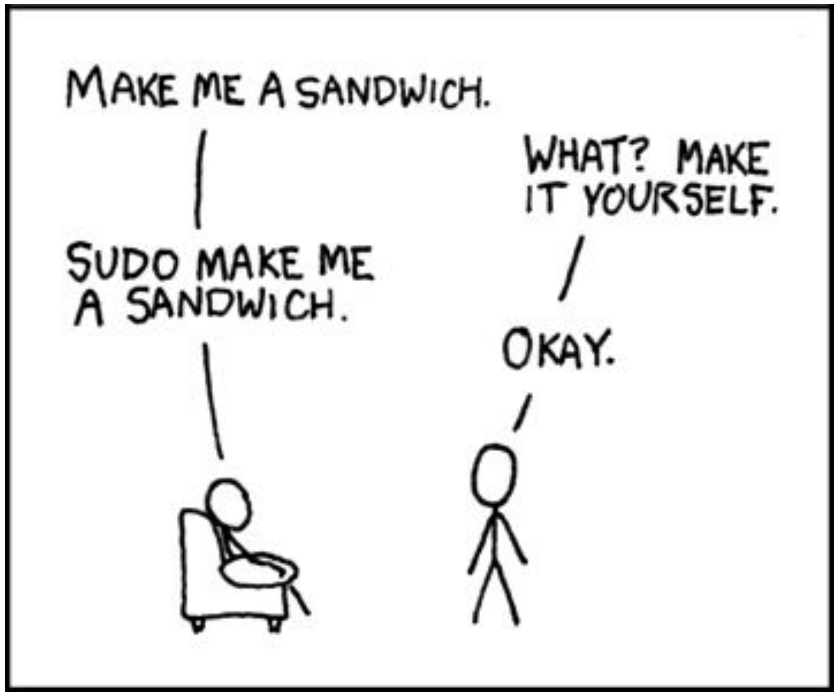
Current state of rootless dockerd

Akihiro Suda (@_AkihiroSuda_)
NTT Software Innovation Center

What is rootless dockerd?

- **Run Docker daemon (and also containers of course) as a non-root user**



- **Don't confuse with:**
  - `sudo`
  - `usermod -aG docker penguin`
  - `docker run --user`
  - `dockerd --userns-remap`

- **Experimentally supported since Docker v19.03**
  https://get.docker.com/rootless

Image: https://xkcd.com/149/

Why?

- **For Cloud-Native envs:**
  - To mitigate potential vulnerability of container runtimes and orchestrator

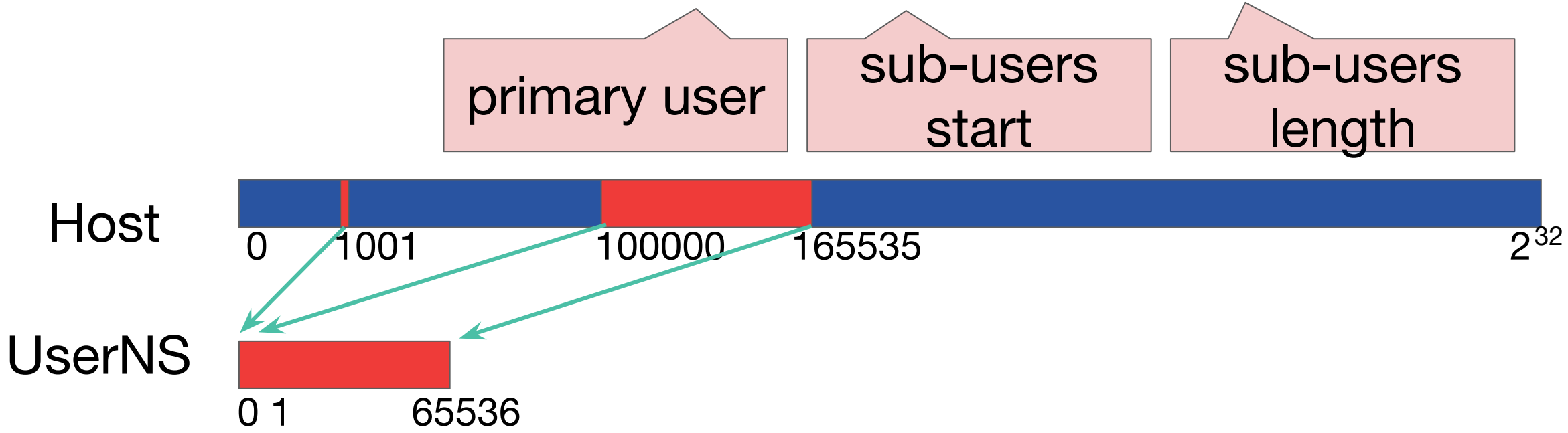
- **For HPC envs:**
  - To run containers without the risk of breaking other users environments

How it works: User Namespaces

- **User namespaces allow non-root users to pretend to be the root**
- **Root-in-UserNS can have “fake” UID 0 and also create other namespaces (MountNS, NetNS..)**
- **Unlike Singularity, NetNS can be unshared**
  - By using either usermode TCP/IP stack (VPNKit, slirp4netns) or SETUID binary (lxc-user-nic)

System requirements: `/etc/{subuid,subgid}`

- **If `/etc/subuid` contains "`1001:100000:65536`"**

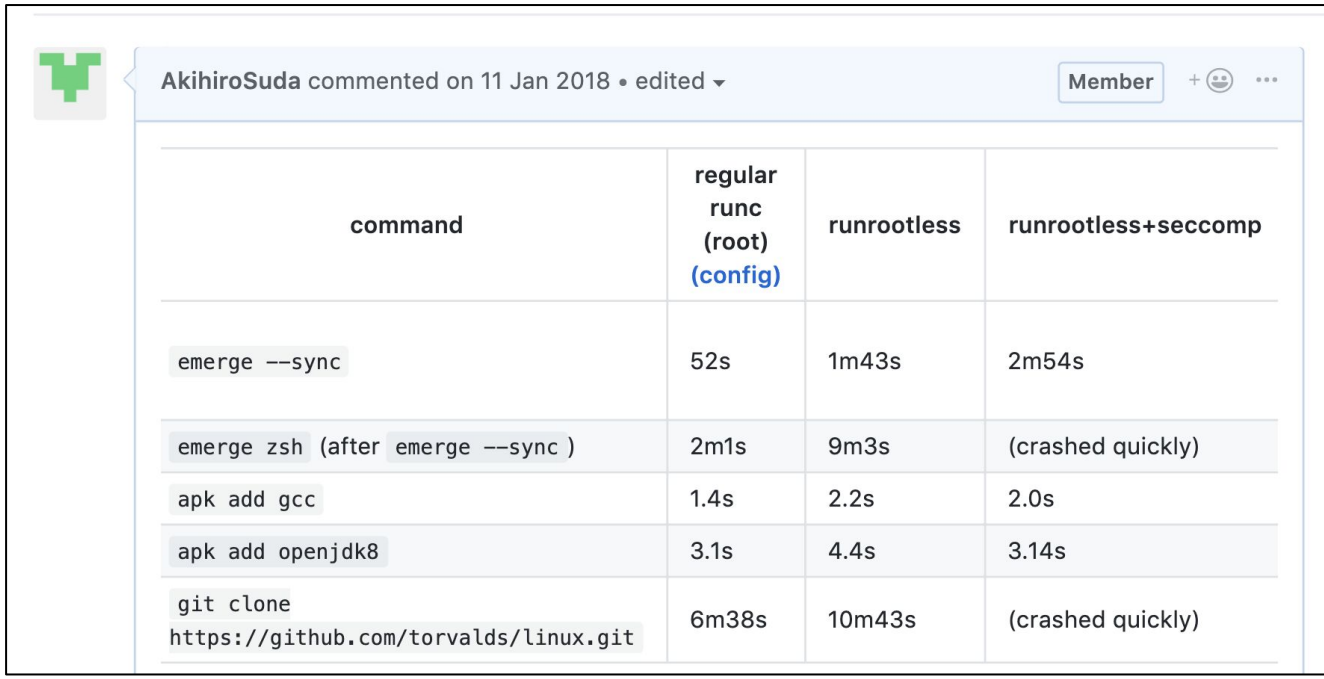


- **Having 65536 sub-users should be enough for most containers**

Unresolved issues (Contribution wanted!)

- **Hard to maintain `subuid` & `subgid` in LDAP/AD envs**
  - NSS module is being under discussion
    https://github.com/shadow-maint/shadow/issues/154
  - Single-mapping mode w/o `subuid` & `subgid` is also under discussion
    - uses ptrace and xattrs (slow!)
    - seccomp could be used for acceleration

https://github.com/rootless-containers/runrootless



AkihiroSuda commented on 11 Jan 2018 • edited — Member

command	regular runc (root) (config)	runrootless	runrootless+seccomp
`emerge --sync`	52s	1m43s	2m54s
`emerge zsh` (after `emerge --sync`)	2m1s	9m3s	(crashed quickly)
`apk add gcc`	1.4s	2.2s	2.0s
`apk add openjdk8`	3.1s	4.4s	3.14s
`git clone https://github.com/torvalds/linux.git`	6m38s	10m43s	(crashed quickly)

Unresolved issues (Contribution wanted!)

Innovative R&D by NTT

- **Lacks cgroup**
  - cgroup2 (unified-mode) supports unprivileged mode but migration may take a few years… or even more
  - For cgroup1, `pam_cgfs` could be used instead, but not available in Fedora / RHEL due to a security concern

- **Kernel / VM / HW may have vulns**
  - Not suitable for real multi-tenancy
  - gVisor might able to mitigate some of them

NTT