Analyse du contenu d'une image

Enoncé

1. Télécharger l'image *mongo:3.6* en local
2. Quelles sont les différentes étapes de constructions de l'image

Comparez ces étapes avec le contenu du Dockerfile utilsé pour builder cette image.

3. Inspectez l'image
4. En utilisant la notation Go template, listez les ports exposés
5. Exportez l'image *mongo:3.6* dans un tar

- Extrayez le contenu de cette archive avec la commande *tar -xvf*, qu'observez-vous ?
- Extrayez le contenu d'une des layers, qu'observez-vous ?

3. Supprimez l'image *mongo:3.6*

Correction

1. La commande suivante permet de téléchargez l'image *mongo:3.6* en local

```
$ docker image pull mongo:3.6
```

2. Les étapes de constructions de l'image peuvent être visualisées avec la sous-commande *history*:

```
$ docker image history mongo:3.6
IMAGE               CREATED             CREATED BY
SIZE                COMMENT
b0389c0d7ab3        10 days ago         /bin/sh -c #(nop)  CMD ["mongod"]
0B
<missing>           10 days ago         /bin/sh -c #(nop)  EXPOSE 27017
0B
<missing>           10 days ago         /bin/sh -c #(nop)  ENTRYPOINT ["docker-
```

```
entry…   0B
<missing>           10 days ago         /bin/sh -c #(nop) COPY
file:aede91d254349505…   10.5kB
<missing>           10 days ago         /bin/sh -c #(nop)  VOLUME [/data/db
/data/co…   0B
<missing>           10 days ago         /bin/sh -c mkdir -p /data/db
/data/configdb …   0B
<missing>           10 days ago         /bin/sh -c set -x  && apt-get update  &&
apt…   276MB
<missing>           10 days ago         /bin/sh -c echo "deb
http://$MONGO_REPO/apt/…   68B
<missing>           10 days ago         /bin/sh -c #(nop)  ENV
MONGO_VERSION=3.6.9      0B
<missing>           10 days ago         /bin/sh -c #(nop)  ENV MONGO_MAJOR=3.6
0B
<missing>           10 days ago         /bin/sh -c #(nop)  ENV
MONGO_PACKAGE=mongodb…   0B
<missing>           10 days ago         /bin/sh -c #(nop)  ARG
MONGO_REPO=repo.mongo…   0B
<missing>           10 days ago         /bin/sh -c #(nop)  ARG
MONGO_PACKAGE=mongodb…   0B
<missing>           10 days ago         /bin/sh -c set -ex;  export
GNUPGHOME="$(mkt…   1.74kB
<missing>           10 days ago         /bin/sh -c #(nop)  ENV
GPG_KEYS=2930ADAE8CAF…   0B
<missing>           10 days ago         /bin/sh -c mkdir /docker-entrypoint-
initdb.d    0B
<missing>           10 days ago         /bin/sh -c set -ex;   apt-get update;
apt-g…   11.8MB
<missing>           10 days ago         /bin/sh -c #(nop)  ENV
JSYAML_VERSION=3.10.0    0B
<missing>           10 days ago         /bin/sh -c #(nop)  ENV GOSU_VERSION=1.10
0B
<missing>           10 days ago         /bin/sh -c set -eux;  apt-get update;
apt-g…   8.51MB
<missing>           10 days ago         /bin/sh -c groupadd -r mongodb &&
useradd -r…   329kB
<missing>           10 days ago         /bin/sh -c #(nop)  CMD ["bash"]
0B
<missing>           10 days ago         /bin/sh -c #(nop) ADD
file:6d6f6f123e45697d3…   55.3MB
```

Si l'on regarde le Dockerfile utilisé pour la création de l'image (https://github.com/docker-library/mongo/blob/cac8a53d000f9e9f537438b976b719ad1b5bad3c/3.6/Dockerfile), on peut voir que chaque entrée de la commande précédente correspond à une instruction du Dockerfile.

3. La commande suivante permet d'obtenir l'ensemble des informations de l'image

```
$ docker image inspect mongo:3.6
[
```

```
    {
        "Id":
"sha256:b0389c0d7ab339f37863b9278cee754d7f6ac2c283d9825a3841cf4cac84c81d",
        "RepoTags": [
            "mongo:3.6"
        ],
        "RepoDigests": [

"mongo@sha256:5fe6e4f950b783cffb5ff4bba40d4c900ba0a68e87eb69e9c49c88f9316f7fe5"
        ],
        "Parent": "",
        "Comment": "",
        "Created": "2018-12-29T00:56:39.026981017Z",
        "Container":
"6aa57f2648513d1abc7b76dea62771992de8f88d3a43ac978ff49a29a4d1df54",
        "ContainerConfig": {
            "Hostname": "6aa57f264851",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "ExposedPorts": {
                "27017/tcp": {}
            },
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [

"PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "GOSU_VERSION=1.10",
                "JSYAML_VERSION=3.10.0",
                "GPG_KEYS=2930ADAE8CAF5059EE73BB4B58712A2291FA4AD5",
                "MONGO_PACKAGE=mongodb-org",
                "MONGO_REPO=repo.mongodb.org",
                "MONGO_MAJOR=3.6",
                "MONGO_VERSION=3.6.9"
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "#(nop) ",
                "CMD [\"mongod\"]"
            ],
            "ArgsEscaped": true,
            "Image":
"sha256:14a7d71e30ec0ffedbc9198092de253d5612554617a2524072332ab69c641f64",
            "Volumes": {
                "/data/configdb": {},
                "/data/db": {}
            },
            "WorkingDir": "",
            "Entrypoint": [
                "docker-entrypoint.sh"
            ],
```

```
            "OnBuild": null,
            "Labels": {}
        },
        "DockerVersion": "18.06.1-ce",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "ExposedPorts": {
                "27017/tcp": {}
            },
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
"PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "GOSU_VERSION=1.10",
                "JSYAML_VERSION=3.10.0",
                "GPG_KEYS=2930ADAE8CAF5059EE73BB4B58712A2291FA4AD5",
                "MONGO_PACKAGE=mongodb-org",
                "MONGO_REPO=repo.mongodb.org",
                "MONGO_MAJOR=3.6",
                "MONGO_VERSION=3.6.9"
            ],
            "Cmd": [
                "mongod"
            ],
            "ArgsEscaped": true,
            "Image":
"sha256:14a7d71e30ec0ffedbc9198092de253d5612554617a2524072332ab69c641f64",
            "Volumes": {
                "/data/configdb": {},
                "/data/db": {}
            },
            "WorkingDir": "",
            "Entrypoint": [
                "docker-entrypoint.sh"
            ],
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 351694343,
        "VirtualSize": 351694343,
        "GraphDriver": {
            "Data": {
                "LowerDir":
"/var/lib/docker/overlay2/51a62829053651f62baba3471b4b846435d5052f773d2e31787e97539
                "MergedDir":
```

```
    "/var/lib/docker/overlay2/d56e6c030760c5f94281fb7ba6d2a11451a218eb082e72e84a406dc1f
                "UpperDir":
    "/var/lib/docker/overlay2/d56e6c030760c5f94281fb7ba6d2a11451a218eb082e72e84a406dc1f
                "WorkDir":
    "/var/lib/docker/overlay2/d56e6c030760c5f94281fb7ba6d2a11451a218eb082e72e84a406dc1f
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [
    "sha256:7b4e562e58dcb7fbe1e27bb274f0ff8bfeb2fd965203380436e159df9f218900",
    "sha256:80e1e737ddc7c63e673d5dbd509bf280dd5cf9620bc2089788897d39b9f82bba",
    "sha256:0919ea181fd688e90d51cd8dbfa732b043812521586f9a1265ea6b55815caca7",
    "sha256:d1a1ebd36f8e94ed8bde67a0111e5f14c896724caa1eb713903b3161c9cbbfe7",
    "sha256:7537b2900a9754dfb0449d02938adb8c63790d6e5e5c0e60979876e6d198cacf",
    "sha256:74a50a517c42a003ddb900805139f6a76c74f1e28f351daafa477362b2306608",
    "sha256:b7bf8f515235af39f615b6b359be94c3cd67982cee6ad2ec96b1422ba0677c67",
    "sha256:7d9786c7a804f743b969238ab0931207d18f537bd4d9b294ff1bef93864db9b3",
    "sha256:b6c14e542ccdae0aff48034501ecf48c0e5a7c3585283e270a9789880a675ff0",
    "sha256:98bb570d574a06ee34dfe5c5b9297e1d8cc34ed92c5900917e404921d2df349d"
            ]
        },
        "Metadata": {
            "LastTagTime": "0001-01-01T00:00:00Z"
        }
    }
]
```

> l'image *mongo:3.6* ayant souvent des mises à jour mineures, il est possible que le résultat que vous obtenez soit différent de celui présenté si dessus, notamment en ce qui concerne les sha256 des layers constituant l'image.

Cette commande donne énormément d'information, il est souvent utile d'utiliser les Go templates pour extraire seulement l'information dont on a besoin.

4. En utilisant la notation Go template, la liste des ports exposés peut être obtenue avec la

commande suivante

```
$ docker image inspect --format '{{ json .ContainerConfig.ExposedPorts }}'
mongo:3.6 | jq .
{
  "27017/tcp": {}
}
```

5. Une image peut être exportée dans un fichier tar, la commande suivante permet de créer le fichier d'archive *mongo-3.6.tar* à partir de l'image *mongo:3.6*

```
$ docker save -o mongo-3.6.tar mongo:3.6
```

La commande suivante permet d'ouvrir cette archive et de voir son contenu

```
$ tar -xvf mongo-3.6.tar
01af69ae1508e511c43863a36fde7c4af3e3921e3642014dca94176c25f75640/
01af69ae1508e511c43863a36fde7c4af3e3921e3642014dca94176c25f75640/VERSION
01af69ae1508e511c43863a36fde7c4af3e3921e3642014dca94176c25f75640/json
01af69ae1508e511c43863a36fde7c4af3e3921e3642014dca94176c25f75640/layer.tar
03c80614eba044e8a9d66593444a2f4fffb8104c0c485cb7906d41da54318fd4/
03c80614eba044e8a9d66593444a2f4fffb8104c0c485cb7906d41da54318fd4/VERSION
03c80614eba044e8a9d66593444a2f4fffb8104c0c485cb7906d41da54318fd4/json
03c80614eba044e8a9d66593444a2f4fffb8104c0c485cb7906d41da54318fd4/layer.tar
2a74bf2c1137a78c4895926fb93d4c7bbd505cd2e57263cb8d212044b5f6b0ee/
2a74bf2c1137a78c4895926fb93d4c7bbd505cd2e57263cb8d212044b5f6b0ee/VERSION
2a74bf2c1137a78c4895926fb93d4c7bbd505cd2e57263cb8d212044b5f6b0ee/json
2a74bf2c1137a78c4895926fb93d4c7bbd505cd2e57263cb8d212044b5f6b0ee/layer.tar
38ab4beb01c9880d13e8b66fd2cc4658eb9e153874911c1d42dc2af0f1efe795/
38ab4beb01c9880d13e8b66fd2cc4658eb9e153874911c1d42dc2af0f1efe795/VERSION
38ab4beb01c9880d13e8b66fd2cc4658eb9e153874911c1d42dc2af0f1efe795/json
38ab4beb01c9880d13e8b66fd2cc4658eb9e153874911c1d42dc2af0f1efe795/layer.tar
3b670de8f89dba660888508fd86fa06157933659ba1c5fb9d5ddd9e7df9d8ebd/
3b670de8f89dba660888508fd86fa06157933659ba1c5fb9d5ddd9e7df9d8ebd/VERSION
3b670de8f89dba660888508fd86fa06157933659ba1c5fb9d5ddd9e7df9d8ebd/json
3b670de8f89dba660888508fd86fa06157933659ba1c5fb9d5ddd9e7df9d8ebd/layer.tar
43eeaad8c2057368e27b3ec3831709a65dbf92ad68ba45848c66dfb4e78afa96/
43eeaad8c2057368e27b3ec3831709a65dbf92ad68ba45848c66dfb4e78afa96/VERSION
43eeaad8c2057368e27b3ec3831709a65dbf92ad68ba45848c66dfb4e78afa96/json
43eeaad8c2057368e27b3ec3831709a65dbf92ad68ba45848c66dfb4e78afa96/layer.tar
7022b9486a7d823c9b9dcfb35272f5e83e10b4731d1b3d1758dbbdf9a9149b89/
7022b9486a7d823c9b9dcfb35272f5e83e10b4731d1b3d1758dbbdf9a9149b89/VERSION
7022b9486a7d823c9b9dcfb35272f5e83e10b4731d1b3d1758dbbdf9a9149b89/json
7022b9486a7d823c9b9dcfb35272f5e83e10b4731d1b3d1758dbbdf9a9149b89/layer.tar
b0389c0d7ab339f37863b9278cee754d7f6ac2c283d9825a3841cf4cac84c81d.json
cb07db998e70caf570e4891b495b72d713ff6bc8af704c3a4b7482d8854c3bba/
cb07db998e70caf570e4891b495b72d713ff6bc8af704c3a4b7482d8854c3bba/VERSION
```

```
cb07db998e70caf570e4891b495b72d713ff6bc8af704c3a4b7482d8854c3bba/json
cb07db998e70caf570e4891b495b72d713ff6bc8af704c3a4b7482d8854c3bba/layer.tar
cc440668afd949c6b89be513757db5ff179206ab53d85810aaa8ee0ae9e965d1/
cc440668afd949c6b89be513757db5ff179206ab53d85810aaa8ee0ae9e965d1/VERSION
cc440668afd949c6b89be513757db5ff179206ab53d85810aaa8ee0ae9e965d1/json
cc440668afd949c6b89be513757db5ff179206ab53d85810aaa8ee0ae9e965d1/layer.tar
feec457e2115a95ab951645e36eec9816415ee557b856f6d1251597a18d211d9/
feec457e2115a95ab951645e36eec9816415ee557b856f6d1251597a18d211d9/VERSION
feec457e2115a95ab951645e36eec9816415ee557b856f6d1251597a18d211d9/json
feec457e2115a95ab951645e36eec9816415ee557b856f6d1251597a18d211d9/layer.tar
manifest.json
repositories
```

On peut remarquer ici que cette archive est un ensemble de layer, pour chacune d'entre elle on a un répertoire contenant:

- une archive (fichier layer.tar)
- un fichier json
- un fichier VERSION

Si l'on extrait le contenu de l'une de ces layers, on obtient un système de fichier. L'ensemble de ces layers, et donc de ces systèmes de fichiers, constitue le système de fichier global de l'image.

Exemple de contenu de l'une des layers de l'image:

```
$ tar -xvf
7022b9486a7d823c9b9dcfb35272f5e83e10b4731d1b3d1758dbbdf9a9149b89/layer.tar
bin/
bin/bash
bin/cat
bin/chacl
bin/chgrp
bin/chmod
bin/chown
bin/cp
bin/dash
bin/date
bin/dd
bin/df
bin/dir
bin/dmesg
...
```

6. L'image peut être supprimée avec la commande suivante

```
$ docker image rm mongo:3.6
Untagged: mongo:3.6
```