



# containerdの最近の機能とv2

Container Runtime Meetup #5 (Feb. 21)

日本電信電話株式会社  
ソフトウェアイノベーションセンタ  
徳永 航平

- コンテナ関連OSSのメンテナ
  - containerd(reviewer)、BuildKit(maintainer)
  - stargz-snapshotter(maintainer): コンテナイメージを高速にpullするプラグイン(containerd non-core subproject)
- 興味: コンテナ、WebAssembly
  - 最近はコンテナをWasm VM(ブラウザやWasiランタイムなど)で実行する技術にも取り組む(container2wasm)



# containerdの概要



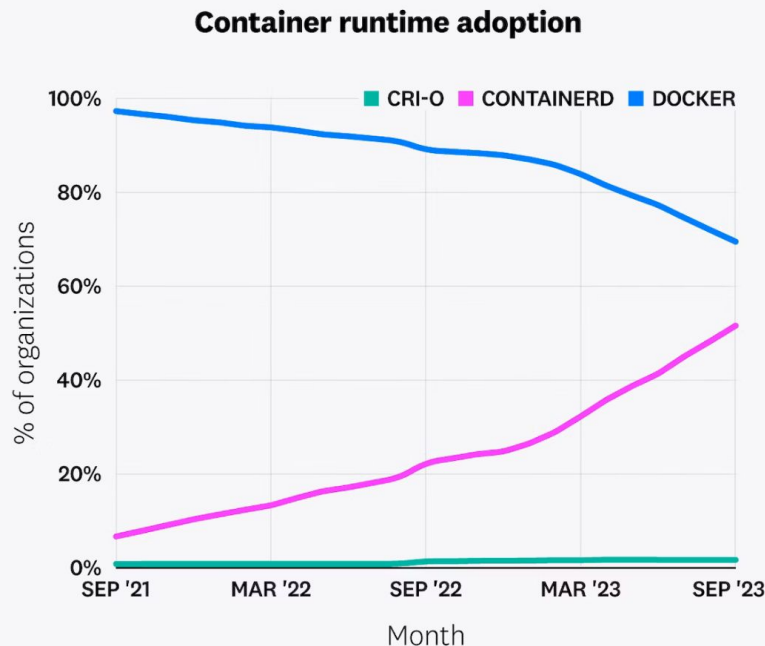
<https://github.com/containerd/containerd>

- オープンソースなコンテナランタイム (CNCF graduated)
- Kubernetes、Dockerはじめ様々なツールでコンテナ等の管理に用いられる

## 採用プロジェクト例

- コンテナ関連ツール: Docker/Moby、BuildKit
- マネージドKubernetes: Google Kubernetes Engine、Azure Kubernetes Service、Amazon Elastic Kubernetes Service
- Kubernetesディストリビューション: k3s、kind、minikube

# containerdの採用拡大



**CONTAINERD  
ADOPTION HAS  
MORE THAN  
DOUBLED IN THE  
LAST YEAR**

Source: Datadog

Datadog社による調査「10 INSIGHTS ON REAL-WORLD CONTAINER USE」より引用(2024/2/13)  
: <https://www.datadoghq.com/container-report/>

# さまざまなツールからのcontainerd利用

## コンテナ開発ツール

**nerdctl**

高速rootless、eStargzなど先進機能を備えたDocker互換ツール

**Docker**

containerdベースのコンテナ管理ツール

## コンテナ運用基盤

**Kubernetes**

分散環境上のコンテナ管理

## その他コンテナを使うツール

**BuildKit**

コンテナイメージビルダ

...

containerd API

containerd API

CRI

containerd API

**containerd**

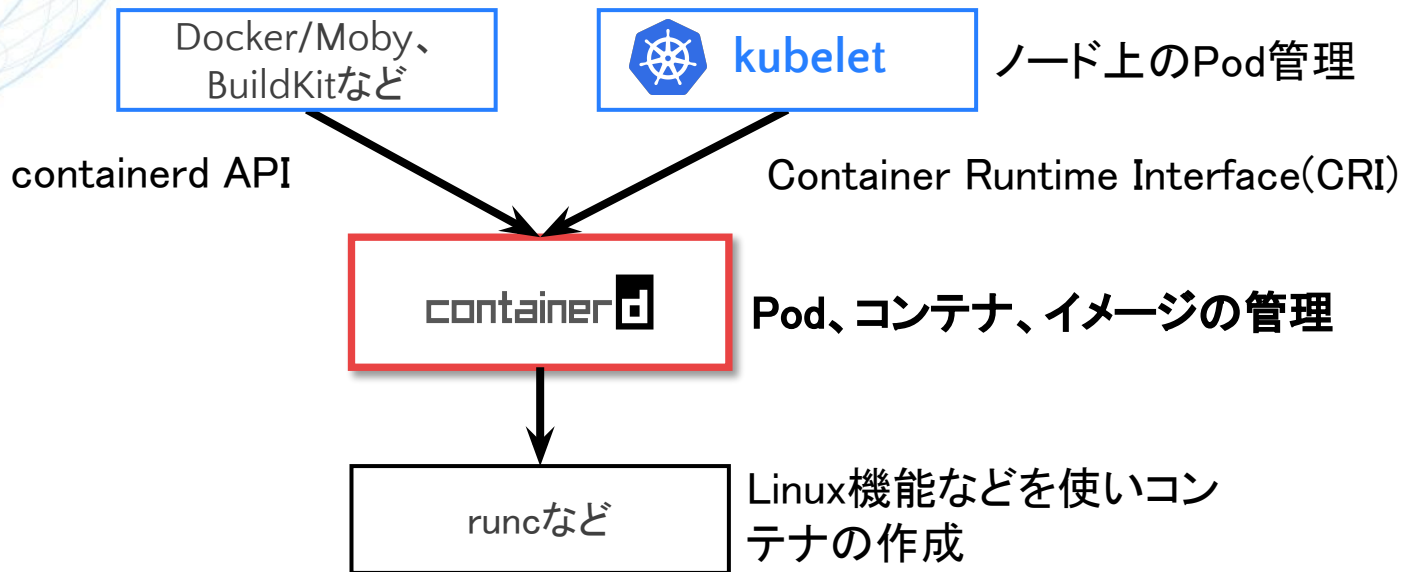
汎用のコンテナ管理フレームワーク

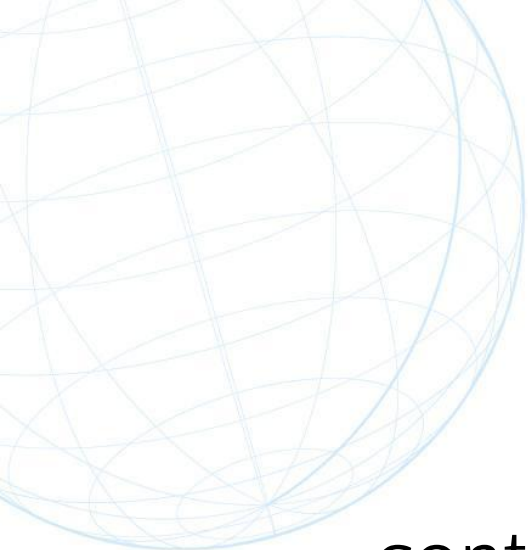
**runcなど低レベルランタイム**

Linuxなどの機能を使いコンテナ作成

# containerdの2つのクライアントAPI

- **containerd API**: Docker、nerdctl、BuildKitなどが使う
- **Container Runtime Interface(CRI)**: Kubernetesが使う
  - 各ノードのkubeletはCRIを通じてランタイム操作・Pod管理



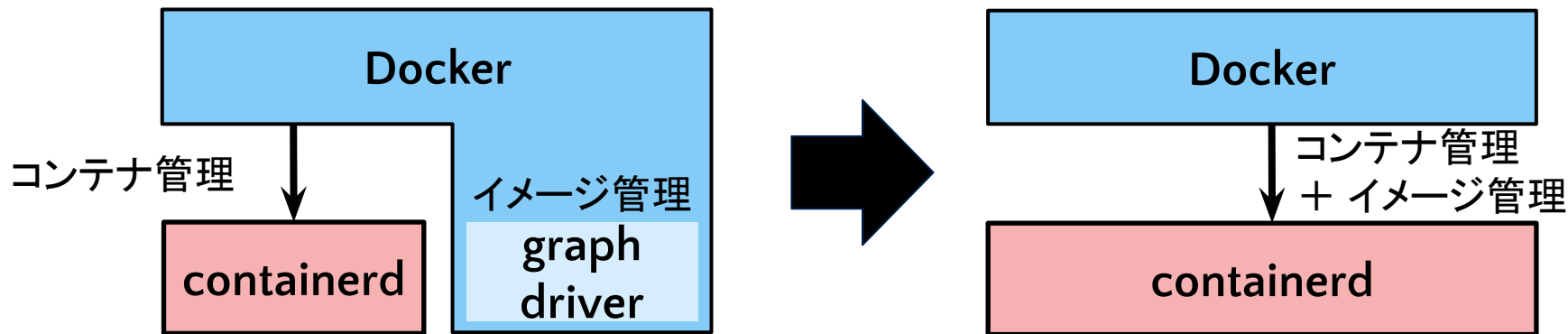


# containerdまわりの最近の話題



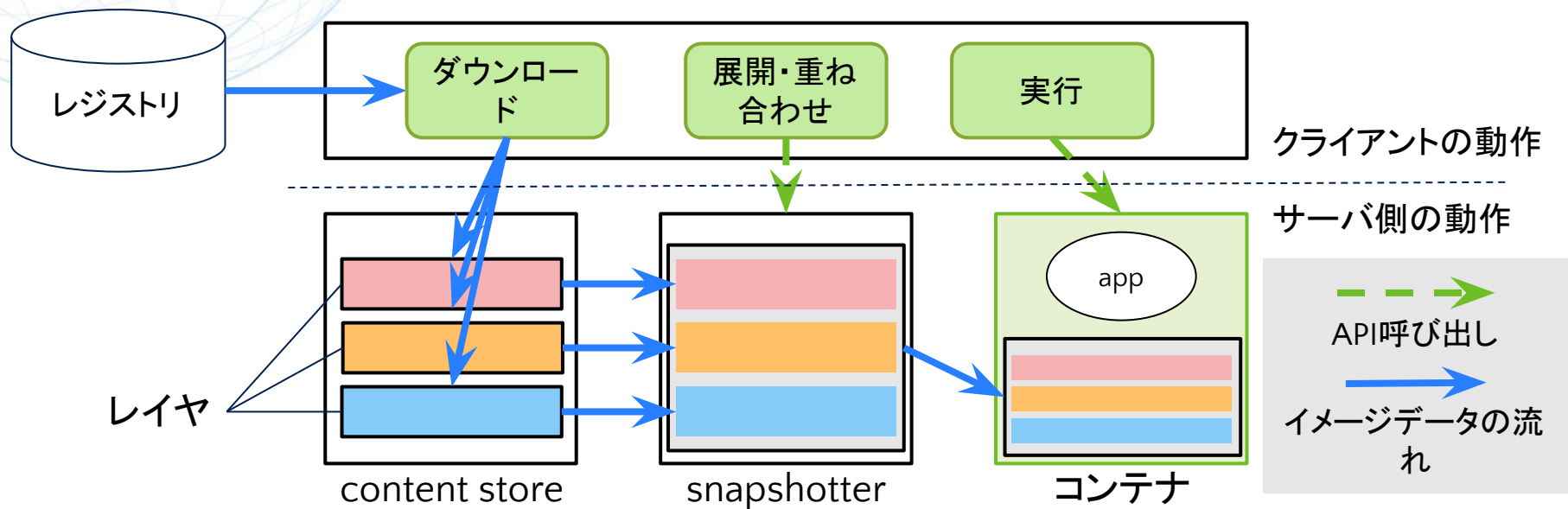
# Dockerとcontainerdのimage store統合

- Dockerは従来からコンテナ管理にcontainerdを利用してきた
- Docker (>= v24)はイメージ管理機能にもcontainerdを利用できるようになった(experimental)
- Dockerはcontainerdのイメージ管理プラグイン(snapshotter)を活用できる



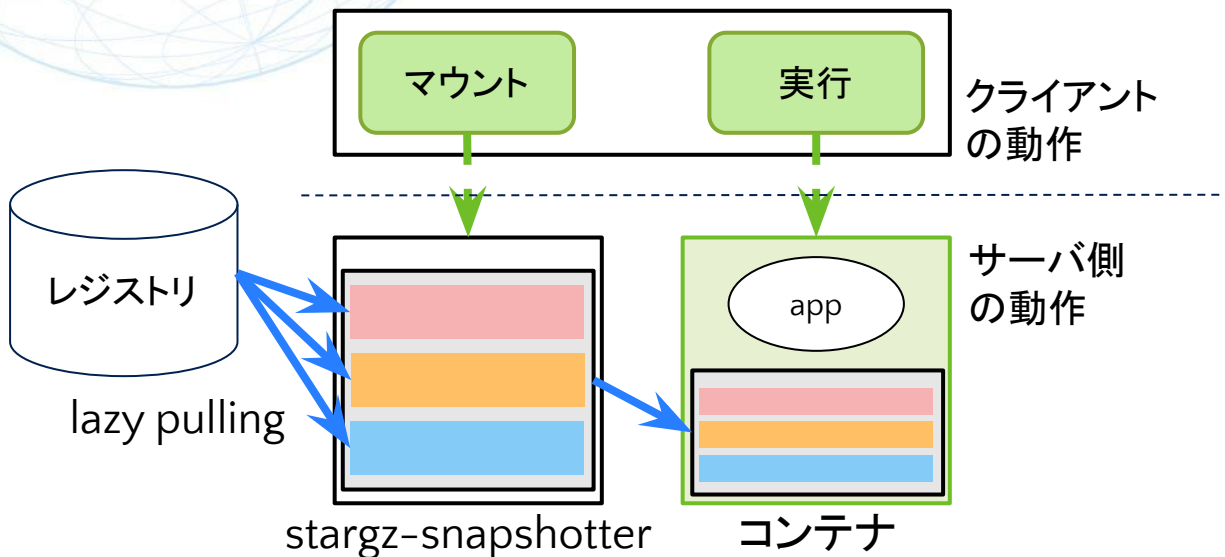
# containerd snapshotter(Dockerからも利用可)

- イメージのレイヤを展開・重ね合わせたもの(snapshots)を管理
- コンテナの起動時はsnapshotがrootfsとして使われる
- さまざまなファイルシステム等が利用可能
  - overlaysfs, btrfs, blockfile, devmapper, lcow, native, windows



# Lazy pullingによる高速なイメージpull(Dockerからも利用可)

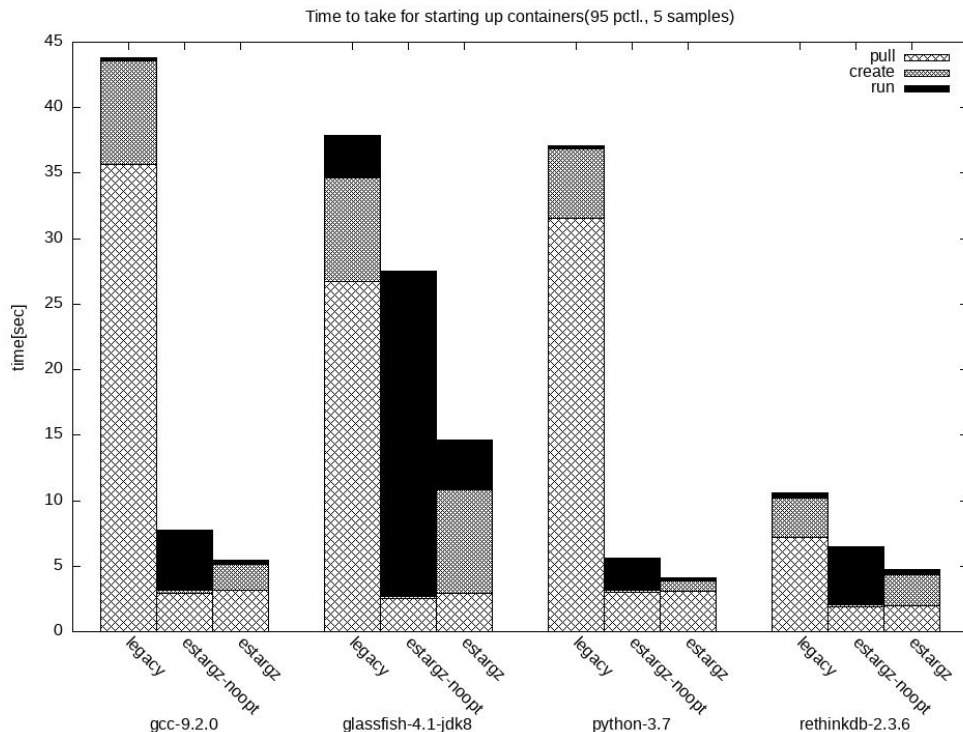
- Lazy pulling: イメージ全体がノード上にpullされるのを待たず、起動に必要な一部のファイルだけダウンロードしコンテナを起動
- “Remote Snapshotter”プラグインを通じてLazy pullingが利用可能
- Docker v24もcontainerdを通じてremote snapshotterを利用可能



## 様々なremote snapshotters

- stargz-snapshotter
- soci-snapshotter
- nydus-snapshotter
- overlaybd-snapshotter
- CVMFS snapshotter

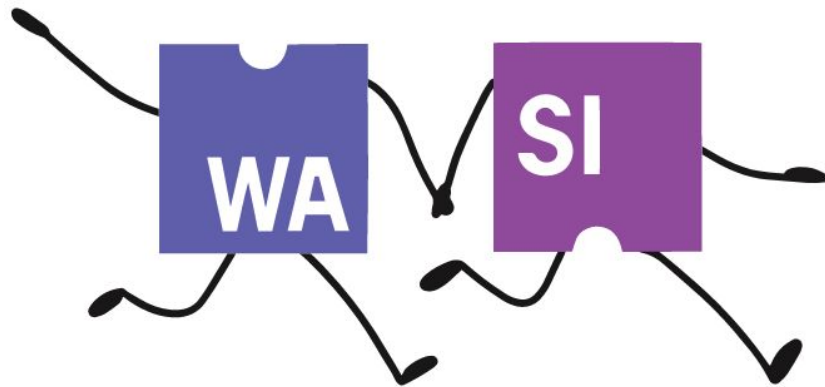
# Stargz Snapshotter(containerd + ctrの例)



<https://github.com/containerd/stargz-snapshotter/blob/d7e16d46bf960597fb1e46f22a47bfc5bf074a8e/README.md>

# WebAssembly(Wasm)実行を可能にするrunwasi

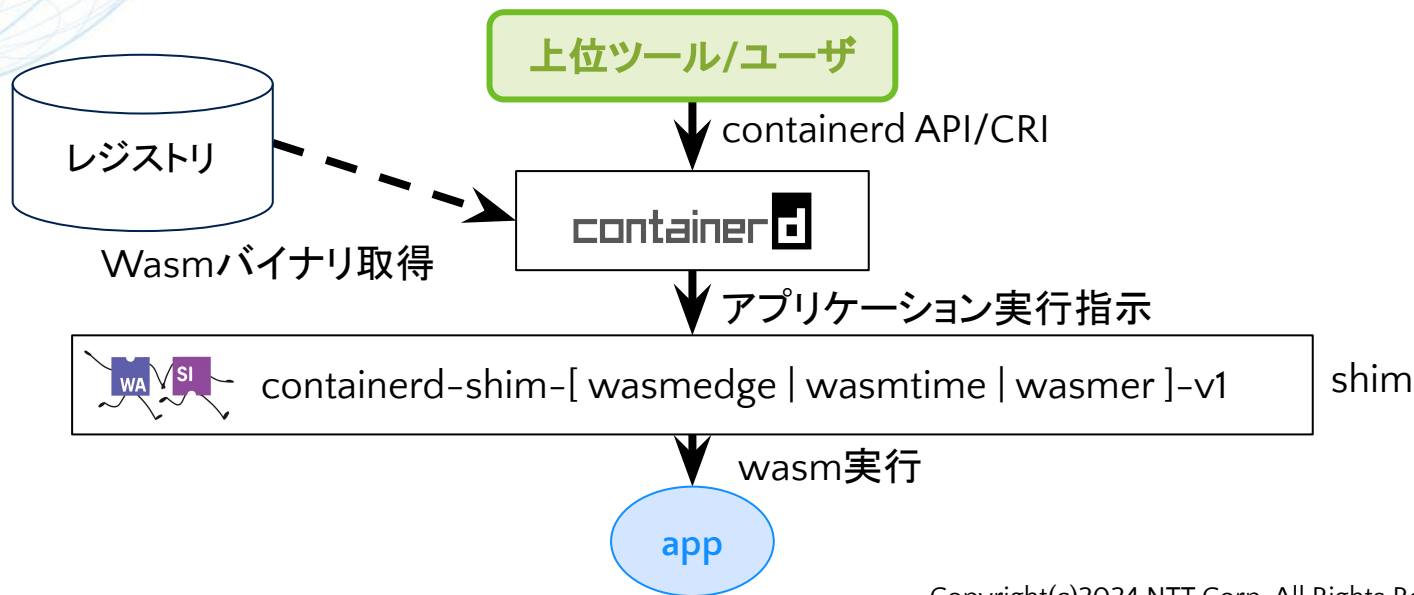
<https://github.com/containerd/runwasi>



- containerdサブプロジェクト
- containerdでWasm(Wasi)アプリケーションを管理可能
- wasmtime、wasmedge、wasmerを統合済
- 他のwasmホストにも統合するためのライブラリを提供

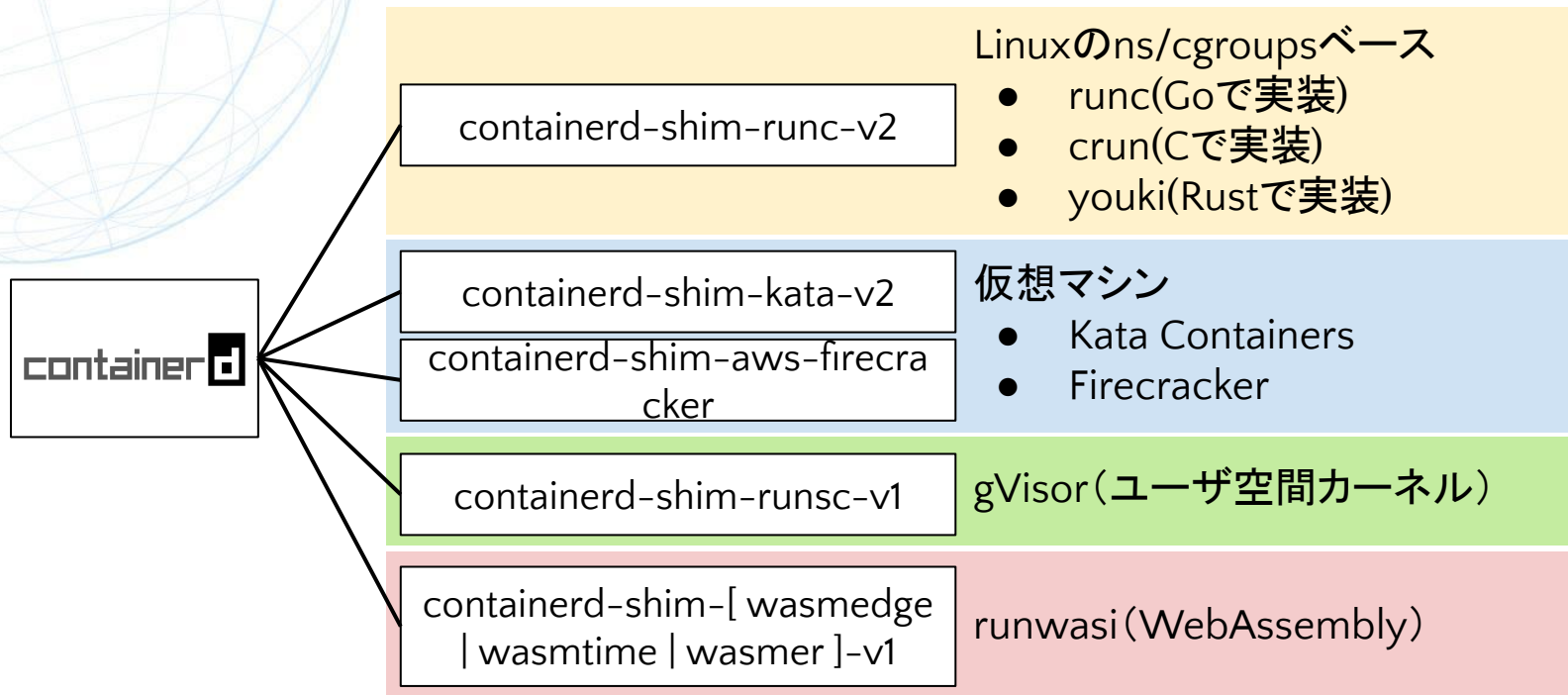
# runwasi: containerd shimによるWasm統合

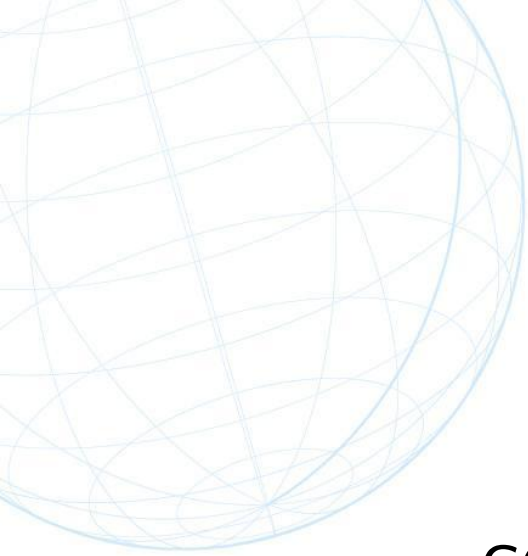
- containedは低レベルランタイムなどを”shim”プラグインを通じて操作
  - shimを実装することで、Linuxベースのコンテナに限らずさまざまなアプリケーションの実行方法に対応可能
- runwasiが提供するWasm実行対応のshimによりcontainerdでWasmアプリケーションの管理が可能



# shimを通じた様々な実行手法のサポート

- shimは従来からWasmに限らず様々なコンテナ実行方法(VMベースのコンテナランタイムなど)をサポートするのに使われてきた



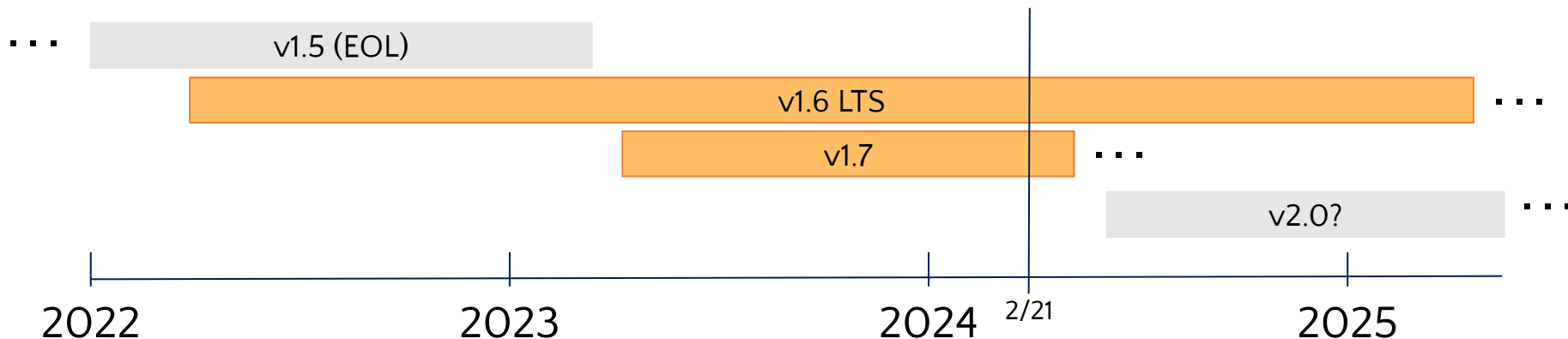


# containerdの最近の機能



# containerd v1.6 Long Term Stable (LTS) release

- v1.6はLong Term Stable (LTS) releaseとして3年以上サポートされる(少なくとも2025/2/15まで。あるいは次回LTS+6ヶ月)
- v1.7は2024/2/21現在の最新リリース(少なくとも2024/3/10までサポート。あるいはv2.0+6ヶ月)
  - v1.6と同じEOLまで延長されそう(?)
- v2.0は次回のリリース。2024/2/21現在でv2.0.0-beta.2。



<https://github.com/containerd/containerd/blob/main/RELEASES.md#support-horizon>

# containerd v2.0 (v2.0.0-beta.2現在)

- 2024/2/21現在でv2.0.0-beta.2
- 2017/12(v1.0.0)以来のメジャーバージョンリリース
- deprecated機能の削除
- 実験的機能の正式サポート化

※v2.0は2024/2/21現在で開発中(beta.2)のため、今回紹介する機能は正式リリース時には変更されている可能性があります

# containerd v2.0: 非推奨機能の削除

- deprecatedになっていた機能が削除される予定

Component	Deprecation release	Target release for removal	Recommendation
Runtime V1 API and implementation ( <code>io.containerd.runtime.v1.linux</code> )	containerd v1.4	containerd v2.0 ✓	Use <code>io.containerd.runc.v2</code>
Runc V1 implementation of Runtime V2 ( <code>io.containerd.runc.v1</code> )	containerd v1.4	containerd v2.0 ✓	Use <code>io.containerd.runc.v2</code>
Built-in <code>aufs</code> snapshotter	containerd v1.5	containerd v2.0 ✓	Use <code>overlayfs</code> snapshotter
Container label <code>containerd.io/restart.logpath</code>	containerd v1.5	containerd v2.0 ✓	Use <code>containerd.io/restart.loguri</code> label
<code>cri-containerd-*.tar.gz</code> release bundles	containerd v1.6	containerd v2.0 ✓	Use <code>containerd-*.tar.gz</code> bundles
Pulling Schema 1 images ( <code>application/vnd.docker.distribution.manifest.v1+json</code> )	containerd v1.7	containerd v2.0	Use Schema 2 or OCI images
CRI <code>v1alpha2</code>	containerd v1.7	containerd v2.0 ✓	Use CRI <code>v1</code>
Legacy CRI implementation of podsandbox support	containerd v2.0	containerd v2.0 ✓	
Go-Plugin library ( <code>*.so</code> ) as containerd runtime plugin	containerd v2.0	containerd v2.1	Use external plugins (proxy or binary)

<https://github.com/containerd/containerd/blob/v2.0.0-beta.2/RELEASES.md#deprecated-features>

# containerd v2.0: 実験的機能の正式サポート化

- 実験的機能として位置づけられていた機能が正式サポートになる

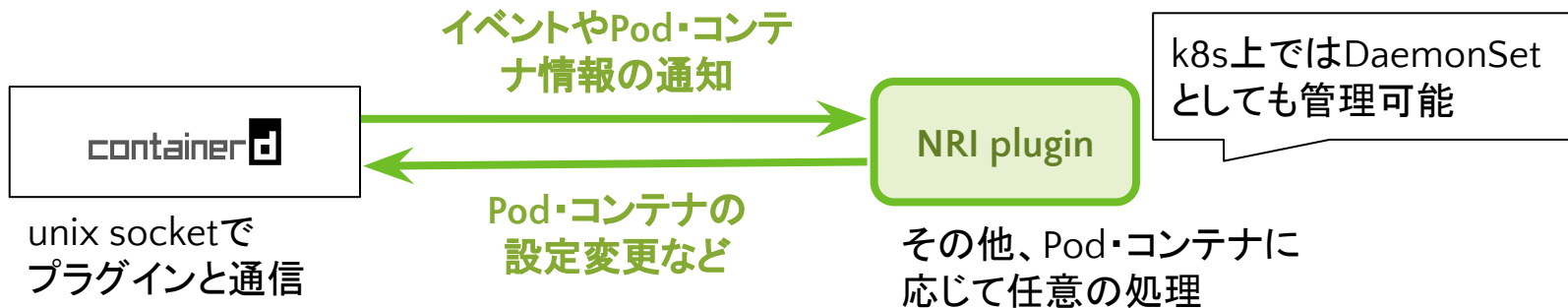
Component	Initial Release	Target Supported Release
<a href="#">Sandbox Service</a>	containerd v1.7	containerd v2.0
<a href="#">Sandbox CRI Server</a>	containerd v1.7	containerd v2.0
<a href="#">Transfer Service</a>	containerd v1.7	containerd v2.0
<a href="#">NRI in CRI Support</a>	containerd v1.7	containerd v2.0
<a href="#">gRPC Shim</a>	containerd v1.7	containerd v2.0
<a href="#">CRI Runtime Specific Snapshotter</a>	containerd v1.7	containerd v2.0
<a href="#">CRI Support for User Namespaces</a>	containerd v1.7	containerd v2.0

<https://github.com/containerd/containerd/blob/v2.0.0-beta.2/RELEASES.md#experimental-features>

# Node Resource Interface(NRI)

<https://github.com/containerd/nri>

- コンテナへのデバイス追加やCPU・メモリ制限など、リソース設定などに変更を施せるプラグイン
- containerdはPodやコンテナ作成などのイベントをプラグインに都度通知
- プラグインはイベントに応じてコンテナのリソース設定などを変更
- v2.0から、CRI使用時にNRIを利用可能



# Node Resource Interface(NRI)

## Container Adjustment

During container creation plugins can request changes to the following container parameters:

- annotations
- mounts
- environment variables
- OCI hooks
- rlimits
- linux
  - devices
  - resources
    - memory
      - limit
      - reservation
      - swap limit
      - kernel limit
      - kernel TCP limit
      - swappiness
      - OOM disabled flag
      - hierarchical accounting flag
      - hugepage limits

コンテナ作成時にプラグイン  
から変更可能な設定項目

<https://github.com/containerd/nri?tab=readme-ov-file#container-adjustment>

- CPU
  - shares
  - quota
  - period
  - realtime runtime
  - realtime period
  - cpuset CPUs
  - cpuset memory
- Block I/O class
- RDt class

# Node Resource Interface(NRI)

コンテナ作成後、プラグインから変更可能な設定項目

<https://github.com/containerd/nri?tab=readme-ov-file#container-updates>

- resources

- memory

- limit
    - reservation
    - swap limit
    - kernel limit
    - kernel TCP limit
    - swappiness
    - OOM disabled flag
    - hierarchical accounting flag
    - hugepage limits

- CPU

- shares
    - quota
    - period
    - realtime runtime
    - realtime period
    - cpuset CPUs
    - cpuset memory
  - Block I/O class
  - RDT class

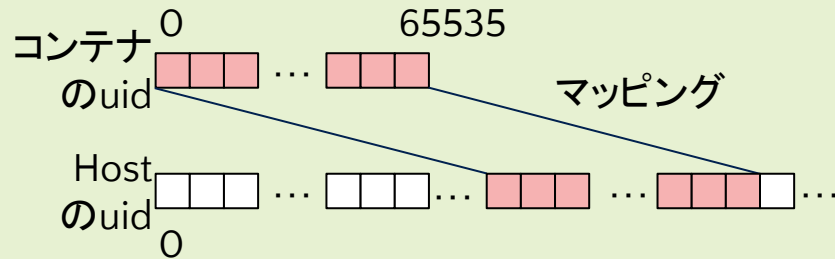
# CRI User Namespace

Kubernetes v1.28 alpha機能

- KEP-127: [Support User Namespaces](#)
- Podをホストと異なるuser nsで実行
  - コンテナ内でrootで実行されるプロセスをホスト側ではnon-rootユーザに対応させ権限を与えない
- ファイルシステムのuid,gidマッピングにはLinuxのID mapped mountを利用
- ランタイムやkubeletなどk8sコンポーネントの非特権ユーザ実行はNon-Goal
  - [KEP-2033: Kubelet-in-UserNS \(aka Rootless mode\)](#)で取り組まれている

```
apiVersion: v1
kind: Pod
metadata:
  name: userns
spec:
  hostUsers: false
  containers:
  - name: shell
    command: ["sleep", "infinity"]
    image: debian
```

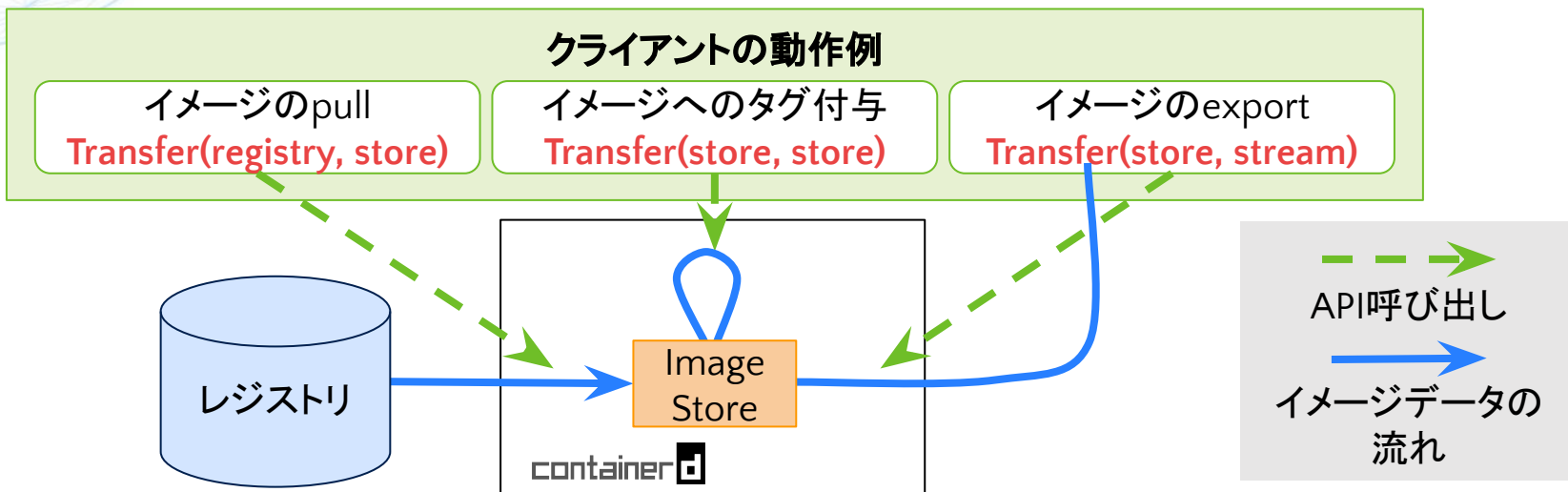
<https://kubernetes.io/docs/tasks/configure-pod-container/user-namespaces/#create-pod>





```
Transfer(ctx context.Context, source interface{}, destination interface{}, opts ...Opt) error
```

- イメージをcontainerd内のある場所から別の場所にする機能
- v2.0ではイメージ検証プラグインがtransfer serviceに追加された
  - <https://github.com/containerd/containerd/blob/v2.0.0-beta.2/docs/image-verification.md>
- レジストリ通信をクライアントだけでなくサーバでも扱える



# Transfer Serviceで実装されるイメージ操作

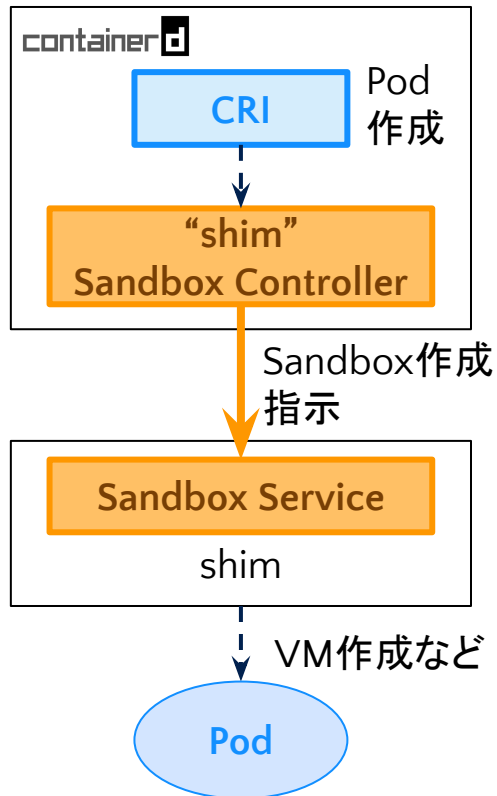
- Transfer Serviceに指定するSource、Destinationの組み合わせで様々なイメージ操作が実装される

Source	Destination	Description	Local Implementation Version
Registry	Image Store	"pull"	1.7
Image Store	Registry	"push"	1.7
Object stream (Archive)	Image Store	"import"	1.7
Image Store	Object stream (Archive)	"export"	1.7
Object stream (Layer)	Mount/Snapshot	"unpack"	Not implemented
Mount/Snapshot	Object stream (Layer)	"diff"	Not implemented
Image Store	Image Store	"tag"	1.7
Registry	Registry	mirror registry image	Not implemented

# Sandbox Service

- コンテナの集合(Podなど)をshimで管理可能
  - 従来のshimは個別のコンテナを管理
- shimはコンテナの集合の管理機能を”Sandbox Service”としてcontainerdに提供可能
- 低レベルランタイムがPodごとの隔離追加(e.g. VM作成)などしやすく、実装を単純化できる
- containerd v2.0のCRI機能では2つの実装が利用可能
  - “shim”: shimのSandbox Serviceを利用
  - “podsandbox”: shimのSandbox Serviceを利用しない。pauseコンテナを使う。(デフォルト)

“shim”モードの例



# containerd v2.0: その他正式サポートになる機能

- gRPC shim
  - <https://github.com/containerd/containerd/pull/8052>
  - containerdとshim間の通信プロトコルに、新たにgRPCが利用可能
  - これまではttrpcという、軽量を重視したプロトコルのみ対応していた (Go, Rust実装がある): <https://github.com/containerd/ttrpc>
  - ttrpc実装がない言語 (非Go、非Rust) もshimを作りやすくなる
- CRI Runtime Specific Snapshotter
  - <https://github.com/containerd/containerd/pull/6899>
  - CRIにて、利用する低レベルランタイムごとに異なるsnapshotterを指定できる

## containerdまわりの動向

- Docker (>= v24)によるcontainerd image store統合(experimental)で、snapshotterやlazy pullingをDockerでつかえるようになる
- runwasiによりcontainerdからWasmアプリケーションを管理可能

## containerdの最近の機能

- v1.6はLTSとしてサポートされる
- v2.0は非推奨機能の削除や実験的機能の正式サポートが含まれる予定
  - v2.0-beta.2:

<https://github.com/containerd/containerd/releases/tag/v2.0.0-beta.2>