



DMA-buf cgroup accounting for proxy allocators

Attributing buffers to the right consumer

Albert Esteve

Red Hat

T.J. Mercier

Google

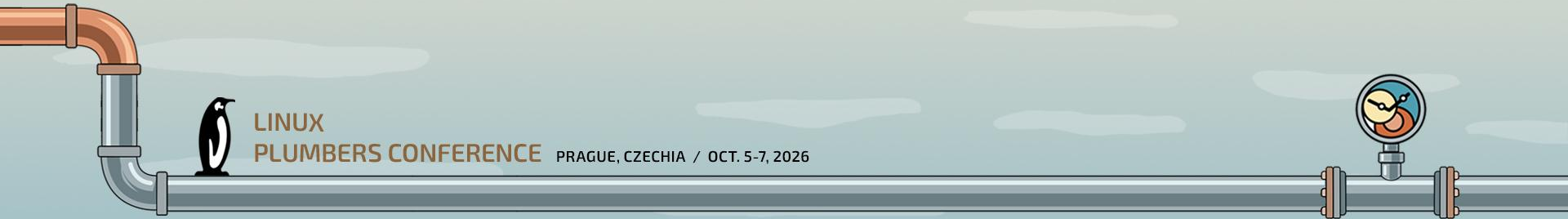


LINUX
PLUMBERS
CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

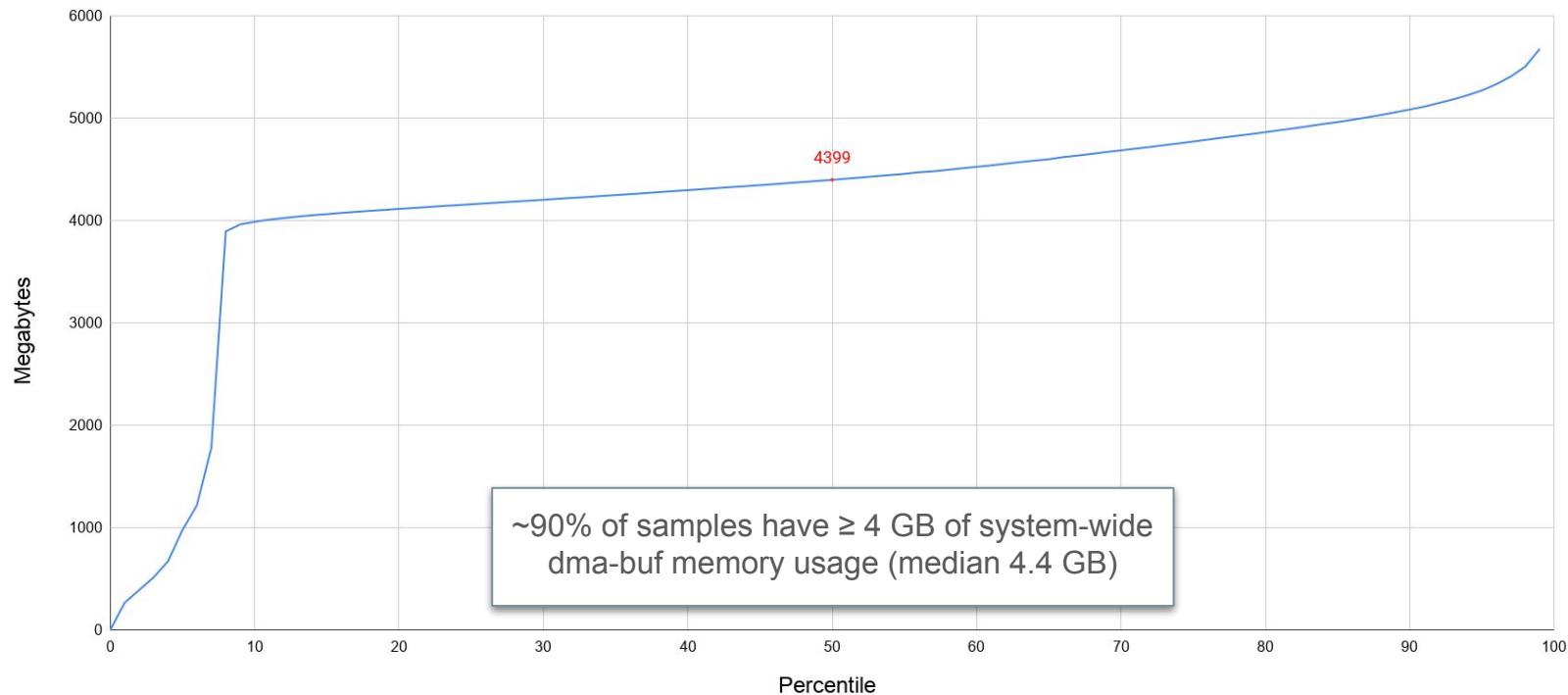
Outline

- Motivation
- Problem statement
- Previous work
- Talking points



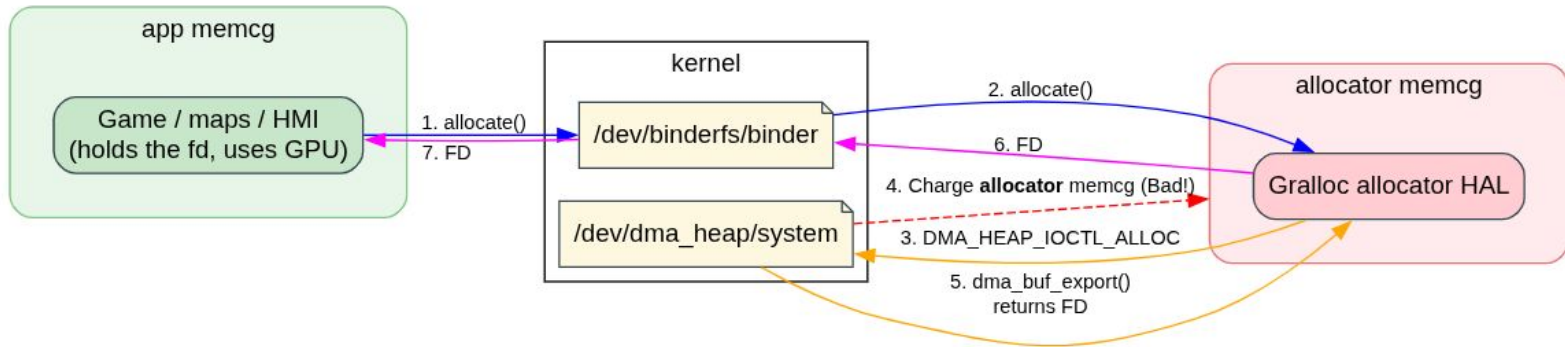
Motivation

System-wide DMA-BUF memory use on Android



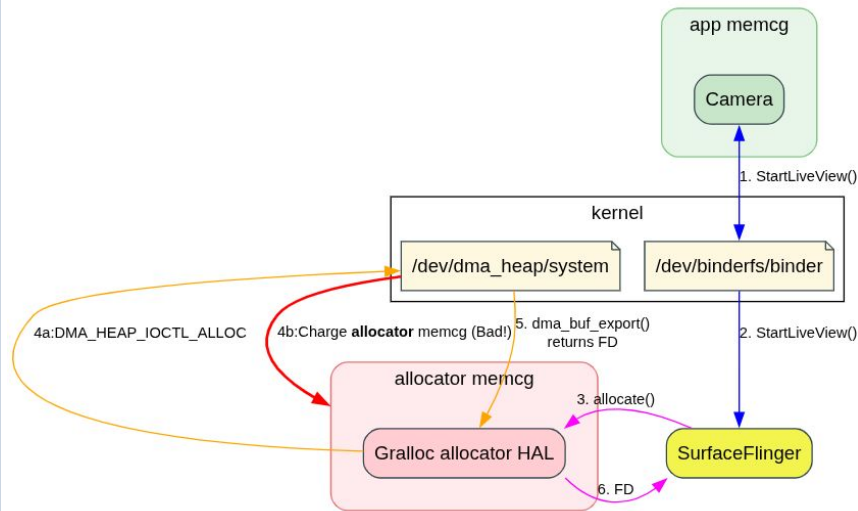
The problem - Gralloc

- App calls `allocate()` over Binder; it does not talk to dma-heap
- Gralloc HAL issues `DMA_HEAP_IOCTL_ALLOC`
- **Charge hits the `allocator memcg` (current), not the app**
- Same dma-buf FD goes from HAL to Binder, and then sent to the app
- **App `memory.max` misses the buffers**



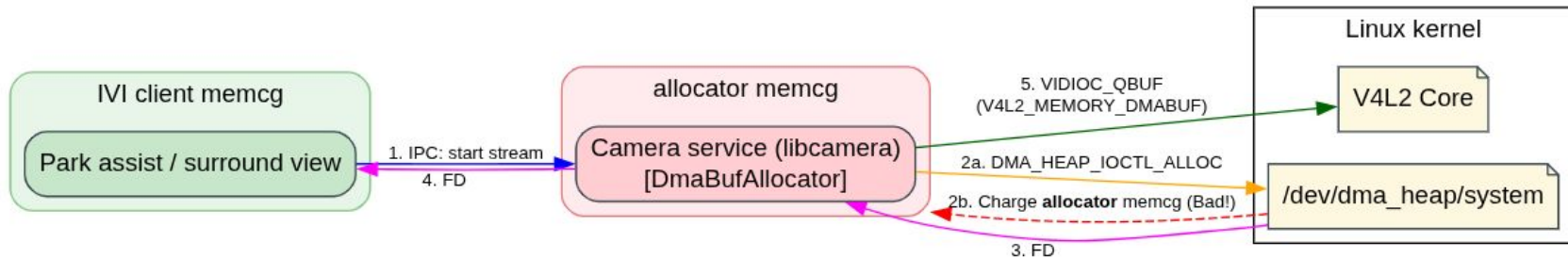
The problem - Gralloc (multi hop)

- Camera app calls `StartLiveView()`
 - Never talks to dma-heap or gralloc
- SurfaceFlinger handles the request and calls allocate on the gralloc HAL
- Gralloc issues `DMA_HEAP_IOCTL_ALLOC`
- **Charge hits the allocator memcg** (current), not the app nor SurfaceFlinger
- Same dma-buf FD returns to HAL, and then back to SurfaceFlinger
- **App memory.max misses the buffers**



The problem - libcamera

- IVI client request a camera stream (PipeWire)
- Camera service (DmaBufAllocator) issues `DMA_HEAP_IOCTL_ALLOC`
- **Charge hits the allocator memcg (current)**, the FD returns to the service
- FD is shared to the client once, then buffers are queued (`VIDIOC_QBUF`)
- **IVI client memory.max misses the buffers**



Previous work

[PATCHv2 0/4] Track exported dma-buffers with memcg

23 Jan 2023 by T.J. Mercier

Proposal

- `memory.stat dmabuf + try_charge` at `dma_buf_export()` (uncharge on free)
- Charge sticks to **one** cgroup for the life of the buffer
- `dma_buf_transfer_charge()` moves it to another cgroup
- HAL opts in with Binder `XFER_CHARGE`; SELinux via `binder_transfer_file`

In-thread discussed points

- When to charge
 - Reviewers: export size is not “memory used”
→charge at attach/map
 - However: real exporters allocate at export
 - If wait until map→unmapped / leaked FDs are never billed
- Transient hit on the allocator
 - HAL can OOM before Binder transfer
- Transfer is Binder-only
- Single owner (framework limitation)



LINUX
PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

Previous work

[\[PATCH RFC 0/5\] memcg: dma-buf per-cgroup accounting via pid_fd](#)

12 May 2026 by Albert Esteve

Proposal

- Keep TJ's `memory.stat` dmabuf counter
- Do **not** charge at `dma_buf_export()`
- Charge at `DMA_HEAP_IOCTL_ALLOC` (new parameter)
 - `charge_pid_fd` set → bill process's memcg
 - `charge_pid_fd` unset → default behavior

```
struct dma_heap_allocation_data alloc = {  
    .len          = buffer_size,  
    .fd_flags     = O_RDWR | O_CLOEXEC,  
    .charge_pid_fd = pidfd,  
};  
ioctl(heap_fd, DMA_HEAP_IOCTL_ALLOC, &alloc);
```

- Charge sticks to one cgroup for life of buffer, even if FD is shared
- Control achieved through LSM + SELinux

```
allow allocator_app_t client_app_t:dma_heap charge_to;
```

In-thread discussed points

- Android has 3 usecases: covers 2 trivially, 3rd (multi-hop) is doable
- Charging responsibility shifts to userspace + uAPI change
- Heaps-only not enough?
 - Move more drivers to heaps
 - Allow charge transfers
- **alloc vs transfer (+ pidfd vs cgroupfd)**: `pidfd` at `alloc` avoids transient bill, `transfer` is more general
 - `ioctl(buf_fd, DMA_BUF_IOCTL_XFER_CHARGE, pidfd/cgroupfd)` → “charge this FD to that cgroup”
- Single owner (framework limitation)

Talking points

1. How does kernel know which memcg to charge?

- Need to modify uAPI (extend `DMA_HEAP_IOCTL_ALLOC`)
- Handle: `pidfd` (perfect fit for Android) vs `cgroupfd` (more precise)
- Could be both?

```
if (!IS_ERR(pidfd_pid(fget(charge_fd)))) {...}
else if (!IS_ERR(cgroup_get_from_fd(charge_fd))) {...}
memfd_create2(name, flags, charge_fd);
```

2. Who has permission to charge a remote memcg?

- LSM? `Cap` (e.g., `CAP_SYS_ADMIN` or new one)?
- Hierarchy (`parent→child`, fails on `HAL→app` scenario)?

3. Charge at allocation or transfer?

- Allocation avoids transient bill

Transfer is more general:

- `ioctl(buf_fd, DMA_BUF_IOCTL_XFER_CHARGE, pidfd/cgroupfd)`
- Memcg charge transfer was deprecated and removed from memcg v2 (due to complex synchronization during charge movement)

4. What is chargeable?

- Userspace mmap of buffers is `VM_PFNMAP`: vma/page walker will not find a folio to move
- **Some heaps have no chargeable folio at all** (no-map carveouts are not buddy pages, so `__GFP_ACCOUNT/ folio->memcg_data` never happens)

```
aoc_reserve: aoc@94000000 {
    reg = <0x0 0x94000000 0x03000000>; /* 48 MB */
    no-map;
};
```
- Some schemes must charge something other than page (`dma_buf` size? `dmem?`): `sensor_direct_heap`, `aaudio_playback_heap`, `aaudio_capture_heap`, `audio_offload_heap`