Rootless Docker in TSD

haatveit

University Center for Information Technology (USIT), University of Oslo

What?

Running containers without elevating privileges

Motivation

- Making containers more practical for
  - our **users**
  - us
- Improving security

Old TSD Docker installations

- `dockerd` still running as root
- Daemon *default* setting runs containers in user namespace
- User namespaces not being a standardized non-production ID range
- User control over the daemon restricted by complex sudo ruleset for `docker` CLI

Rootless Docker in TSD

- `dockerd` running as non-root user(s) – usually a local system account
  - Allowed usage of user namespaces in high UID/GID ranges, which don't overlap with what's used outside of containers
  - No project user ends up with wider privileges or access than they should

Rootless Docker in TSD

- Users control `dockerd` via authenticated TCP
  - mTLS for daemon access (preconfigured for convenience)
  - Non-restrictive usage of `docker` CLI
  - Tools like Compose just work™
- Granular access possible via client certificate access control

Non-root containers: challenges

- NFS doesn't understand user namespaces
  - This means local VM storage required for container store
- Namespaces have to be set up on a per-user basis
  - Hence the sharing of an unprivileged system user
- ESS (new storage solution) not yet tested

Future: podman

Podman is a container engine from Red Hat.

- Supported on RHEL
- Actively targeting the non-root use case
- As of podman 3.0.0 from Februrary 2021, now has Docker v1.40 API compatibility