Isolated Dynamic User Namespaces

Stéphane Graber (Canonical)
<stgraber@ubuntu.com>

Christian Brauner (Canonical)
<christian.brauner@ubuntu.com>

Isolated Id Mappings

- LXD has supported isolated id mappings since 2016.
  - Each container has its own, non-overlapping id mapping.
  - Limited to a full POSIX (65536) range by default.
  - Isolated id mappings only isolated within LXD instance not globally.
    - Another container runtime or user can trivially create overlapping mappings.

Isolated Id Mappings In Userspace

- Could isolated id mappings be coordinated in userspace? No.
  - No coordination method exists and is cumbersome to implement.
  - We tried to have that discussion.
  - Userspace contract would need to be adhered to by anyone using user namespaces → Basically impossible.
    - Most container runtimes ignore /etc/sub{g,u}id.
    - systemd advocates and ignores /etc/sub{g,u}id completely too.
- Size limitation of the ranges is becoming a problem.
  - Default size of 65536 isn't enough these days.
  - Network authentication commonly uses very high uid/gid in seemingly random ranges. As do a variety of services.
  - To be safe with most cases, we'd need a range of 10000000 limiting the total number of containers on the system to less than 500.

Kernel Enforced Id Mappings: Keeping track of mappings

- First approach was to introduce new sysctl or boot option to switch kernel into isolated id mapping mode.
  - Only allow allocation of contiguous maps (no holes or complex maps).
  - Track active mappings via IDRs and lookup maps by starting id.
    - Refuse if map is active and allow if map is not active.

Kernel Enforced Id Mappings: Keeping track of mappings

- Needs method to lookup free id mappings or random free map assigned at user namespace creation time.
- Would break old applications when running in that mode.
- Severely limits number of container that can be run.
- Seem hackish.

Kernel Enforced Id Mappings: Going 64bit

- Discussed and design between Eric, Stéphane, Aleksa, and I.
  - Switch id types _in the kernel_ to 64bit.
  - Lower 32bit continue to be used by userspace, upper 32bit used by the kernel.
  - Introduce new clone3(CLONE_NEWUSER_ISOLATED) generating a new kernel-side 32bit integer (upper 32bit of 64bit kuid_t).
    - Allow to specify owner uid/gid during clone3() and default to effective uid/gid.

Kernel Enforced Id Mappings: Going 64bit

- In this mode uid_map/gid_map are full range (unsigned 32bit)
  - Allows to support post-POSIX range users that allocate high-range uid/gid (LDAP, systemd, etc).
  - Full unsigned 32bit uid/gid range, compatible with every Linux workload.
  - No need for different container runtimes to collaborate on uid/gid ranges and benefits everyone.
  - Trivial nesting because of removed need to split existing range.
  - Simplified usage of user namespace for newcomers → Finally increase adoption.
  - Clear owner for a user namespace will make monitoring/interacting way easier.
  - 64bit uid/gid invisible from userspace.
    - Use owner uid/gid to give a credential to use when interacting with a different isolated namespace.