docker pull

Estimated reading time: 8 minutes

Description

Pull an image or a repository from a registry

Usage

```
docker pull [OPTIONS] NAME[:TAG|@DIGEST]
```

Options

Name, shorthand	Default	Description
`--all-tags , -a`		Download all tagged images in the repository
`--disable-content-trust`	`true`	Skip image verification
`--platform`		**experimental (daemon)** (https://docs.docker.com/engine/reference/commandline/dockerd/#daemon-configuration-file) **API 1.32+** (https://docs.docker.com/engine/api/v1.32/) Set platform if server is multi-platform capable

Parent command

Command	Description
docker (https://docs.docker.com/engine/reference/commandline/docker)	The base command for the Docker CLI.

Extended description

Most of your images will be created on top of a base image from the Docker Hub (https://hub.docker.com) registry.

Docker Hub (https://hub.docker.com) contains many pre-built images that you can `pull` and try without needing to define and configure your own.

To download a particular image, or set of images (i.e., a repository), use `docker pull` .

Proxy configuration

If you are behind an HTTP proxy server, for example in corporate settings, before open a connect to registry, you may need to configure the Docker daemon's proxy settings, using the `HTTP_PROXY` , `HTTPS_PROXY` , and `NO_PROXY` environment variables. To set these environment variables on a host using `systemd` , refer to the control and configure Docker with systemd (https://docs.docker.com/engine/admin/systemd/#http-proxy) for variables configuration.

Concurrent downloads

By default the Docker daemon will pull three layers of an image at a time. If you are on a low bandwidth connection this may cause timeout issues and you may want to lower this via the `--max-concurrent-downloads` daemon option. See the daemon documentation (https://docs.docker.com/engine/reference/commandline/dockerd/) for more details.

Examples

Pull an image from Docker Hub

To download a particular image, or set of images (i.e., a repository), use `docker pull`. If no tag is provided, Docker Engine uses the `:latest` tag as a default. This command pulls the `debian:latest` image:

```
$ docker pull debian

Using default tag: latest
latest: Pulling from library/debian
fdd5d7827f33: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:e7d38b3517548a1c71e41bffe9c8ae6d6d29546ce46bf62159837aad072c90aa
Status: Downloaded newer image for debian:latest
```

Docker images can consist of multiple layers. In the example above, the image consists of two layers; `fdd5d7827f33` and `a3ed95caeb02`.

Layers can be reused by images. For example, the `debian:jessie` image shares both layers with `debian:latest`. Pulling the `debian:jessie` image therefore only pulls its metadata, but not its layers, because all layers are already present locally:

```
$ docker pull debian:jessie

jessie: Pulling from library/debian
fdd5d7827f33: Already exists
a3ed95caeb02: Already exists
Digest: sha256:a9c958be96d7d40df920e7041608f2f017af81800ca5ad23e327bc402626b58e
Status: Downloaded newer image for debian:jessie
```

To see which images are present locally, use the `docker images` (https://docs.docker.com/engine/reference/commandline/images/) command:

```
$ docker images

REPOSITORY   TAG      IMAGE ID       CREATED      SIZE
debian       jessie   f50f9524513f   5 days ago   125.1 MB
debian       latest   f50f9524513f   5 days ago   125.1 MB
```

Docker uses a content-addressable image store, and the image ID is a SHA256 digest covering the image's configuration and layers. In the example above, `debian:jessie` and `debian:latest` have the same image ID because they are actually the *same* image tagged with different names. Because they are the same image, their layers are stored only once and do not consume extra disk space.

For more information about images, layers, and the content-addressable store, refer to understand images, containers, and storage drivers (https://docs.docker.com/engine/userguide/storagedriver/imagesandcontainers/).

Pull an image by digest (immutable identifier)

So far, you've pulled images by their name (and "tag"). Using names and tags is a convenient way to work with images. When using tags, you can `docker pull` an image again to make sure you have the most up-to-date version of that image. For example, `docker pull ubuntu:14.04` pulls the latest version of the Ubuntu 14.04 image.

In some cases you don't want images to be updated to newer versions, but prefer to use a fixed version of an image. Docker enables you to pull an image by its *digest*. When pulling an image by digest, you specify *exactly* which version of an image to pull. Doing so, allows you to "pin" an image to that version, and guarantee that the image you're using is always the same.

To know the digest of an image, pull the image first. Let's pull the latest `ubuntu:14.04` image from Docker Hub:

```
$ docker pull ubuntu:14.04

14.04: Pulling from library/ubuntu
5a132a7e7af1: Pull complete
fd2731e4c50c: Pull complete
28a2f68d1120: Pull complete
a3ed95caeb02: Pull complete
Digest: sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb2
Status: Downloaded newer image for ubuntu:14.04
```

Docker prints the digest of the image after the pull has finished. In the example above, the digest of the image is:

```
sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb2
```

Docker also prints the digest of an image when *pushing* to a registry. This may be useful if you want to pin to a version of the image you just pushed.

A digest takes the place of the tag when pulling an image, for example, to pull the above image by digest, run the following command:

```
$ docker pull ubuntu@sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb2

sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb2: Pulling from library/ubuntu
5a132a7e7af1: Already exists
fd2731e4c50c: Already exists
28a2f68d1120: Already exists
a3ed95caeb02: Already exists
Digest: sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb2
Status: Downloaded newer image for ubuntu@sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f8787
6677c5cb2
```

Digest can also be used in the `FROM` of a Dockerfile, for example:

```
FROM ubuntu@sha256:45b23dee08af5e43a7fea6c4cf9c25ccf269ee113168c19722f87876677c5cb2
MAINTAINER some maintainer <maintainer@example.com>
```

> **Note**: Using this feature "pins" an image to a specific version in time. Docker will therefore not pull updated versions of an image, which may include security updates. If you want to pull an updated image, you need to change the digest accordingly.

Pull from a different registry

By default, `docker pull` pulls images from Docker Hub (https://hub.docker.com). It is also possible to manually specify the path of a registry to pull from. For example, if you have set up a local registry, you can specify its path to pull from it. A registry path is similar to a URL, but does not contain a protocol specifier (`https://`).

The following command pulls the `testing/test-image` image from a local registry listening on port 5000 (`myregistry.local:5000`):

```
$ docker pull myregistry.local:5000/testing/test-image
```

Registry credentials are managed by docker login (https://docs.docker.com/engine/reference/commandline/login/).

Docker uses the `https://` protocol to communicate with a registry, unless the registry is allowed to be accessed over an insecure connection. Refer to the insecure registries (https://docs.docker.com/engine/reference/commandline/dockerd/#insecure-registries) section for more information.

Pull a repository with multiple images

By default, `docker pull` pulls a *single* image from the registry. A repository can contain multiple images. To pull all images from a repository, provide the `-a` (or `--all-tags`) option when using `docker pull`.

This command pulls all images from the `fedora` repository:

```
$ docker pull --all-tags fedora

Pulling repository fedora
ad57ef8d78d7: Download complete
105182bb5e8b: Download complete
511136ea3c5a: Download complete
73bd853d2ea5: Download complete
....

Status: Downloaded newer image for fedora
```

After the pull has completed use the `docker images` command to see the images that were pulled. The example below shows all the `fedora` images that are present locally:

```
$ docker images fedora

REPOSITORY   TAG         IMAGE ID        CREATED      SIZE
fedora       rawhide     ad57ef8d78d7    5 days ago   359.3 MB
fedora       20          105182bb5e8b    5 days ago   372.7 MB
fedora       heisenbug   105182bb5e8b    5 days ago   372.7 MB
fedora       latest      105182bb5e8b    5 days ago   372.7 MB
```

Cancel a pull

Killing the `docker pull` process, for example by pressing `CTRL-c` while it is running in a terminal, will terminate the pull operation.

```
$ docker pull fedora

Using default tag: latest
latest: Pulling from library/fedora
a3ed95caeb02: Pulling fs layer
236608c7b546: Pulling fs layer
^C
```

> **Note**: Technically, the Engine terminates a pull operation when the connection between the Docker Engine daemon and the Docker Engine client initiating the pull is lost. If the connection with the Engine daemon is lost for other reasons than a manual interaction, the pull is also aborted.