

Sharing Space Between super and userdata Partitions on Android

Yangtao Li (frank.li@vivo.com)

Xirui Zhang, Chuang Pan, Liang Zheng



LINUX
PLUMBERS
CONFERENCE

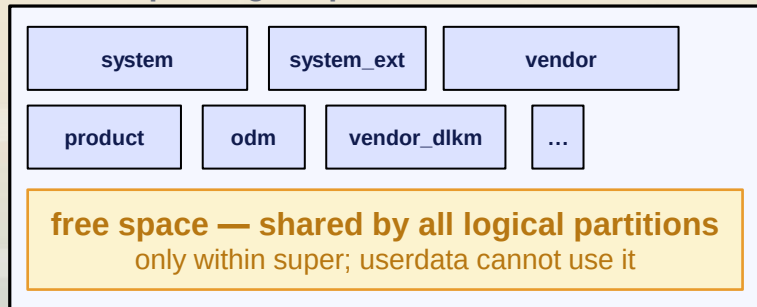
PRAGUE, CZECHIA / OCT. 5-7, 2026

Background: Android partitions current status

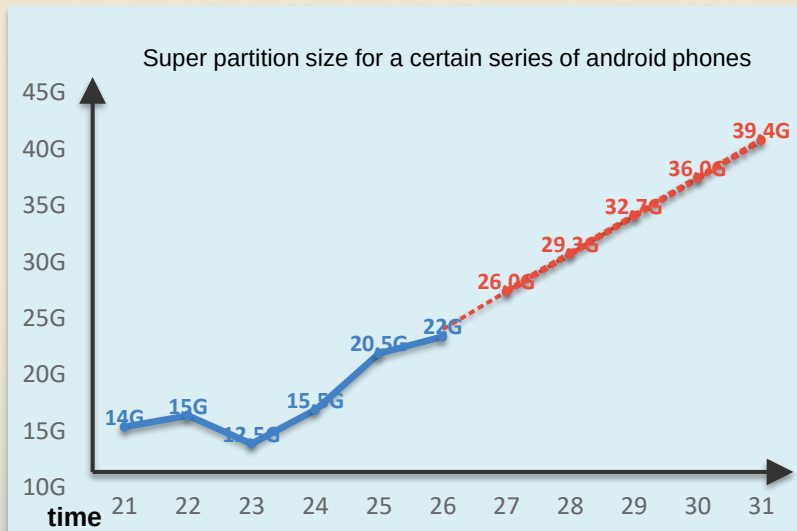
Typical device (UFS)



Inside super: logical partitions share one



Fixed GPT layout decided at the factory.



Super has experienced rapid growth in the past and will continue to do so in the future.



LINUX
PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



Problem: Super size is a one-time guess

super is too small

- System images keep growing: AI models, camera features, vendor modules, security updates
- No room for updated logical partitions → OTA limited
- Even with Virtual A/B (COW in userdata), the merge still needs room in super

super is too large

- Reserved space sits unused — forever
- Directly reduces the storage visible to users
- Cannot be given back to userdata after shipment (no safe GPT resize path)

Our fleet: Insufficient super headroom blocks continuous major OTA upgrades.

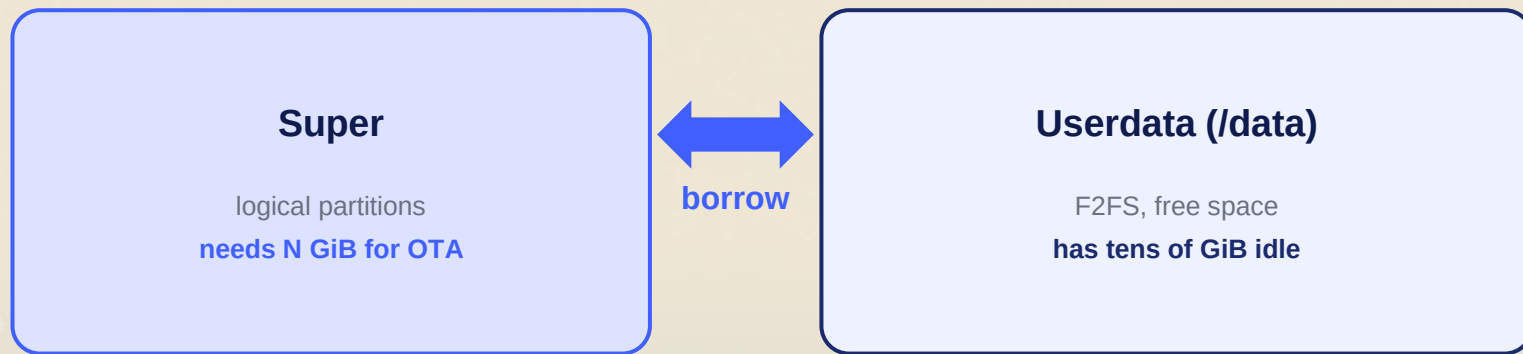


LINUX
PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026



Goal: Elastic Space Sharing between super and userdata



- **Compatible**

Maintain compatibility with legacy devices (wherever possible)

- **Online**

Grow (and shrink) super while the system is running

- **Efficient**

Fine-grained space allocation management

Without extra I/O overhead

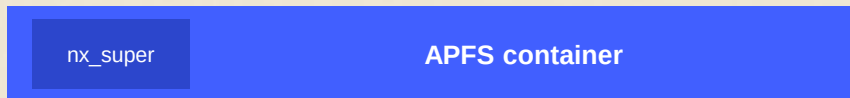


Solution: Trade-off Table

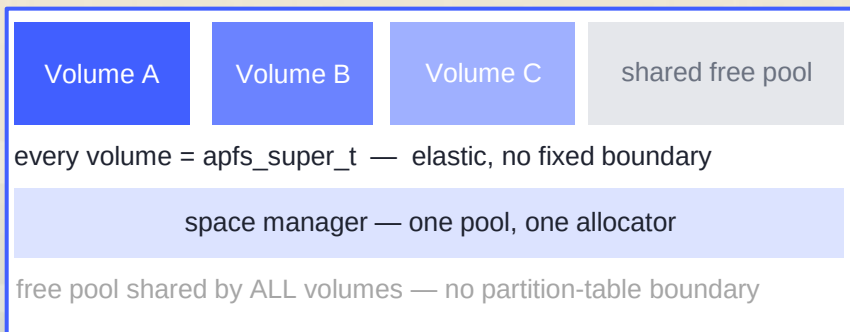
Approach	Idea	Compatible	Online	Efficient
Backing file	Create backing files on userdata partition as LpMetadataBlockDevices of the super partition	✓	✓	✓
One container	Move userdata into super, super is the only container (APFS-like)	✗	✓	✓
dm-thin	Create thin volumes from a unified thin pool	✗	✓	✗
Pinning device	control based on the F2FS device-aliasing-file mechanism	✗	✓	✗

Solution: One container

APFS — one container holds ALL volumes

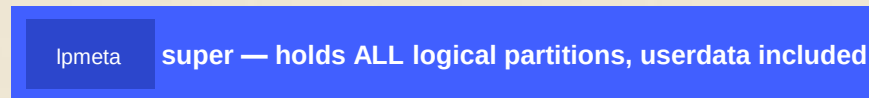


`nx_super` → `oid_t nx_fs_oid[NX_MAX_FILE_SYSTEMS]`

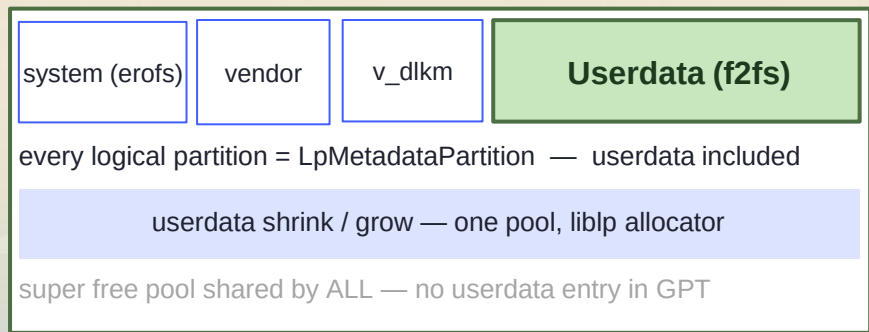


1 container · 1 metadata object
volumes discovered via `nx_fs_oid[]`

Android — one super holds ALL logical partitions



`lpmeta` → `struct LpMetadataPartition[]` (first_extent_index ~ num_extents)



1 super · liblp owns all space
resize = `lpmeta` + dm table (+ f2fs resize)

All in one: `nx_super` : volumes ≡ `lpmeta` : logical partitions.
One container, one metadata object, one shared free pool — userdata is just another member.



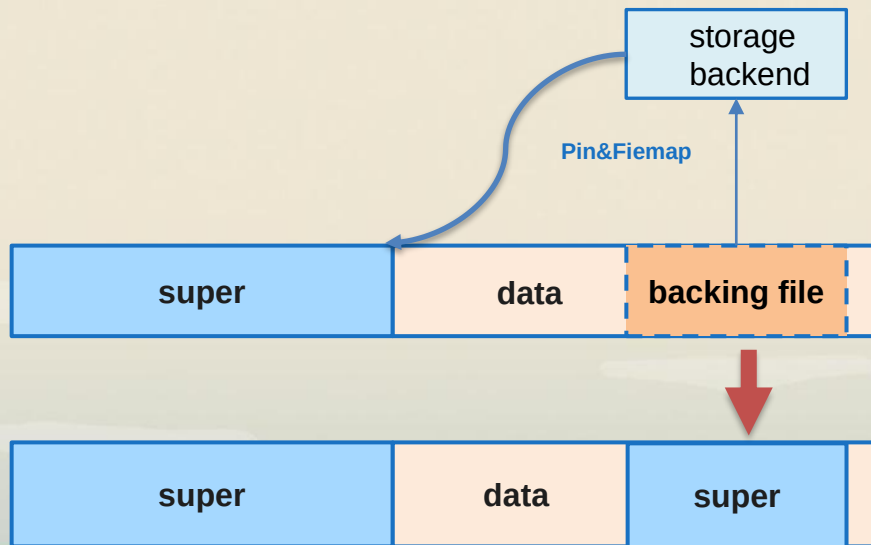
LINUX
PLUMBERS CONFERENCE

PRAGUE, CZECHIA / OCT. 5-7, 2026

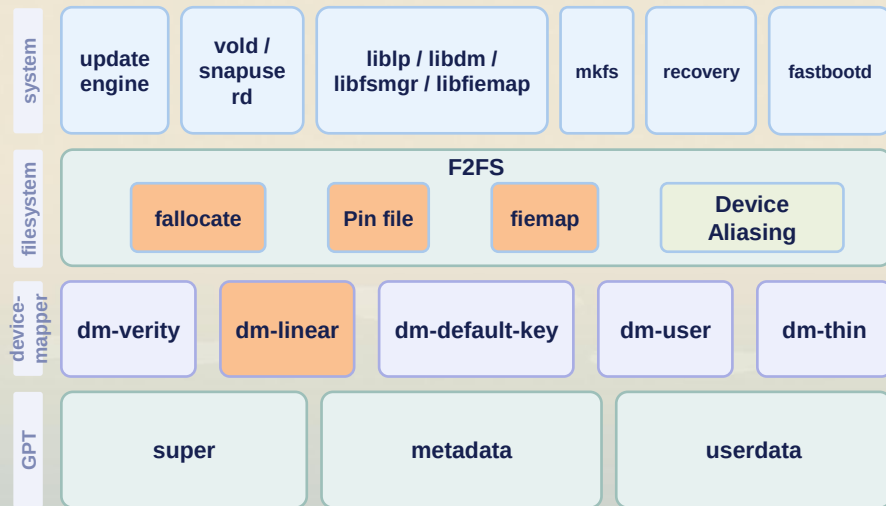


Solution: Backing file

Idea: a file on userdata lends its blocks to super



Stack: components on the backing-file path



LINUX
PLUMBERS CONFERENCE

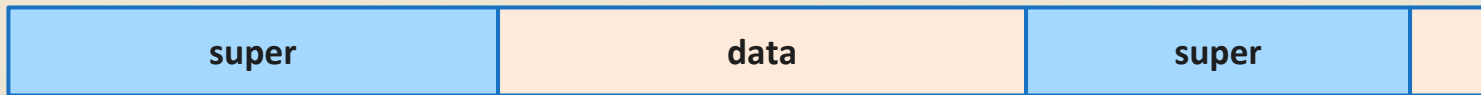
PRAGUE, CZECHIA / OCT. 5-7, 2026



How do we preserve the backing file when the data partition is formatted?

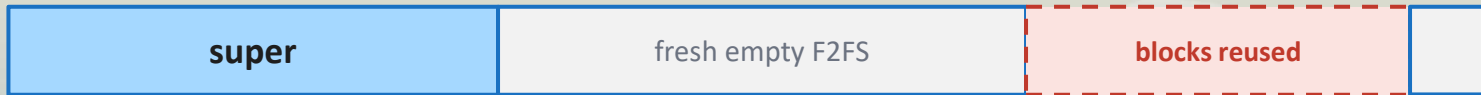
We must track the file's physical extents and protect them.
Dynamic, device-specific allocations with varying LBA addresses increase complexity.

① before format



mkfs.f2fs · factory reset

② after format



super's dm-linear table still points here → stale mapping, silent corruption
the file entry, the pin and the extent list are all gone

