rally-ci Documentation

Release latest

May 25, 2015

Contents

The simplest way to install rally-ci is pull docker image.

First you need to install docker. Installing docker in ubuntu may be done by following:

```
$ sudo apt-get update
$ sudo apt-get install docker.io
$ sudo usermod -a -G docker `id -u -n` # add yourself to docker group
```

NOTE: re login is required to apply users groups changes and actually use docker.

Pull docker image:

```
$ docker pull rallyforge/rally-ci
```

Or you may want to build rally-ci image from source:

```
$ cd ~/sources/rally-ci # cd to rally-ci sources on your system
$ docker build -t myrally .
```

Create volume-directory:

```
$ mkdir ~/rally-ci-volume
```

This directory will be used to store configuration and logs.

CHAPTER 1

Run in simulation mode

```
$ cd ~/rally-ci-volume
$ ln -s /etc/rally-ci/simulation-config.yaml config.yaml
$ docker run -p 10022:22 -p 10080:80 -v $HOME/rally-ci-volume:/home/rally rallyforge/rally-ci
```

Now you can point your browser to http://localhost:10080/ and see a real time status of the service.

CHAPTER 2

Run in production mode

You should create a ssh key and upload it to gerrit (https://review.openstack.org/):

```
$ cd ~/rally-ci-volume
$ mkdir ~/.ssh
$ ssh-keygen -f .ssh/id_rsa # create ssh keypair
$ vi config.yaml # create configuration file
$ sudo chown -R 65510 .
$ sudo chmod -R g+w .
$ cat .ssh/id_rsa.pub # copy and paste this key to gerrit
```

And run container:

```
$ docker run -p 10022:22 -p 10080:80 -v $HOME/rally-ci-volume:/home/rally rallyforge/rally-ci
```

All logs may be found in ~/rally-ci-volume/. You may want to see the rally-ci logs in real time:

```
$ tail -f ~/rally-ci-volume/rally-ci.err.log
```

The rally-ci service will listen tcp ports:

- 10022 ssh service (for emergency situations)
- 10080 web service (jobs logs and realtime status of the service)

You may expose web and ssh ports to any numbers. Just use "-p 8080:80" to expose web to port 8080 instead of 10080.

CHAPTER 3

Configuration file

3.1 Structure of configuration file

Configuration file is yaml object (dictionary). Each key of this object represents name of section. Value of this object represents configuraion of this section.

Nearly all objects contain "module" key. The value of this key is plugin to be used to do all work.

3.2 Stream section

Stream is plugin to be used for collecting events.

3.2.1 Available stream plugins

rallyci.streams.gerrit

Standard stream for receiving gerrit events.

Sample config:

```
stream:
    module: rallyci.streams.gerrit
    username: joe
    hostname: review.openstack.org
    port: 29418
```

rallyci.streams.fake

Used for testing. Will read events from file line by line.

Sample config:

```
stream:
    module: rallyci.streams.fake
    path: /path/to/json/file/with/events.json
```

3.3 Loggers section

Loggers are used to log scripts output.

3.3.1 Available loggers

rallyci.loggers.file

Logs scripts output to local files.

Sample config:

```
loggers:
  file:
    module: rallyci.loggers.logfile
    path: /store/log/rally-ci/
```

3.4 Environments section

Each environment performs some actions and export environment variables.

3.4.1 Available environments

rallyci.environments.event

This environment is used to export gerrit event variables to script's env.

Sample config:

```
module: rallyci.environments.event
export-event:
  GERRIT_PROJECT: change.project
  GERRIT_REF: patchSet.ref
```

rallyci.environments.dummy

Simple environment to export any static variables. Does not have any configuration at this level. All configuration is done in "jobs" section (see full config example).

Sample config:

```
dummy:
  module: rallyci.environments.dummy
```

3.5 Nodepools section

Nodepools are used to manage worker nodes.

3.5.1 Available nodepools

rallyci.nodepools.fair

Return node with less running jobs.

Sample configuraion:

```
nodepools:
  localdocker:
    module: rallyci.nodepools.fair
    tasks_per_node: 2
    nodes:
      - hostname: worker1.net
        username: rally
        port: 33
      - hostname: worker1.net
        username: admin
        key: /home/rally/.ssh/superkey
```

The config above has two nodes in pool. First node has non standard ssh port.

3.6 Runners section

Runners are used to run scripts on VM's or containers created on nodes from nodepools. Containers or VM's are created by runner according to runner's configuration.

3.6.1 Available runners

rallyci.runners.docker

Run jobs in docker containers. Build images from dockerfiles hardcoded in config:

```
runners:
  localdocker:
    nodepool: localdocker
    module: rallyci.runners.docker
    images:
      ubuntu-dev: |
        FROM ubuntu:14.04
        MAINTAINER Sergey Skripnick <sskripnick@mirantis.com>
        RUN apt-get update && apt-get install python2.7-dev
        RUN useradd -u 65510 -m rally
        USER rally
        WORKDIR /home/rally
        RUN mkdir openstack && cd openstack && \
            git clone git://git.openstack.org/openstack/rally.git
```

rallyci.runners.fake

Used for testing. Does nothing but sleeping random delays. Always returns success.

rallyci.runners.lxc

Work in progress.

rallyci.runners.virsh

Work in progress.

3.7 Scripts section

Scripts may be used for running tests and building images.

Sample scripts section:

```
scripts:
  git_checkout:
    interpreter: /bin/bash -xe -s
    data: |
      cd $GERRIT_PROJECT && git checkout master && git pull
      git fetch https://review.openstack.org/$GERRIT_PROJECT $GERRIT_REF
      git checkout FETCH_HEAD && git rebase master
  run_tox:
    interpreter: /bin/bash -xe -s
    data: |
      tox -epy27
```

3.8 Jobs section

Jobs definitions. Key is the name of job, value is configuration.

Configuration consist of following sections:

- envs
- runner

Sample jobs section:

```
jobs:
  py27:
    envs:
      - name: event
      - name: dummy
        export:
          RCI_TOXENV: py27
    runner:
      name: localdocker
      image: ubuntu-dev
      scripts:
        - git_checkout
        - run_tox
```

3.9 Projects section

This sections descibes which jobs run for which projects:

```
projects:
  "openstack/nova":
    jobs:
      - pep8
      - py27
  "openstack/designate"
    jobs:
      - py34
      - rally
```

Full working sample may be found in source code tree in file etc/sample-config.yaml.

Example full configuration:

```
---
stream:
    module: rallyci.streams.gerrit
    username: CHANGEME
    hostname: review.openstack.org
    port: 29418

loggers:
  file:
    module: rallyci.loggers.logfile
    path: /home/rally/ci-logs/

environments:
  event:
    module: rallyci.environments.event
    export-event:
      GERRIT_PROJECT: change.project
      GERRIT_REF: patchSet.ref
  dummy:
    module: rallyci.environments.dummy

nodepools:
  localdocker:
    module: rallyci.nodepools.fair
    tasks_per_node: 2
    nodes:
      - hostname: localhost

runners:
  localdocker:
    nodepool: localdocker
    module: rallyci.runners.docker
    images:
      ubuntu-dev: |
        FROM ubuntu:14.04
        MAINTAINER Sergey Skripnick <sskripnick@mirantis.com>
        RUN apt-get update
        RUN apt-get -y install git python2.7 bash-completion python-dev libffi-dev \
        libxml2-dev libxslt1-dev libssl-dev libpq-dev
        RUN apt-get -y install python-pip
        RUN pip install tox==1.6
```

```
        RUN useradd -u 65510 -m rally
        USER rally
        WORKDIR /home/rally
        RUN git config --global user.email "rally-ci@mirantis.com" && \
            git config --global user.name "Mirantis Rally CI"
        RUN mkdir openstack && cd openstack && \
            git clone git://git.openstack.org/openstack/rally.git

scripts:
  git_checkout:
    interpreter: /bin/bash -xe -s
    data: |
      env
      cd $GERRIT_PROJECT && git checkout master && git pull
      git fetch https://review.openstack.org/$GERRIT_PROJECT $GERRIT_REF
      git checkout FETCH_HEAD && git rebase master || true
      git clean -fxd -e .tox -e *.egg-info
      git diff --name-only master
  tox:
    interpreter: /bin/bash -xe -s
    data:
      cd $GERRIT_PROJECT && tox -e$RCI_TOXENV

jobs:
  py27:
    envs:
      - name: event
      - name: dummy
        export:
          RCI_TOXENV: py27
    runner:
      name: localdocker
      image: ubuntu-dev
      scripts:
        - git_checkout
        - tox

projects:
  "openstack/rally":
    jobs:
     - py27
```

The configuration above will run tox -epy27 on each patch in openstack/rally.