Pulp Docker Documentation

Release 1.0.2

Pulp Team

September 11, 2015

Contents

This project adds support to Pulp for managing Docker images.

CHAPTER 1

User Guide

1.1 Installation

1.1.1 Prerequisites

The only requirement is to meet the prerequisites of the Pulp Platform. Please see the Pulp User Guide for prerequisites including repository setup.

Note: If you are installing on Fedora, you will need to install `docker-io` instead of `docker`. See the official Docker Documentation for more information.

1.1.2 Server

```
$ sudo yum install pulp-docker-plugins
```

Then run `pulp-manage-db` to initialize the new type in Pulp's database.

```
$ sudo -u apache pulp-manage-db
```

Then restart each pulp component, as documented in the Pulp User Guide.

1.1.3 Admin Client

```
$ sudo yum install pulp-docker-admin-extensions
```

1.2 Release Notes

Contents:

1.2.1 1.0 Release Notes

1.0.1

This is a minor release with some bug fixes.

Bugfixes

See the bugs fixed in 1.0.1.

1.0.0

The Pulp team is pleased to release version 1.0.0 of the Docker plugins for Pulp.

Upgrade

To upgrade, simply follow these steps (substituting for systemctl as appropriate, if you are not using systemd):

1. Stop all Pulp services on every machine that is part of the installation:

```
$ for s in {pulp_workers,pulp_resource_manager,pulp_celerybeat,httpd,goferd}; do sudo systemctl
```

2. Upgrade the Pulp packages on every machine:

```
$ sudo yum update
```

3. Apply database migrations:

```
$ sudo -u apache pulp-manage-db
```

4. Start the Pulp services:

```
$ for s in {pulp_workers,pulp_resource_manager,pulp_celerybeat,httpd,goferd}; do sudo systemctl
```

Bugfixes

See the bugs fixed in 1.0.0.

1.3 Configuration

1.3.1 Importer Configuration

The Docker importer is configured by editing `/etc/pulp/server/plugins.conf.d/docker_importer.json`. This file must be valid JSON.

The importer supports the settings documented in Pulp's importer config docs.

1.4 Concepts

1.4.1 Repository and Tags

A docker repository is a collection of images that can have tags. A pulp repository likewise is a collection of docker images. Tags are a property of the repository and can be modified with the command `pulp-admin docker repo update` and its `--tag` option.

1.4.2 Upload

An upload operation potentially includes multiple layers. When doing a `docker save`, a tarball is created with the requested repository and all of its ancestor layers. When uploading that tarball to pulp, each layer will be added to the repository as a unit. The tags will also be added to the repository, overwriting any previous tags of the same name.

1.5 Recipes

1.5.1 Configuring Crane with pulp_docker

The Crane project can be used to make docker repositories hosted by Pulp available to the docker client. This allows a `docker pull` to be performed against data that is published by the Pulp server.

If Crane is being run on the same server that is running Pulp, there is one setting that must be configured in Crane in order for it to find the information that is published by Pulp. In the /etc/crane.conf the `data_dir` parameter must be set to the location that the pulp publish is placing metadata files. By default this is the `/var/lib/pulp/published/docker/app/` directory. Crane will check the `data_dir` for updates periodically. Full documentation for /etc/crane.conf can be found in the Crane readme.

1.5.2 Upload To Pulp

To upload a docker image to pulp, first you must save its repository with docker. Note that the below command saves all of the images and tags in the `busybox` repository to a tarball:

```
$ sudo docker pull busybox
$ sudo docker save busybox > busybox.tar
```

Then create a pulp repository and run an upload command with `pulp-admin`:

```
$ pulp-admin docker repo create --repo-id=busybox
Repository [busybox] successfully created

$ pulp-admin docker repo uploads upload --repo-id=busybox -f busybox.tar
+----------------------------------------------------------------------+
                            Unit Upload
+----------------------------------------------------------------------+

Extracting necessary metadata for each request...
[==================================================] 100%
Analyzing: busybox.tar
... completed

Creating upload requests on the server...
[==================================================] 100%
Initializing: busybox.tar
... completed

Starting upload of selected units. If this process is stopped through ctrl+c,
the uploads will be paused and may be resumed later using the resume command or
canceled entirely using the cancel command.

Uploading: busybox.tar
[==================================================] 100%
2825216/2825216 bytes
```

```
... completed

Importing into the repository...
This command may be exited via ctrl+c without affecting the request.


[\]
Running...

Task Succeeded


Deleting the upload request...
... completed
```

There are now four new images in the pulp repository:

```
$ pulp-admin docker repo list
+----------------------------------------------------------------------+
                          Docker Repositories
+----------------------------------------------------------------------+

Id:                  busybox
Display Name:        busybox
Description:         None
Content Unit Counts:
  Docker Image: 4
```

During an image upload, you can specify the id of an ancestor image that should not be uploaded to the repository. In this case, the masked ancestor and any ancestors of that image will not be imported:

```
$ pulp-admin docker repo create --repo-id tutorial
Repository [tutorial] successfully created

$ pulp-admin docker repo uploads upload --repo-id tutorial
-f /home/skarmark/git/pulp1/pulp/tutorial.tar
--mask-id 'f38e479062c4953de709cc7f08fa8f85bec6bc5d01f03e340f7caf2990e8efd1'
+----------------------------------------------------------------------+
                          Unit Upload
+----------------------------------------------------------------------+

Extracting necessary metadata for each request...
[==================================================] 100%
Analyzing: tutorial.tar
... completed

Creating upload requests on the server...
[==================================================] 100%
Initializing: tutorial.tar
... completed

Starting upload of selected units. If this process is stopped through ctrl+c,
the uploads will be paused and may be resumed later using the resume command or
canceled entirely using the cancel command.

Uploading: tutorial.tar
[==================================================] 100%
353358336/353358336 bytes
... completed
```

```
Importing into the repository...
This command may be exited via ctrl+c without affecting the request.


[\]
Running...

Task Succeeded


Deleting the upload request...
... completed
```

There are now only two images imported into the pulp repository, instead of five total images in the tar file:

```
$ pulp-admin docker repo list
+----------------------------------------------------------------------+
                        Docker Repositories
+----------------------------------------------------------------------+

Id:                  tutorial
Display Name:        tutorial
Description:         None
Content Unit Counts:
    Docker Image: 2
```

1.5.3 Publish

The `busybox` repository uploaded above can be published for use with Crane.

First the docker repository name must be specified, which can be different than the `repo_id`. The repository name should usually have a namespace, a `/`, and then a name. Other than the slash between the namespace and the name, it is required that this field can contain only lower case letters, integers, hyphens, and periods. The command below sets the repository name to `pulpdemo/busybox`:

```
$ pulp-admin docker repo update --repo-id=busybox --repo-registry-id=pulpdemo/busybox
This command may be exited via ctrl+c without affecting the request.


[\]
Running...
Updating distributor: docker_web_distributor_name_cli

Task Succeeded



[\]
Running...
Updating distributor: docker_export_distributor_name_cli

Task Succeeded
```

Then a publish operation can be executed:

```
$ pulp-admin docker repo publish run --repo-id=busybox
+----------------------------------------------------------------------+
```

```
                    Publishing Repository [busybox]
+----------------------------------------------------------------------+

This command may be exited via ctrl+c without affecting the request.


Publishing Image Files.
[==================================================] 100%
4 of 4 items
... completed

Making files available via web.
[-]
... completed


Task Succeeded
```

Crane can now be run on the same machine serving the docker repository through its docker-registry-like read-only API.

1.5.4 Export

The `busybox` repository can also be exported for a case where Crane will be run on a different machine, or the image files will be hosted by another service:

```
$ pulp-admin docker repo export run --repo-id=busybox
+----------------------------------------------------------------------+
                    Publishing Repository [busybox]
+----------------------------------------------------------------------+

This command may be exited via ctrl+c without affecting the request.


Publishing Image Files.
[==================================================] 100%
4 of 4 items
... completed

Saving tar file.
[-]
... completed


Task Succeeded
```

This produces a tarball at `/var/lib/pulp/published/docker/export/repo/busybox.tar` which contains both a JSON file for use with crane, and the static image files to which crane will redirect requests. See the Crane documentation for how to use that tarball.

1.5.5 Sync

The pulp-docker plugin supports syncing from upstream repositories as of version 0.2.1. For example:

```
$ pulp-admin docker repo create --repo-id=synctest --feed=https://index.docker.io --upstream-name=bus
Repository [synctest] successfully created
```

```
$ pulp-admin docker repo sync run --repo-id synctest
+----------------------------------------------------------------------+
                    Synchronizing Repository [synctest]
+----------------------------------------------------------------------+

This command may be exited via ctrl+c without affecting the request.


Retrieving metadata
[\]
... completed

Copying units already in pulp
[-]
... completed

Downloading remote files
[-]
... completed

Saving images and tags
[-]
... completed


Task Succeeded
```

Once this is complete, the data in the remote repository is now in your local Pulp instance.

CHAPTER 2

Technical Reference

2.1 Importer

ID: `docker_importer`

2.1.1 Configuration

The following options are available to the docker importer configuration.

`mask_id` Supported only as an override config option to a repository upload command, when this option is used, the upload command will skip adding given image and any ancestors of that image to the repository.

`feed` The URL for the docker repository to import images from

`upstream_name` The name of the repository to import from the upstream repository

2.2 Distributor Configuration

2.2.1 Web Distributor

Type ID: `docker_distributor_web`

The Web distributor is used to publish a docker repository in a way that can be consumed and served by Crane directly. By default the *redirect file* is stored as `/var/lib/pulp/published/docker/app/<reponame>.json`, and the repo data itself is stored in `/var/lib/pulp/published/docker/web/<repo_id>/`.

The global configuration file for the docker_web_distributor plugin can be found in `/etc/pulp/server/plugins.conf.d/docker_distributor.json`.

All values from the global configuration can be overridden on the local config.

Supported keys

`docker_publish_directory` The publish directory used for this distributor. The web server should be configured to serve <publish_directory>/web. The default value is `/var/lib/pulp/published/docker`.

`protected` if "true" requests for this repo will be checked for an entitlement certificate authorizing the server url for this repository; if "false" no authorization checking will be done. This defaults to true.

redirect-url The server URL that will be used when generating the redirect map for connecting the docker API to the location the content is stored. The value defaults to `https://<server_name_from_pulp_server.conf>/pulp/docker/<repo_name>`.

repo-registry-id The name that should be used for the repository when it is served by Crane. If specified it will be used for the `repository` field in the *redirect file*. If a value is not specified, then repository id is used.

2.2.2 Export Distributor

Type ID: `docker_distributor_export`

The export distributor is used to save the contents of a publish into a tar file that can be moved easily for instances where Crane is running on a different server than your pulp instance. By default the *redirect file* is stored in the root of the tar file as `<reponame>.json`, and the repo data itself is stored in the `/<repo_id>/` sub directory of the tar file.

The global configuration file for the docker_export_distributor plugin can be found in `/etc/pulp/server/plugins.conf.d/docker_distributor_export.json`.

All values from the global configuration can be overridden on the local config.

Supported keys

docker_publish_directory The publish directory used for this distributor. The web server should be configured to serve <publish_directory>/export. The default value is `/var/lib/pulp/published/docker`.

export_file The fully qualified path and name of the tar file that will be created by the export. This defaults to `<docker_publish_directory>/export/repo/<repo_id>.tar`

protected if "true" requests for this repo will be checked for an entitlement certificate authorizing the server url for this repository; if "false" no authorization checking will be done.

redirect-url The URL where image files for this repository are served. Crane will join this URL with `<image_id>/<filename>`

repo-registry-id The name that should be used for the repository when it is served by Crane. If specified it will be used for the `repository` field in the *redirect file*. If a value is not specified, then repository id is used. Docker requires that this field contains only lower case letters, integers, hyphens, and periods. Additionally a single slash can be used to namespace the repo.

2.2.3 Redirect File

The distributors generate a json file with the details of the repository contents.

The file is JSON formatted with the following keys

- **type** *(string)* - the type of the file. This will always be "pulp-docker-redirect"
- **version** *(int)* - version of the format for the file. Currently version 1
- **repository** *(string)* - the name of the repository this file is describing
- **repo-registry-id** *(string)* - the name that will be used for this repository in the Docker registry
- **url** *(string)* - the url for access to the repositories content
- **protected** *(bool)* - whether or not the repository should be protected by an entitlement certificate.
- **images** *(array)* - an array of objects describing each image/layer in the repository

  - **id** *(str)* - the image id for the image

- **tags** *(obj)* - an object containing key, value paris of "tag-name":"image-id"

Example Redirect File Contents:

```
{
 "type":"pulp-docker-redirect",
 "version":1,
 "repository":"docker",
 "repo-registry-id":"redhat/docker",
 "url":"http://www.foo.com/docker",
 "protected": true,
 "images":[
   {"id":"48e5f45168b97799ad0aafb7e2fef9fac57b5f16f6db7f67ba2000eb947637eb"},
   {"id":"511136ea3c5a64f264b78b5433614aec563103b4d4702f3ba7d4d2698e22c158"},
   {"id":"769b9341d937a3dba9e460f664b4f183a6cecdd62b337220a28b3deb50ee0a02"},
   {"id":"bf747efa0e2fa9f7c691588ce3938944c75607a7bb5e757f7369f86904d97c78"}
   ],
 "tags": {"latest": "769b9341d937a3dba9e460f664b4f183a6cecdd62b337220a28b3deb50ee0a02"}
}
```

2.3 Tags

Tags on images are managed via the repository object. In the `tags` sub object of the `scratchpad` object, a list of key value pairs for each tag & Image ID are stored as shown below.

Example Repository Object:

```
{
...
 "scratchpad": {
   ...
   "tags": [
       { "tag": "latest",
         "image_id": "48e5f45168b97799ad0aafb7e2fef9fac57b5f16f6db7f67ba2000eb947637eb"}
   ]
}
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search