A Docker alapjai

Takács Ákos

PTE Egyetemi Könyvtár és Tudásközpont

Docker alapjai

VIRTUALIZÁCIÓS MEGOLDÁSOK, ÚT A DOCKERIG

A virtualizáció előnyei

- Idő,- pénz- és helytakarékos
- Rugalmasabb gazdálkodás az erőforrásokkal
- Automatikus áthelyezés a gépek között
- Növelhető rendelkezésre állás
- Tetszőleges, új környezetek létrehozása, sablonok használata, másolás akár percek alatt

Virtuális gép vs. konténer

Szempont	Virtuális gép	Konténer
Hardver virtualizáció	van	nincs
Saját vendég kernel	van	nincs
Gazda OS-től eltérő vendég OS	van	nincs
Opcionális izoláció (pl net, pid)	nincs	van
Akár egyetlen process izolációja	nincs	van
Akár egyetlen bináris izolációja	nincs	van
Rendszerindítás	percek	Akár másodpercek

Konténerek története

- 1979 – chroot rendszerhívás bevezetése a Unix 7-ben
- 1982 – Bill Joy beépítette a BSD-be
- 1991 – Bill Cheswick kezdte el használni a jail elnevezést
- 2000 – "jail" parancs bevezetése a FreeBSD-ben
- 2005 – Sun kiadta a Solaris Containers-t, mint "chroot szteroidon"
- 2008 – LXC, avagy Linux Containers megjelenése
- 2013 febr. – Linux Kernel 3.8, teljes User namespace támogatással -> LXC nagyobb népszerűsége
- 2013 márc. - Docker megjelenése
- 2013 okt. - LMCTFY (Let Me Contain That For You) a Google-től

Névterek

- **PID:** A hoszton levő processzeket nem látjuk, csak a névtérben levőket
- **NET:** Saját hálózati interfészek a névtérnek. A hoszt interfészei nem láthatók.
- **UTS:** A névtérben levő hosztnév nem azonos a hoszt hosztnevével.
- **MNT:** Saját fájlrendszer gyökér
- **IPC:** Processzek közti kommunikáció. Pl. Osztott memória
- **USER:** A névtéren belül a felhasználó id-je különbözik a névtéren kívülitől

LXC és LXD

- **LXC**
  - A Linux kernel képességeit használja ki
  - Cgroups: Erőforrások limitálása és priorizálása
  - Namespaces: Izoláció egy processz szemszögéből (PID, UTS, IPC, NET, MNT, USER)
  - Virtuális gépek helyettesítése. Az OS van a központban, nem az alkalmazás.
- **LXD**
  - Konténer menedzser az LXC-re építve
  - Egyszerűsíti az LXC használatát
  - Az LXD csak daemon, az lxc parancsot kell használni

LXC és LXD

Funkció	LXC	LXD
Konténer létrehozása	"lxc-create"	"lxc init" ("lxc launch")
Konténer indítása	"lxc-start"	"lxc start" ("lxc launch")
Parancs futtatása konténerben	"lxc-execute"	"lxc exec"
Konténer leállítása	"lxc-stop"	"lxc stop"
Konténer törlése	"lxc-destroy"	"lxc delete"

Docker alapjai

DOCKER MŰKÖDÉSE ÉS ALAPFOGALMAK

A Dockerről általában

- 2013. márciusában jelent meg
- 0.9-es verzióig az LXC-t használta a háttérben
- 0.9-es verziótól saját fejlesztésű "libcontainer" driver és "nsenter"
- 1.8-tól "deprecated" az LXC exec driver
- Alkalmazás virtualizáció
- Micro service-ek
- Deployment és terheléselosztás egyszerűsítése
- A Docker képességeinek bővítése egyedi driver-ekkel
- Egyszerűbb reprodukálás

Komponensek

- Docker Engine (Docker CE és Docker EE)
  - Docker Daemon
  - Docker Client
  - Docker API
- Docker Compose
- Docker Swarm
- Docker Machine
- Docker Registry (Docker HUB, letölthető imageként. Tyúk és tojás esete)

Objektumok: általános

- **Image:** Csak olvasható fájlrendszer + metaadatok (Template)
- **Container:** Egy Image-ből létrehozott virtuális környezet írható fájlrendszerrel
- **Volume:** A Docker által kezelt, perzisztens, konténerek között megosztható könyvtár
- **Network:** A konténerekhez rendelt saját hálózati interfészek kezelése.
- **Plugin:** Egyedi naplózás, volume megoldások

Objektumok: swarm

- **Node:** Egy klaszterben lévő Docker Host-ok
- **Service:** Egy klaszterben indított szolgáltatás. Egy konténer egy vagy több példányban indítva.
- **Stack:** Egymással kapcsolatban álló, összetartozó szolgáltatások összessége.
- **Config:** Egy klaszterben a service konténereivel megosztható konfigurációs fájl.
- **Secret:** A Config-hoz hasonló, de a hálózaton titkosítva küldött adatok. Pl jelszavak

Ábra az architektúráról

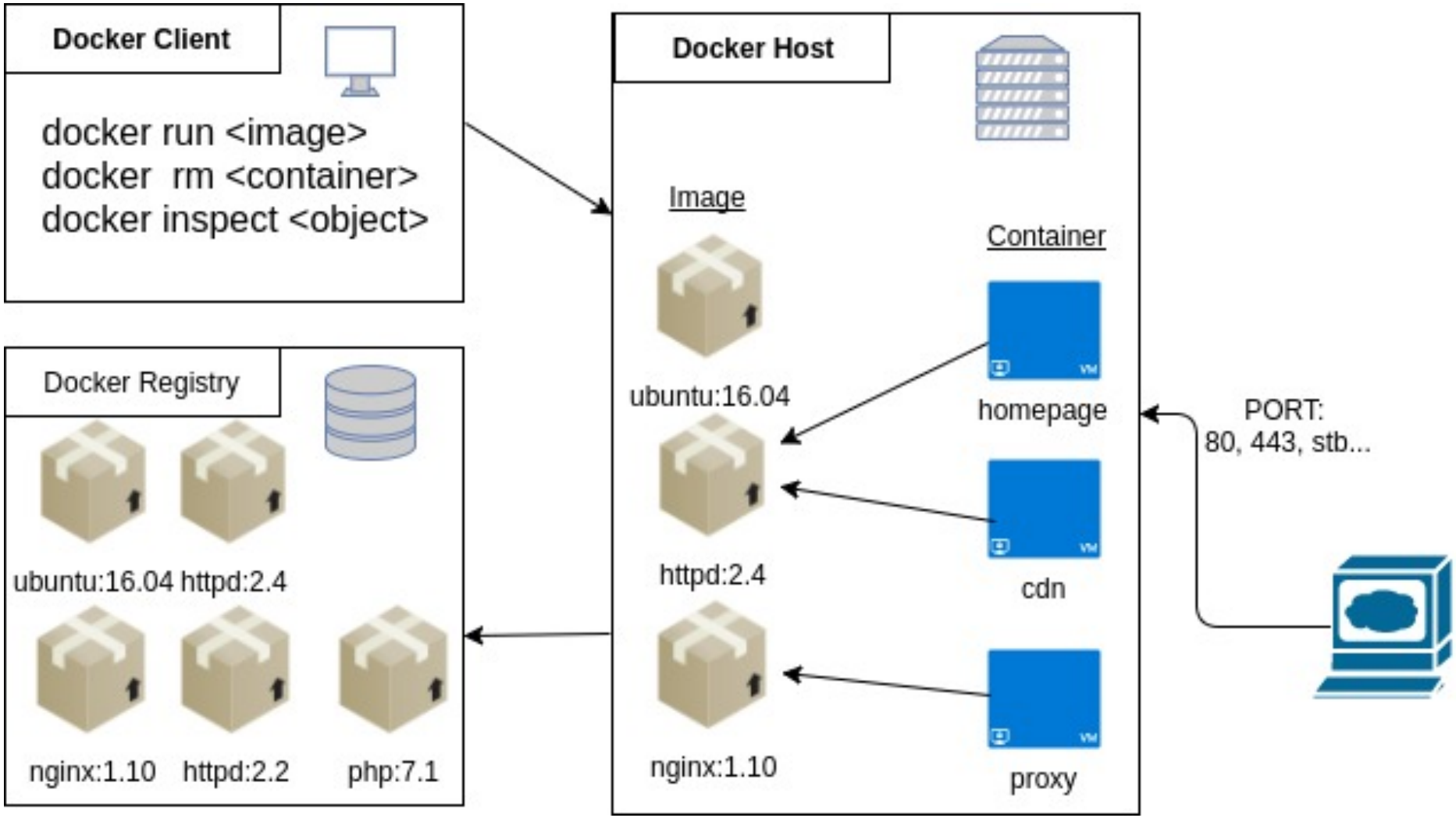

Image rétegek

Image rétegek: debian törlés

Image rétegek: php 7.1 törlés

Image rétegek: rimelek/php 7.1 törlés

Image rétegek: rimelek/php 7.0 törlés

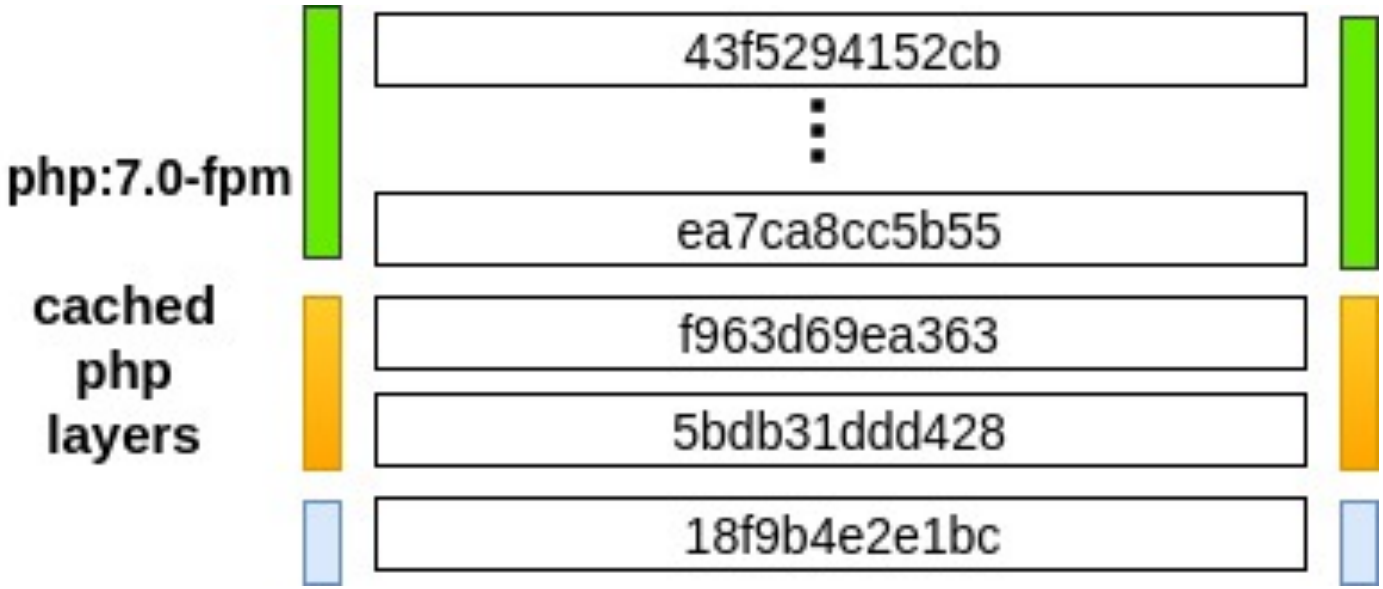

Image rétegek: php 7.0 törlés

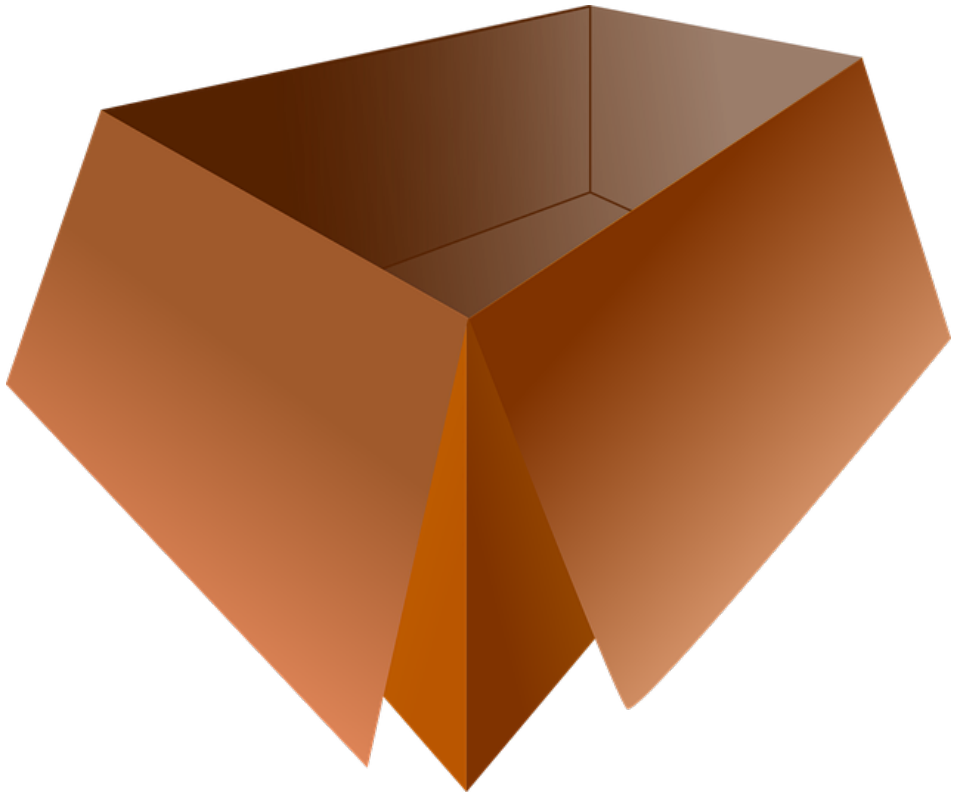

Image nevének felépítése

"registry/owner/image:tag"

Image és alapértelmezések

"registry": docker.io

"owner": library

"tag": latest

"docker.io/library/hello-world:latest"

=

"hello-world"

Storage driver

- Konténerek fájlrendszerének tárolása, összefűzése és írása
- Blokkszintű és fájlszintű storage driverek.
- Az AUFS a legrégebbi, de nem mindenhol támogatott
- Különböző esetekben eltérő hatékonyságúak
- Alapértelmezés prioritási sorrend szerint

- Aufs
- Btrfs
- Zfs
- Overlay2
- Overlay
- Devicemapper
- Vfs

Logging driver

- A konténerben a standard output és standard error átirányításának kezelése
- Alapértelmezett a "json-file"
- Beállítható a Docker daemon-ban vagy konténerenként
- A "docker logs" csak a "json-file" és "journald" drivert támogatja.

- None
- Json-file
- Syslog
- Journald
- Gelf
- Fluentd
- Awslogs
- Splunk
- Etwlogs
- Gcplogs

Docker, kernel és biztonság

- Névterek és cgroup-ok
- Az összes konténer ezt a kernelt kapja meg, nincs sajátja
- Legalább 3.10-es Linux kernel kell
- Korlátozható képességek (Pl. "net_bind_service" stb.)

Docker alapjai

A DOCKER KLIENS MŰKÖDÉSE

Docker Engine és utasítások

"docker run hello-world"

Docker Engine és utasítások

"docker run httpd:2.4"

Docker Engine és utasítások

"docker run -p "8080:80" httpd:2.4"

Docker Engine és utasítások

"docker run -d -p "8080:80" httpd:2.4"

Docker Engine és utasítások

"docker run -d -p "8080:80" \

-v $(pwd):/usr/local/apache2/htdocs \

httpd:2.4"

Docker Engine és utasítások

```
"docker run -d -p "8080:80" \
-v $(pwd):/usr/local/apache2/htdocs \
--name httpd-test \
httpd:2.4"
```

Docker Engine és utasítások

"docker container ls -a"

Docker Engine és utasítások

"docker container stop laughing_khorana"

"docker container rm laughing_khorana"

Vagy

"docker container rm -f laughing_khorana"

Docker Engine és utasítások

"docker container stop httpd-test"

"docker container start httpd-test"

Vagy

"docker container start -a httpd-test

Docker Engine és utasítások

"docker container logs httpd-test"

Vagy

"docker container logs -f httpd-test

Docker Compose és utasítások

```
version: "3.1"

services:
  httpd:
    image: httpd:2.4
    container_name: httpd-test
    ports:
      - "8080:80"
    volumes:
      - .:/usr/local/apache2/htdocs
```

Docker Compose és utasítások

"docker-compose up -d"

Docker Compose és utasítások

"docker-compose logs"

Docker Compose és utasítások

"docker-compose down"

Docker Compose és utasítások

"docker-compose start"

Docker Compose és utasítások

"docker-compose stop"

Docker Swarm

- A Docker saját klaszter menedzsere
- Manager és Worker gépek (node)
- Szolgáltatásokat indítunk (service)
- Címkézhető node-ok és szabályok
- Megérti a Docker Compose fájlt is

Docker Swarm

"docker swarm init"

Docker Swarm

"docker swarm join --token SWMTKN-1-4iyqdryfwvsoe07gnecj6k25vfb7f7o1j7yd0d96qmvw23cwr8-e2zreykvo67tr2igmweqoyg7d 192.168.1.106:2377"

Docker alapjai

MAGAS RENDELKEZÉSRE ÁLLÁS ÉS LEHETŐSÉGEK

Kizárólag Dockerrel

- Healthcheck definiálása
  - Konténer létrehozásakor ("docker run", "docker create", "docker service create")
  - Image létrehozásakor Dockerfile-ban
  - Docker Compose-zal a docker-compose.yml fájlban.
- Restart Policy
- Docker Swarm klaszter
  - Több példány futtatása több gépen.
  - Ha kiesik egy node, vagy leáll egy konténer, újraindul akár másik gépen.
  - Reverse Proxy (HA Proxy, NginX proxy)
- Naplózás külső központi szerverre (Keresés, értesítés)

További eszközökkel

- Speciális operációs rendszerek
  - CoreOS
  - Atomic Host (Fedora, CentoOS, Red Hat)
  - RancherOS
- Szoftverek
  - Kubernetes
  - OpenShift (Origin/Enterprise)
  - Rancher

Docker alapjai

HOL LEHET DOCKERT FUTTATNI?

Ingyenes lehetőségek

- Docker Playground (http://play-with-docker.com/)
- Saját VPS (+Docker Cloud, 1 saját node ingyenes)
- Saját gépen (akár windowson)

Fizetős szolgáltatások

- Google Compute Engine
- Amazon Web Services
- Microsoft Azure
- Digital Ocean
- OpenStack

Felhasznált képek

- Docker logo:
  - https://www.docker.com/brand-guidelines (Docker Media Kit)
- Image rétegek és architektúra rajza:
  - https://www.draw.io
- További képek forrása:
  - https://pixabay.com

Ajánlott források

- https://docs.docker.com
- https://www.youtube.com/user/dockerrun
- https://devopsakademia.hu
- https://netacademia.hu
- https://it-sziget.hu/tag/docker

Gyakorlat

- A virtuális gépeket szolgáltatja: Cloud For Education
- Operációs rendszer: Ubuntu 16.04
- Feladatok és szkriptek elérése: http://ld.it-sziget.hu
- Docker image-ek forrása
  - https://hub.docker.com/explore (https://strore.docker.com)
  - https://hub.docker.com/r/jwilder
  - https://hub.docker.com/r/itsziget

Kapcsolat

Takács Ákos

E-mail: kapcsolat@it-sziget.hu

Weboldal: https://it-sziget.hu