SegmentFault D-Day 2021 · 开源开放与新技术创新

容器镜像构建原理和最佳实践

张晋涛 @ API7.AI

- @张晋涛
- Apache APISIX committer
- Kubernetes ingress-nginx reviewer
- 『K8S 生态周报』发起人
- GitHub: tao12345666333

Agenda

- 容器镜像是什么
- 容器镜像构建原理
- 容器镜像构建最佳实践
- 多样的容器镜像构建方案

容器镜像是什么

- manifest
- layers

```
➜  ~ docker pull debian
Using default tag: latest
latest: Pulling from library/debian
bb7d5a84853b: Pull complete
Digest: sha256:4d6ab716de467aad58e91b1b720f0badd7478847ec7a18f66027d0f8a329a43c
Status: Downloaded newer image for debian:latest
docker.io/library/debian:latest
➜  ~ mkdir -p debian-image
➜  ~ docker image save -o debian-image/debian.tar debian
➜  ~ ls debian-image
debian.tar
➜  ~ tar -C debian-image -xf debian-image/debian.tar
➜  ~ tree -I debian.tar debian-image
debian-image
├── 0d587dfbc4f4800bfe9ab08662e8396ffc37060c493f8ef24b2823fef3320df6.json
├── 3c6848d3d983bc0db9c8750311dcc9b9b5efcd71c084a8ffc1fea7ba6b3d9805
│   ├── json
│   ├── layer.tar
│   └── VERSION
├── 53b67ec39af0bd928c4a92be63ffc2c0341914b8092a4db051e3abeb34e48414
│   ├── json
│   ├── layer.tar
│   └── VERSION
├── f776cfb21b5e06bb5b4883eb15c09ab928a411476b8275293c7f96d09b90f7f9.json
├── manifest.json
└── repositories

2 directories, 10 files
```

容器镜像是什么


```
➜  ~ cat debian-image/manifest.json | jq
[
  {
    "Config": "f776cfb21b5e06bb5b4883eb15c09ab928a411476b8275293c7f96d09b90f7f9.json",
    "RepoTags": [
      "debian:latest"
    ],
    "Layers": [
      "3c6848d3d983bc0db9c8750311dcc9b9b5efcd71c084a8ffc1fea7ba6b3d9805/layer.tar"
    ]
  },
  {
    "Config": "0d587dfbc4f4800bfe9ab08662e8396ffc37060c493f8ef24b2823fef3320df6.json",
    "RepoTags": null,
    "Layers": [
      "53b67ec39af0bd928c4a92be63ffc2c0341914b8092a4db051e3abeb34e48414/layer.tar"
    ]
  }
]
```

容器镜像是什么


```
→  ~ cat debian-image/f776cfb21b5e06bb5b4883eb15c09ab928a411476b8275293c7f96d09b90f7f9.json | jq -r  '. |
{rootfs: .rootfs, history: .history}'
{
  "rootfs": {
    "type": "layers",
    "diff_ids": [
      "sha256:62a747bf1719d2d37fff5670ed40de6900a95743172de1b4434cb019b56f30b4"
    ]
  },
  "history": [
    {
      "created": "2021-10-12T01:20:30.273959207Z",
      "created_by": "/bin/sh -c #(nop) ADD file:aea313ae50ce6474a3df142b34d4dcba4e7e0186ea6fe55389cb2ea903b9ebbb
in / "
    },
    {
      "created": "2021-10-12T01:20:30.89167925Z",
      "created_by": "/bin/sh -c #(nop)  CMD [\"bash\"]",
      "empty_layer": true
    }
  ]
}
```

- 为启动容器提供必要的文件
- 记录各层的操作
- 记录各层的配置

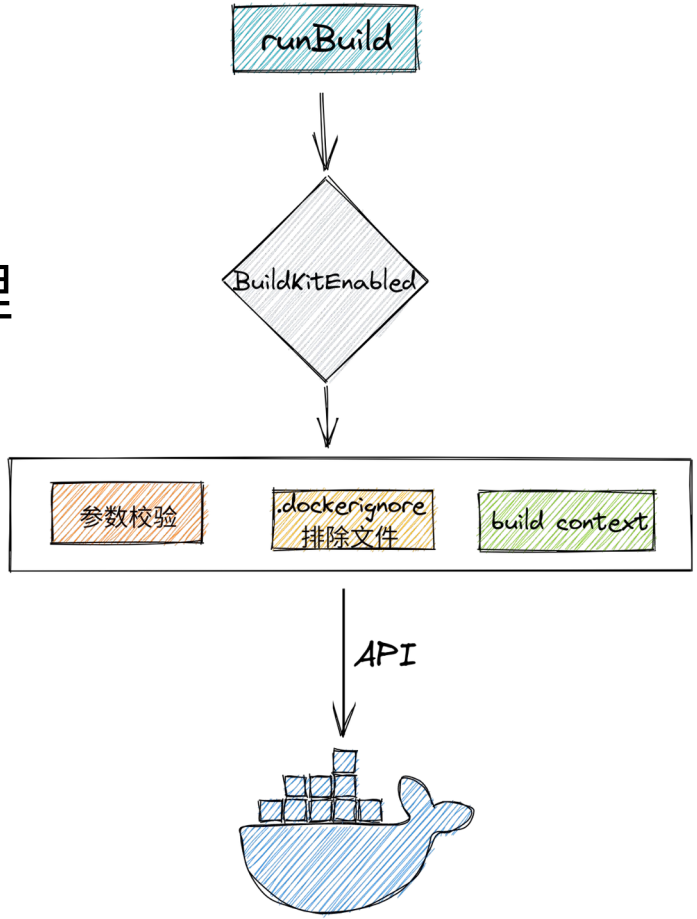
容器镜像构建原理

- 从容器构建
  - 简单，直接
  - 不可追溯
- 从 Dockerfile 构建
  - 可追溯
  - 可编程

镜像构建过程和原理


```
(MoeLove) ➜  x ls
bar  Dockerfile  foo
(MoeLove) ➜  x DOCKER_BUILDKIT=0 docker build -f Dockerfile .
Sending build context to Docker daemon  15.41kB
Step 1/3 : FROM scratch
 --->
Step 2/3 : COPY foo foo
 ---> a2af45d66bb5
Step 3/3 : COPY bar bar
 ---> cc803c675dd2
Successfully built cc803c675dd2
(MoeLove) ➜  x docker history cc803c675dd2
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
5cc4fd13e9ea   38 seconds ago   /bin/sh -c #(nop) COPY file:4eb5c5901a63f9ab…   4B
de77c607fbbc   40 seconds ago   /bin/sh -c #(nop) COPY file:8d7ea209a266ec18…   4B
```

- C/S 架构
- dockerd 进行实际构建
- docker CLI 进行前置处理



容器镜像构建最佳实践

- 利用缓存

```
FROM debian

RUN apt update
RUN apt install -y openjdk-8-jdk

COPY . /app

CMD [ "java", "-jar", "/app/target/gs-spring-boot-0.1.0.jar" ]
```

- 部分拷贝

```
FROM debian

RUN apt update
RUN apt install -y openjdk-8-jdk

COPY target/gs-spring-boot-0.1.0.jar /app/

CMD [ "java", "-jar", "/app/gs-spring-boot-0.1.0.jar" ]
```

慎用包管理器


```
FROM debian

RUN apt update && apt install -y openjdk-8-jdk

COPY target/gs-spring-boot-0.1.0.jar /app/

CMD [ "java", "-jar", "/app/gs-spring-boot-0.1.0.jar" ]
```

```
FROM debian

RUN apt update && apt install -y --no-install-recommends openjdk-8-jdk

COPY target/gs-spring-boot-0.1.0.jar /app/

CMD [ "java", "-jar", "/app/gs-spring-boot-0.1.0.jar" ]
```

```
root@5a23eb858163:/# apt install  --no-install-recommends openjdk-8-jdk | grep 'additional disk space will be
used'
...
After this operation, 344 MB of additional disk space will be used.
^C
root@5a23eb858163:/# apt install openjdk-8-jdk | grep 'additional disk space will be used'
...
After this operation, 548 MB of additional disk space will be used.
^C
```

清理包管理器缓存

```
(MoeLove) ➜  docker run --rm -it debian
root@cd857c3ab882:/# apt -qq  update
All packages are up to date.
root@cd857c3ab882:/# du -sh /var/lib/apt/lists/
16M     /var/lib/apt/lists/
root@cd857c3ab882:/#
```

```
FROM debian

RUN apt update && apt install -y --no-install-recommends openjdk-8-jdk \
        && rm -rf /var/lib/apt/lists/*

COPY target/gs-spring-boot-0.1.0.jar /app/

CMD [ "java", "-jar", "/app/gs-spring-boot-0.1.0.jar" ]
```

- 选择合适基础镜像
- 易用性
- 安全性
- 性能

- 保持构建环境一致性
- 步骤分离

```
FROM maven:3.6.1-jdk-8-alpine

WORKDIR /app

COPY pom.xml /app/
COPY src /app/src

RUN mvn -e -B package

CMD [ "java", "-jar", "/app/target/gs-spring-boot-0.1.0.jar" ]
```

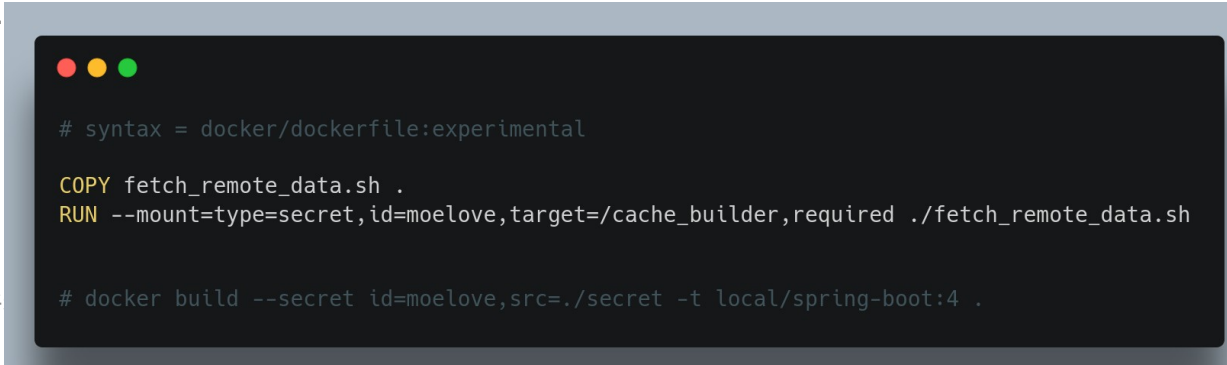
```
FROM maven:3.6.1-jdk-8-alpine

WORKDIR /app

COPY pom.xml /app/
RUN mvn dependency:go-offline
COPY src /app/src

RUN mvn -e -B package

CMD [ "java", "-jar", "/app/target/gs-spring-boot-0.1.0.jar" ]
```

➢ 多阶段构建

```
FROM maven:3.6.1-jdk-8-alpine AS builder

WORKDIR /app

COPY pom.xml /app/
RUN mvn dependency:go-offline
COPY src /app/src
RUN mvn -e -B package

FROM builder AS dev

RUN apk add --no-cache vim

FROM openjdk:8-jre-alpine

COPY --from=builder /app/target/gs-spring-boot-0.1.0.jar /

CMD [ "java", "-jar", "/gs-spring-boot-0.1.0.jar" ]
```

```
FROM maven:3.6.1-jdk-8-alpine AS builder

WORKDIR /app

COPY pom.xml /app/
RUN mvn dependency:go-offline
COPY src /app/src
RUN mvn -e -B package

FROM openjdk:8-jre-alpine

COPY --from=builder /app/target/gs-spring-boot-0.1.0.jar /

CMD [ "java", "-jar", "/gs-spring-boot-0.1.0.jar" ]
```

- 不推荐
  - 将密码硬编码写入代码中
- 一般做法
  - 通过环境变量的方式构建
- 推荐
  - Buildkit 挂载特性



```
# syntax = docker/dockerfile:experimental

COPY fetch_remote_data.sh .
RUN --mount=type=secret,id=moelove,target=/cache_builder,required ./fetch_remote_data.sh


# docker build --secret id=moelove,src=./secret -t local/spring-boot:4 .
```

➢ 挂载 SSH 密钥

```
# syntax = docker/dockerfile:experimental
FROM alpine

# 安装必要的包
RUN apk add --no-cache git openssh-client

# 创建必要的目录 .ssh 由于要使用 ssh 连接，所以需要使用 ssh-keyscan 先获取 public SSH host key
# 当然也可以给 .ssh/config 写配置文件来跳过验证，但容易带来安全问题，不推荐
RUN mkdir -p -m 0700 ~/.ssh && ssh-keyscan github.com >> ~/.ssh/known_hosts

# clone 私有项目仓库，并创建分支
RUN --mount=type=ssh,required git clone git@github.com:tao12345666333/moe.git \
        && cd moe \
        && git checkout -b release
```

```
(MoeLove) ➜  d eval $(ssh-agent)
Agent pid 28184
(MoeLove) ➜  d ssh-add ~/.ssh/id_rsa
Enter passphrase for /home/tao/.ssh/id_rsa:
Identity added: /home/tao/.ssh/id_rsa (/home/tao/.ssh/id_rsa)
(MoeLove) ➜  d docker build --ssh=default -t local/ssh .
[+] Building 0.5s (10/10) FINISHED
 => [internal] load build definition from Dockerfile                                0.1s
 => => transferring dockerfile: 96B                                                 0.0s
 => [internal] load .dockerignore                                                   0.1s
 => => transferring context: 2B                                                     0.0s
 => resolve image config for docker.io/docker/dockerfile:experimental               0.0s
 => CACHED docker-image://docker.io/docker/dockerfile:experimental                  0.0s
 => [internal] load metadata for docker.io/library/alpine:latest                    0.0s
 => [1/4] FROM docker.io/library/alpine                                             0.0s
 => CACHED [2/4] RUN apk add --no-cache git openssh-client                          0.0s
 => CACHED [3/4] RUN mkdir -p -m 0700 ~/.ssh && ssh-keyscan github.com >> ~/.ssh/known_hosts  0.0s
 => CACHED [4/4] RUN --mount=type=ssh,required git clone git@github.com:tao12345666333/moe.git  0.0s
 => exporting to image                                                              0.0s
 => => exporting layers                                                             0.0s
 => => writing image sha256:35d3ded5595a48de50054121feed13ebadf9b5e73b6cefeeba4215e1a20a20fd  0.0s
 => => naming to docker.io/local/ssh
```

多样的容器镜像构建方案

- BuildKit
  - 可以无特权运行
  - 可扩展的前端格式，支持 Dockerfile
  - 多样的输出格式
  - 插件化架构
  - 高效的缓存管理
  - 并发优化

- kaniko
  - Dockerfile 兼容
  - 可运行在多种环境（包括， K8s 的 Pod 内）

- Buildah
  - Dockerfile 兼容
  - OCI 镜像构建
  - 无需特权

- 容器镜像是什么
- 容器镜像构建原理
- 容器镜像构建最佳实践
- 多样的容器镜像构建方案

THANK YOU

QUESTIONS?

CONTACT ME： zhangjintao@apache.org