Dynamically Allocated Binder Devices

Linux Plumbers Conference, 2018
Vancouver, Canada

Christian Brauner
christian@brauner.io
christian.brauner@ubuntu.com
@brau_ner
https://brauner.github.io

Binder Devices & IPC Namespaces

- Goal is to use binder devices in different ipc namespaces (e.g. containers)
- Major obstacles:
  - Binder not available as kernel module
    Needs to be compiled into the kernel
  - No dynamic device allocation
    Currently number of binder devices determined at compile time
  - No per-ipc namespace devices
    All binder devices belong to the initial ipc namespace

Attaching IPC namespace on open()

- Pros:
  - Almost trivial to implement
  - Same binder device can be used in different ipc namespaces and open() creates ipc namespace context
- Cons:
  - Unclear semantics: what happens if a binder fd openen in ipc_ns_1 gets setns to ipc_ns_2 and is repopened through /proc/<pid>/fd/<binder-fd>?
  - This introduced an implicit and only partially functional binder device namespace
  - Other IPC mechanisms such as mqueue or shm implement this very differently
  - Requires changes to create_ipc_ns()
  - Doesn't allow to dynamically change the number of binder devices at runtime

/dev/binder-control

- Pros:
  - Follows proven /dev/loop-control design that Kay died years back
  - Dynamic allocation/deallocation of binder devices at runtime
- Cons:
  - More difficult to implement (but not too much I reckon)
  - (Should/Need to request dedicated major number from kernel?)
  - Doesn't solve the problem how to have per-ipc namespace binder devices

binderfs

- Pros:
  - Same features as the /dev/binder-control solution
  - Dynamic allocation/deallocation of binder devices at runtime
  - Allows for per-ipc namespace binder devices
  - Aligns with mqueue and /dev/shm implementations
- Cons:
  - More difficult to implement (but also not too difficult)
  - (Should/Need to request dedicated major number from kernel?)
  - Opt-in compile-time option

Dynamically Allocated Binder Devices

Linux Plumbers Conference, 2018
Vancouver, Canada

Christian Brauner
christian@brauner.io
christian.brauner@ubuntu.com
@brau_ner
https://brauner.github.io