Revisiting How Kernels Invoke Initrds

Linux Plumbers Conference 2024, Vienna

Initrd, initramfs

- I say „initrd“, but I mean „initramfs“, which is actually more of an „inittmpfs“ these days

Status Quo

- Series of (typically compressed) cpio archives
- Loaded into memory pre-kernel
- Unpacked into tmpfs in kernel, early on
- First used by firmware loader, CPU microcode, others
- Finally /init executable is invoked in it

Status Quo 2

- Initrd then does its thing
- Eventually / is overmounted with final rootfs
- Final init process invoked
- Background: initrd tmpfs is emptied (rm -rf)
- Superblock remains loaded/mounted, just not visible

Problems with this

- In a UKI world, initrd might get *large*
- Unpacking/decompressing touches every byte: slow, and forces synchronous behaviour
- Emptying initrd when transitioning is slow, again
- pivot_root() not possible
- The whole thing is mutable, but everyone loves immutable these days

Ideas

- Maybe: stop doing cpio
- Instead do squashfs/erofs: pre-boot puts it in memory somewhere, kernel mounts it directly, done
- Problem with that: series of cpios would have to become series of erofs, overlayed. But: cpio has benefit of being something we can create „on-the-fly“ in systemd-stub and similar.

Ideas 2

- Maybe introduce „rootfs“ as proper superblock fs, that is entirely empty, and is always the root of the mount tree. Then cpio tmpfs/squashfs/erofs are mounted on top of that.
- pivot_root() would work, and we can get rid of original fs simply by unmounting it.

More Ideas?

- Anyone?