



25

# DEVLILLE

## 2025



25

# C'EST L'HISTOIRE D'UN.E OPS

Qui gère des images Docker "de base"

Qui sont utilisées par 260 micro-services



25

# UNE FAILLE DE SÉCURITÉ DANS UNE IMAGE DE BASE

Implique de mettre à jour l'image de base et la re-construire

```
docker image build && docker image push
```

Il faut mettre à jour 260 `Dockerfile`

Et relancer 260 pipelines de CI

Ça va prendre des jours



## BIENVENUE DANS L'ENFER OPS





25



si seulement on pouvait mettre à jour les images facilement  
sans tout re-build



# LE PARADIS OPS

Mise à jour des images sans re-build

application



vieux certif  
java 23



nouveaux certif  
java 24



faille  
ubuntu



sans faille  
ubuntu



25

# CRANE

Rebase d'images Docker / OCI



**Julien Wittouck**

Freelance @CodeKaio

Associé @Ekit3

Teacher @univ-lille

Team @Cloud-Nord

Technical Writer @ENI



# IMAGES OCI ET MANIFESTS

9e8e73ef

3afff29d

393ad826

5a7813e0

81bd62b8



# IMAGES OCI ET MANIFESTS

> bash

```
# docker inspect  
docker image inspect eclipse-temurin:24
```

> bash

```
TOKEN=$(curl --silent "https://auth.docker.io/token?service=registry.docker.io&scope=repository:library/eclipse-temurin:pull")  
curl \\  
  -H "Accept: application/vnd.docker.distribution.manifest.v2+json" \\  
  -H "Authorization: Bearer $TOKEN" \\  
  https://registry-1.docker.io/v2/library/eclipse-temurin/manifests/sha256:393ad826422f9dbee85d460
```









# UNE IMAGE OCI

`{}` json

```
{
  "schemaVersion": 2,
  "mediaType": "application/vnd.oci.image.manifest.v1+json",
  "config": {
    "mediaType": "application/vnd.oci.image.config.v1+json",
    "digest": "sha256:57db53168f8cca5b3a2cbf1bb2c89a97c48762a501f481f53380567a00a9f1f6",
    "size": 6071
  },
  "layers": [
    {
      "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
      "digest": "sha256:5a7813e071bfadf18aaa6ca8318be4824a9b6297b3240f2cc84c1db6f4113040",
      "size": 29754290
    },
    {
      "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
      "digest": "sha256:d1504bee8985780f3a868793411f26933d71ee828bc50880d128017596324321",
      "size": 21318286
    },
    {
      "mediaType": "application/vnd.oci.image.layer.v1.tar+gzip",
      "digest": "sha256:7b5a3507f742036dd166ee8b5a8ceeb35d451eacd472188cb43fdf7844a9e06b",
      "size": 165317796
    }
  ]
}
```



# MANIPULER LES IMAGES OCI

-  Manipuler directement le manifest JSON `{ }`
-  Pull / Push du manifest depuis le registry
-  Réagencer les layers
-  Sans rebuild, ni même avoir besoin de `docker`



# REBASE D'IMAGI

9e8e73ef

3afff29d

old-base  
393ad826

5a7813e0

81bd62b8

new-base  
8e2a1c85

694bb4ff

9e6d86c6



# REBASE D'IMAGE

Nouvelle image en manipulant le manifest JSON

{ } json

```
{
  "layers" : [
    { "digest": "sha256:9e8e73ef"
    { "digest": "sha256:3afff29d"
    { "digest": "sha256:393ad826"
    { "digest": "sha256:5a7813e0"
    { "digest": "sha256:81bd62b8"
  ]
}
```

9e8e73ef

3afff29d

393ad826

5a7813e0

81bd62b8

{ } json

```
{
  "layers" : [
    { "digest": "sha256:9e8e73ef"
    { "digest": "sha256:3afff29d"
    { "digest": "sha256:8e2a1c85"
    { "digest": "sha256:694bb4ff"
    { "digest": "sha256:9e6d86c6"
  ]
}
```

9e8e73ef

3afff29d

8e2a1c85


694bb4ff

9e6d86c6



# CAS D'USAGES CONCRETS



- Patchs des images sans les rebuild 
- Montées de version de runtime 
- Ajout ou suppression de packages 
- Mise à jour de certificats 



# CRANE

 [google/go-containerregistry](https://github.com/google/go-containerregistry)

manipulations d'images et de registries

commandes intéressantes :

> bash

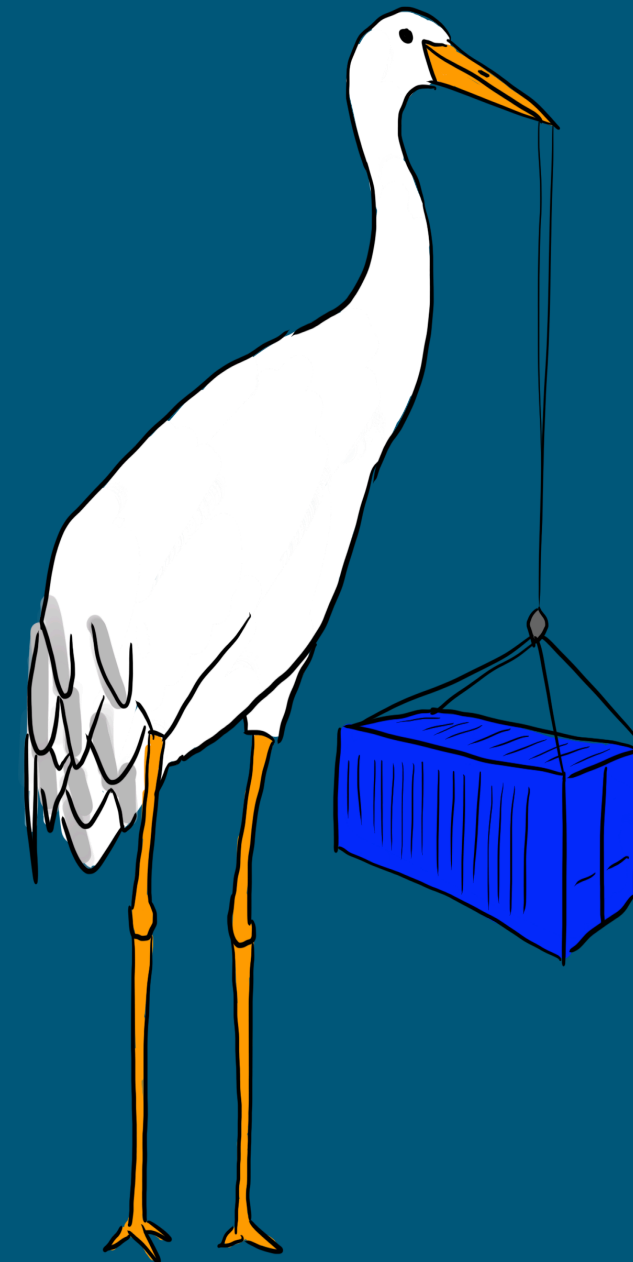
```
# récupération d'un manifest
crane manifest
# calcul du digest d'une image
crane digest
```

> bash

```
# rebase d'une image
crane rebase
```

> bash

```
# modification générale d'un manifest
crane mutate
```





# crane rebase

> bash

```
crane rebase IMAGE \  
  --old_base OLD_BASE \  
  --new_base NEW_BASE \  
  --tag NEW_TAG
```

IMAGE

9e8e73ef

3afff29d

393ad826

5a7813e0

81bd62b8

OLD\_BASE

393ad826

5a7813e0

81bd62b8

NEW\_BASE

8e2a1c85

694bb4ff

9e6d86c6

NEW\_TAG

9e8e73ef

3afff29d

8e2a1c85

694bb4ff

9e6d86c6



# DÉMO

## Rebase d'une image Java 23 -> 24

> bash

```
docker container run --rm -it $REGISTRY/java-app:jdk-23
```

> bash

```
crane manifest $REGISTRY/java-app:jdk-23 | jq ".layers[].digest"  
crane manifest $REGISTRY/eclipse-temurin:23 | jq ".layers[].digest"
```

> bash

```
crane rebase $REGISTRY/java-app:jdk-23 \  
  --old_base $REGISTRY/eclipse-temurin:23 \  
  --new_base $REGISTRY/eclipse-temurin:24 \  
  --tag $REGISTRY/java-app:jdk-24
```

> bash

```
crane manifest $REGISTRY/java-app:jdk-24 | jq ".layers[].digest"  
crane manifest $REGISTRY/eclipse-temurin:24 | jq ".layers[].digest"
```





# DÉMO

## Ajout d'un agent OpenTelemetry

> bash

```
docker container run --rm -it $REGISTRY/java-app:jdk-23
```

> bash

```
crane rebase $REGISTRY/java-app:jdk-23 \  
  --old_base $REGISTRY/eclipse-temurin:23 \  
  --new_base $REGISTRY/eclipse-temurin-with-otel:24 \  
  --tag $REGISTRY/java-app:jdk-24-otel
```

> bash

```
crane manifest $REGISTRY/java-app:jdk-24-otel | jq ".layers[].digest"
```



# POUR RIGOLER 🥰

## Détection d'une image de base

> bash

```
docker scout quickview $REGISTRY/keycloak
```

## Rebase de l'image

> bash

```
crane rebase $REGISTRY/keycloak \  
--old_base $REGISTRY/redhat/ubi9-micro:9.5 \  
--new_base $REGISTRY/ubuntu \  
--tag $REGISTRY/keycloak:ubuntu
```

## Rebase complètement pété 💥

> bash

```
crane rebase $REGISTRY/java-app:jdk-23 \  
--old_base $REGISTRY/eclipse-temurin:23 \  
--new_base $REGISTRY/node:22 \  
--tag $REGISTRY/java-app:node-22
```



```
--old_base --new_base
```

Paramètres un peu casse-pied

## ANNOTATIONS HINTS

Annotations définies dans la spec OCI : [opencontainers/image-spec#annotations.md](https://opencontainers.org/spec#annotations)

```
org.opencontainers.image.base.digest
```

```
org.opencontainers.image.base.name
```

old\_base et new\_base inférées à partir des annotations



# ANNOTATIONS HINTS

Annoter les images avec `docker image build`

> bash

```
DIGEST=$(crane digest $REGISTRY/eclipse-temurin:23)
```

> bash

```
docker image build --tag $REGISTRY/java-app:jdk-23 \  
  --annotation "org.opencontainers.image.base.name=$REGISTRY/eclipse-temurin:23" \  
  --annotation "org.opencontainers.image.base.digest=$DIGEST"
```



# DÉMO

Annoter les images avec `crane mutate`  
Annoter l'image Java 23

> bash

```
crane manifest $REGISTRY/java-app:jdk-23 | jq
```

> bash

```
DIGEST=$(crane digest $REGISTRY/eclipse-temurin:23)
crane mutate $REGISTRY/java-app:jdk-23 \
  -a "org.opencontainers.image.base.name=$REGISTRY/eclipse-temurin:23" \
  -a "org.opencontainers.image.base.digest=$DIGEST"
```

> bash

```
crane manifest $REGISTRY/java-app:jdk-23 | jq
```



# DÉMO

## Rebase simplifié

> bash

```
crane manifest $REGISTRY/java-app:jdk-23 | jq
```

> bash

```
docker image pull eclipse-temurin:23.0.2_7-jdk  
docker image tag eclipse-temurin:23.0.2_7-jdk $REGISTRY/eclipse-temurin:23  
docker image push $REGISTRY/eclipse-temurin:23
```

> bash





```
# rebase avec écrasement du tag !  
crane rebase $REGISTRY/java-app:jdk-23
```

> bash



```
crane manifest $REGISTRY/java-app:jdk-23 | jq ".layers[].digest"  
crane manifest $REGISTRY/eclipse-temurin:23.0.2_7-jdk | jq ".layers[].digest"
```



# LIMITES

- aucune garantie de fonctionnement des images 
- reports à effectuer sur les Dockerfile 
- impossible de couper à partir d'une layer arbitraire 
- cherry-pick ? 

# BONNES PRATIQUES

- toujours re-tagguer les images 
- tester les images rebasées  (avec Container Structure Test)



25

# MERCI !



**Julien Wittouck**

 @codeka.io

 julien-wittouck