WeIRDO Documentation

Release 0.2

David Moreau Simard

Sep 27, 2017

Contents

We [**I**nstall | **I**ntegrate | **I**mprove] **RDO**: An open initiative to improve RDO with it's community for it's community.

Testing OpenStack is hard:

- There is *currently* almost 50 official OpenStack projects
- There is over 1000 source and binary packages built and provided by the official RDO repositories
- These packages are bundled, integrated or provided by many different installers and vendors
- There are countless deployment use cases ranging from simple private clouds to large scale complex and highly available public clouds

Providing test coverage to make sure that each and every project and use case works well is a hard problem to solve.

WeIRDO is an idea that is about leveraging the effort the community puts towards their own CI initiative to improve the coverage that upstream RDO provides. This is about testing as many ways of installing, configuring and integrating OpenStack as possible without re-inventing the wheel.

WeIRDO takes the gate jobs outside of the gate to test what RDO ships before it's packages even make it to the upstream OpenStack CI.

Let's work together to make RDO better.

CHAPTER 1

Where can I find the WeIRDO CI Jobs ?

All RDO CI jobs are publicly available and the ones from WeIRDO are no exception.

WeIRDO is currently used in two different implementations:

- ci.centos.org: Periodic jobs to test and promote RDO trunk packages
- review.rdoproject.org: Gate jobs for WeIRDO itself (and it's roles) and for other projects

For more details on those implementations, see the jenkins job configuration documentation.

CHAPTER 2

Table of Contents

Contributing

WeIRDO was built from the ground up with the highest standards and best practices in mind just like OpenStack.

To try and keep it that way, commits to WeIRDO are first reviewed through Gerrit and are gated against jobs to ensure the patch does not break anything.

Gate jobs

- **gate-weirdo-ansible-lint** checks that playbooks, roles and tasks are clean and "compile"
- **gate-weirdo-docs** checks that documentation can be built properly
- **gate-weirdo-integration-*** runs the integration tests implemented in WeIRDO whenever relevant (i.e, only run Packstack integration tests when commits involve code that touches Packstack)

Getting started

The process is very similar to what someone would expect when contributing to an OpenStack project except WeIRDO leverages the gerrit instance at review.rdoproject.org.

You can submit a patch to WeIRDO by doing something like this:

```
# git-review is required and can be installed through pip
pip install git-review
git clone https://github.com/rdo-infra/weirdo.git
cd weirdo
# < Some hacking occurs >
git commit -a -m "Fixing things"
git review
```

How does it work ?

In a nutshell

WeIRDO is a collection of Ansible roles and playbooks.

WeIRDO, through Ansible, will:

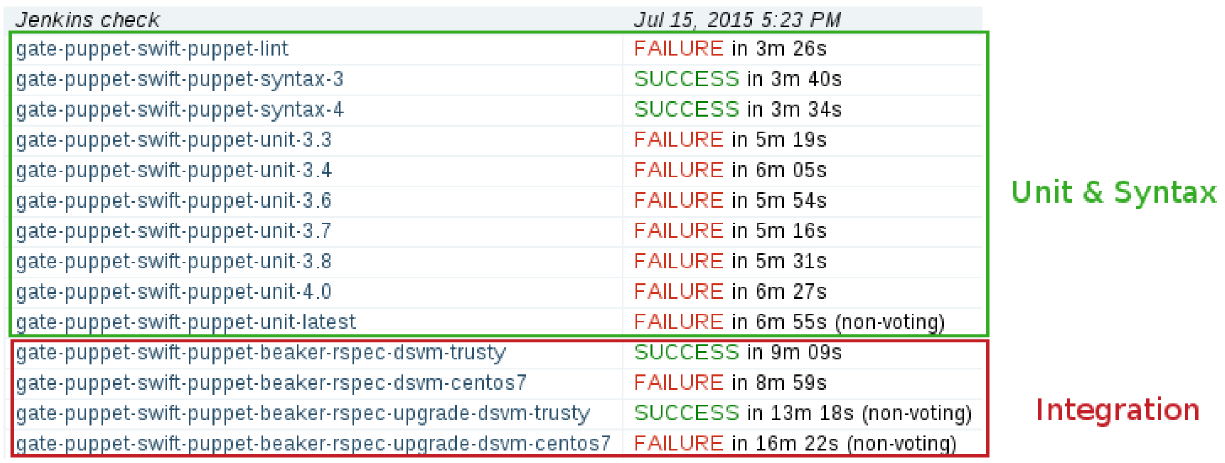
- Connect to a node specified in it's inventory (this can be a remote node or localhost)
- Generate a similar environment to the gate (setup configuration, packages, dependencies)
- Run the gate job (i.e, `run_tests.sh` or `tox` provided by the upstream project)

WeIRDO, by itself, doesn't use or require Jenkins. When run inside a job, however, it will be able to report success or failure.

Details on how the Jenkins jobs are configured is available here.

Example: Puppet-Openstack

Puppet-Openstack gates every commit that is done with syntax, unit and integration tests.



Jenkins check	Jul 15, 2015 5:23 PM
gate-puppet-swift-puppet-lint	FAILURE in 3m 26s
gate-puppet-swift-puppet-syntax-3	SUCCESS in 3m 40s
gate-puppet-swift-puppet-syntax-4	SUCCESS in 3m 34s
gate-puppet-swift-puppet-unit-3.3	FAILURE in 5m 19s
gate-puppet-swift-puppet-unit-3.4	FAILURE in 6m 05s
gate-puppet-swift-puppet-unit-3.6	FAILURE in 5m 54s
gate-puppet-swift-puppet-unit-3.7	FAILURE in 5m 16s
gate-puppet-swift-puppet-unit-3.8	FAILURE in 5m 31s
gate-puppet-swift-puppet-unit-4.0	FAILURE in 6m 27s
gate-puppet-swift-puppet-unit-latest	FAILURE in 6m 55s (non-voting)
gate-puppet-swift-puppet-beaker-rspec-dsvm-trusty	SUCCESS in 9m 09s
gate-puppet-swift-puppet-beaker-rspec-dsvm-centos7	FAILURE in 8m 59s
gate-puppet-swift-puppet-beaker-rspec-upgrade-dsvm-trusty	SUCCESS in 13m 18s (non-voting)
gate-puppet-swift-puppet-beaker-rspec-upgrade-dsvm-centos7	FAILURE in 16m 22s (non-voting)

These tests ensure there is no regression when someone contributes to one of the many modules involved in Puppet-Openstack.

While the syntax and unit checks are supplied by each respective puppet module, the integration tests are provided through the puppet-openstack-integration project. Ultimately, these are all configured to run by Jenkins through project-config and are triggered when a patchset is submitted to Gerrit.

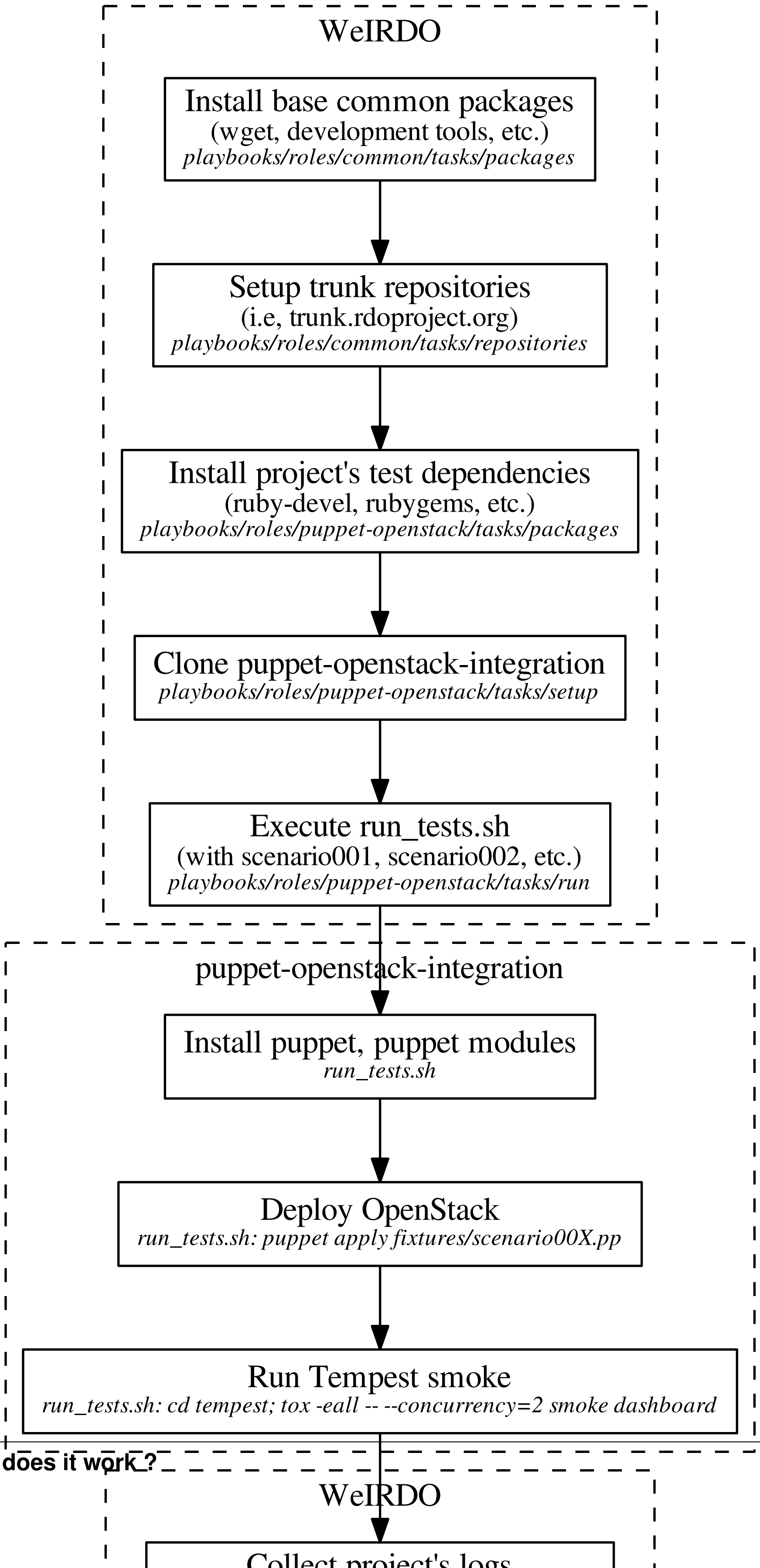
The integration tests are of particular interest since the puppet modules will install OpenStack libraries, clients and services with the packages provided through RDO. Tempest is currently the only exception and is installed from source.

Puppet-Openstack uses these integration tests to ensure their puppet modules work well.

WeIRDO leverages these integration tests to ensure that the packages provided by RDO work well.

Test implementation

This is what the WeIRDO implementation looks like for puppet-openstack:


WeIRDO
Install base common packages
(wget, development tools, etc.)
playbooks/roles/common/tasks/packages
Setup trunk repositories
(i.e, trunk.rdoproject.org)
playbooks/roles/common/tasks/repositories
Install project's test dependencies
(ruby-devel, rubygems, etc.)
playbooks/roles/puppet-openstack/tasks/packages
Clone puppet-openstack-integration
playbooks/roles/puppet-openstack/tasks/setup
Execute run_tests.sh
(with scenario001, scenario002, etc.)
playbooks/roles/puppet-openstack/tasks/run
puppet-openstack-integration
Install puppet, puppet modules
run_tests.sh
Deploy OpenStack
run_tests.sh: puppet apply fixtures/scenario00X.pp
Run Tempest smoke
run_tests.sh: cd tempest; tox -eall -- --concurrency=2 smoke dashboard
WeIRDO
Collect project's logs

Ansible Playbooks

Summary of the playbooks:

- **logs-ci-centos**: Wrapper used on CI jobs run on the ci.centos.org environment to upload logs to the artifact server
- **puppet-openstack-scenario001**: Runs scenario001.pp with run_tests.sh provided by puppet-openstack-integration
- **puppet-openstack-scenario002**: Runs scenario002.pp with run_tests.sh provided by puppet-openstack-integration
- **puppet-openstack-scenario003**: Runs scenario003.pp with run_tests.sh provided by puppet-openstack-integration
- **packstack-scenario001**: Runs scenario001.sh with run_tests.sh provided by Packstack
- **packstack-scenario002**: Runs scenario002.sh with run_tests.sh provided by Packstack
- **packstack-scenario003**: Runs scenario003.sh with run_tests.sh provided by Packstack

Ansible roles

ansible-role-weirdo-common

This role centralizes common tasks used across different roles for the WeIRDO project.

Role variables

For default values, see **'defaults/main.yml'_**

- `base_packages`: A list of packages to install by default
- `debug_packages`: A list of packages meant for debug purposes to install by default
- `job_name`: Name of the job (defaults to Jenkins-provided environment variable)
- `build_numer`: Build number of the job (defaults to Jenkins-provided environment variable)
- `log_root`: Root directory where the debug logs will be sent to
- `log_destination`: Subdirectory where the debug logs will be sent to
- `log_paths`: Log paths to recover by default
- `debug_commands`: Hash of debug commands that will be run and output to log files

Included tasks

- `logs`: Recovers basic logs by default
- `packages`: Ensures required dependencies are installed
- `setup`: Sets up log directory, configures sysstat
- `rescue`: Task that gets included in rescue blocks to notify different handlers

Dependencies

ansible-role-ci-centos to be installed as `ci-centos` when the role ci-centos is used to provision nodes.

Example playbook

```
- name: Run packstack scenario001 tests
  hosts: openstack_nodes
  force_handlers: True
  roles:
    - role: "common"
    - { role: "packstack", test: "scenario001" }
    - { role: "common", action: "logs" }
```

ansible-role-weirdo-puppet-openstack

This role provides an implementation of the puppet-openstack-integration gate jobs for the WeIRDO project.

Supported gate jobs

- gate-puppet-openstack-integration-scenario001-dsvm-centos7
- gate-puppet-openstack-integration-scenario002-dsvm-centos7
- gate-puppet-openstack-integration-scenario003-dsvm-centos7

Role variables

For default values, see **'defaults/main.yml'_**

- `project`: Name of the project, analogous to the role name
- `openstack_release`: Name of the OpenStack release to select which trunk repository to use by default
- `version`: Version/tag/branch of puppet-openstack-integration to clone
- `manage_repos`: Whether puppet-openstack-integration should manage repository setup
- `repository`: URL to the puppet-openstack-integration repository
- `clone_path`: Path where puppet-openstack-integration will be cloned
- `delorean_url`: URL to the delorean repository .repo file.
- `delorean_deps_url`: URL to the delorean-deps repository .repo file.
- `copy_puppet_logs_url`: URL to the copy_puppet_logs.sh script for puppet-openstack-integration log retrieval
- `puppet_workspace`: Path where puppet-openstack-integration should believe the jenkins workspace is at
- `required_packages`: Required packaged dependencies to install prior to running the tests
- `gem_packages`: Required packaged gems to install prior to running the tests

Included tasks

- `repositories`: Ensures required repositories are setup
- `packages`: Ensures required dependencies are installed
- `setup`: Clones the puppet-openstack-integration repository and prepares log retrieval
- `logs`: Retrieves logs with the copy_puppet_logs.sh script
- `run`: Runs the integration tests

Dependencies

ansible-role-weirdo-common to be installed as `common`

Example playbook

```
- name: Run puppet-openstack-integration scenario001 tests
  hosts: openstack_nodes
  force_handlers: True
  roles:
    - { role: "puppet-openstack", test: "scenario001" }
```

ansible-role-weirdo-packstack

This role provides an implementation of the Packstack gate jobs for the WeIRDO project.

Supported gate jobs

- gate-packstack-integration-scenario001-tempest-dsvm-centos7
- gate-packstack-integration-scenario002-tempest-dsvm-centos7
- gate-packstack-integration-scenario003-tempest-dsvm-centos7

Role variables

For default values, see **‘defaults/main.yml‘_**

- `project`: Name of the project, analogous to the role name
- `openstack_release`: Name of the OpenStack release to select which trunk repository to use by default
- `version`: Version/tag/branch of Packstack to clone
- `repository`: URL to the Packstack repository
- `clone_path`: Path where Packstack will be cloned
- `install_from_source`: Whether Packstack should be installed from source or from packages
- `delorean_url`: URL to the delorean repository .repo file.
- `delorean_deps_url`: URL to the delorean-deps repository .repo file.
- `packstack_workspace`: Path where Packstack should believe the jenkins workspace is at

Included tasks

- `setup`: Clones the Packstack repository and prepares log retrieval
- `run`: Runs the integration tests

Dependencies

ansible-role-weirdo-common to be installed as `common`

Example playbook

```
- name: Run packstack scenario001 tests
  hosts: openstack_nodes
  force_handlers: True
  roles:
    - { role: "packstack", test: "scenario001" }
```

Using WeIRDO with Jenkins

There are currently two different implementations of WeIRDO jobs that are run through Jenkins.

- Periodic jobs in a pipeline for trunk package testing and promotion on ci.centos.org
- Gate jobs against WeIRDO itself as well as against other projects on review.rdoproject.org

ci.centos.org: Pure jenkins and JJB on external bare metal

The configuration for the jobs that leverage WeIRDO on ci.centos.org to test RDO trunk repositories can be found in the rdo-infra/ci-config repository.

The WeIRDO jobs themselves and their results are publicly available on https://ci.centos.org/view/rdo/view/promotion-pipeline/

ci.centos.org provides an API, called Duffy, to manage ephemeral bare metal nodes to run jobs on. As such, the jobs that run on this environment have particular requirements and are documented in the rdo-infra/ci-config repository.

In this use case, a task is executed before WeIRDO runs to request a bare metal node and an inventory is created with that node in it. After that, WeIRDO runs with the generated inventory file, logs are collected and uploaded to ci.centos.org/artifacts and the node is destroyed.

Troubleshooting ci.centos.org jobs

A logs.html file will be saved as an artifact of the job. This logs.html file is actually just a redirection to the location where the logs are located for this particular job.

review.rdoproject.org: JJB with Gerrit, Zuul and Nodepool virtual machines

review.rdoproject.org is an environment driven by Gerrit, Zuul and Jenkins and jobs run on ephemeral jenkins slaves provided by Nodepool, it is a software factory instance that attempts to provide the same experience as the one provided by openstack-infra for their continuous integration ecosystem.

The configuration for the different components can be found in the config repository.

In this use case, nodepool takes care of the node provisioning and destruction before and after the job and the jenkins slave is effectively “localhost”. To run WeIRDO on localhost, we create a minimal inventory file with localhost as the target and pre-install some dependencies beforehand. Once the job is over, artifacts and logs are uploaded to Swift.

Troubleshooting review.rdoproject.org jobs

WeIRDO runs from gate jobs in review.rdoproject.org’s Gerrit. Clicking on a Job in a Jenkins comment in a review will lead you to the Jenkins job result and console. To access the job logs, click on the console log and the link to the logs will be available at the very bottom of the log, provided by zuul-swift-uploader.