Kontejnery

Ondřej Smola

04.04.2022 | CTC

Open Container Initiative (OCI)

- Založeno v roce 2015 v době začátku masivní popularity nástrojů pro orchestraci kontejnerů (Kubernetes)
- Požadavky a standardy na OS pro běh kontejnerů

- https://opencontainers.org/

- OCI Image Specification
- OCI Runtime Specification

Komponenty

- Obraz (Image)
- Registr
- Kontejner
  - Jádro (Engine)
  - Běhové prostředí

Obraz

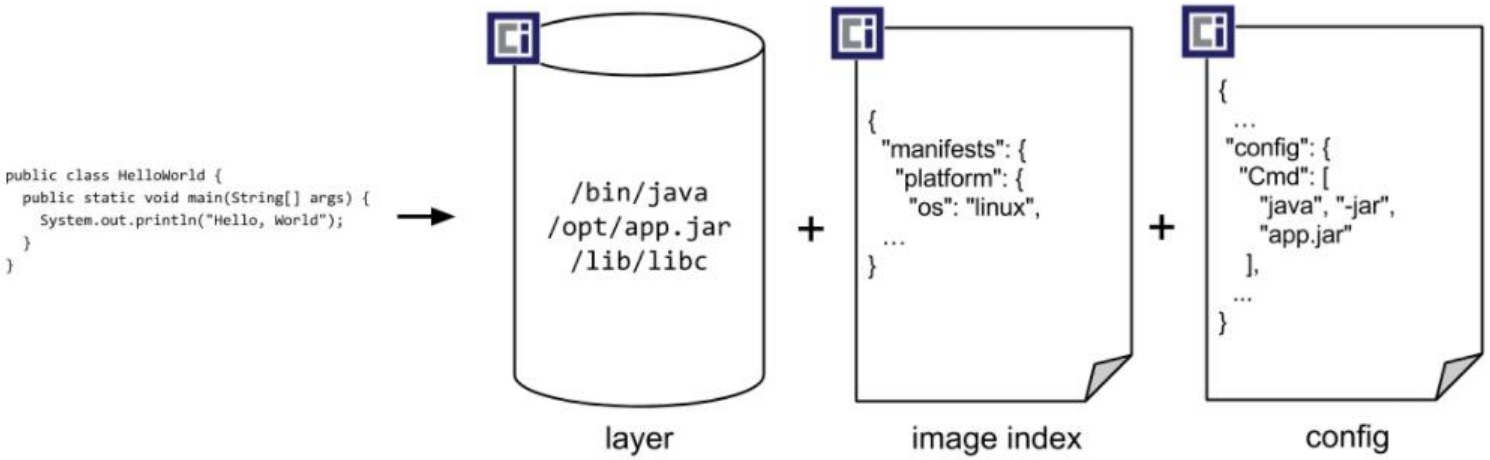
- Základní funkční prvek
- Skládá se z
  - Manifestu – definice všech komponent které dohromady tvoří image
  - Rozložení (Layout) – definuje strukturu vrstev souborového systému
  - Vrstvy souborového systému
  - Index (volitelný) – index manifestů

- plná OCI specifikace

```
FROM ubuntu:21.04
COPY . /app
RUN make /app
CMD python /app/app.py
```

Obraz 2

- “pojmenovaný balíček”, který lze jednoduše kopírovat, sdílet a upravovat
- Přenositelné mezi platformami (Ubuntu, CentOS, Fedora …)



Pojmenování

- Plné jméno
  - docker.io/library/ubuntu:20.04
  - REGISTR/SKUPINA/PROJEKT:TAG
- Projekt
  - Slouží k oddělení projektů
  - Např. ubuntu nebo nginx
- Tag
  - Uživatelský: 1.2.4
  - Hash: e1f5733f050b2488a17b7…
  - Speciální: latest

Registr

- Centrální úložiště pro obrazy (aka Github/Gitlab pro kontejnery)
  - Cloudové
    - Dockerhub
    - Azure Container Registry
  - Privátní
    - Harbor
    - Docker Registry
- Může obsahovat více skupin
- Pokud není nakonfigurováno jinak je implicitní nastavení
  - Registr: docker.io
  - Skupina: library
- docker pull ubuntu:20.04 == docker.io/library/ubuntu:20.04

Kontejnery

- Skupina izolovaných procesů běžících na jednom stroji se společnou sadou vlastností

- Izolace aplikací
  - Procesor, paměť, disk, síť, oprávnění, souborový systém
  - Vytvoření izolovaného prostředí (sandbox)
- Jednoduchá distribuce
  - Balíček obsahující všechny prostředky potřebné pro běh
  - Systémové balíčky a knihovny, konfigurační soubory, skripty
- Sjednocený způsob pro práci
  - Spuštění, zastavení, výpis, přístup
  - Přístup k výstupu, metrikám

Jádro

- Role
  - Přijímá požadavky klientů (z příkazové řádky nebo přes API)
  - Stahuje a rozbaluje obrazy
  - Spouští kontejnery s pomocí běhového prostředí
- Docker (2013)
- Containerd
  - Základem byl původní Docker, implementace v Go
  - Přesunut pod Open Container Initiative (OCI)
- Cri-O
  - Alternativa od Red Hat
  - Zaměření na Kubernetes

Běhové prostředí

- Využívané jádrem pro běh kontejnerů
- OCI specifikace s referenční implementací **runc**
  - Existují alternativní implementace (crun, kata, gVisor)

- Role
  - Použití připraveného mount-pointu a metadat
  - Komunikace s jádrem pro vytvoření procesu
  - Nastavení izolace pomocí namespace (další slide)

- https://developers.redhat.com/blog/2018/02/22/container-terminology-practical-introduction#container_runtime

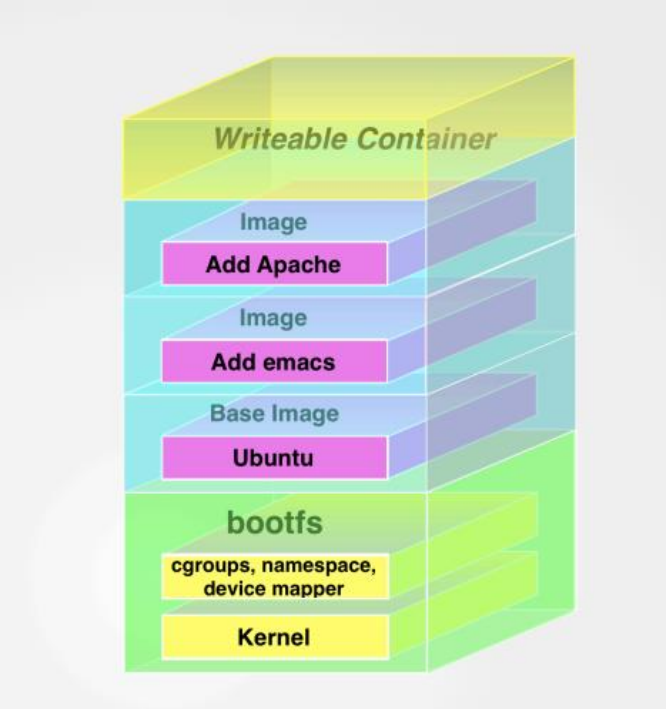
Linux namespace

- Izolace prostředků v jádře od verze 2.4 (pouze mnt, 2002)
- Pro potřeby kontejnerů kompletní od jádra 3.8 (2013)
  - *mnt* - *izolace na úrovni mnt pointů*
  - *net* - *izolace na úrovni virtuálních síťových vrstev*
    - *privátní IP, dns, firewall, sokety*
    - *veth pair bridge*
  - *user* - jiné opravnění vně/uvnitř (rootless)
  - *cgroup* - *izolace systémových prostředků (cpu, mem, gpu, …)*
  - *ipc (inter process komunikace), uts (hostname), pid (hierarchie a izolace)*

- *Lze je komponovat - nsenter*

Union filesystem

- Skládá se z vrstev
  - BootFs (přečten při startu)
  - RootFs (např. ubuntu, pouze čtení)
  - Uživatelské vrstvy (pouze čtení)
  - Vrstva běžícího kontejneru (čtení i zápis)



- Spojeny pomocí Union mount

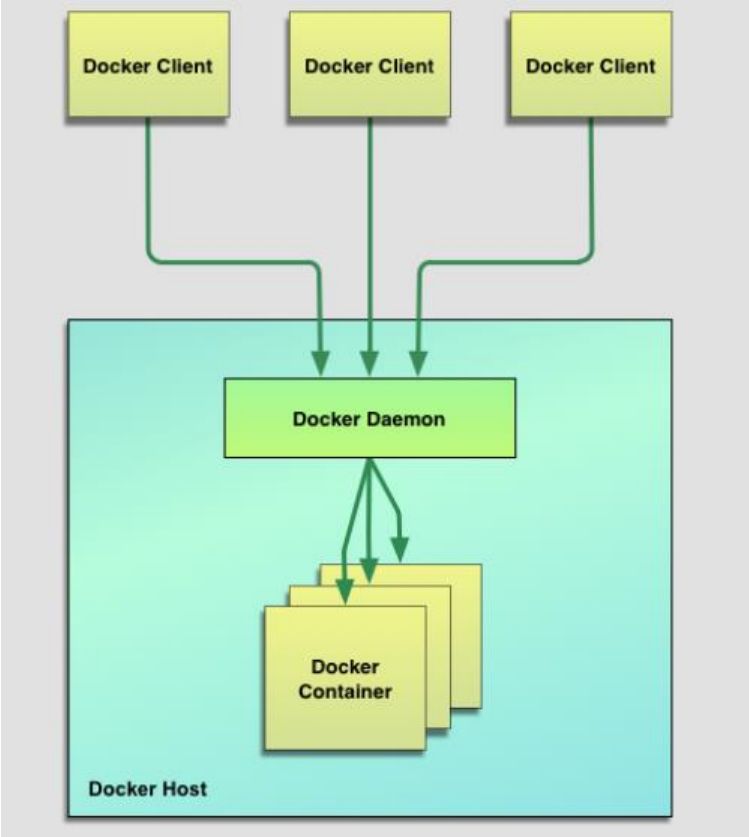
- Zápis pomocí metody COW (copy on write)
  - Upravený soubor se nakopíruje do zápisové vrstvy, původní verze v předchozí vrstvě se nezmění

Docker

Klient a server

- Práce s dockerem probíhá pomocí
  - CLI programu docker
  - REST API
- Serverem je dockerd (démon)
- Server může běžet
  - lokálně /var/run/docker.sock
  - vzdáleně tcp://0.0.0.0:2375
    - musí být povoleno
    - komunikace není v základu zabezpečena!!!

>> docker version

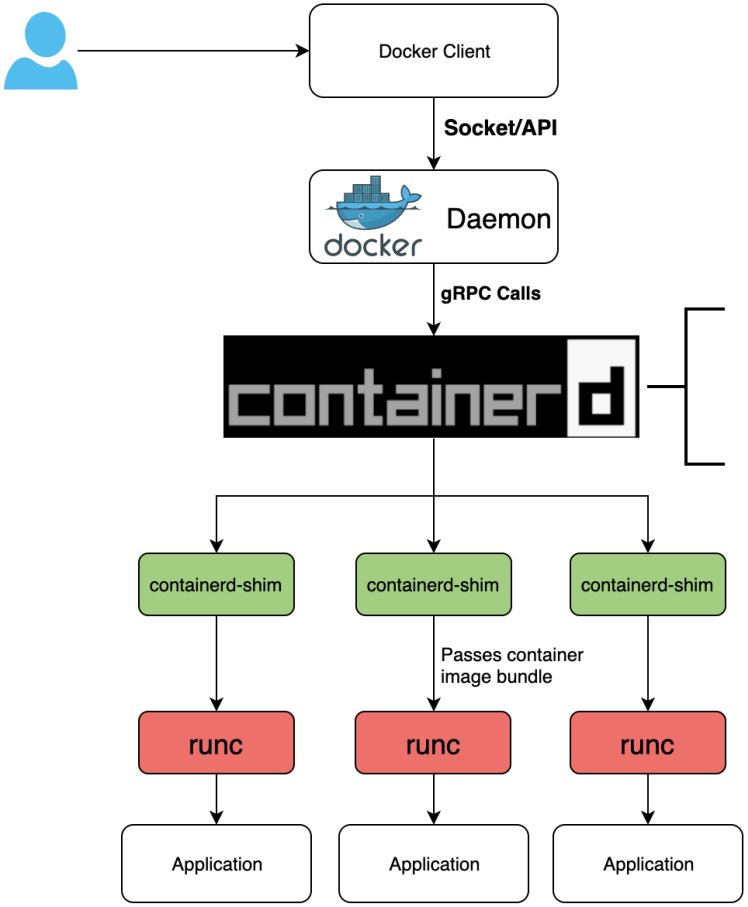


Docker -> Containerd (OCI)

- Containerd vzniklo oddělením enginu dockeru jako samostatného open source řešení

```
apt-get install docker-ce docker-ce-cli containerd.io
```

Komunikace

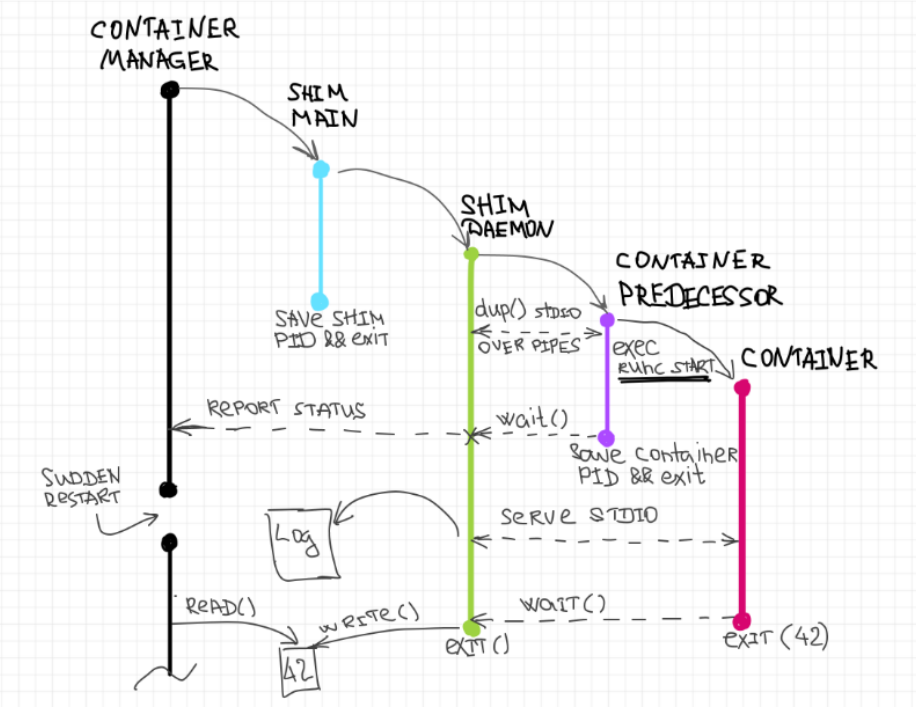


- High level container runtime
- Responsible for container lifecycle management
- Responsible for container image management
- unpacks the image into an OCI runtime bundle and shells out to `runc` to run it

- OCI container runtime spec reference implementation
- Container runtime code
- Create containers

Co je to „shim“

- Prvek mezi správcem (containerd, cri-o) a běhovým prostředím (runc)
- Umožňuje
  - Přežít restart správce
  - Správce nemusí spravovat dlouho běžící volání po startu
  - Synchronizace stavu
    - Exit kód
    - Chyba v CLI argumentech



Síť

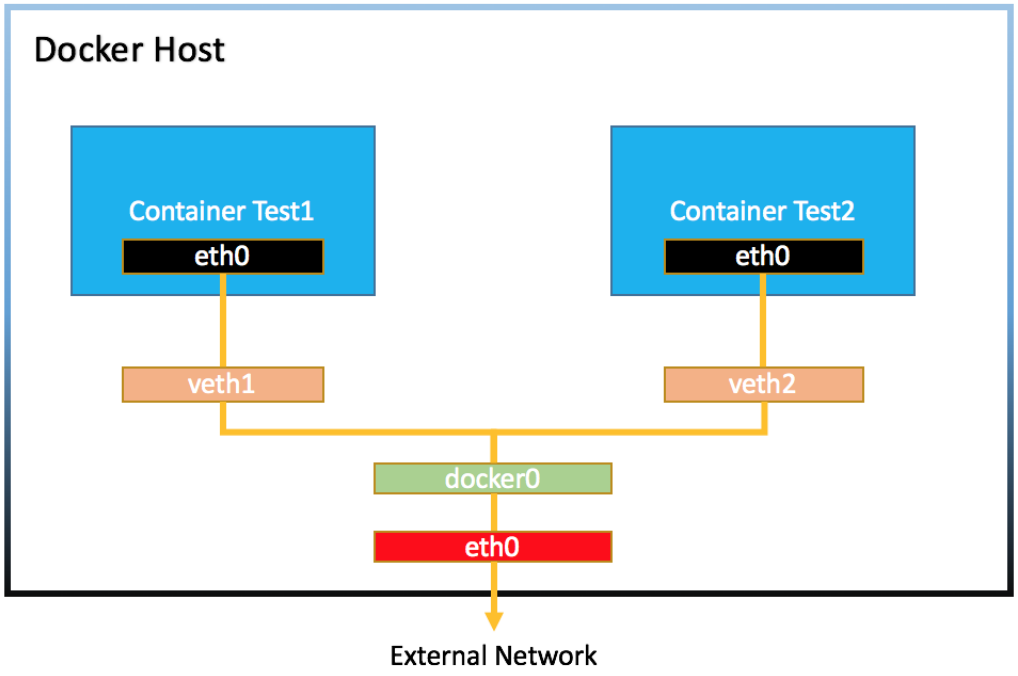
- Ovladače
  - Bridge
    - Propojení kontejnerů v rámci jednoho stroje
  - Host
    - Bez izolace
  - Overlay
    - Propojení kontejnerů v rámci více strojů
  - None
    - Bez externí konektivity
  - Externí pluginy

Konfigurace

- Mapování portů
  - definice (host:kontejner) pomocí volby -p
    - 8080:80
    - 192.168.1.100:8080:80
    - 8080:80/udp
  - Lze mapovat více portů
- Hostname
  - Pokud neuveden je shodný s ID kontejneru
- DNS server

Bridge

- Pokud neuvedeno jinak, je použit bridge “bridge”
- docker network create
- z příkazové řádky volba –network



Úložiště

- Volume
  - docker volume create
  - spravováno Dockerem
  - lze je sdílet mezi kontejnery
  - pojmenované nebo anonymní
- Bind mount
  - sdílený adresář mezi hostem a kontejnerem
  - umožňuje upravit soubory hosta
- Tmpfs
  - souborový systém v paměti, ztracen při restartu