CHINA LINUX KERNEL
中国Linux内核开发者大会

華中科技大學
网络空间安全学院
School of Cyber Science and Engineering, HUST

第19届中国Linux内核开发者大会

赞助单位

华中科技大学 | HUAWEI | 阿里云 | FUJITSU | 腾讯云
oppo | 火山引擎 | vivo | 龙芯中科 LOONGSON TECHNOLOGY | 蚂蚁集团 ANT GROUP

支持单位

清华大学 Tsinghua University | 迪捷软件 DIGIPROTO | OpenEuler | OpenAnolis 龙蜥社区 | OpenCloudOS

支持社区&媒体

CSDN | 云巅论剑 | sf 思否 | 阅码场 yomocode | InfoQ 极客传媒
51CTO | 开源江湖

2024年10月　湖北 • 武汉

華中科技大學

EROFS文件系统最小化内存占用及申请时延的优化方案

郭纯海 vivo文件系统工程师

目录

Part 1

背景介绍

EROFS文件系统应用痛点

EROFS (Enhanced Read-Only File System) 是新近几年被广泛使用的内核原生只读压缩文件系统。相比其它非压缩文件系统，它在设计实现时，需要额外申请更多内存资源用于解压和管理操作。

应用痛点

低内存场景下，存在应用启动变慢和使用卡顿这类的性能问题

从一个应用场景出发

某个应用启动慢的场景拆解：

– 分析EROFS申请page分配耗时数据显示：次数极少的slowpath，page分配耗时占比却高达64%~84%

应用启动子阶段	page分配耗时(ms)	page分配次数	slowpath耗时(ms)	slowpath耗时占比	slowpath次数	slowpath次数占比
X子阶段	111.4	13537	72	64.6%	39	0.3%
Y子阶段	144.8	1388	123	84.9%	48	3.4%

总结：EROFS的性能会受内存压力影响

EROFS文件系统痛点原由

EROFS压缩文件系统读取文件示例（Page4 ~ Page7 共4个 Page）：

① 从存储设备读取2个block到2个压缩page中 — **优势：** 减少I/O数量

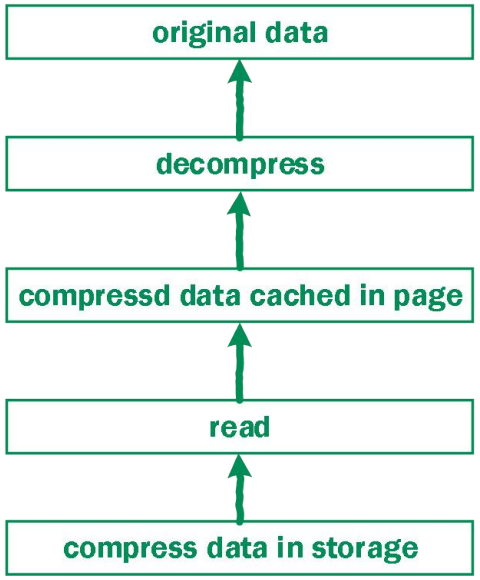
② 把这2个压缩page解压成7个page后，返回后4个page给应用（前3个page会被释放）

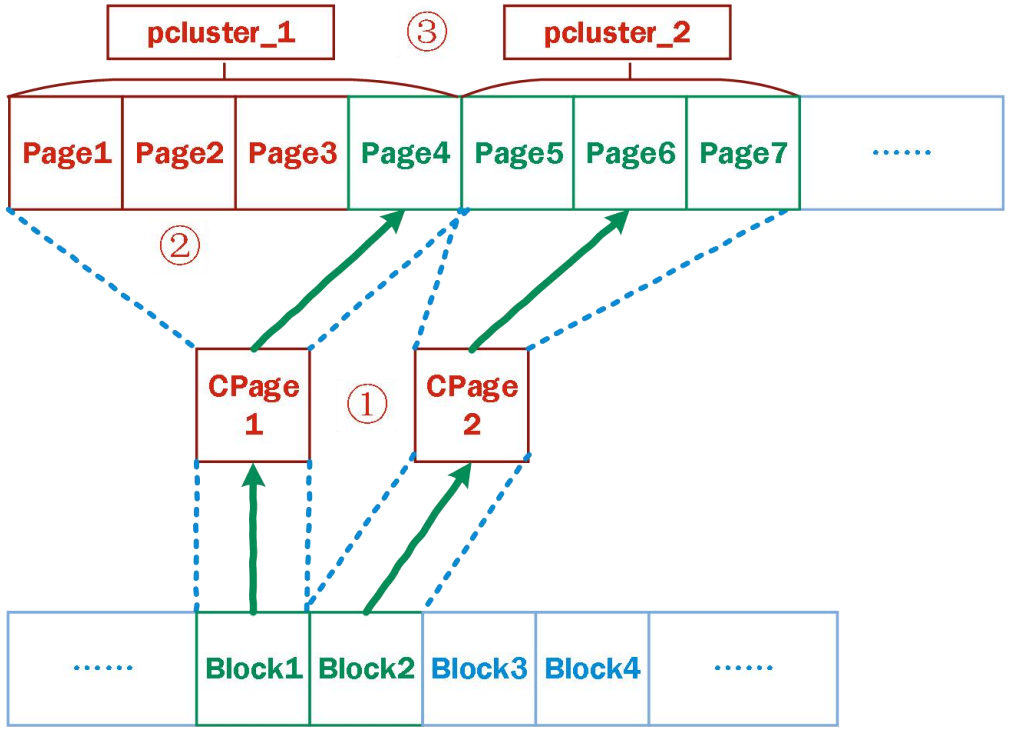
③ 其中涉及两个pcluster，对应需要申请2个pcluster结构体进行管理

劣势：每次读取都可能需要申请以上3种内存资源

内存占用量（分蛋糕游戏）

内存分配次数（抢凳子游戏）





Part 2

优化方案

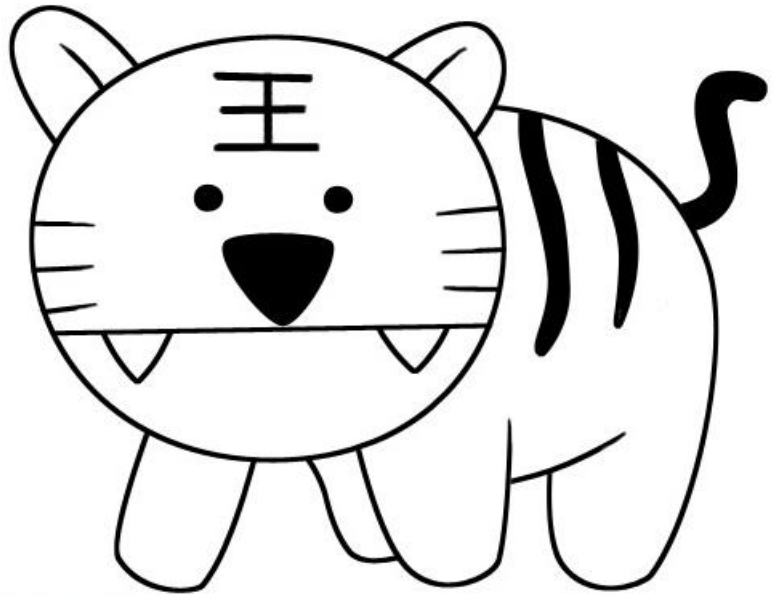
EROFS使用的三种内存资源

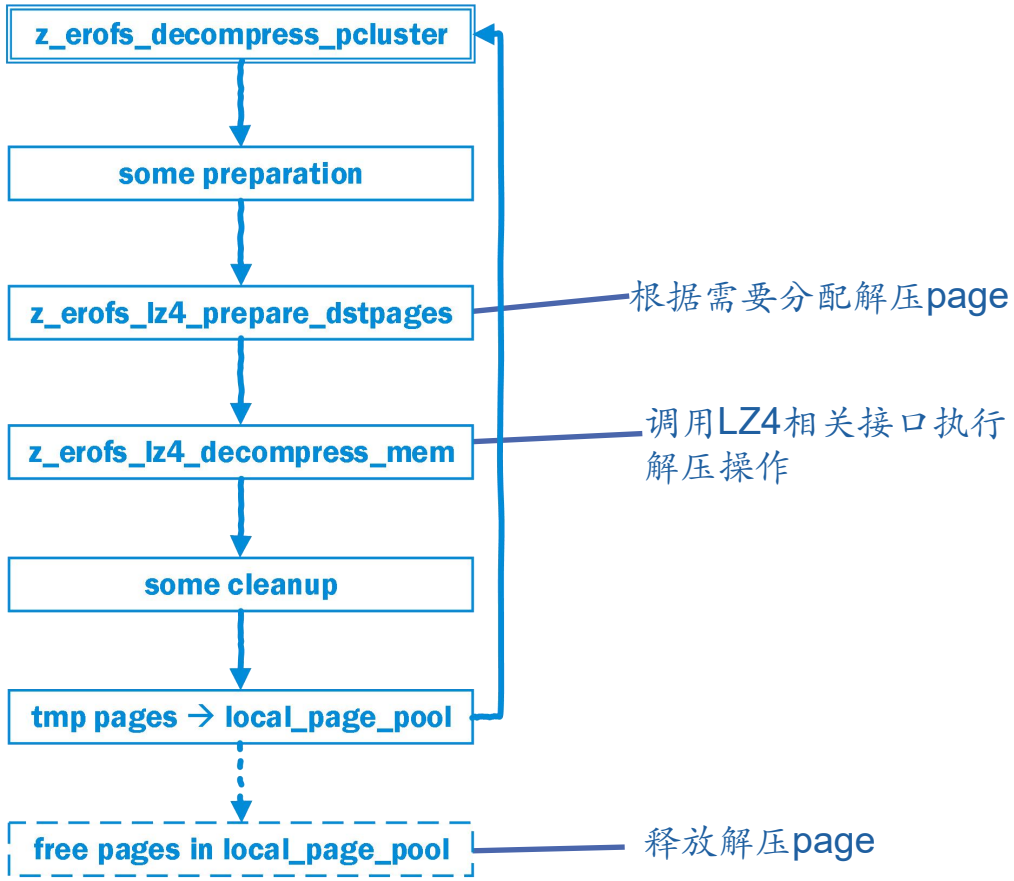
- 解压使用的page
- 保存压缩数据的page
- 关键结构体的缓存

解压page的分析

EROFS解压page特点

- page分配次数多（占比超过45%）
- 实际内存占用量少（快速申请，快速释放）





解压page的优化方案

- **优化方案**

  建立一个EROFS专用的小型page pool，分配解压page

  - **优势**
    - 系统内存page分配数量减少
    - 低内存场景，解压page分配耗时减少
  - **劣势**
    - 需要引入常驻page——最大占用量会不会太大？ 实测256K内存基本可以满足相关场景
      - 多线程并发
      - LZ4滑动窗口大小

解压page优化方案的收益

低内存场景下，应用启动过程中解压page申请耗时对比数据：

软件版本	解压page申请耗时(ms)
默认无优化版本	3434
page pool 优化版本	21
耗时减少比例	99.39%

结论：低内存场景下，使用专有page pool，解压page申请耗时减少99%，数据提升显著

解压page优化方案：不使用page pool

解压page申请耗时长原因进一步探索：

- 原因一：使用 __GFP_NOFAIL 申请page（一直等）
  - 针对预读优化：

    __GFP_NOFAIL → GFP_NOWAIT | __GFP_NORETRY
  - 优化效果：page申请耗时减少约20%（见下表）
- 原因二：page申请次数多
  - 优化方法：减少LZ4压缩滑动窗口大小
  - 优化效果：1/4滑动窗口大小 → page申请耗时减少约40%（见下表）
    - 实测压缩率基本没变化：9,117,044 KB → 9,113,096 KB (99.95%)

page申请耗时(ms)	原始版本	预读优化版本	优化比例
原始滑动窗口大小	3364	2684	-20.2%
1/4 滑动窗口大小	2079	1610	-22.5%
优化比例	-38.1%	-40%	

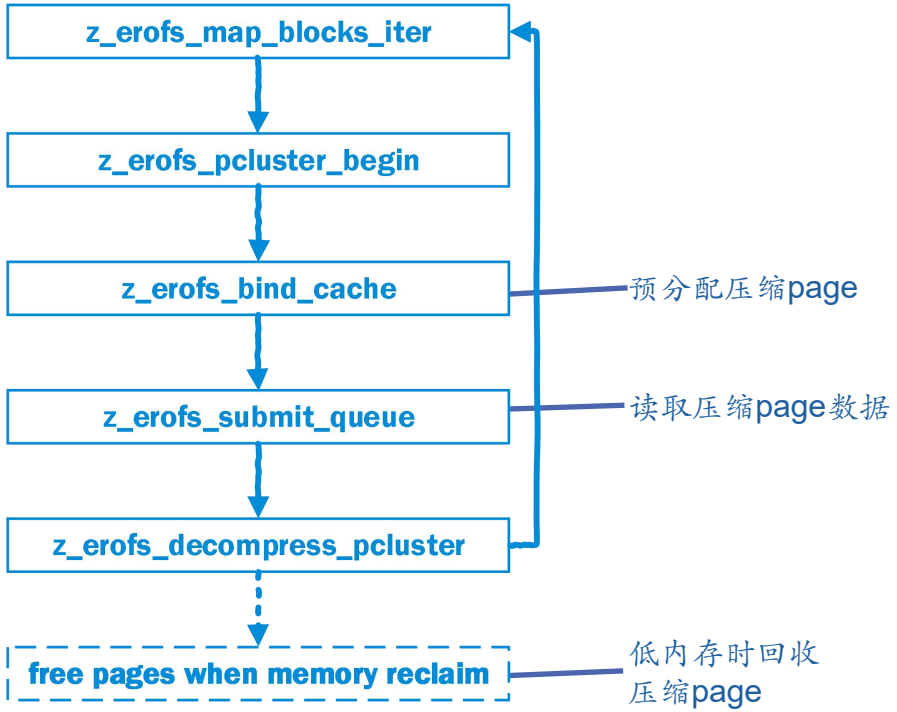
不使用page pool方案优化效果：减少约52%的page申请耗时 (3364 →1610)

补充：EROFS最新主线已经彻底去掉__GFP_NOFAIL，预期还有优化

压缩page的分析

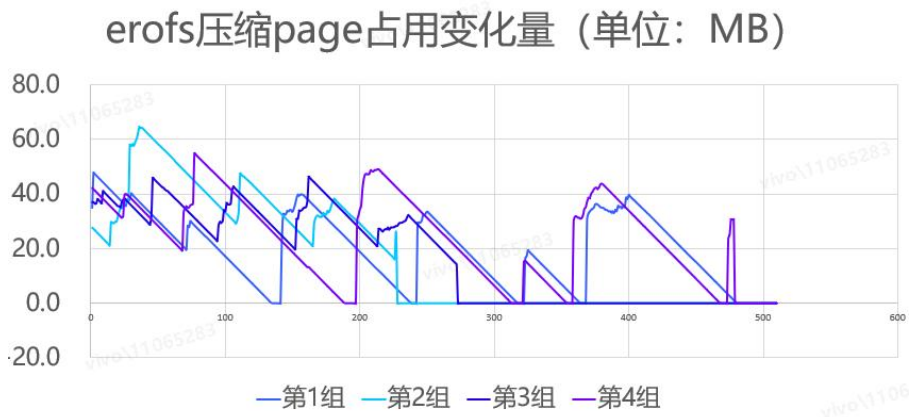
EROFS压缩page特点

- page分配次数多（占比超过45%）
- 内存实际占用量多（上限由内存回收阈值决定）
- 低内存场景：回收时快速下降，后续又随着使用快速上升



命中率分析

命中率很低（平均4%）→ 缓存的价值较低



压缩page低内存场景的优化方案和收益

低内存场景优化方案：

- 即时释放压缩page（不缓存） → 快速申请和释放
- 从page pool中分配

低内存场景优化收益

- 系统内存占用量和分配次数减少接近100%
- 压缩page分配耗时减少99%

代价：不缓存压缩page会增加3.5%左右的I/O量

关键结构体缓存的优化和收益

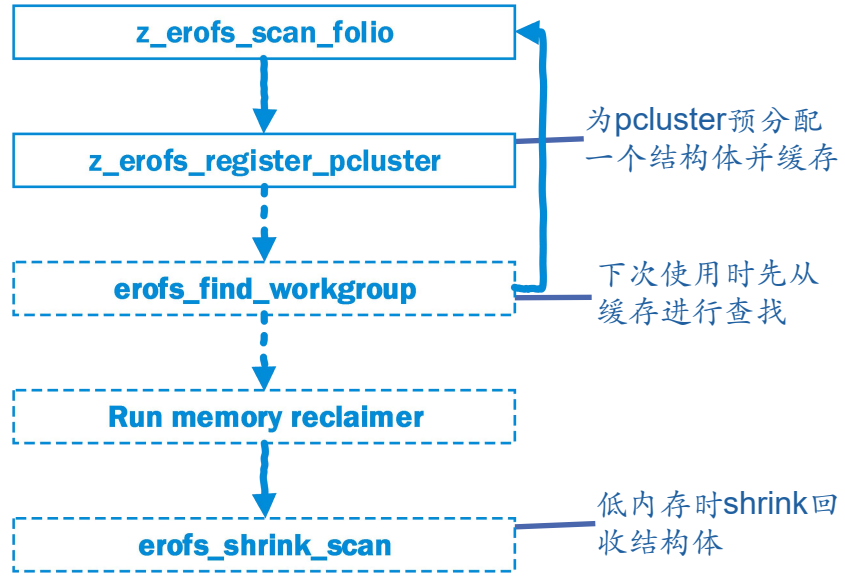
- pcluster结构体的特点
  - 分配次数多（手机开机后时申请50+万个）
  - 内存占用量不低（手机开机后占用接近80MB）
  - **命中率低（平均7%），创建成本小（无需I/O）→ 缓存价值低**

- 优化方法
  - 内存宽松场景：只缓存包含压缩page的pcluster结构体

    收益：分区拷贝实验的对比数据显示：
    - 减小内存占用：约95%
    - 减少内存回收开销：约95%
  - 低内存场景：即时释放（不缓存）

    收益：内存占用和回收开销减少接近100%

```
struct z_erofs_pcluster {
    struct erofs_workgroup obj;                  对应的blkaddr
    unsigned int length;                         原始数据长度
    unsigned int pclustersize;                   压缩数据长度
    unsigned char algorithmformat;               压缩算法
    struct z_erofs_bvec compressed_bvecs[];      压缩page信息
    ......
}
```



优化收益总结

- 优化收益

综合前述，EROFS内存优化后低内存场景的收益情况如下：

内存资源类型	低内存场景收益
解压page优化	page分配耗时减少99%
压缩page优化	page分配耗时减少99% 系统内存占用量和分配次数减少近100%
关键结构体缓存的优化	系统内存占用量和回收次数减少近100%

优化提交汇总

开源提交：

https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=79f504a2cd3c0b7d953d0015618a2a41559a2cfd
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=0f6273ab46375b62c8dd5c987ce7c15877602831
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=d6db47e571dcaecaeaafa8840d00ae849ae3907b
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=f36f3010f67611a45d66e773bc91e4c66a9abab5
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=cacd5b04e24c74a813c694ec7b26a1a370b5d666
https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=d9281660ff3ffb4a05302b485cc59a87e709aefc
https://lore.kernel.org/linux-erofs/20240930140424.4049195-1-guochunhai@vivo.com/T/#u

Thank You~