OpenAnolis
龙 蜥 社 区

EROFS over FSCACHE: 内核原生的镜像加速方案

徐静波 <jefflexu@linux.alibaba.com>　阿里云操作系统开发工程师

OpenAnolis
龙 蜥 社 区

01 Introduction

为什么需要容器镜像加速？

02 EROFS over FSCACHE

Introduction to EROFS

EROFS-based RAFSv6 image format

FSCACHE-based lazy pulling

Test & Performance

03 New Features

failover

share domain

daemonless

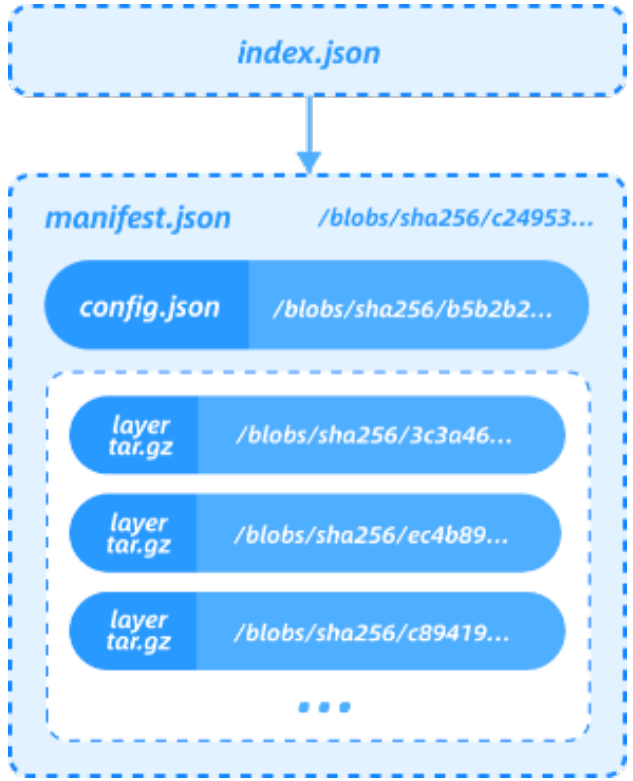
为什么需要容器镜像加速？

"*pulling packages accounts for 76% of container start time, but only 6.4% of that data is read*" [1]

镜像的按需加载 (lazy pulling)

OCI 镜像格式

- 一个容器由多层 (layer) 组成，层与层之间的去重 (layer deduplication)
- 每一层是 file archival 格式，e.g. tar + gzip
- 遍历整个 tar+gzip stream 以构建 directory tree，non-seekable
- 访问其中一个文件，必须下载整个 tar+gzip archival
- 不支持 lazy pulling



[1] Slacker: Fast Distribution with Lazy Docker Containers [fast16]

业界按需加载方案

		Leadership	backend(s)	Deduplication	OCI image compatiblity	Security / Rootless	Blob update
File-based format	CRFS / estargz	Google, NTT	FUSE	Layer	✅	✅	✅ (easy to add/update files for each blob)
	SOCI	AWS	FUSE	Layer	✅	✅	
	Nydus	Alibaba Cloud, Ant Group, Bytedance	EROFS or FUSE	Chunk	✅ (via estargz) 🚧 (reusing OCI blob, by recoding deflate sliding window like SOCI)	✅ (FUSE) 🚧 (EROFS)	
Block-based format (like qcow2)	DADI	Alibaba Cloud	Tcmu + random local filesystem	Layer	🚫 (need to generate a filesystem and convert to its block-based format anyway)	🚫 (not self-contained; can be attacked by random crafted fs)	🚫 (impossible to update an immediate layer and combine images)

按需加载方案：用户态 vs 内核态

用户态方案 (FUSE/TCMU)

- 在用户态定义/解析镜像格式
- 文件访问过程，(check cache hit/miss) 必须由内核态切换到用户态 (even in cache hit)
- 容器镜像已经下载到本地 (不需要按需加载)，频繁的内核态/用户态切换开销
- 性能相比 OCI 差

内核态方案：EROFS over FSCACHE

- (in-kernel EROFS-based) RAFSv6 (Registry Acceleration File System) image format (v5.15, v5.16)
- (in-kernel) FSCACHE-based lazy pulling (v5.19)

- 在内核态 check cache hit/miss
- cache hit 的时候不需要再由内核态切换到用户态
- 与 OCI 相近的性能表现

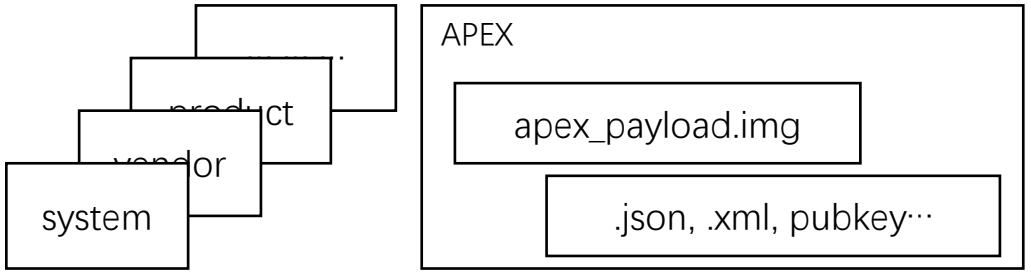
EROFS (Enhanced Read-Only Filesystem)

OpenAnolis 龙蜥社区

- 2017 开始开发，v4.19 进入 staging，v5.4 正式合入主线

- Page-sized (4KiB) block-based filesystem
- tail-packing inline (non-compressed) (space-saving + better performance)
- DAX support (v5.15)
- support for RAFSv6 image format (container scenario)
  - chunk-based data layout and deduplication (v5.15)
  - multi device (blob) support (v5.16)
- fscache support (v5.19)
- packed_inode & fragments (compressed) (v6.1)
- variable-length deduplication with rolling hash (compressed) (v6.1)

- 高性能的只读文件系统场景
  - Android 系统分区[1] & APEX[2]
  - 容器镜像 (Nydus[3] 镜像分发服务)

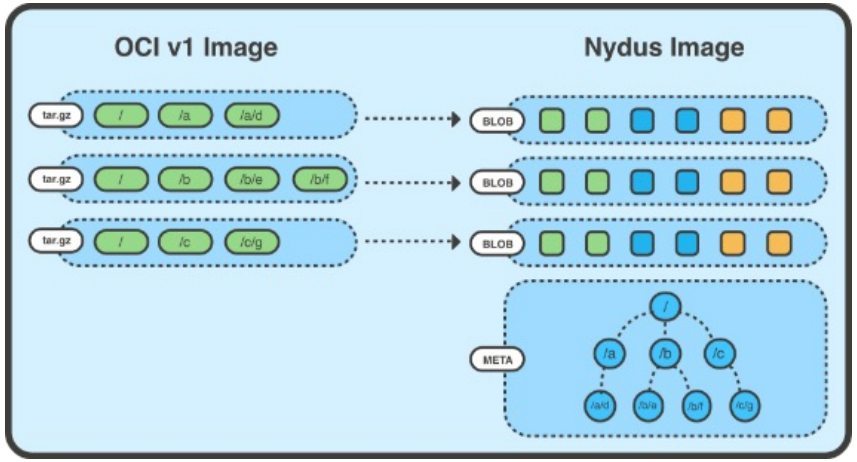


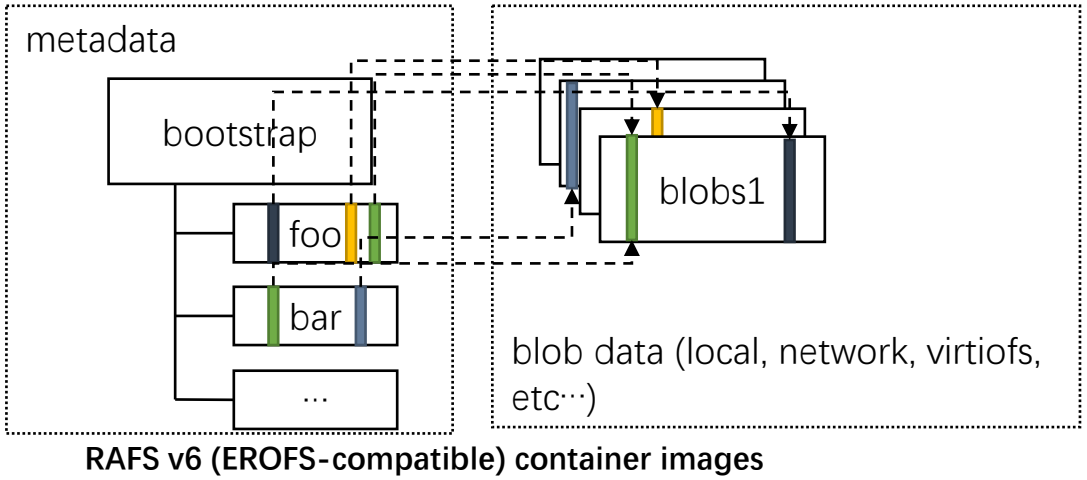
Android Smartphones

[1] https://source.android.com/docs/core/architecture/kernel/erofs
[2] https://source.android.com/devices/tech/ota/apex
[3] https://github.com/dragonflyoss/image-service

RAFSv6 image format

- 一个容器由一个 bootstrap (metadata) 和 多个 blob (data) 组成，每个 blob 代表一层
- 每个文件切分为多个 chunk，支持 (文件之间、层之间) chunk 粒度去重
- blob 内存储 chunk 数据





RAFS v6 (EROFS-compatible) container images

FSCACHE-based lazy pulling

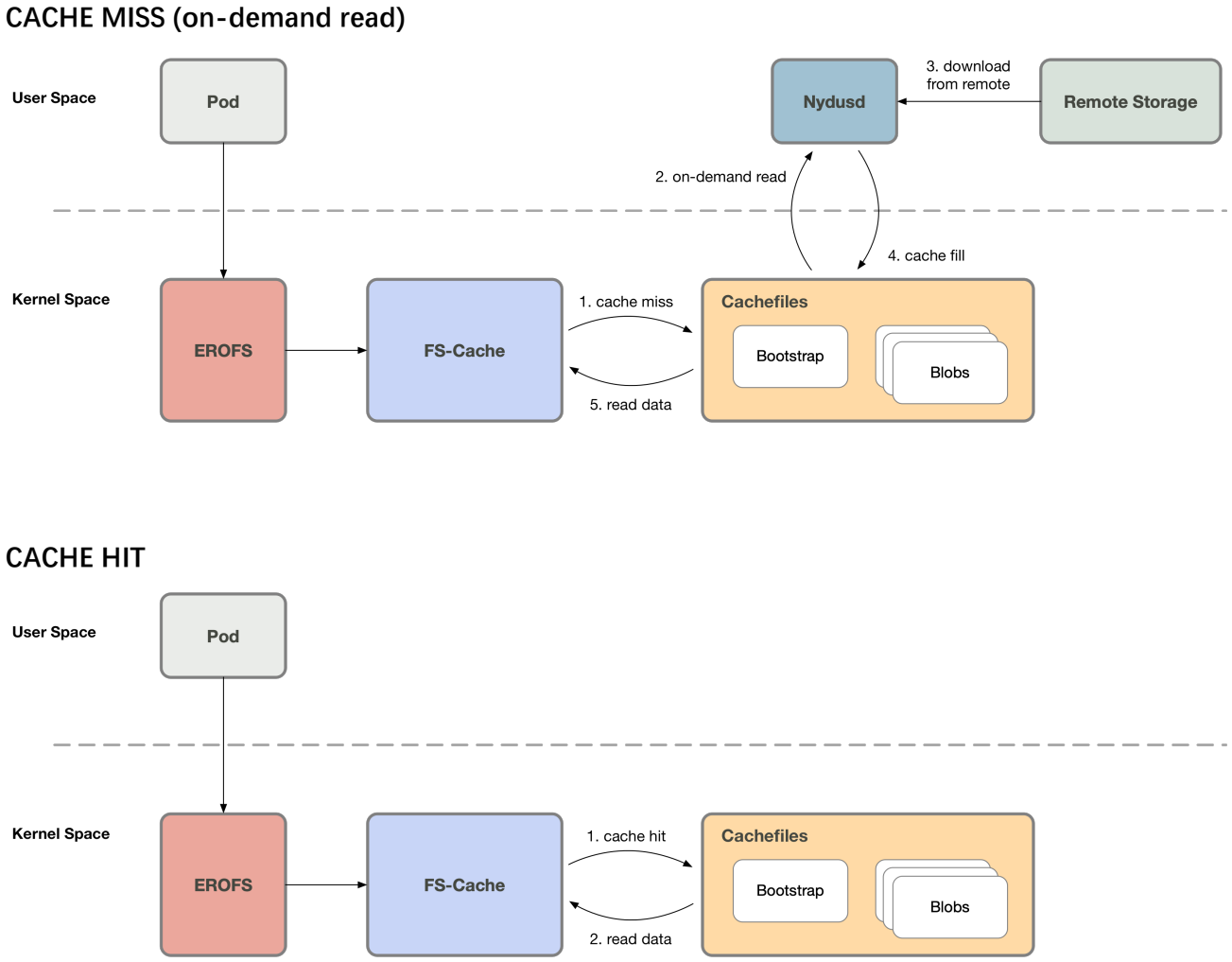
- fscache 文件缓存方案，应用于网络文件系统，e.g. NFS
- fscache 实现 (内核态的) 容器镜像缓存管理，e.g. check cache hit/miss

cache miss

- 进程: 通知用户态 daemon 处理按需加载的请求，睡眠等待
- daemon: 处理按需加载请求，拉取数据，写入 bootstrap/blob 文件，通知进程
- 进程: 从 bootstrap/blob 文件读取数据

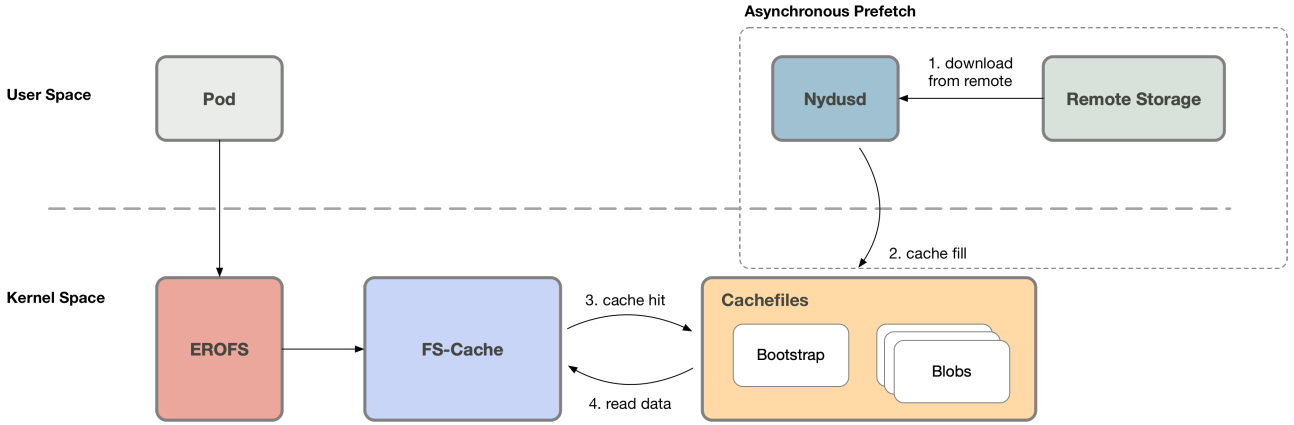
cache hit

- 内核态内 fscache 判断 cache hit，直接从 bootstrap/blob 文件读取数据
- 全程处于内核态



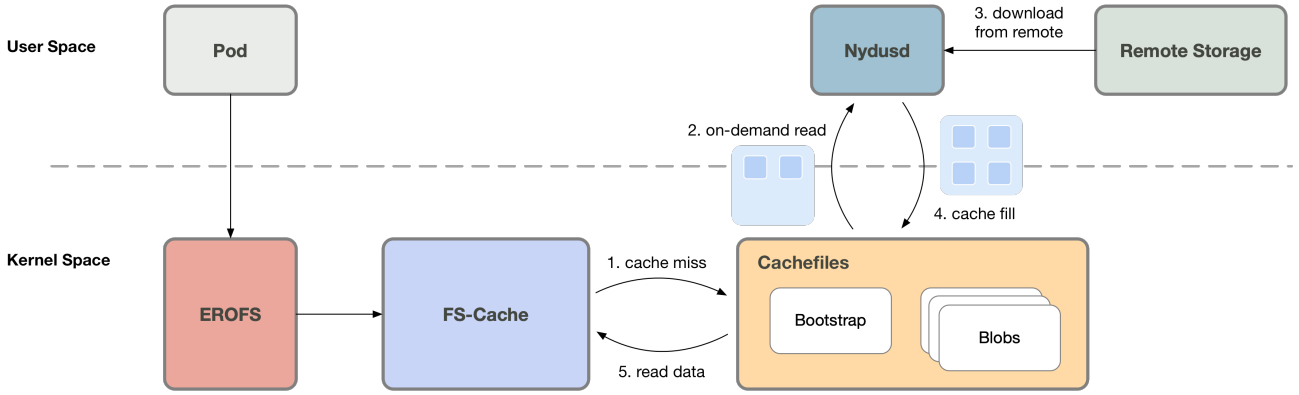
异步预取

- 当尚未触发 cache miss 的时候，用户态 daemon 就可以开始下载数据
- 之后当访问预取范围内的数据时，直接从缓存文件读取数据，而不会再切换到用户态

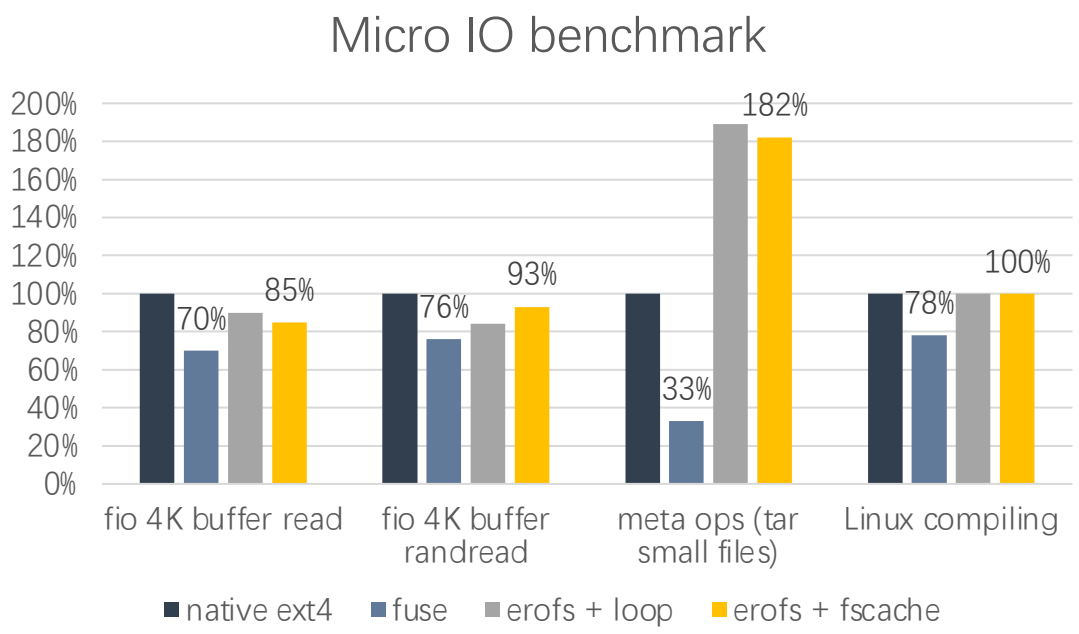


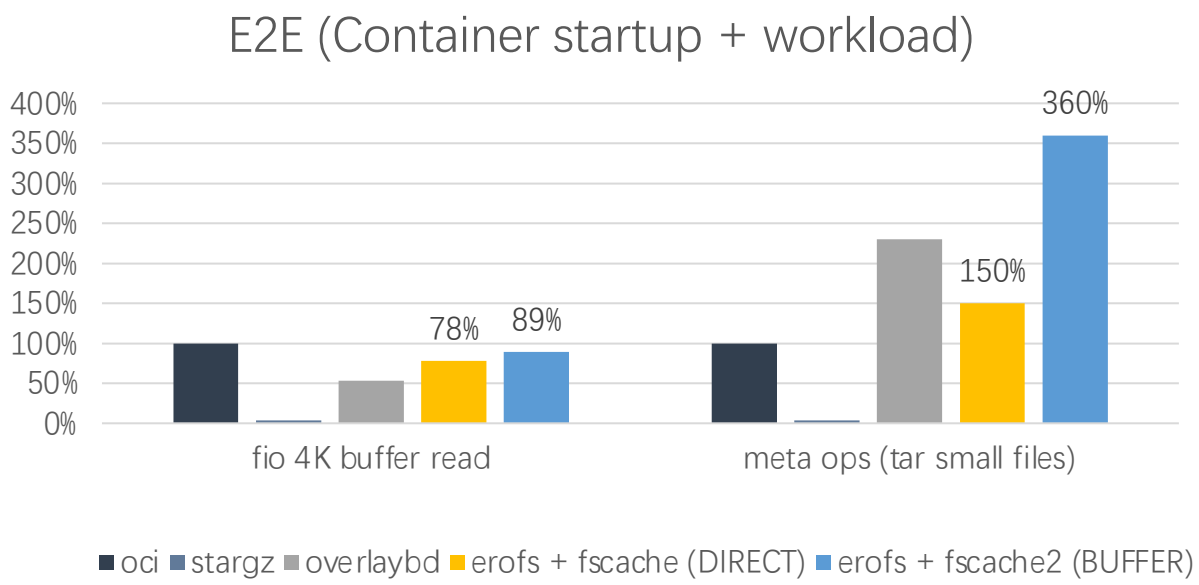
daemon 预读

- 当触发 cache miss 时，用户态 daemon 可以一次性下载比当前实际请求的数据量更多的数据 (cache miss 4K，下载 1MB)
- 之后当访问预取范围内的数据时，直接从缓存文件读取数据，而不会再切换到用户态



PERFORMANCE





OpenAnolis
龙 蜥 社 区

龙蜥操作系统 (Anolis OS) 内核 Cloud Kernel 4.19/5.10

https://gitee.com/anolis/cloud-kernel

龙蜥 OpenAnolis 社区交流 2 群
34 人

扫一扫群二维码，立刻加入该群。

OpenAnolis 高性能存储 SIG
131 人

扫一扫群二维码，立刻加入该群。

Nydus Image Service
147 人

扫一扫群二维码，立刻加入该群。