Dynamic Partitions

Device-mapper and dm-linear

Problem...

Android runs out of space in read-only partitions.

boot	64M
system	2G
vendor	1G
product	512M
userdata	64G

Everything is overprovisioned

Some partitions run out of space after ~3 major releases

Problem...

If a partition has free space, we cannot share it.
Each partition is signed and verified and maybe owned by a different provider.

boot	64M
system	2G
vendor	1G
product	512M
userdata	64G

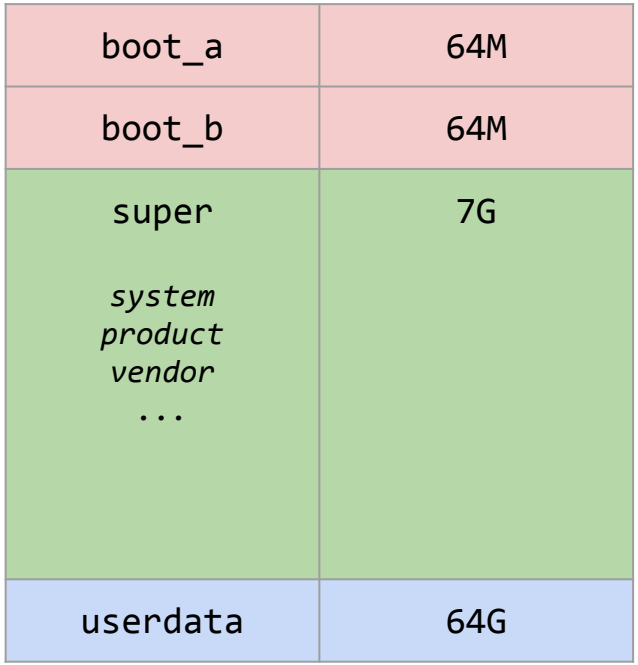
Why not rewrite the GPT?

GPT is **G**UID **P**artition **T**able, the fixed partition layout. Why not resize partitions and write a new GPT, like gparted?

- Android does not specify a partitioning system.
- Rewriting GPT is risky; we can't risk userdata becoming inaccessible.

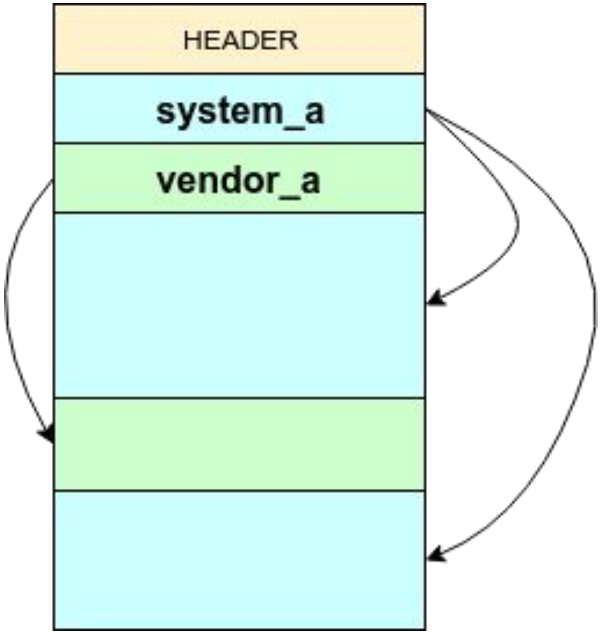
Solution: Device-Mapper

One big partition, allocate partitions in user-space with device-mapper.



Implementation

Super partition has metadata in its first sectors to describe partition names and extents.



DM_TABLE for system_a:

dm-linear <region1>
dm-linear <region3>

DM_TABLE for vendor_a:

dm-linear <region>

Changes to boot sequence

- Bootloader used to skip initramfs. We can't do that since the kernel doesn't understand our partitioning system.
- Ramdisk is stored in the boot partition; init reads super partition metadata.
- fstab mounting code knows how to find our partitions

Over-the-air updates

- Partition management handled by "liblp" library in userspace.
- Update code uses liblp to resize, delete, create partitions without needing to modify GPT.

Fastboot

fastboot protocol is used to flash devices from the bootloader.

But … bootloaders do not have device-mapper.

Now, developers will boot to userspace fastboot to flash dynamic partitions.

.. This is too slow for factory flashing, so we can also pre-generate an image of the super partition.

Device-Mapper flexibility

dm-verity continues to work, stacked on top of the "logical" partition.

We can retrofit older devices by allocating extents within existing partitions.

E.g. we don't need "super", we can span our dynamic partitions across system, vendor, product, etc, for devices where we can't modify GPT.

Performance

No measurable performance impact up to hundreds of extents.

I/O overhead measurable into thousands of extents.

Questions?

Why not LVM?

Problems specific to Android…

- Need to generate factory images
- Non-A/B updates need to survive power loss
- A/B updates cannot modify the source partitions
- Needed a quota mechanism for partition owners
- Leave door open for simple bootloader interaction