Docker Registry Yönetimi

Hakan Özler
@ozlerhakan
hakan.ozler@kodcu.com

docker

docker SWARM

docker COMPOSE

docker DOCKER DATACENTER

docker DOCKER CLOUD

docker REGISTRY

Google

IBM

amazon

redhat

Microsoft

Distribution aka Registry 2.0

- Open-source Go Lang implementation
  - https://github.com/docker/distribution
- Image store & distribution service
  - allows to build your own internal registry
- Much like a Git repository
- An image repository consists of
  - Images
  - Tags
  - Layers
  - Metadata
- Registry API v2

Glossary

digest	a content addressable image identifier, sha256 based digest
manifest	a signature of an image, describes the components of an image
repository	contains a bunch of images and metadata
image	build-time construct, combination of layers
tag	identifier of an image, alias
layer	an image, a file, part of an image
blob	binary large object, an image layer

Public Registry Services

Docker's default registry, Docker Hub https://hub.docker.com/

Docker EE - Docker Store - Docker Trusted Registry

Quay https://quay.io/

JFrog Artifactory https://www.jfrog.com/artifactory/

Amazon EC2 Container Registry https://aws.amazon.com/ecr/

https://git.io/v5ppZ

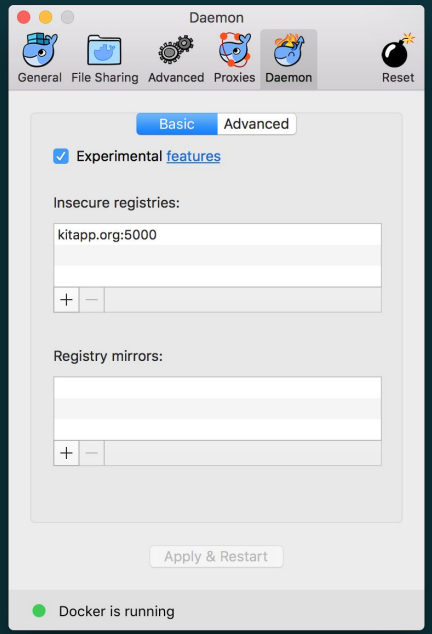
#1 Personal Usage

```
# docker-compose-registry-step-1.yaml
version: '3.2'
services:
   registry:
      restart: always
      image: registry:2
      ports:
       - 5000:5000 # insecure centralize
      volumes:
       - type: bind
         source: ./datastore/first-db-path
         target: /var/lib/registry

$ docker-compose -f docker-compose-registry-step-1.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-1.yaml down -v
```

Access Insecure Registries

```
$ /usr/bin/dockerd \
    --insecure-registry kitapp.org:5000 \
    ...
# in daemon.json
{
  "insecure-registries" : [
    "kitapp.org:5000"
  ]
}
$ docker info
...
Insecure Registries:
 kitapp.org:5000
 127.0.0.0/8
```



GET /v2/

```
# using httpie
$ http --print b example.org:5000/v2/
# using cURL
$ curl -s example.org:5000/v2/ \
       -H "Content-Type: application/json" | jq
HTTP/1.1 200 OK
…
Docker-Distribution-Api-Version: registry/2.0

{}
```

GET /v2/_catalog

```
# using httpie
$ http --print b example.org:5000/v2/_catalog
# using cURL
$ curl -s example.org:5000/v2/_catalog \
       -H "Content-Type: application/json" | jq
{
    "repositories": [
        "alpine",
        "ubuntu"
    ]
}
```

HEAD /v2/<name>/manifests/<reference>

```
$ http --print h HEAD example.org:5000/v2/alpine/manifests/latest \
Accept:application/vnd.docker.distribution.manifest.v2+json
HTTP/1.1 200 OK

Content-Type: application/vnd.docker.distribution.manifest.v1+prettyjws
…
Docker-Content-Digest:
sha256:2bd98bd132f6a3ef3ffbc1f0f6a0d9a65d7930d5dd879af25bddf7628ade61b5
Docker-Distribution-Api-Version: registry/2.0
…
Etag: "sha256:2bd98bd132f6a3ef3ffbc1f0f6a0d9a65d7930d5dd879af25bddf7628ade61b5"
```

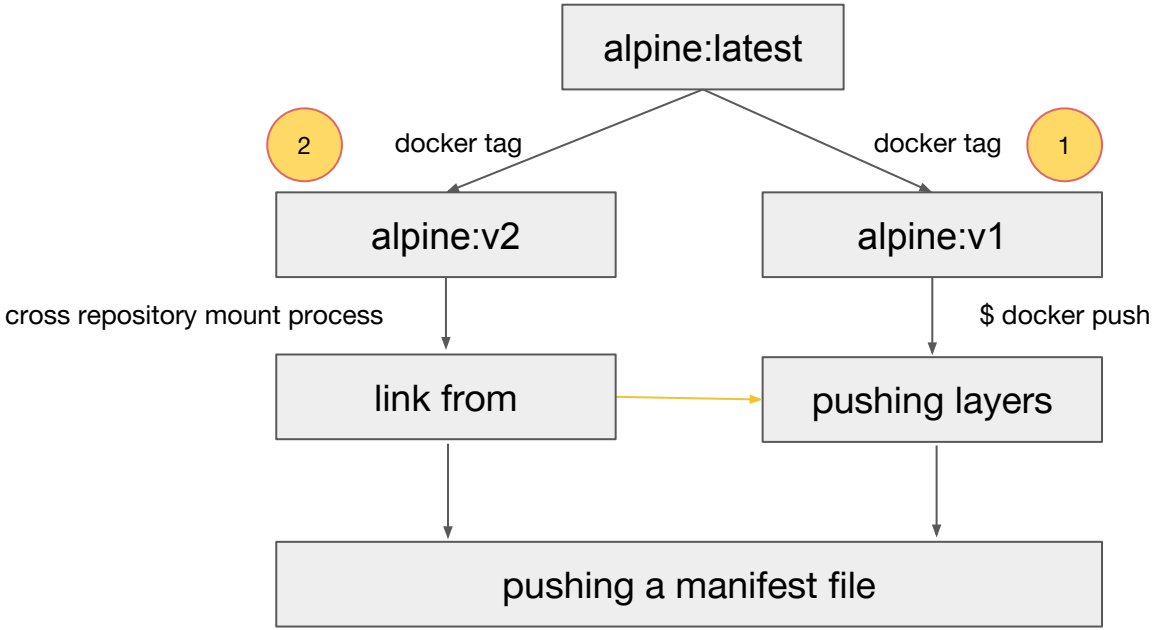
GET /v2/<name>/manifests/<reference>

```
$ http example.org:5000/v2/alpine/manifests/latest \
Accept:application/vnd.docker.distribution.manifest.v2+json
HTTP/1.1 200 OK
Content-Type: application/vnd.docker.distribution.manifest.v2+json
Docker-Content-Digest:
sha256:2bd98bd132f6a3ef3ffbc1f0f6a0d9a65d7930d5dd879af25bddf7628ade61b5
{
     "config": {
     "digest":
"sha256:7328f6f8b41890597575cbaadc884e7386ae0acc53b747401ebce5cf0d624560",
        "mediaType": "application/vnd.docker.container.image.v1+json",
        "size": 1520
    },
     …
}
```

GET /v2/<name>/blobs/<reference> (docker pull)

```
$ http example.org:5000/v2/alpine/manifests/latest \
Accept:application/vnd.docker.distribution.manifest.v2+json
{
    ...
    "layers": [
        {
            "Digest": "sha256:6d987f6f42797d81a318c40d442369ba3dc124883a0964d40b0c8f4f7561d913",
            "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
            "size": 1990402 # 1.9MB
        }
    ],
      ...
}
$ curl -s \
example.org:5000/v2/alpine/blobs/sha256:6d987f6f42797d81a318c40d442369ba3dc124883a0964d40b0c8f4f7561d913
> alpine.latest.tar
$ docker import alpine.latest.tar alpine:tar
```

POST /v2/<name>/blobs/uploads/ (docker push)



https://docs.docker.com/registry/spec/api/#pushing-an-image

GET /v2/<name>/tags/list

```
# using httpie
$ http --print b example.org:5000/v2/alpine/tags/list
# using cURL
$ curl -s example.org:5000/v2/alpine/tags/list \
       -H "Content-Type: application/json" | jq
{

    "name": "alpine",
    "tags": [
        "latest"
    ]
}
```

DELETE /v2/<name>/manifests/<digest>

```
# step 2
$ docker run … \
-e REGISTRY_STORAGE_DELETE_ENABLED=true … registry:2
# in /etc/docker/registry/config.yml
version: 0.1
…
storage:
  delete:
    enabled: true
…
$ http DELETE \
<IP>:5000/v2/alpine/manifest/sha256:2bd98bd132f6a3ef3ffbc1f0f6a0d9a65d7930d5dd879af
25bddf7628ade61b5
HTTP/1.1 202 Accepted
Content-Length: 0
```

Registry File System

#3 Reserve Proxy

```
# docker-compose-registry-step-3.yaml
services:
  nginx:
    build:
      context: nginx/step3/
      dockerfile: Dockerfile
…
    restart: "on-failure:5"
    depends_on: # not compatible in swarm mode
     - registry
  registry:
    restart: always
    image: registry:2
…
…
$ docker-compose -f docker-compose-registry-step-3.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-3.yaml down -v
```

#4 TLS Enabled on Nginx

```
# docker-compose-registry-step-4.yaml
$ docker-compose -f docker-compose-registry-step-4.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-4.yaml down -v

$ http --verify no --print b https://example.org/v2/_catalog
{
    "repositories": [
        "alpine"
    ]
}

$ reg ls https://example.org/
Repositories for example.org
REPO                TAGS
alpine              3.6
```

https://github.com/jessfraz/reg

#5 TLS Enabled on Registry

```
# docker-compose-registry-step-5.yaml
services:
  registry:
    …
    environment:
      REGISTRY_HTTP_TLS_CERTIFICATE: /certs/live/example.org/fullchain.pem
      REGISTRY_HTTP_TLS_KEY: /certs/live/example.org/privkey.pem
    …
    ports:
      - target: 5000
        published: 443
        protocol: tcp
        mode: host

$ docker-compose -f docker-compose-registry-step-5.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-5.yaml down -v
```

#6 Authentication on Nginx

```
$ docker run --rm -it --entrypoint htpasswd registry:2 -nB joe

# docker-compose-registry-step-6.yaml

$ docker-compose -f docker-compose-registry-step-6.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-6.yaml down -v

# accessing a basic auth and tls active registry

$ reg --username joe --password doe  -r https://example.org/ ls
Repositories for example.org
REPO                TAGS
alpine              3.6

$ curl -XGET -s https://example.org/v2/_catalog -u joe:doe

$ http --print b --auth joe:doe https://example.org/v2/_catalog
```

#7 Authentication on Registry

```
$ docker run --rm -it --entrypoint htpasswd registry:2 -nB joe
services:
  registry:
    …
    environment:
      REGISTRY_AUTH: htpasswd
      REGISTRY_AUTH_HTPASSWD_PATH: /auth/pwd
      REGISTRY_AUTH_HTPASSWD_REALM: registry.example.org

# docker-compose-registry-step-7.yaml
$ docker-compose -f docker-compose-registry-step-7.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-7.yaml down -v

$ reg --username joe --password doe  -r https://example.org tags redis
latest
```

#8 Blob Storage

```
services:
  ...
  registry:
    restart: always
    image: registry:2
    environment:
      REGISTRY_STORAGE_DELETE_ENABLED: 'true'
      REGISTRY_STORAGE: s3
      REGISTRY_STORAGE_S3_ACCESSKEY: ABCDEFGH
      REGISTRY_STORAGE_S3_SECRETKEY: asmdjanskdaj/asdnjaskda
      REGISTRY_STORAGE_S3_REGION: us-west-2
      REGISTRY_STORAGE_S3_BUCKET: registry.event
      REGISTRY_HEALTH_STORAGEDRIVER_ENABLED: 'false'

# docker-compose-registry-step-8.yaml
$ docker-compose -f docker-compose-registry-step-8.yaml up -d --build
$ docker-compose -f docker-compose-registry-step-8.yaml down
```

OS Registry Projects

https://github.com/docker/migrator

https://github.com/jessfraz/reg

https://github.com/SUSE/Portus

https://github.com/kwk/docker-registry-frontend

https://github.com/atcol/docker-registry-ui

https://github.com/mkuchin/docker-registry-web

https://github.com/klausmeyer/docker-registry-browser

Thanks! :)

Hakan Özler
@ozlerhakan
hakan.ozler@kodcu.com