clusterdock Documentation

Release 1.6.0

Dima Spivak

Mar 19, 2018

Contents

---

clusterdock is a Python 3 project that enables users to build, start, and manage Docker container-based clusters. It uses a pluggable system for defining new types of clusters using folders called *topologies* and is a swell project, if I may say so myself.

CHAPTER 1

"I hate reading, make this quick."

Before doing anything, install a recent version of Docker to your machine. Next, clone a clusterdock topology to your machine. For this example, we'll use the nodebase topology. Assuming that you've already installed **clusterdock**, you could start a 2-node cluster:

```
$ git clone https://github.com/clusterdock/topology_nodebase.git
$ clusterdock start topology_nodebase
2017-08-03 10:04:18 PM clusterdock.models   INFO     Starting cluster on network␣
↪(cluster) ...
2017-08-03 10:04:18 PM clusterdock.models   INFO     Starting node node-1.cluster ...
2017-08-03 10:04:19 PM clusterdock.models   INFO     Starting node node-2.cluster ...
2017-08-03 10:04:20 PM clusterdock.models   INFO     Cluster started successfully␣
↪(total time: 00:00:01.621).
```

To list cluster nodes:

```
$ clusterdock ps

For cluster `famous_hyades` on network cluster the node(s) are:
CONTAINER ID     HOST NAME            PORTS              STATUS        CONTAINER NAME␣
↪          VERSION    IMAGE
a205d88beb       node-2.cluster                          running       nervous_
↪sinoussi        1.3.3      clusterdock/topology_nodebase:centos6.6
6f2825c596       node-1.cluster       8080->80/tcp       running       priceless_
↪franklin      1.3.3      clusterdock/topology_nodebase:centos6.6
```

To SSH into a node and look around:

```
$ clusterdock ssh node-1.cluster
[root@node-1 ~]# ls -l / | head
total 64
dr-xr-xr-x   1 root root 4096 May 19 20:48 bin
drwxr-xr-x   5 root root  360 Aug  4 05:04 dev
drwxr-xr-x   1 root root 4096 Aug  4 05:04 etc
drwxr-xr-x   2 root root 4096 Sep 23  2011 home
dr-xr-xr-x   7 root root 4096 Mar  4  2015 lib
```

```
dr-xr-xr-x   1 root root 4096 May 19 20:48 lib64
drwx------   2 root root 4096 Mar  4  2015 lost+found
drwxr-xr-x   2 root root 4096 Sep 23  2011 media
drwxr-xr-x   2 root root 4096 Sep 23  2011 mnt
[root@node-1 ~]# exit
```

To see the complete usage message for the topology:

```
$ clusterdock start topology_nodebase -h
usage: clusterdock start [-h] [--node-disks map] [--always-pull]
                         [--namespace ns] [--network nw] [-o sys] [-r url]
                         [--nodes node [node ...]]
                         topology

Start a nodebase cluster

positional arguments:
  topology              A clusterdock topology directory

optional arguments:
  -h, --help            show this help message and exit
  --always-pull         Pull latest images, even if they're available locally
                        (default: False)
  --namespace ns        Namespace to use when looking for images (default:
                        clusterdock)
  --network nw          Docker network to use (default: cluster)
  -o sys, --operating-system sys
                        Operating system to use for cluster nodes (default:
                        centos6.6)
  -r url, --registry url
                        Docker Registry from which to pull images (default:
                        None)

nodebase arguments:
  --node-disks map      Map of node names to block devices (default: None)

Node groups:
  --nodes node [node ...]
                        Nodes of the nodes group (default: ['node-1',
                        'node-2'])
```

When you're done and want to clean up:

```
$ clusterdock manage nuke
2017-08-03 10:06:28 PM clusterdock.actions.manage INFO     Stopping and removing␣
↪clusterdock containers ...
2017-08-03 10:06:30 PM clusterdock.actions.manage INFO     Removed user-defined␣
↪networks ...
```

CHAPTER 2

More pages with words on them

2.1 Installation

2.1.1 From pip

To install clusterdock, run this command in your terminal:

```
$ pip3 install clusterdock
```

This is the preferred method to install clusterdock, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.1.2 From sources

The sources for clusterdock can be downloaded from its Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/clusterdock/clusterdock
```

Or download the tarball:

```
$ curl  -OL https://github.com/clusterdock/clusterdock/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```

2.2 API reference

2.2.1 clusterdock package

clusterdock.models module

This module contains the main abstractions used by clusterdock topologies to bring up clusters.

class `clusterdock.models.`**`Cluster`**(**nodes*)

The central abstraction for interacting with Docker container clusters. No Docker behavior is actually invoked until the start method is called.

Parameters **`*nodes`** – One or more *`clusterdock.models.Node`* instances.

`execute`(*command*, ***kwargs*)

Execute a command on every *`clusterdock.models.Node`* within the *`clusterdock.models.Cluster`*.

Parameters

- **`command`** (`str`) – Command to execute.
- **`**kwargs`** – Additional keyword arguments to pass to *`clusterdock.models.Node.execute()`*.

Returns

A `collections.OrderedDict` of `str` instances (the FQDN of the node) mapping to the `collections.namedtuple` instances returned by *`clusterdock.models.Node.execute()`*.

`start`(*network*, *pull_images=False*, *update_etc_hosts=True*)

Start the cluster.

Parameters

- **`network`** (`str`) – Name of the Docker network to use for the cluster.
- **`pull_images`** (`bool`, optional) – Pull every Docker image needed by every *`clusterdock.models.Node`* instance, even if it exists locally. Default: `False`
- **`update_etc_hosts`** (`bool`) – Update the /etc/hosts file on the host with the hostname and IP address of the container. Default: `True`

class `clusterdock.models.`**`Node`**(*hostname*, *group*, *image*, *ports=None*, *volumes=None*, *devices=None*, ***create_container_kwargs*)

Class representing a single cluster host.

Parameters

- **`hostname`** (`str`) – Hostname of the node.
- **`group`** (`str`) – *`clusterdock.models.NodeGroup`* to which the node should belong.
- **`image`** (`str`) – Docker image with which to start the container.
- **`ports`** (`list`, optional) – A list of container ports to expose to the host. Elements of the list could be integers (in which case a random port on the host will be chosen by the Docker daemon) or dictionaries (with the key being the host port and the value being the container port). Default: `None`

- **volumes** (`list`, optional) – A list of volumes to create for the node. Elements of the list could be dictionaries of bind volumes (i.e. key: the absolute path on the host, value: the absolute path in the container) or strings representing the names of Docker images from which to get volumes. As an example, `[{'/var/www': '/var/www'}, 'my_super_secret_image']` would create a bind mount of `/var/www` on the host and use any volumes from `my_super_secret_image`. Default: `None`
- **devices** (`list`, optional) – Devices on the host to expose to the node. Default: `None`
- ****create_container_kwargs** – Any other keyword arguments to pass directly to `docker.api.container.create_container()`.

DEFAULT_CREATE_CONTAINER_KWARGS = {'volumes': ['/etc/localtime'], 'detach': True}

DEFAULT_CREATE_HOST_CONFIG_KWARGS = {'cap_add': ['ALL'], 'security_opt': ['seccomp=u

commit (*repository*, *tag=None*, *push=False*, ***kwargs*)
Commit the Node's Docker container to a Docker image.

Parameters

- **repository** (`str`) – The Docker repository to commit the image to.
- **tag** (`str`, optional) – Docker image tag. Default: `None`
- **push** (`bool`, optional) – Push the image to Docker repository. Default: `False`
- ****kwargs** – Additional keyword arguments to pass to `docker.models.Containers.Container.commit()`

execute (*command*, *user='root'*, *quiet=False*, *detach=False*)
Execute a command on the node.

Parameters

- **command** (`str`) – Command to execute.
- **user** (`str`, optional) – User with which to execute the command. Default: `root`
- **quiet** (`bool`, optional) – Run the command without showing any output. Default: `False`
- **detach** (`bool`, optional) – Run the command in detached mode. Default: `False`

Returns A `collections.namedtuple` instance with *exit_code* and *output* attributes.

get_file (*path*)
Get file from the node.

Parameters **path** (`str`) – Absolute path to file.

Returns A `str` containing the contents of the file.

put_file (*path*, *contents*)
Put file on the node.

Parameters

- **path** (`str`) – Absolute path to file.
- **contents** (`str`) – The contents of the file.

start (*network*, *cluster_name=None*)
Start the node.

Parameters

- **network** (`str`) – Docker network to which to attach the container.
- **cluster_name** (`str`, optional) – Cluster name to use for the Node. Default: `None`

stop(*remove=True*)
Stop the node and optionally removing the Docker container.

Parameters **remove** (`bool`, optional) – Remove underlying Docker container. Default: `True`

class `clusterdock.models.`**NodeGroup**(*name*, **nodes*)
Abstraction representing a collection of Nodes that it could be useful to interact with enmasse. For example, a typical HDFS cluster could be seen as consisting of a 1 node group consisting of hosts with NameNodes and an n-1 node group of hosts with DataNodes.

Parameters

- **name** (`str`) – The name by which to refer to the group.
- ***nodes** – One or more *`clusterdock.models.Node`* instances.

execute(*command*, ***kwargs*)

Execute a command on every *`clusterdock.models.Node`* within the *`clusterdock.models.NodeGroup`*.

Parameters

- **command** (`str`) – Command to execute.
- ****kwargs** – Additional keyword arguments to pass to *`clusterdock.models.Node.execute()`*.

Returns

A `collections.OrderedDict` of `str` instances (the FQDN of the node) mapping to the `collections.namedtuple` instances returned by *`clusterdock.models.Node.execute()`*.

clusterdock.utils module

Various utilities to be used by other modules.

`clusterdock.utils.`**DEFAULT_TIMEOUT = 60**

`clusterdock.utils.`**DEFAULT_TIME_BETWEEN_CHECKS = 1**

`clusterdock.utils.`**generate_cluster_name**()
Generate a random cluster name.

`clusterdock.utils.`**get_clusterdock_label**(*cluster_name=None*)
Generate a clusterdock meta data label in json format. Meta data such as: clusterdock package name, version, location of clusterdock install, etc.

Args:

cluster_name (`str`, optional): Cluster name to attach to meta data label. Default: `None`

Returns: (json): clusterdock meta data label

`clusterdock.utils.`**get_container**(*hostname*)
Get running Docker container for a given hostname.

`clusterdock.utils.`**get_containers**(*clusterdock=False*)
Get Docker containers.

Parameters **clusterdock** (bool, optional) – clusterdock containers only. Default: `False`

Returns List of containers.

Return type (list)

`clusterdock.utils.`**`join_url_parts`**(**parts*)

Join a URL from a list of parts. See http://stackoverflow.com/questions/24814657 for examples of why urllib.parse.urljoin is insufficient for what we want to do.

`clusterdock.utils.`**`max_len_list_dict_item`**(*list_dict*, *attr*)

Returns max length of a given attribute from a list of dict items.

`clusterdock.utils.`**`nested_get`**(*dict_*, *keys*)

Utility function that returns the value of a sequence of nested keys.

Example

```
>>> details = {'name': {'first': {'english': 'Dima'}}}
>>> nested_get(details, ['name', 'first', 'english'])
'Dima'
```

Parameters

- **dict** (dict) – Dictionary to access.
- **keys** (list) – A list of keys to access in a nested manner.

Returns The value.

`clusterdock.utils.`**`print_topology_meta`**(*topology_name*, *quiet=False*)

Given a toplogy name, relative to current directory, print its meta info.

`clusterdock.utils.`**`version_str`**(*version*)

Convert a version tuple or string to a string. Will return major.minor.release kind of format.

`clusterdock.utils.`**`version_tuple`**(*version*)

Convert a version string or tuple to a tuple. Will return (major, minor, release) kind of format.

`clusterdock.utils.`**`wait_for_condition`**(*condition*, *condition_args=None*, *condition_kwargs=None*, *time_between_checks=1*, *timeout=60*, *time_to_success=0*, *success=None*, *failure=None*)

Wait until a condition is satisfied (or timeout).

Parameters

- **condition** – Callable to evaluate.
- **condition_args** (*optional*) – A list of args to pass to the `condition`. Default: `None`
- **condition_kwargs** (*optional*) – A dictionary of kwargs to pass to the `condition`. Default: `None`
- **time_between_checks** (int, optional) – Seconds between condition checks. Default: *DEFAULT_TIME_BETWEEN_CHECKS*
- **timeout** (int, optional) – Seconds to wait before timing out. Default: *DEFAULT_TIMEOUT*

- **time_to_success** (`int`, optional) – Seconds for the condition to hold true before it is considered satisfied. Default: `0`
- **success** (*optional*) – Callable to invoke when `condition` succeeds. A `time` variable will be passed as an argument, so can be used. Default: `None`
- **failure** (*optional*) – Callable to invoke when timeout occurs. `timeout` will be passed as an argument. Default: `None`

Raises `TimeoutError`

2.3 Authors

clusterdock is written and maintained by Dima Spivak with invaluable contributions from Srid Banoor and Kirti Velankar.

2.3.1 Special thanks

- Ruthie Spivak, for tolerating/marrying me.
- StreamSets, Inc., for using this in production long before it was stable (which it still probably isn't).
- Cloudera, for employing me during the Hackathon during which I wrote the first version of this project.
- Coke Zero, for being chiefly responsible for my being awake between the hours of 9 am and 5 pm.

2.3.2 Other credits

- Our logo was derived from an icon created by Freepik from Flaticon.

2.4 Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

2.4.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/clusterdock/framework/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

Write Documentation

clusterdock could always use more documentation, whether as part of the official clusterdock docs, in docstrings, or even on the web in blog posts, articles, and such.

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/clusterdock/clusterdock/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

2.4.2 Get Started!

Ready to contribute? Here's how to set up *clusterdock* for local development.

1. Fork the *clusterdock* repo on GitHub.
2. Clone your fork locally:

   ```
   $ git clone git@github.com:your_name_here/clusterdock.git
   ```

3. Install your local copy into a virtualenv. Assuming you have virtualenv installed and keep your virtual environments within a folder in your home directory, this is how you would set up your fork for local development:

   ```
   $ virtualenv ~/virtualenvs/clusterdock
   $ cd clusterdock/
   $ python setup.py develop
   ```

4. Create a branch for local development:

   ```
   $ git checkout -b name-of-your-bugfix-or-feature
   ```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8:

   ```
   $ flake8 clusterdock
   ```

   To get flake8, just pip install it into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Summary of your change"
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

2.4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
2. The pull request should work for Python 3.3, 3.4 and 3.5, and for PyPy.

2.5 History

2.5.1 1.6.0 (2018.03.19)

- Add –port argument functionality to clusterdock start.

2.5.2 1.5.0 (2018.03.09)

- Add support for build action.
- Use Docker labels for clusterdock nodes and clusters.
- Enhance clusterdock manage action.
- Add clusterdock ps action.
- Add clusterdock cp action.

2.5.3 1.4.0 (2018.02.21)

- Add nodes to /etc/hosts during start.

2.5.4 1.3.3 (2018.02.08)

- Fix docker-py dependency to 2.7.0.

2.5.5 1.3.2 (2017.11.13)

- Added support for executing commands in detached mode.

2.5.6 1.3.1 (2017.11.07)

- Fixed broken fix of volume handling from previous release.

2.5.7 1.3.0 (2017.11.01)

- Fixed handling of duplicate networks.
- Made `clusterdock.models.Node.execute()` run commands in a shell (using `/bin/sh` by default).
- Fixed handling of volumes passed to `clusterdock.models.Node`.

2.5.8 1.2.0 (2017.10.23)

- Changed return type of `clusterdock.models.Cluster.execute()` and `clusterdock.models.NodeGroup.execute()`.
- Added support for node devices.

2.5.9 1.1.0 (2017.09.21)

- Updated `clusterdock.models.Node.execute()` to return a namedtuple with the command's exit code and output.
- Fixed bug around `quiet` argument to `clusterdock.models.Node.execute()`.
- Added support for specifying `host:container` port mappings when creating a node.
- Added `ip_address` attribute to `clusterdock.models.Node`.

2.5.10 1.0.7 (2017.09.18)

- Removed `DEFAULT_NAMESPACE` to let topologies define their own.

2.5.11 1.0.6 (2017.09.04)

- Added `clusterdock.models.Node.put_file()` and `clusterdock.models.Node.get_file()`.
- Made `network` an instance attribute of `clusterdock.models.Cluster`.

2.5.12 1.0.5 (2017.09.02)

- Added logic to pull missing images to `clusterdock.models`.

2.5.13 1.0.4 (2017.09.02)

- Fixed missing install requirement.

2.5.14 1.0.3 (2017.09.02)

- Cleaned up `clusterdock.models.Node` API.
- Added wait_for_permission and join_url_parts utility functions.

2.5.15 1.0.2 (2017.08.04)

- Updated how Cluster and Node objects are initialized.
- Added project logo.
- Doc improvements.

2.5.16 1.0.1 (2017.08.03)

- First release on PyPI.

Python Module Index

Index