NTT

Why was nerdctl made?

https://github.com/containerd/nerdctl

Akihiro Suda, NTT

nerdctl: contaiNERD CTL

NTT

- Docker-compatible CLI for contai**nerd** (and buildkitd)
  `alias docker=nerdctl`

- **Made for facilitating new experiments in the containerd platform**

- The experimental features can be safely turned off for production environments (`export NERDCTL_EXPERIMENTAL=0`)

- Useful for debugging Kubernetes nodes too

https://github.com/containerd/nerdctl

Demo

```
$ nerdctl run ...

$ nerdctl compose up ...
```

Novel features of containerd & nerdctl

NTT

- Lazy-pulling with eStargz/SOCI/Nydus/OverlayBD
- Registry-less P2P image distribution with IPFS *
- Image encryption with OCIcrypt *
- Image signing with Cosign / Notation *
- “Real” read-only mounts with mount_setattr
- Slirp-less rootless containers with bypass4netns *
- Interactive debugging of Dockerfiles, with buildg *

* = Experimental

Lazy-pulling with eStargz/SOCI/Nydus/OverlayBD

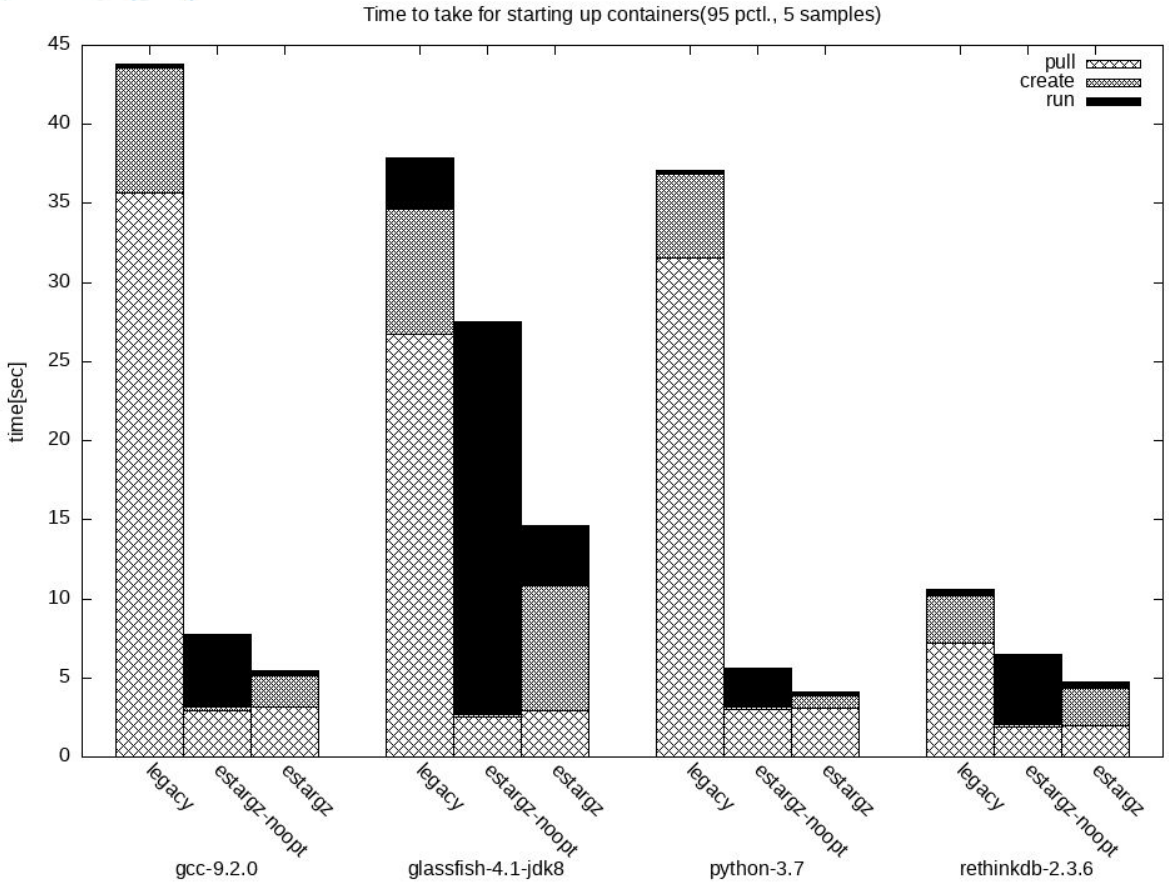
NTT

- Lazy-pulling: pulling image contents on demand
- No need to pull an entire image
- Several formats are being proposed

Format	Implementation for containerd	Description
eStargz	github.com/containerd/stargz-snapshotter	Optimizes gzip granularity for seek()-ability; Forward compatible with OCI v1 tar.gz
SOCI	github.com/awslabs/soci-snapshotter	Captures a checkpoint of tar.gz decoder state; Forward compatible with OCI v1 tar.gz
Nydus	github.com/containerd/nydus-snapshotter	An alternate image format; Not compatible with OCI v1 tar.gz
OverlayBD	github.com/containerd/overlaybd	Block devices as container images; Not compatible with OCI v1 tar.gz

https://github.com/containerd/nerdctl/blob/main/docs/stargz.md

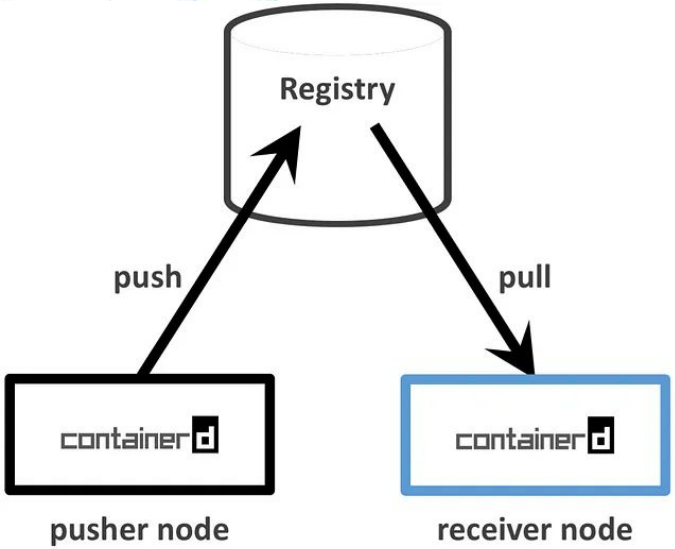
Lazy-pulling with eStargz/SOCI/Nydus/OverlayBD



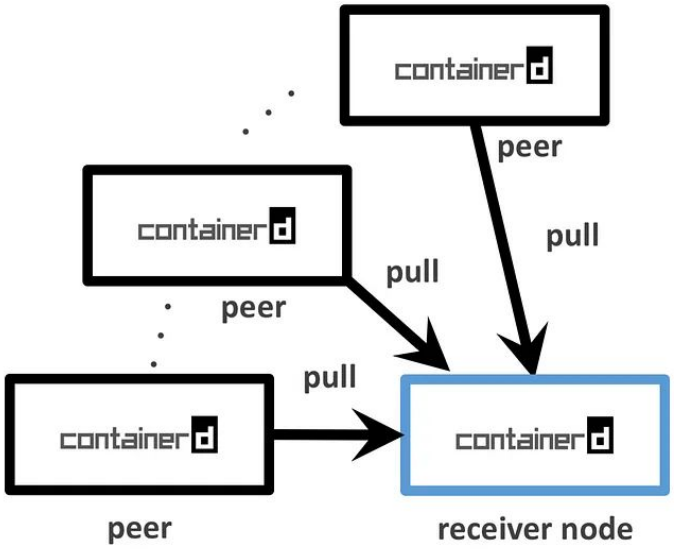
https://github.com/containerd/stargz-snapshotter/blob/v0.14.3/docs/images/benchmarking-result-ecdb227.png

Registry-less P2P image distribution with IPFS

```
$ nerdctl pull ipfs://<CID>
```



Registry-based image distribution



IPFS-based image distribution

https://medium.com/nttlabs/nerdctl-ipfs-975569520e3d
https://github.com/containerd/nerdctl/blob/main/docs/ipfs.md

Experimental

Image encryption with OCIcrypt

NTT

- OCI Image Layers can be encrypted to protect secret files

  (OCI Image Config, e.g., env vars, are not encrypted)

```
$ nerdctl image encrypt --recipient=jwe:mypubkey.pem foo example.com/foo:encrypted
$ nerdctl push example.com/foo:encrypted
```

- Encrypted layers can be decrypted transparently

```
$ nerdctl pull example.com/foo:encrypted
```

https://github.com/containerd/nerdctl/blob/main/docs/ocicrypt.md

Experimental

Image signing with Cosign / Notation

- Docker CLI supports `DOCKER_CONTENT_TRUST` (Notary v1), but it didn't see wide adoption

- `DOCKER_CONTENT_TRUST` is not implemented for nerdctl

- Instead, nerdctl supports Cosign and Notation ("Notary v2")

```
$ nerdctl pull --verify=(cosign|notation)
```

https://github.com/containerd/nerdctl/blob/main/docs/cosign.md
https://github.com/containerd/nerdctl/blob/main/docs/notation.md

“Real” read-only mounts with mount_setattr

NTT

- Surprisingly `docker run -v /mnt:/mnt:ro` doesn’t really make `/mnt` inside the container read-only
  - Submounts like `/mnt/usbdisk` are writable

- nerdctl supports RRO: recursively-read-only mounts

  `nerdctl run -v /mnt:/mnt:rro`

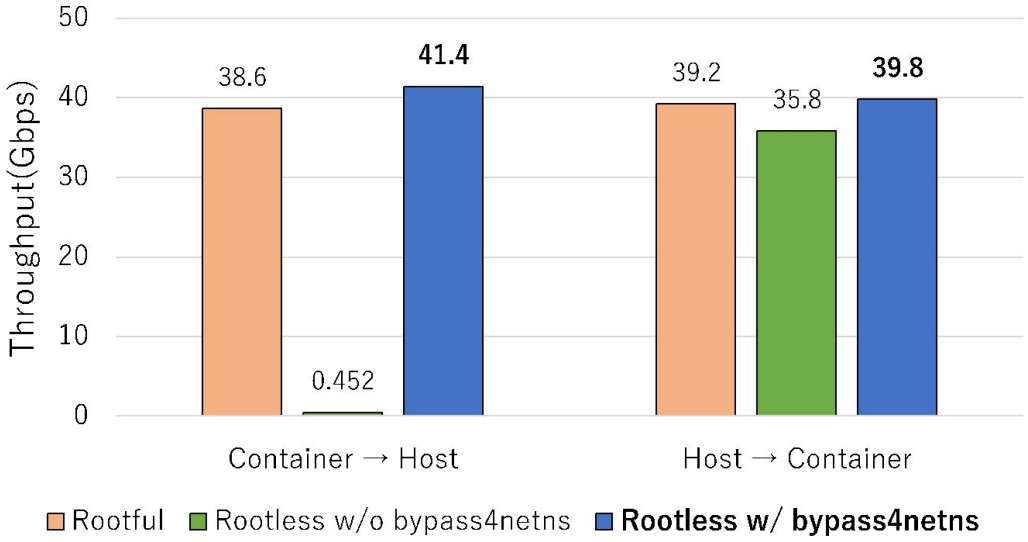
  – Implemented using `MOUNT_ATTR_RDONLY` (kernel >= 5.12)

Slirp-less rootless containers with bypass4netns

NTT

```
$ nerdctl run --label=nerdctl/bypass4netns=1
```

- Bypass slirp4netns (usermode TCP/IP), by using `SECCOMP_IOCTL_NOTIF_ADDFD`
- Even faster than rootful



https://github.com/containerd/nerdctl/blob/main/docs/rootless.md#bypass4netns
https://speakerdeck.com/mt2naoki/ip-communications-in-rootless-containers-by-socket-switching?slide=4

Experimental

Interactive debugging of Dockerfiles, with buildg

NTT

- Breakpoints for Dockerfiles

```
$ nerdctl builder debug .
....
 =>   1| FROM ubuntu AS dev
      2| RUN echo hello > /hello
      3| RUN echo world > /world
      4|
(buildg) break 3
(buildg) continue
...
Breakpoint[0]: reached line: Dockerfile:3
(buildg) exec /bin/sh
#
```

https://github.com/containerd/nerdctl/blob/main/docs/builder-debug.md
https://github.com/ktock/buildg

Other features

NTT

- Namespacing: `nerdctl --namespace=<NS> ps`
- Connecting a container to multiple networks at once:

  `nerdctl run --net foo --net bar ...`
- AppArmor for rootless containers

  (Although still needs `sudo nerdctl apparmor load`)
- FreeBSD containers
- Windows containers
- And more...

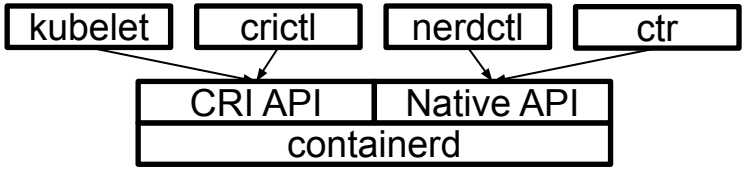
FAQ: Why not ctr?

NTT

- ctr: a CLI included in containerd
- ctr is often misbelieved to be the standard CLI for containerd
- But ctr is just a debugging tool written for containerd developers
- ctr lacks lots of high-level features that are present in Docker and nerdctl
  - Exposing ports (`nerdctl run -p <PORT>`)
  - Re-startable containers w/ networking (`nerdctl run --restart=always`)
  - Support for `~/.docker/config.json` , `docker-credential-ecr-login`, etc.
  - Reading logs of background containers (`nerdctl logs`)

FAQ: Why not crictl?

- crictl is similar to ctr but uses a different API
- crictl has similar restrictions as ctr



- CRI API has very tight scope compared to the containerd native API
- Negotiation for modifying the CRI API is relatively hard

FAQ: Why not Docker?

NTT

- Docker doesn't/didn't use the containerd API for image management
  - Can't/Couldn't be used for experimenting lazy-pulling, etc.
  - Work is in progress toward Docker v24+
- Release cycle is/was slow
  - `Docker v19.03: Jul 2019`
  - `Docker v20.10: Dec 2020`
  - `Docker v23:    Feb 2023`
- Not trying to "defeat" Docker; Experiments in nerdctl are being ported to Docker too

FAQ: Why not Podman/CRI-O ?

NTT

- Podman doesn't work as a Kubernetes runtime

- CRI-O doesn't support non-CRI API, and lacks Docker-like CLI
  - `podman ps` , etc. works for CRI-O too, but still Podman != CRI-O

- Not trying to "defeat" Podman/CRI-O either;

  Experiments in containerd/nerdctl are being ported to Podman/CRI-O too
  - e.g., eStargz

Getting started

- https://github.com/containerd/nerdctl/releases
- `nerdctl-full-<VERSION>-linux-amd64.tar.gz` contains all the dependencies (containerd, runc, ...)

```
$ sudo systemctl enable --now containerd
$ sudo nerdctl run -d --name nginx -p 80:80 nginx:alpine
```

```
$ containerd-rootless-setuptool.sh install
$ nerdctl run -d --name nginx -p 8080:80 nginx:alpine
```

Getting started

- macOS users should try Lima
  - Lima: Linux Machine, originally designed as “nerdctl Machine”
  - CNCF Sandbox Project
  - Automatic port forwarding & host filesystem mounting

```
$ brew install lima
$ alias nerdctl=nerdctl.lima
$ nerdctl run -d --name nginx -p 80:80 nginx:alpine
```

- nerdctl is also included in AWS Finch, Colima, Rancher Desktop, ...

Finch

Colima

RANCHER DESKTOP

Wrap-up

- nerdctl = contaiNERD CTL
- Made for facilitating new experiments in the containerd platform
  - Lazy-pulling with eStargz/SOCI/Nydus/OverlayBD
  - Registry-less P2P image distribution with IPFS
  - Image encryption with OCIcrypt
  - Image signing with Cosign / Notation
  - “Real” read-only mounts with mount_setattr
  - Slirp-less rootless containers with bypass4netns
  - Interactive debugging of Dockerfiles, with buildg
  - And more!

https://github.com/containerd/nerdctl Slack: `#containerd` at slack.cncf.io