引擎](#)停止acache_client。
- 修改acache_client.service文件。

文件路径：“/usr/lib/systemd/system/acache_client.service”

```
[root@localhost ~]# cat /usr/lib/systemd/system/acache_client.service
[Unit]
Description=acache client

[Service]
Type=simple
WorkingDirectory=/usr/local/bin/
ExecStart=/usr/local/bin/acache_client --logsize=20480 --loglevel=4
Restart=on-failure
RestartSec=1s

[Install]
WantedBy=multi-user.target
```

⚠ 注意

不建议在acache_client使用过程中删除“/var/log/prefetch/”目录下的acache_client.log文件，会导致日志无法正常生成。

如果出现误删，请重启acache_client恢复。

4.8 获取 IO 信息功能

智能预取用户态程序acache_client提供获取IO信息功能。

步骤1 将IO信息重定向到文件中。

```
acache_client -d > acache_client_trace
```

文件信息内容如下，包括：op（IO状态），time（时间），majorDev（主设备号），minorDev（次设备号），offset（IO地址），length（IO长度）。

```
acache_client version: 0.2.0
iopattern version: 0.2.0
got dev size 20975616
  op,           time(ns),majorDev,minorDev,      offset(B),    length(B)
  L,      26448852970,      8,      176,    2142962597888,      524288
  L,           5360780,      8,      16,    2004934557696,      196608
  L,     27759593669890,      8,     160,    2006765998080,      262144
  L,           6078680,      8,     128,    2006236037120,      262144
  L,     27759447995700,      8,      48,    2005744787456,     1048576
  L,           6366450,      8,     192,    2003712000000,      524288
  R,     27760108177460,      8,     160,    2999999852544,        4096
  L,           27220,      8,     160,    2999999852544,        4096
  R,     27760108373420,      8,     160,    2007357083648,      262144
```

----结束

4.9 故障处理

4.9.1 磁盘故障

问题现象

系统中出现故障磁盘。

处理步骤

步骤1 停止智能预取。

大数据场景业务请参考[4.5 停止大数据智能预取](#)，分布式存储场景业务请参考[4.3 停止分布式存储智能预取](#)。

步骤2 重新更换磁盘。

步骤3 启动智能预取。

大数据场景业务请参考[4.4 启动大数据智能预取](#)，分布式存储场景业务请参考[4.2 启动分布式存储智能预取](#)。

----结束

4.9.2 缓存盘故障

问题现象

系统中出现缓存盘故障时。

处理步骤

步骤1 停用智能预取。

大数据场景业务请参考[4.5 停止大数据智能预取](#)，分布式存储场景业务请参考[4.3 停止分布式存储智能预取](#)。

步骤2 重新更换缓存盘。

步骤3 启动智能预取。

大数据场景业务请参考[4.4 启动大数据智能预取](#)，分布式存储场景业务请参考[4.2 启动分布式存储智能预取](#)。

----结束

4.9.3 sequential_cutoff 功能缺失

问题现象

使用CentOS 7.6官方内核时，系统中的sequential_cutoff功能缺失，会出现如下现象：


```
# echo 4M > /sys/block/bcache0/bcache/sequential_cutoff  
-bash: /sys/block/bcache0/bcache/sequential_cutoff: Permission denied
```

处理步骤

出现这种现象的原因是CentOS 7.6官方内核没有把sequential_cutoff功能编译进去，解决办法是获取官方内核，修改内核config配置文件。

- 步骤1 将CONFIG_BCACHE宏定义打开。
- 步骤2 重新编译内核。
- 步骤3 升级内核。
- 步骤4 按照[3 软件编译和安装](#)编译安装bcache。

----结束

4.9.4 服务器重启后出现 bcache 设备异常时的处理

问题现象

服务器异常重启后，出现无法正常make-bcache，正常卸载bcache.ko的错误。

处理步骤

- 步骤1 查看是否bcache被应用程序使用了。

```
lsmod | grep bcache
```
- 步骤2 若bcache已经被应用程序使用了的情况下请排查以下地方。
 - 1. SSD缓存盘是否已经被创建。
如果已经创建了缓存盘bcache，请先注销缓存盘，参考[停用bcache ssd缓存盘](#)步骤执行。
 - 2. 当前是否存在bcache设备。
如果存在，如需更换bcache驱动，请先停止bcache设备，参考[停用bcache盘](#)步骤执行。
 - 3. 后端盘 “/sys/block/sd[\$i]” 目录是否存在bcache目录。
如果存在bcache目录，则执行以下命令。

```
echo 1 > /sys/block/sd[$i]/bcache/stop
```
 - 4. acache_client服务程序是否启动。
如果启动，则停止应用程序。

```
systemctl stop acache_client
```

----结束

4.9.5 Ramdisk 做缓存盘时服务器重启后无法自动注册 bcache

问题现象

Ramdisk做缓存盘时服务器重启后无法自动注册bcache，现象是lsblk命令执行后，各后端盘的bcache盘缺失，原因是当服务器重启时，Ramdisk上保存的bcache信息会丢失，此时在自动注册bcache时，Ramdisk上没有有效的bcache信息，也就无法自动注册bcache。

处理步骤

在出现以上现象，无法自动注册bcache的情况下，可以使用以下步骤手动注册bcache：

步骤1 确认/dev/sdb是否有bcache盘。

```
lsblk /dev/sdb
```

- 当显示结果如下时，表示/dev/sdb有bcache分区，注册bcache已经完成，结束操作。

```
[root@localhost ~]# lsblk /dev/sdb
NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdb          8:16   0   7.3T  0 disk
└─bcache1    252:16   0   7.3T  0 disk
```

须知

例子中显示/dev/sdb的分区名是bcache1，不论bcache后面跟的数值是多少，都认为/dev/sdb已经有bcache分区了。

- 当显示结果如下时，表示/dev/sdb没有bcache分区，bcache盘没有注册完成，继续执行以下操作。

```
[root@localhost ~]# lsblk /dev/sdb
NAME MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
sdb   8:16   0   7.3T  0 disk
```

步骤2 确认bcache是否处于running状态。

```
cat /sys/block/sdb/bcache/running
0
```

running值为0时，表示不处于running状态；值为1时，表示处于running状态。

步骤3 如果步骤2的running结果为0，则执行此步骤以停止bcache，否则bcache已经注册完成，结束操作。

```
echo 1 > /sys/block/sdb/bcache/stop
```

步骤4 执行4.4 启动大数据智能预取。

----结束

A 修订记录

发布日期	修订记录
2021-03-30	第二次正式发布。 <ul style="list-style-type: none">更新acache_client和hcache软件包获取路径。新增故障处理4.9.5 Ramdisk做缓存盘时服务器重启后无法自动注册bcache。
2021-01-20	第一次正式发布。