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

華中科技大學 | 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

内存压缩大页(ZHP)优化

ZRAM Huge Page， Based on Android Smartphone

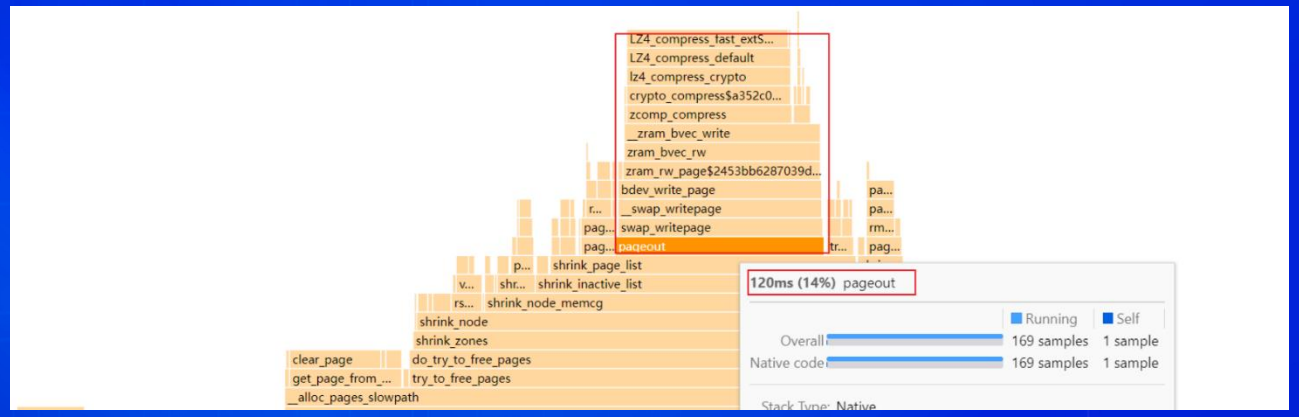
袁晓峰

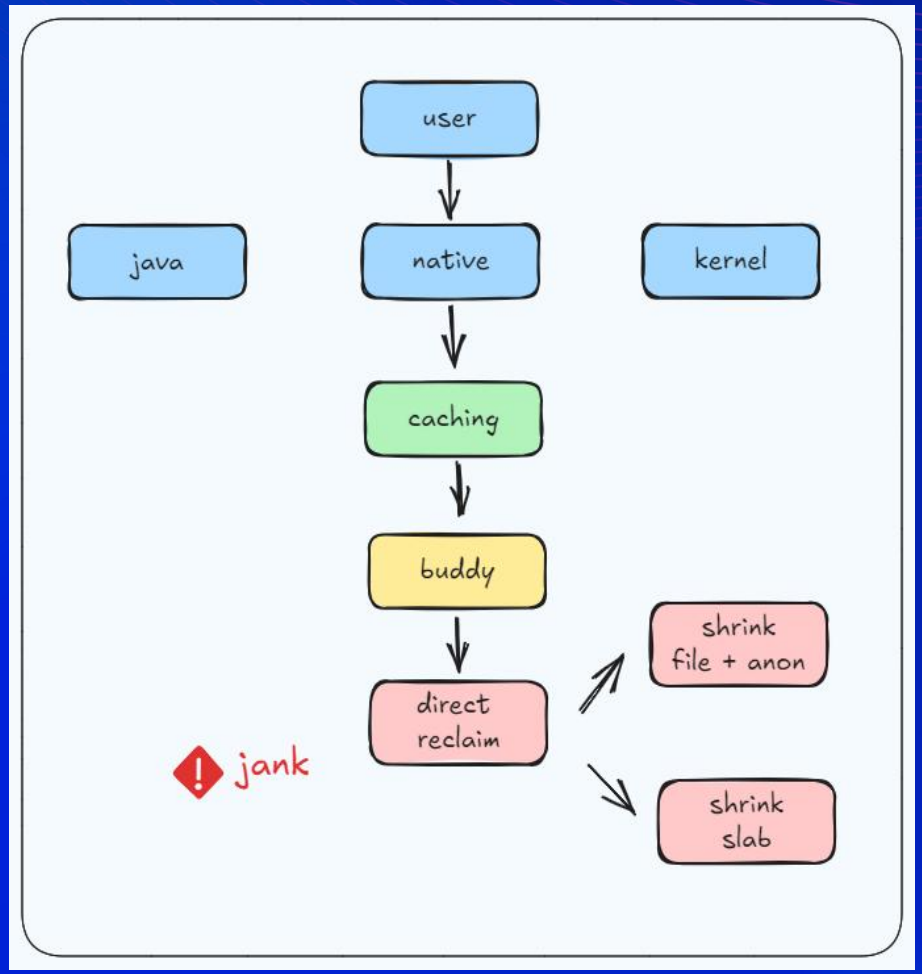
vivo性能优化专家

目录

1.1 背景 —— 内存&性能

- 内存是影响用户体验的**关键系统资源**
- direct reclaim会**阻塞**关键流程
- zram压缩匿名页**耗时长**





*** During an application startup, 14% of the time is spent on zram compression**

1.2 背景 —— zram压缩效率

影响zram压缩效率的软件因素：

- 压缩速率
- 压缩比
  - **压缩算法**
  - **内存数据**

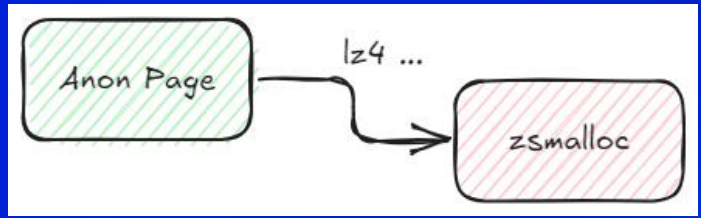
ZHP（Zram Huge Page），内存压缩大页

是指经zram压缩后的size大于huge_class_size

压缩后仍需占用一个page

Compressor name	Ratio	Compression	Decompress.
zstd 1.5.6 -1	2.887	510 MB/s	1580 MB/s
zlib 1.2.11 -1	2.743	95 MB/s	400 MB/s
brotli 1.0.9 -0	2.702	395 MB/s	430 MB/s
zstd 1.5.6 --fast=1	2.437	545 MB/s	1890 MB/s
zstd 1.5.6 --fast=3	2.239	650 MB/s	2000 MB/s
quicklz 1.5.0 -1	2.238	525 MB/s	750 MB/s
lzo1x 2.10 -1	2.106	650 MB/s	825 MB/s
lz4 1.9.4	2.101	700 MB/s	4000 MB/s
lzf 3.6 -1	2.077	420 MB/s	830 MB/s
snappy 1.1.9	2.073	530 MB/s	1660 MB/s

* **https://github.com/facebook/zstd**



2.1 ZHP现状 — system

一次开机后的匿名内存状态分布，以lz4压缩算法为例：

- 约2.7%内存压缩后无法节省内存（ZHP），约占压缩后8.6%的内存
- 约20%内存的压缩占比在50%以上，即经压缩节省的内存 < 50% PAGE_SIZE，约占压缩后39%内存

algorithm	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	ZHP	50%+
lz4	13.92%	10.05%	13.08%	19.70%	23.00%	12.36%	3.71%	1.59%	1.09%	0.86%	0.63%	2.66%	20.25%
lzo	9.25%	7.76%	11.85%	34.01%	21.61%	6.83%	3.65%	1.61%	1.12%	1.15%	1.17%	2.70%	15.53%
lzo-rle	14.08%	11.21%	16.71%	26.54%	18.36%	5.86%	2.88%	1.35%	1.03%	0.90%	1.08%	3.08%	13.10%

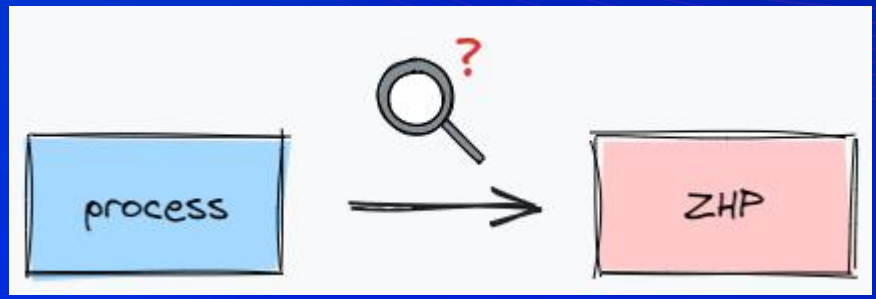
after compression

algorithm	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%	ZHP	50%+
lz4	0.01%	3.27%	8.49%	19.16%	29.82%	20.03%	7.23%	3.62%	3.54%	2.79%	2.04%	8.63%	39.26%
lzo	0.01%	2.42%	7.37%	31.71%	26.86%	10.62%	6.81%	3.51%	3.49%	3.58%	3.62%	5.42%	31.63%
lzo-rle	0.01%	3.96%	11.78%	28.04%	25.86%	10.33%	6.10%	3.35%	3.63%	3.16%	3.79%	10.84%	30.36%

* Anonymous page status after boot

2.2 ZHP现状 — process

process	Anon pages	ZHP pages	ZHP Rate
logd	3523	1818	51.60%
fingerprint@3.1	630	141	22.38%
sectee@1.0-serv	799	148	18.50%
diag_mdlog	447	69	15.44%
Input method	19274	2469	12.81%
vusbd	2042	190	9.31%
composer-service	4903	411	8.38%
provider@1.0-se	9501	707	7.44%
surfaceflinger	6064	272	4.49%
om.vivo.gallery	15330	656	4.28%
d.process.acore	13537	473	3.49%
com.tencent.mm	23126	805	3.48%
droid.ugc.aweme	49924	1020	2.04%
system_server	54778	1044	1.91%



- 进程ZHP占匿名页比例平均约为**2%**
- 部分进程ZHP占比很高（>10%），如native进程logd、输入法应用等
- 部分进程ZHP量很多（1000+），如系统进程system_server等

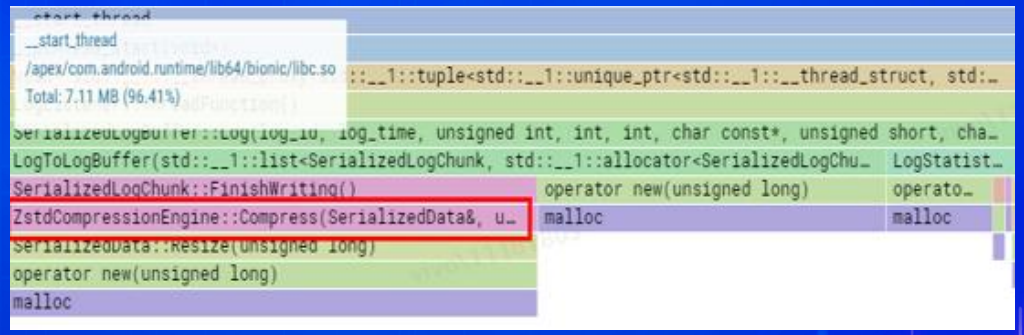
2.3 ZHP现状 — vma

vma	Anon pages	ZHP pages	ZHP Rate
[anon:scudo:primary]	373	366	98.12%
[anon:scudo:primary]	339	332	97.94%
[anon:scudo:primary]	276	270	97.83%
[anon:scudo:primary]	379	57	15.04%
[anon:scudo:primary]	30	29	96.67%
[anon:scudo:primary]	93	23	24.73%
[anon:scudo:primary]	29	18	62.07%

*** logd vma related**

vma	Anon pages	ZHP pages	ZHP Rate
[anon:dalvik-free list large object space]	3032	2050	67.61%
[anon:scudo:primary]	187	77	41.18%
[anon:scudo:primary]	167	54	32.34%
[anon:scudo:primary]	331	44	13.29%
[anon:scudo:primary]	300	41	13.67%
[anon:scudo:primary]	112	31	27.68%

*** input method app vma related**



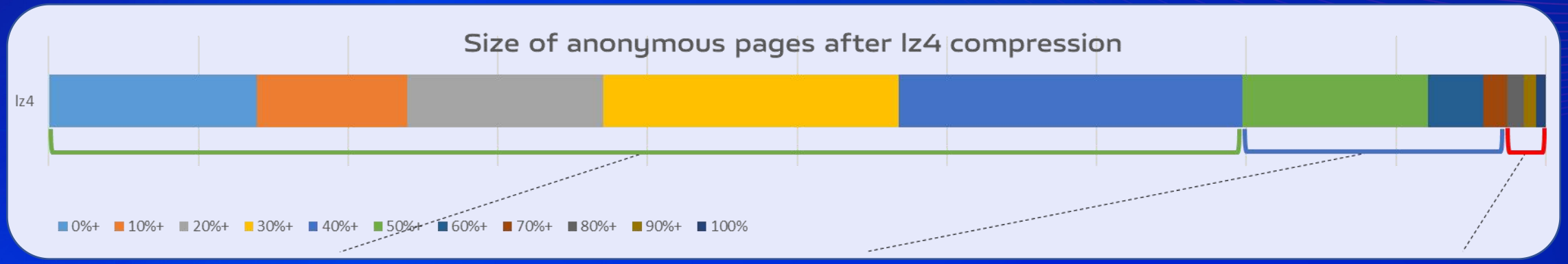
- **logd**存储压缩后数据的vma，ZHP占比很高
- 部分应用**LOS**区域的ZHP比例较高，这里存储的是size较大的java对象
- ZHP类型与应用的**业务行为**关系较大

3 ZHP优化探索

状态识别 + 场景关联 → 提升性能体验

- 定义匿名页的**压缩状态**
- **快速识别**出匿名页的 压缩状态
- 将压缩状态 与 **进程或vma关联**
- 识别的状态与 **用户场景关联**

3.1 ZHP优化探索 —— 分级压缩标识



Zram Small Page（ZSP）

压缩小页

压缩后size < 50%

有利于压缩 -- 前台及关键进程优先

Zram Medium Page（ZMP）

压缩中页

压缩后size为50%- huge_class_size之间

可以压缩 -- 后台进程优先

Zram Huge Page（ZHP）

压缩大页

压缩后size≥huge_class_size

不用压缩 -- 节省CPU资源

Suspected Zram Huge Page（SZHP）

疑似压缩大页

大概率（90%+）为ZHP

可以压缩 -- 后台进程优先

3.2 ZHP优化探索 —— 状态识别

关联识别

对高概率为ZHP的vma进行标记

识别及时 覆盖率低

主动识别

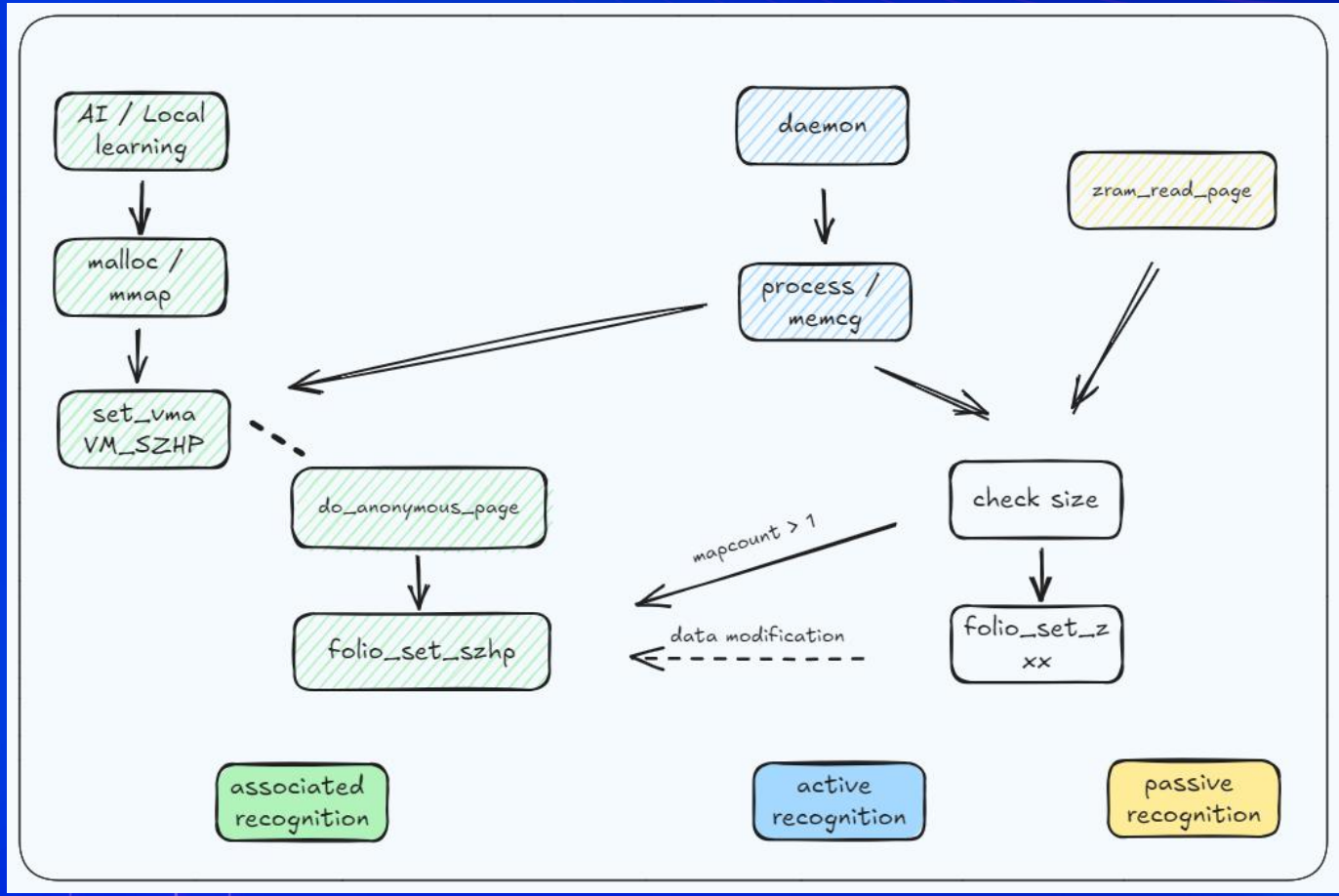
daemon线程主动识别压缩状态

覆盖率高 CPU开销

被动识别

利用原始解压流程识别压缩状态

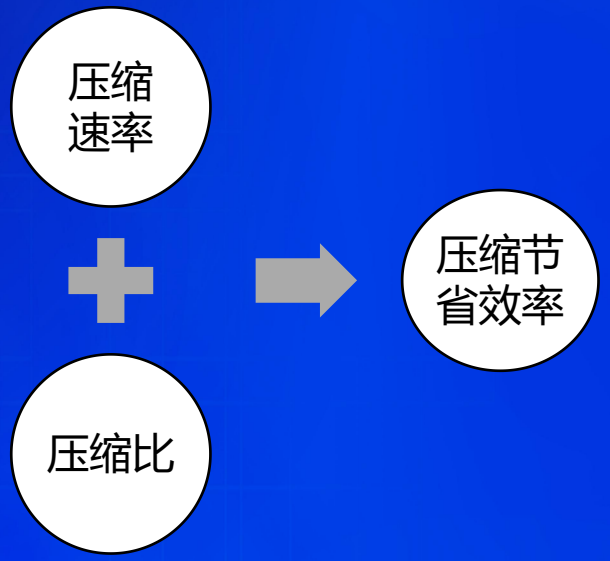
开销可控 覆盖率低



3.3 ZHP优化探索 —— 场景关联

- 针对**ZHP**，zram压缩时直接跳过压缩操作，执行后面的copy操作；
- 针对**ZSP**，由前台和关键线程优先压缩；
- 针对**ZMP、SZHP**，优先由后台线程压缩。

3.4 ZHP优化探索 —— 优化效果



内存压缩节省效率 （Saved Size Speed）

Saved Size = PAGE_SIZE – Compressed Size

优化项	优化率
内存压缩节省效率	fg 30%+ bg 8%+
应用启动	5% - 15%

谢谢！