# Contents

1 Overview 1

2 Contents 3
   2.1 Project Introduction 3
   2.2 Development Documentation 488
   2.3 Installation and Operations Manual 525

3 Indices and tables 549
This project provides scalable and reliable IT automation using SaltStack for installing and operating wide variety of services and resources. Project provides standards to define service models and processes with ability to reuse these components in varying contexts.
2.1 Project Introduction

Here you will find documentation relevant to architecture and goals of the project. Existing formula ecosystem and underlying metadata standards.

2.1.1 Overview

Chapter 1. Overview

Home SaltStack-Formulas Project Introduction

Project Objectives

- **Collateral Goodies**

Project provides standards to define service models and processes with ability to reuse these components in varying contexts. Metadata model shared across all services let us explore underlying relationships that ease the management of infrastructures across whole life-span.

The project has little different objectives compare to official salt-formulas. The general orientation of project may be similar to the official salt formulas but the major differences lie at meta-data model and clear decomposition which being consistent across all formulas in SaltStack-Formulas project.

Collateral Goodies

Adhering to the standards allows further services to be declared and configured in dynamic way, consuming metadata of surrounding services. This include following domains:
• Dynamic monitoring: Event collecting, telemetry with dashboards, alarms with notifications
• Dynamic backup: Data backuping and restoring
• Dynamic security: Firewall rules, router configurations
• Dynamic documentation, topology visualizations
• Dynamic audit profiles and beacons

All these can be generated out of your existing infrastructure without need for any further parametrisation.

---

Project History

• Beginnings
• tcp cloud Era
• openstack-salt Project
• saltstack-formulas Project

Beginnings

The initial formula structure was created in 2013. The formulas were not even called formulas back then, but states. It was time of great confusion and the quality of newly created salt-formulas was low.

tcp cloud Era

The majority of formulas were rewritten to current standard structure and were used in production for cloud deployments. All the formulas were open-sourced and support metadata were instroduces in 2015.

openstack-salt Project

OpenStack-Salt project was OpenStack Big Tent initiative project in 2015/16 and provided resources for installing and operating OpenStack deployments. It used subset of the formulas and project was abandoned when tcp cloud was bought by Mirantis.
**saltstack-formulas Project**

The scope of current project is much wider than management of OpenStack installations and provides generic formula ecosystem capable of managing multiple heterogenous infrastructures.

---

**Introduction to SaltStack**

### Pillar Metadata

SaltStack-Formulas uses Salt configuration platform to install and manage infrastructures. Salt is an automation platform that greatly simplifies system and application deployment. Salt uses service *formulas* to define resources written in the YAML language that orchestrate the individual parts of system into the working entity.

**Pillar Metadata**

Pillar is an interface for Salt designed to offer global values that are distributed to all minions. The ext_pillar option allows for any number of external pillar interfaces to be called to populate the pillar data.

Pillars are tree-like structures of data defined on the Salt Master and passed through to the minions. They allow confidential, targeted data to be securely sent only to the relevant minion. Pillar is therefore one of the most important systems when using Salt.

---

2.1. **Project Introduction**
2.1.2 Quick Start

Chapter 2. Quick start

Home SaltStack-Formulas Project Introduction

Deployment Preparation Guidelines

- Salt Master Formulas
- Salt Master Metadata

Let’s consider simple deployment of single configuration node with one application and one database node.

- Config node [salt master]
- Application node [python app]
- Database node [postgres db]

To start the simple deployment you need first setup the Salt master. Installation of salt minions on controlled nodes is then very simple.

Salt Master Formulas

States are delivered by formulas and are stored in /srv/salt/env/<env>/ directory. Environment can be either production [prd] or development [dev]. This directory is correlates with salt_files root for given environment. You can serve multiple environments from single salt master at once, but this setup is not recommended.

Usually production environment formulas are delivered by packages and development environment formulas are delivered by git sourced formulas.

```
/srv/salt/env/<env>/
|-- service1/
  |-- itit.sls
  |-- role1/
    | |-- service.sls
    | `-- resource.sls
    `-- role2.sls
-- service2/
  |-- itit.sls
  `-- role.sls
```

For example basic linux, python-app and openssh services for development environment in a little shortened version.

```
/srv/salt/env/dev/
|-- linux/
  | |-- itit.sls
  | |-- system/
  | | |-- repo.sls
  | | `-- user.sls
  | `-- network/
  |   |-- interface.sls
  |   `-- host.sls
```

(continues on next page)
More about structure and layout of the formulas can be found in Development documentation.

**Salt Master Metadata**

Metadata then define what state formulas in given specific context are projected to managed nodes.

Following trees shows simple metadata structure for simple python application deployment. Important parameters are `cluster_name` labeling individual deployments and `cluster.domain` giving the deployment nodes domain part of the FQDN.

```
|-- python-app/
 | |-- itit.sls
 | |-- server.sls
 `-- openssh/
    | |-- itit.sls
    | |-- server.sls
    `-- client.sls
```

You start with defining single node `cfg.cluster.domain` in nodes directory and that is core node pointing to your `cluster.deploy.infra.config` class.

Content of the `nodes/cfg.cluster.domain.yml` file:

```
classes:
  - cluster.deploy.infra.config
parameters:
```

(continues on next page)
Contains pointer to class `cluster.deploy.infra.config` and some basic parameters.

Content of the `classes/cluster/deploy/infra/config.yml` file:

```yaml
classes:
- system.openssh.client
- system.salt.master.git
- system.salt.master.formula.git
- system.reclass.storage.salt
- cluster.cluster_name
parameters:
  _param:
    salt_master_base_environment: dev
    reclass_data_repository: git@git.domain.com:reclass-models/salt-model.git
    salt_master_environment_repository: "https://github.com/salt-formulas"
    reclass_data_revision: master
    reclass_config_master: ${_param:infra_config_deploy_address}
    single_address: ${_param:infra_config_address}
reclass:
  storage:
    node:
      python_app01:
        name: app01
        domain: ${_param:cluster_domain}
        classes:
          - cluster.$(_param:cluster_name).python_app.application
        params:
          salt_master_host: ${_param:reclass_config_master}
          single_address: ${_param:python_application_node01_single_address}
          database_address: ${_param:python_database_node01_single_address}
      python_dbs01:
        name: dbs01
        domain: ${_param:cluster_domain}
        classes:
          - cluster.$(_param:cluster_name).python_app.database
        params:
          salt_master_host: ${_param:reclass_config_master}
          single_address: ${_param:python_database_node01_single_address}
```

More about structure and layout of the metadata can be found in Metadata chapter.
Bootstrap Salt-Formulas infrastructure

- **TL;DR**
  - **Quick bootstrap**
    - Bootstrap salt-master
    - Bootstrap salt-minion
  - **Advanced usage**
    - Additional bootstrap ENV variables
    - Bootstrap Salt Master in a container for model validation purposes
    - To verify the model (reclass model)

This document’s describes scripted way to configure Salt Master node.
To setup the environment according to Quickstart Configure specification.

**TL;DR**

We uses and script that provide functions to install and configure required primitives and dependencies.
Script with function library is to:

- install and configure *salt master* and *minions*
- install and configure reclass
- bootstrap *salt master* with *salt-formulas* common prerequisites in mind
- validate reclass the model / pillar for all nodes

**Note:** This script is expected to convert to salt formula in a longterm perspective.

Expected usage in shortcut is:

```
git clone https://github.com/salt-formulas/salt-formulas-scripts /srv/salt/scripts
source /srv/salt/scripts/bootstrap.sh
```

Use one of the functions or follow the “setup()” which is executed by default:

- source_local_envs()
- install_reclass()
- clone_reclass()
- configure_pkg_repo()
- configure_salt_master()
- configure_salt_minion()
- install_salt_formula_git()
- install_salt_formula_pkg()
- install_salt_master_pip()
- install_salt_master_pkg()
* install_salt_minion_pip()
* install_salt_minion_pkg()

* verify_salt_master()
* verify_salt_minion()
* verify_salt_minions()

Quick bootstrap

Bootstrap salt-master

(expects salt-formulas reclass model repo)

```
git clone https://github.com/salt-formulas/salt-formulas-scripts /srv/salt/scripts

git clone <model-repository> /srv/salt/reclass
cd /srv/salt/reclass
git submodule update --init --recursive

cd /srv/salt/scripts

CLUSTER_NAME=regionOne HOSTNAME=cfg01 DOMAIN=infra.ci.local ./bootstrap.sh

# OR just
HOSTNAME=cfg01 DOMAIN=infra.ci.local ./bootstrap.sh
```

**Note:** Creates $PWD/.salt-master-setup.sh.passed if successfully passed the “setup script” with the aim to avoid subsequent setup.

Bootstrap salt-minion

This is mostly just to makeweight as configure minion as a super simple task that can be achieved by other means as well.

```
export HTTPS_PROXY="http://proxy.your.corp:8080"; export HTTP_PROXY=$HTTPS_PROXY

export MASTER_HOSTNAME=cfg01.infra.ci.local || export MASTER_IP=10.0.0.10
export MINION_ID=$((hostname -f)) || export HOSTNAME=prx01 DOMAIN=infra.ci.local

install_salt_minion_pkg
```

Advanced usage

The script is fully driven by environment variables. That’s Pros and known Cons of course.
Additional bootstrap ENV variables

(for full list of options see the bootstrap.sh source)

```bash
# reclass
export RECLASS_ADDRESS=<repo url>  ## if not already cloned in /srv/salt/reclass >

# formula
export FORMULAS_BRANCH=master
export FORMULAS_SOURCE=git

# system / host / salt master minion id
export HOSTNAME=cfg01
export DOMAIN=infra.ci.local
# Following variables are calculated from the above if not provided
#export MINION_ID
#export MASTER_HOSTNAME
#export MASTER_IP

# salt
export BOOTSTRAP_SALTSTACK_OPTS=" -dX stable 2016.3"
export EXTRA_FORMULAS="prometeus"
SALT_SOURCE=${SALT_SOURCE:-pkg}
SALT_VERSION=${SALT_VERSION:-latest}

# bootstrap control
export SALT_MASTER_BOOTSTRAP_MINIMIZED=False
export CLUSTER_NAME=#{cluster %}

# workarounds (forked reclass)
export RECLASS_IGNORE_CLASS_NOTFOUND=False
export EXTRA_FORMULAS="prometheus telegraph"
```

Bootstrap Salt Master in a container for model validation purposes

We use this to check the model during CI. The example count’s with using forked version of Reclass
<https://github.com/salt-formulas/reclass> with additional features, like ability to ignore missing classes during the bootstrap.

To spin a container we uses a kitchen-test framework. The setup required you may find in the Testing formulas section
<../develop/testing-formulas.html#requirements

Assume you have a repository with your reclass model. Add to this repository following files. Both files can be found at salt-formulas/deploy/model <https://github.com/salt-formulas/salt-formulas/tree/master/deploy/model> repo.

```yaml
---
name: docker
use_sudo: false
volume:
  - <%= ENV['PWD'] %>/tmp/kitchen

provisioner:
  name: shell
```

(continues on next page)
script: verify.sh

platforms:
<% `find classes/cluster -maxdepth 1 -mindepth 1 -type d | tr '_' '-' |sort -u`. 
- split().each do |cluster| %>
  - name: <%= cluster %>
    driver_config:
      # image: ubuntu:16.04
      image: tcpcloud/salt-models-testing # With preinstalled dependencies (faster)
      platform: ubuntu
      hostname: cfg01.<%= cluster %>.local
      provision_command:
        - apt-get update
        - apt-get install -y git curl python-pip
        - git clone https://github.com/salt-formulas/salt-formulas-scripts /srv/salt/
        - cd /srv/salt/scripts; git pull -r; cd -
          # NOTE: Configure ENV options as needed, example:
          - echo "
            export BOOTSTRAP=1;
            export CLUSTER_NAME=<%= cluster %>; 
            export FORMULAS_SOURCE=pkg; 
            export RECLASS_VERSION=dev; 
            export RECLASS_IGNORE_CLASS_NOTFOUND=True; 
            export EXTRA_FORMULAS="";
          " > /kitchen.env
          #export RECLASS_SOURCE_PATH=/usr/lib/python2.7/site-packages/reclass;
          #export PYTHONPATH=$RECLASS_SOURCE_PATH:$PYTHONPATH;
  - scripts
    - cd /srv/salt/scripts; git pull -r; cd -
      # NOTE: Configure ENV options as needed, example:
      - echo "
          export BOOTSTRAP=1;\n          export CLUSTER_NAME=<%= cluster %>;\n          export FORMULAS_SOURCE=pkg;\n          export RECLASS_VERSION=dev;\n          export RECLASS_IGNORE_CLASS_NOTFOUND=True;\n          export EXTRA_FORMULAS="";\n        " > /kitchen.env
      #export RECLASS_SOURCE_PATH=/usr/lib/python2.7/site-packages/reclass;\n      #export PYTHONPATH=$RECLASS_SOURCE_PATH:$PYTHONPATH;\n  <% end %>

suites:
- name: cluster

verify.sh:

#!/bin/bash

# ENV variables for MASTER_HOSTNAME composition
# export HOSTNAME=${`hostname -s`}
# export DOMAIN=${`hostname -d`}
cd /srv/salt/scripts; git pull -r || true; source bootstrap.sh || exit 1

# BOOTSTRAP
if [[ $BOOTSTRAP =~ ^{True|true|1|yes}$ ]]; then
  # workarounds for kitchen
  test ! -e /tmp/kitchen || (mkdir -p /srv/salt/reclass; rsync -avh /tmp/kitchen/ / 
  /srv/salt/reclass)
  cd /srv/salt/reclass
  # clone latest system-level if missing
  if [[ -e .gitmodules ]]; then
    git submodule update --init --recursive --remote || true
  fi
  source_local_envs
  /srv/salt/scripts/bootstrap.sh
  if [[ -e /tmp/kitchen ]]; then
    sed -i '/export BOOTSTRAP=/d' /kitchen.env; fi
fi

(continues on next page)
# VERIFY
export RECLASS_IGNORE_CLASS_NOTFOUND=False
cd /srv/salt/reclass &&
if [[ -z "$1" ]] ; then
  verify_salt_master &&
  verify_salt_minions
else
  verify_salt_minion "$1"
fi

Then with `kitchen list` command list the models in repository to test and finally converge and salt master instance where you will trigger the validation.

$ kitchen list

<table>
<thead>
<tr>
<th>Instance</th>
<th>Driver</th>
<th>Provisioner</th>
<th>Verifier</th>
<th>Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster-aaa-ha-freeipa</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>Created</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ceph-ha</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-k8s-aio-calico</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-k8s-ha-calico</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-aio-contrail</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-aio-ovs</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-ha-contrail</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-ha-ovs</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-ha-ovs-syndic</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-liberty-dvr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-liberty-ovs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-mitaka-contrail</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-mitaka-dvr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-mitaka-ovs</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-ocata-aio</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-ocata-contrail</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Created</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-ocata-contrail-nfv</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-ost-virt-ocata-dvr</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
To converge an instance:

```bash
$ kitchen converge cluster-ost-virt-ocata-contrail
```

**To verify the model (reclass model)**

You may use a custom module build for this purpose in reclass formula [https://github.com/salt-formulas/salt-formula-reclass](https://github.com/salt-formulas/salt-formula-reclass).

```bash
SUDO salt-call $SALT_OPTS --id=$MASTER_HOSTNAME reclass.validate_yaml
SUDO salt-call $SALT_OPTS --id=$MASTER_HOSTNAME reclass.validate_pillar
SUDO salt-call $SALT_OPTS --id=$MASTER_HOSTNAME grains.item roles
SUDO salt-call $SALT_OPTS --id=$MASTER_HOSTNAME state.show_lowstate
SUDO salt-call --no-color grains.items
SUDO salt-call --no-color pillar.data
SUDO reclass --nodeinfo $HOSTNAME
```

Home SaltStack-Formulas Project Introduction

**Quick Deploy on OpenStack with Heat**

- Available Heat Templates
- Heat Client Setup
  - Installation on Ubuntu
- Connecting to OpenStack Cloud
  - Get Network Resource Name
  - Get Image Resource Name
- Launching the Heat Stack

Single node deployments are a great way to setup an SaltStack-Formulas cloud for:
- a service development environment
- an overview of how all of the OpenStack services and roles play together
• a simple lab deployment for testing

It is possible to run full size proof-of-concept deployment on OpenStack with Heat template, the stack has following requirements for cluster deployment:

• At least 200GB disk space
• 70GB RAM

The single-node deployment has following requirements:

• At least 80GB disk space
• 16GB RAM

Available Heat Templates

The app_single environment consists of three nodes.

<table>
<thead>
<tr>
<th>FQDN</th>
<th>Role</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.openstack.local</td>
<td>Salt master node</td>
<td>10.10.10.200</td>
</tr>
<tr>
<td>control.openstack.local</td>
<td>OpenStack control node</td>
<td>10.10.10.201</td>
</tr>
<tr>
<td>compute.openstack.local</td>
<td>OpenStack compute node</td>
<td>10.10.10.202</td>
</tr>
</tbody>
</table>

Heat Client Setup

The preffered way of installing OpenStack clients is isolated Python environment. To creat Python environment and install compatible OpenStack clients, you need to install build tools first.

Installation on Ubuntu

Install required packages:

```bash
$ apt-get install python-dev python-pip python-virtualenv build-essential
```

Now create and activate virtualenv `venv-heat` so you can install specific versions of OpenStack clients.

```bash
$ virtualenv venv-heat
$ source ./venv-heat/bin/activate
```

Use following `requirements.txt`. Clients were tested with Juno and Kilo Openstack versions.

```plaintext
python-cinderclient>=1.3.1,<1.4.0
python-glanceclient>=0.19.0,<0.20.0
#python-heatclient>=0.6.0,<0.7.0
git+https://github.com/tcpcloud/python-heatclient.git@stable/juno#egg=heatclient
python-keystoneclient>=1.6.0,<1.7.0
python-neutronclient>=2.2.6,<2.3.0
python-novaclient>=2.19.0,<2.20.0
python-swiftclient>=2.5.0,<2.6.0
oslo.config>=2.2.0,<2.3.0
oslo.i18n>=2.3.0,<2.4.0
```

(continues on next page)
oslo.serialization>=1.8.0,<1.9.0
oslo.utils>=1.4.0,<1.5.0

Put requirements into file and install them.

$ pip install -r requirements.txt

If everything goes right, you should be able to use openstack clients, heat, nova, etc.

**Connecting to OpenStack Cloud**

Setup OpenStack credentials so you can use openstack clients. You can download openrc file from Openstack dashboard and source it or execute following commands with filled credentials:

```bash
$ vim ~/openrc
```

```bash
export OS_AUTH_URL=https://<openstack_endpoint>:5000/v2.0
export OS_USERNAME=<username>
export OS_PASSWORD=<password>
export OS_TENANT_NAME=<tenant>
```

Now source the OpenStack credentials:

```
$ source openrc
```

To test your sourced variables:

```
$ env | grep OS
```

Some resources required for heat environment deployment.

**Get Network Resource Name**

The public network is needed for setting up both testing heat stacks. The network ID can be found in Openstack Dashboard or by running following command:

```
$ neutron net-list
```

**Get Image Resource Name**

Image ID is required to run OpenStack Salt lab templates, Ubuntu 14.04 LTS is required as config_image and image for one of the supported platforms is required as instance_image, used for OpenStack instances. To lookup for actual installed images run:

```
$ glance image-list
```

**Launching the Heat Stack**

Download heat templates from this repository.
$ git clone git@github.com:openstack/salt-formulas.git
$ cd doc/source/_static/scripts/

Now you need to customize env files for stacks, see examples in envs directory doc/source/_static/scripts/envs and set required parameters.

Full examples of env files for the two respective stacks:

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home SaltStack-Formulas Project Introduction

Quick Deploy on AWS with CloudFormations

- AWS Client Setup
- Connecting to Amazon Cloud
  - Get the Access Keys
- Launching the CFN Stack

AWS Client Setup

If you already have pip and a supported version of Python, you can install the AWS CLI with the following command:

$ pip install awscli --upgrade --user

You can then verify that the AWS CLI installed correctly by running aws --version.

$ aws --version

Connecting to Amazon Cloud

Get the Access Keys

Access keys consist of an access key ID and secret access key, which are used to sign programmatic requests that you make to AWS. If you don’t have access keys, you can create them from the AWS Management Console. We recommend that you use IAM access keys instead of AWS account root user access keys. IAM lets you securely control access to AWS services and resources in your AWS account.

For general use, the aws configure command is the fastest way to set up your AWS CLI installation.
Launching the CFN Stack

After successful login you can test the credentials.

```bash/aws create-stack --stack-name
```

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home SaltStack-Formulas Project Introduction

Quick Deploy with Vagrant

- **Vagrant Setup**
- **Environment Setup**
  - Minion Configuration Files
  - Vagrant Configuration File
- **Launching Vagrant Nodes**
  - Salt master Bootstrap

Single deployments are a great way to setup an infrastructure for:

- a service development environment
- an overview of how all of the OpenStack services and roles play together
- a simple lab deployment for testing

Although single builds aren’t suitable for large production deployments, they’re great for small proof-of-concept deployments.

It’s strongly recommended to have hardware that meets the following requirements before starting an AIO deployment:

- CPU with hardware-assisted virtualization support
- At least 20GB disk space
- 2GB RAM
Vagrant Setup

Installing Vagrant is extremely easy for many operating systems. Go to the Vagrant downloads page and get the appropriate installer or package for your platform. Install the package using standard procedures for your operating system.

The installer will automatically add vagrant to your system path so that it is available in shell. Try logging out and logging back in to your system (this is particularly necessary sometimes for Windows) to get the updated system path up and running.

Add the generic ubuntu1604 image for virtualbox virtualization.

```
$ vagrant box add ubuntu/xenial64

===> box: Loading metadata for box 'ubuntu/xenial64'
    box: URL: https://atlas.hashicorp.com/ubuntu/xenial4

===> box: Adding box 'ubuntu/xenial64' (v20160122.0.0) for provider: virtualbox
    box: Downloading: https://vagrantcloud.com/ubuntu/boxes/xenial64/versions/20160122.0.0/providers/virtualbox.box

===> box: Successfully added box 'ubuntu/xenial64' (v20160122.0.0) for 'virtualbox'!
```

Environment Setup

The environment consists of three nodes.

<table>
<thead>
<tr>
<th>FQDN</th>
<th>Role</th>
<th>IP</th>
</tr>
</thead>
<tbody>
<tr>
<td>config.cluster.local</td>
<td>Salt master node</td>
<td>10.10.10.200</td>
</tr>
<tr>
<td>service.cluster.local</td>
<td>Managed node</td>
<td>10.10.10.201</td>
</tr>
</tbody>
</table>

Minion Configuration Files

Download salt-formulas

Look at configuration files for each node deployed.

scripts/minions/config.conf configuration:

```yaml
id: config.cluster.local
master: 10.10.10.200
```

scripts/minions/service.conf configuration:

```yaml
id: service.cluster.local
master: 10.10.10.200
```

Vagrant Configuration File

The main vagrant configuration for SaltStack-Formulas deployment is located at scripts/Vagrantfile.
boxes = {
  'ubuntu/xenial64' => {
    'name' => 'ubuntu/xenial64',
    'url' => 'ubuntu/xenial64'
  },
}

Vagrant.configure("2") do |config|
  config.vm.define :cluster_config do |cluster_config|
    cluster_config.vm.hostname = 'config.cluster.local'
    cluster_config.vm.box = 'ubuntu/xenial64'
    cluster_config.vm.box_url = boxes['ubuntu/xenial64']['url']
    cluster_config.vm.network :private_network, ip: "10.10.10.200"
    cluster_config.vm.provider :virtualbox do |vb|
      vb.customize ["modifyvm", :id, "--memory", 512]
      vb.customize ["modifyvm", :id, "--cpus", 1]
      vb.name = 'cluster-config'
      vb.gui = false
    end
    cluster_config.vm.provision :salt do |salt|
      salt.minion_config = "minions/config.conf"
      salt.colorize = true
      salt.bootstrap_options = "-F -c /tmp -P"
    end
  end

  config.vm.define :cluster_service do |cluster_service|
    cluster_service.vm.hostname = 'service.cluster.local'
    cluster_service.vm.box = 'ubuntu/xenial64'
    cluster_service.vm.box_url = boxes['ubuntu/xenial64']['url']
    cluster_service.vm.network :private_network, ip: "10.10.10.201"
    cluster_service.vm.provider :virtualbox do |vb|
      vb.customize ["modifyvm", :id, "--memory", 4096]
      vb.customize ["modifyvm", :id, "--cpus", 1]
      vb.name = 'cluster-service'
      vb.gui = false
    end
    cluster_service.vm.provision :salt do |salt|
      salt.minion_config = "minions/service.conf"
      salt.colorize = true
      salt.bootstrap_options = "-F -c /tmp -P"
    end
  end
end
Launching Vagrant Nodes

Check the status of the deployment environment.

```
$ cd /srv/vagrant-cluster
$ vagrant status

Current machine states:

cluster_config not created (virtualbox)
cluster_service not created (virtualbox)
```

Setup config node, launch it and connect to it using following commands, it cannot be provisioned by vagrant salt, as the salt master is not configured yet.

```
$ vagrant up cluster_config
$ vagrant ssh cluster_config
```

Salt Master Bootstrap

Bootstrap the salt master service on the config node, it can be configured with following parameters:

```
$ export RECLASS_ADDRESS=https://github.com/salt-formulas/salt-formulas-model.git
$ export CONFIG_HOST=config.cluster.local
```

To deploy salt-master from packages, run on config node:

```
$ /vagrant/bootstrap/salt-master-setup.sh
```

Now setup the server node. Launch it using following command:

```
$ vagrant up cluster_service
```

To orchestrate all defined services across all nodes, run following command on config node:

```
$ salt-run state.orchestrate orchestrate
```
2.1.3 Metadata Modelling

Chapter 3. Metadata Authoring Guidelines

Home SaltStack-Formulas Project Introduction

Model-driven Architectures

- Core Principles
- Sample Model Architecture
- Horizontally Scaled Services

We have the formula structures covered, now we can proceed to define how the metadata is modelled and key patterns we need to know to build nice standard models.

Model Driven Architecture (MDA) is an answer to growing complexity of systems controlled by configuration management and orchestration tools. It provides unified node classification with atomic service definitions.

Core Principles

Following table shows the core principles for creating model-driven architectures.

<table>
<thead>
<tr>
<th>Atomicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services are separated with such affinity that allows running them on single node.</td>
<td></td>
</tr>
<tr>
<td>Reusability / Re-</td>
<td>Different services serving the same role can be replaced without affecting connected services.</td>
</tr>
<tr>
<td>placibility</td>
<td></td>
</tr>
<tr>
<td>Service Roles</td>
<td>Services may implement multiple roles, these can be then separated to individual nodes.</td>
</tr>
<tr>
<td>Dynamic Resources</td>
<td>Service metadata is always available for definition of dynamic resources.</td>
</tr>
<tr>
<td>Change Management</td>
<td>The strength lies not in describing the static topology of services but more the process of ongoing updates.</td>
</tr>
</tbody>
</table>

Sample Model Architecture

Following figure show sample system that has around 10 services with some outsourced by 3rd party service providers.

We can identify several subsystem layers within this complex application system.

- Proxy service - Distributing load to application layer
- Application service - Application with caches
- Data persistence - Databases and filesystem storage

Horizontally Scaled Services

Certain services span across multiple application systems. These usually play critical roles in system maintenance and are essential for smooth ongoing operations.

These services usually fit into one of following categories:
2.1. Project Introduction
<table>
<thead>
<tr>
<th>Access / Control</th>
<th>SSH access, orchestration engine access, user authentication.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monitoring</td>
<td>Events and metric collections, alarms, dashboards and notifications.</td>
</tr>
<tr>
<td>Essential</td>
<td>Name services, time services, mail transports, etc …</td>
</tr>
</tbody>
</table>

These horizontal services are not entirely configured directly but rather reuse the metadata of other surrounding services to configure itself (for example metering agent collects metrics to collect for metadata of surrounding services on the same node, node exports also metadata for external metric collector to pick).

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home SaltStack-Formulas Project Introduction

Standard Metadata Layout

- Basic Functional Units (Service Class Level)
- Business Function Unit (System Class Level)
- Product Deployments (Cluster Class Level)
- Node/Cluster Classification (Node Level)

Metadata models are separated into 3 individual layers: service, system and cluster. The layers are firmly isolated from each other and can be aggregated on south-north direction and using service interface agreements for objects on the same level. This approach allows to reuse many already created objects both on service and system layers as building blocks for a new solutions and deployments following the fundamental MDA principles.

Basic Functional Units (Service Class Level)

The services are the atomic units of config management. SaltStack formula or Puppet recipe with default metadata set can be considered as a service. Each service implements one or more roles and together with other services form systems. Following list shows decomposition

- Formula - Set of states that perform together atomic service
- State - Declarative definition of various resources (package, files, services)
- Module - Imperative interaction enforcing defined state for each State

Meta-data fragments for individual services are stored in salt formulas and can be reused in multiple contexts. Service level roles set the granularity of service to certain level, role is limited to 1 virtual machine or container aggregation. Service models are used to provide models with defaults for various contexts. This the low level modelling, where models are directly mapped to the Salt formula functions and get projected to the actual nodes.

Given Redis formula from Gitlab example we set basic set of parameters that can be used for actual service configuration as well as support services configuration.

Basic service metadata is present in metadata/service directory of every service formula.
For example RabbitMQ service in various deployments.

```
rabbitmq/
  `-- metadata/
    `-- service/
      `-- server/
        |-- single.yml
        `-- cluster.yml
```

The metadata fragment `/srv/salt/reclass/classes/service/service-formula` maps to `/srv/salt/env/formula-name/metadata/service` so then you can easily refer the metadata as `service.formula-name.role1.deployment1` class for example.

Example `metadata/service/server/cluster.yml` for the cluster setup PostgreSQL server.

```
parameters:
  postgresql:
    server:
      enabled: true
      bind:
        address: '127.0.0.1'
        port: 5432
        protocol: tcp
      clients:
      - '127.0.0.1'
    cluster:
      enabled: true
      members:
      - node01
      - node02
      - node03
```
Example `metadata/service/server/cluster.yml` for the single PostgreSQL server.

```yaml
parameters:
  postgresql:
    server:
      enabled: true
      bind:
        address: '0.0.0.0'
        port: 5432
        protocol: tcp
```

Example `metadata/service/server/cluster.yml` for the standalone PostgreSQL server.

```yaml
parameters:
  postgresql:
    server:
      enabled: true
      bind:
        address: '127.0.0.1'
        port: 5432
        protocol: tcp
    clients:
    - '127.0.0.1'
```

There are about 140 formulas in several categories. You can look at complete Formula Ecosystem chapter.

**Business Function Unit (System Class Level)**

Aggregation of services performing given role in business IT infrastructure. System level models are the sets of the ‘services’ combined in a such way that the result of the installation of these services will produce a ready-to-use application (system) on integration level. In the ‘system’ model, you can not only include the ‘services’, but also override some ‘services’ options to get the system with the expected functionality.

![System - Service Composition](image)

The systems are usually one of the following type:

**Single**

Usually all-in-one application system on a node (Taiga, Gitlab)

**Multi**

Multiple all-in-one application systems on a node (Horizon, Wordpress)

**Cluster**

Service is part of a cluster (OpenStack controllers, large-scale web applications)
Container

Service is run as Docker container

For example, in the service ‘haproxy’ there is only one port configured by default (haproxy_admin_port: 9600), but the system ‘horizon’ add to the service ‘haproxy’ several new ports, extending the service model and getting the system components integrated with each other.

```
system/
  |-- business-system/
     |   |-- role1/
     |     |   |-- deployment1.yml
     |     |   |-- deployment2.yml
     |   `-- role2/
        |   |-- deployment3.yml
```

For example Graphite server with Carbon collector.

```
system/
  |-- graphite/
     |   |-- server/
     |       |   |-- single.yml
     |       |   |-- cluster.yml
     |   `-- collector/
        |   |-- single.yml
        |   `-- cluster.yml
```

Example classes/system/graphite/collector/single.yml for the standalone Graphite Carbon installation.

```
classes:
  - service.memcached.server.local
  - service.graphite.collector.single
parameters:
  _param:
    rabbitmq_monitor_password: password
  carbon:
    relay:
      enabled: false
```

Example classes/system/graphite/collector/single.yml for the standalone Graphite web server installation. Where you combine your individual formulas to functional business unit of single node scope.

```
classes:
  - service.memcached.server.local
  - service.postgresql.server.local
  - service.graphite.server.single
  - service.apache.server.single
  - service.supervisor.server.single
parameters:
  _param:
    graphite_secret_key: secret
    postgresql_graphite_password: password
    apache2_site_graphite_host: ${_param:single_address}
    rabbitmq_graphite_password: password
    rabbitmq_monitor_password: password
    rabbitmq_admin_password: password
    rabbitmq_secret_key: password
  apache:
```

(continues on next page)
Product Deployments (Cluster Class Level)

Cluster/deployment level aggregating systems directly referenced by individual host nodes or container services. Cluster is the set of models that combine the already created ‘system’ objects into different solutions. We can override any settings of ‘service’ or ‘system’ level from the ‘cluster’ level with the highest priority.

Also, for salt-based environments here are specified the list of nodes and some specific parameters for different nodes (future ‘inventory’ files for salt, future generated pillars that will be used by salt formulas). The actual mapping is defined, where each node is member of specific cluster and is implementing specific role(s) in systems.

If we want not just to re-use an object, we can change its behaviour depending of the requirements of a solution. We define basic defaults on service level, then we can override these default params for specific system needs and then if needed provide overrides per deployment basis. For example, a database engine, HA approaches, IO scheduling policy for kernel and other settings may vary from one solution to another.

Default structure for cluster level has following structure:

```
cluster/
 `-- deployment1/
```

(continues on next page)
Where deployments is usually one datacenter, product realise full business units [OpenStack cloud, Kubernetes cluster, etc]

For example deployment Graphite server with Carbon collector.

Example demo-lab/monitoring/monitor.yml class implementing not only Graphite services but also grafana server and sensu server.

Cluster level classes can be shared by members of the particular cluster or by single node.

Node/Cluster Classification (Node Level)

Servers contain one or more systems that bring business value and several maintenance systems that are common to any node. Services running on single host can be viewed as at following picture.

Nodes generally include cluster level classes which include relevant system classes and these include service level classes which configure individual formulas to work.

The previous figure shows the real composition of individual metadata fragments that form the complete service catalog for each managed node.
ReClass - Recursive Classification

- Resources
- Core Functions
  - Deep Data Merging
  - Parameter Interpolation
- Overriding the Metadata
  - The ‘Soft’ Metadata
  - The ‘Hard’ Metadata

reclass in node centric classifier for any configuration management. When reclass parses a node or class definition and encounters a parent class, it recurses to this parent class first before reading any data of the node (or class). When reclass returns from the recursive, depth first walk, it then merges all information of the current node (or class) into the information it obtained during the recursion.

This means any class may define a list of classes it derives metadata from, in which case classes defined further down the list will be able to override classes further up the list.

Resources

Original reclass implementation:
- https://github.com/madduck/reclass

Forked reclass with many additional features:
- https://github.com/salt-formulas/reclass
- https://github.com/salt-formulas/reclass/tree/develop (develop branch)

Core Functions

reclass is very simple and there are only two main concepts.
Deep Data Merging

When retrieving information about a node, reclass first obtains the node definition from the storage backend. Then, it iterates the list of classes defined for the node and recursively asks the storage backend for each class definition. Next, reclass recursively descends each class, looking at the classes it defines, and so on, until a leaf node is reached, i.e. a class that references no other classes.

Now, the merging starts. At every step, the list of applications and the set of parameters at each level is merged into what has been accumulated so far.

Merging of parameters is done “deeply”, meaning that lists and dictionaries are extended (recursively), rather than replaced. However, a scalar value does overwrite a dictionary or list value. While the scalar could be appended to an existing list, there is no sane default assumption in the context of a dictionary, so this behaviour seems the most logical. Plus, it allows for a dictionary to be erased by overwriting it with the null value.

After all classes (and the classes they reference) have been visited, reclass finally merges the applications list and parameters defined for the node into what has been accumulated during the processing of the classes, and returns the final result.

Parameter Interpolation

Parameters may reference each other, including deep references, e.g.:

After merging and interpolation, which happens automatically inside the storage modules, the `python-application:server:database:host` parameter will have a value of “hostname.domain.com”.

Types are preserved if the value contains nothing but a reference. Hence, the value of `dict_reference` will actually be a dictionary.

Overriding the Metadata

The reclass deals with complex data structures we call ‘hard’ metadata, these are defined in class files mentioned in previous text. These are rather complex structures that you don’t usually need to manage directly, but a special dictionary for so called ‘soft’ metadata was introduced, that holds simple list of most frequently changed properties of the ‘hard’ metadata model. It uses the parameter interpolation function of reclass to achieve defining parameter at single location.

The ‘Soft’ Metadata

In reclass storage is a special dictionary called `_param`, which contains keys that are interpolated to the ‘hard’ metadata models. These soft parameters can be defaulted at system level or on cluster level and or changed at the node definition. With some modifications to formulas it will be also possible to have ETCD key-value store to replace or amended the `_params` dictionary.

<table>
<thead>
<tr>
<th>parameters:</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>_param</code>:</td>
</tr>
<tr>
<td>service_database_host: hostname.domain.com</td>
</tr>
</tbody>
</table>

All of these values are preferably scalar and can be referenced as `$_param:service_database_host` parameter. These metadata are considered cluster level readable and can be overriden by reclass. `set_cluster_param name value module`. 

2.1. Project Introduction
Fig. 2: Parameter interpolation of *soft* parameters to *hard* metadata models
The ‘Hard’ Metadata

This metadata are the complex metadata structures that can contain interpolation stings pointing to the ‘soft’ metadata or containing precise values.

```yaml
parameters:
  python-application:
    server:
      database:
        name: database_name
        host: _param:service_database_host
```

When working with metadata a lot of common patterns emerge over the time. The formulas reuse these patterns to maintain the cross-formula consistency.
Creating Service Metadata

Following points are selected as the most frequently asked questions and try to explain the design patterns behind our metadata modes.

Service Formula Roles

The service roles provide level of separation for the formulas, if your service can be split across multiple nodes you should use the role. You can imagine role as simple kubernetes Kubernetes Pods. For example a sensu formula has following roles defined:

server

Definition of server service that sends commands to the clients and consumes the responses.

client

Client role is installed on each of the client nodes and uses the support metadata concept to get the metadata from installed services.

dashboard

Optional definition of Uchiwa dashboard.

You monitoring node can have all 3 roles running on single node, and that is completely OK.

Scalar Parameters

Always keep in mind that we model the resources not the configurations. However tempting can it be to just iterate the config dictionaries and adding all the values, it is not recommended. This approach prevents further parameter schema definition as well as allowing to add the defaults, etc.

Don’t do following snippet, it may save you some at the start but with at the price of untestable and unpredictable results:

```
Warning:

service:
  role:
    config:
      option1: value1
      ...
      optionN: valueN
```

Common Service Metadata

When some metadata is reused by multiple roles it is possible to add the new virtual common role definition. This common metadata should be then available to every role.

The definition in the pillar metadata file:

```
service:
  common:
    var1: value1
```

(continues on next page)
Modelling and Iterating Resource Sets

Resource sets are resources provided by the service formula, for example for MySQL and PostgreSQL formula database is a resource set, for NGINX or Apache formula a member of resource set is vhost. Users, repositories, packages, jobs, interfaces, routes, mounts etc in the Linux formula are also good example of this pattern.

```
mysql:
  server:
    database:
      database_name:
        param1: value1
        param2: value2
```

Following snippet shows defined virtual hosts for the Nginx.

```
nginx:
  server:
    vhost:
      vhost_name:
        param1: value1
        param2: value2
```

Service Network Binding

You can define the address and port on which the service listens in simple way. For single network binding you can use following code.

```
memcached:
  server:
    enabled: true
    maxconn: 8192
    bind:
      address: 0.0.0.0
      port: 11211
      protocol: tcp
```

Service Backend Structure

When service plugin mechanism allows to add arbitrary plugins to the individual roles, it is advised to use following format. Following snippet shows multiple defined backends, in this case it’s pillar data sources.
salt:
  master:
    pillar:
      engine: composite
      reclass:
        index: 1
        storage_type: yaml_fs
        inventory_base_uri: /srv/salt/reclass
        propagate_pillar_data_to_reclass: False
        ignore_class_notfound: False
      saltclass:
        path: /srv/salt/saltclass
      nacl:
        index: 99

Note: The reason for existence of `engine` parameter is to separate various implementations. For relational databases we can determine what specific database is used to construct proper connection strings.

Client Relationship

The client relationship has form of a dictionary. The name of the dictionary represents the required role [database, cache, identity] and the `engine` parameter then refers to the actual implementation. Following snippet shows single service to service relation.

```yaml
keystone:
  server:
    message_queue:
      engine: rabbitmq
      host: 200.200.200.200
      port: 5672
      user: openstack
      password: redacted
      virtual_host: '/openstack'
      ha_queues: true
```

Following snippet shows backend with multiple members.

```yaml
keystone:
  server:
    cache:
      engine: memcached
      members:
        - host: 200.200.200.200
          port: 11211
        - host: 200.200.200.201
          port: 11211
        - host: 200.200.200.202
          port: 11211
```

SSL Certificates

Multiple service use SSL certificates. There are several possible ways how to obtain a certificate.
TODO

Using Service Support Metadata

You can think of support metadata as the k8s annotations for other services to pickup and be configured accordingly. This concept is heavily used in the definition of monitoring, documentation, etc.

Basics of Support Metadata

In formula there’s `meta` directory, each service that needs to extract some data has file with `service.yml` for example `collectd.yml, telegrag.yml`.

Service Documentation

Following snippet shows how we can provide metadata for dynamic documentation creation for Glance service.

```yaml
doc:
  name: Glance
  description: The Glance project provides services for discovering, registering, and retrieving virtual machine images.
  role:
    {% if pillar.glance.server is defined %}
    {% from "glance/map.jinja" import server with context %}
    server:
      name: server
      endpoint:
        glance_api:
          name: glance-api
          type: glance-api
          address: http://{{ server.bind.address }}:{{ server.bind.port }}
          protocol: http
        glance_registry:
          name: glance-registry
          type: glance-registry
          address: http://{{ server.registry.host }}:{{ server.registry.port }}
          protocol: http
    param:
      bind:
        value: {{ server.bind.address }}:{{ server.bind.port }}
      version:
        name: "Version"
        value: {{ server.version }}
      workers:
        name: "Number of workers"
        value: {{ server.workers }}
      database_host:
        name: "Database"
        value: {{ server.database.user }}@{{ server.database.host }}:{{ server.database.port }}/{{ server.database.name }}
      message_queue_ip:
        name: "Message queue"
        value: {{ server.message_queue.user }}@{{ server.message_queue.host }}:{{ server.message_queue.port }}
      identity_host:
    {%- else %}
    {%- endif %}
```

(continues on next page)
Service monitoring checks

Let's have our memcached service and look how the monitoring is defined for this service.

We start with definitions of metric collections.

```jinja
{% from "memcached/map.jinja" import server with context %}
{% if server.get('enabled', False) %}
agent:
  input:
    procstat:
      process:
        memcached:
          exe: memcached
          servers:
            - address: {{ server.bind.address | replace("0.0.0.0", "127.0.0.1") }}
            port: {{ server.bind.port }}
{% endif %}
```

We also define the functional monitoring for the collected metrics.

```jinja
{% from "memcached/map.jinja" import server with context %}
{% if server.get('enabled', False) %}
server:
  alert:
    MemcachedProcessDown:
      if: >=
        procstat_running(process_name="memcached") == 0
      {% raw %}
        severity: warning
        service: memcached
        annotations:
          summary: 'Memcached service is down'
          description: 'Memcached service is down on node {{ $labels.host }}'
      {% endraw %}
{% endif %}
```

Also the definition of the dashboard for the collected metrics is provided.
This snippet appends panel to the main dashboard at grafana and creates a new dashboard. The prometheus and influxdb time-series are supported out of box throughout all formulas.

**Virtual Machines versus Containers**

The containers and services share great deal of parameters, but the way they are delivered is different across various container platforms.

**Virtual machine service deployment models**

- local deployment
• single deployment
• cluster deployment

**Container metadata requirements**

• Metadata for docker swarm
• Metadata for kubenetes

• Documentation Home
• Project Introduction
• Installation and Operations Manual
• Development Documentation

Home SaltStack-Formulas Project Introduction

**Working with Metadata**

• **Scaling Metadata Models**
  – *Shared Cluster and System Level*
  – *Separate Cluster with Single System Level*
  – *Separate Cluster with Multiple System Levels*

• **Handling Sensitive Metadata**

• **Creating new Models**
  – *Creating a New Formula (Service Level)*
  – *Creating New Business Units (System Level)*
  – *Creating New Deployments (Cluster Level)*

• **Making Changes to Existing Models**
  – *Updating Existing Formula (Service Level)*
  – *Updating Business Unit (System Level)*
  – *Updating Deployment Configurations (Cluster Level)*

Every IT solution can be described by using several layers of objects, where the objects of higher layer are combinations of the objects from lower layers. For example, we may install `apache server` and call it `apache service`, but there are objects that contain multiple services like `apache service`, `mysql service`, and some python scripts (for example keystone), we will call these “keystone system” or “freeipa system” and separate them on a higher (System) layer. The systems represent units of business logic and form working components. We can map systems to individual deployments, where “openstack cluster” consists of “nova system”, “neutron system” and others OpenStack systems and “kubernetes cluster” consists of “etcd system”, “calico system” and few others. We can define and map PaaS, IaaS or SaaS solutions of any size and complexity.
Fig. 3: Decomposition of services, systems and clusters

This model has been developed to cope with huge scopes of services, consisting of hundreds of services running VMs and containers across multiple physical servers or locations. Following text takes apart individual layers and explains them in further detail.

Scaling Metadata Models

Keeping consistency across multiple models/deployments has proven to be the most difficult part of keeping things running smooth over time with evolving configuration management. You have multiple strategies on how to manage your metadata for different scales.

The service level metadata can be handled in common namespace not by formulas itself, but it is recommended to keep the relevant metadata states.

Shared Cluster and System Level

If every deployment only defined on system level, you need to keep copy of all system definitions at all deployments. This is suitable only for small number of deployments.

Separate Cluster with Single System Level

With introduction of new features and services shared deployments does not provide necessary flexibility to cope with the change. Having service metadata provided along formulas helps to deliver the up-to-date models to the deployments, but does not reflect changes on the system level. Also the need of multiple parallel deployments lead to adjusting the structure of metadata with new common system level and only cluster level for individual deployment(s). The cluster layer only contains soft parametrization and class references.
Separate Cluster with Multiple System Levels

When customer is reusing the provided system, but also has formulas and system on its own. Customer is free to create its own system level classes.

In this setup a customer is free to reuse the generic formulas with generic systems. At the same time he’s free to create formulas of it’s own as well as custom systems.

Handling Sensitive Metadata

Sensitive data refers to any information that you would not wish to share with anyone accessing a server. This could include data such as passwords, keys, or other information. For sensitive data we use the GPG renderer on salt master to cipher all sensitive data.

To generate a cipher from a secret use following command:

```bash
$ echo -n "supersecret" | gpg --homedir --armor --encrypt -r <KEY-name>
```

The ciphered secret is stored in block of text within PGP MESSAGE delimiters, which are part of cipher.

```plaintext
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1
-----END PGP MESSAGE-----
```

Following example shows full use of generated cipher for virtually any secret.

```plaintext
parameters:
  _param:
    rabbitmq_secret_key: |
    -----BEGIN PGP MESSAGE-----
    Version: GnuPG v1
    -----END PGP MESSAGE-----
    hQEMAweRHkPCfNeAq9GLTn16hCfXabPwU6bBkB0unOc7i9/etGuVc5CyU9Q6um
    QuetdVQrLO/3krC41geNQdM6D9E8FKonMlgJPy0vC8ggxhj/0/IFFEKmsnv2k6+
    cnEfVexSy7o/U1VOVjyUellMCIJ1az/30RXaME49Cpi6No2+vKD8a4q4n2N1UZcG
    RhkhC0S22zNxOXQ38TBkmtJcpxrqT6YWRTU sjVubW3bVC+uZHgqJHu79wmuN8tz
```

(continues on next page)
As you can see the GPG encrypted parameters can be further referenced with reclass interpolation ${_param:rabbitmq_secret_key} statement.

Creating new Models

Following text shows steps that need to be undertaken to implement new functionality, new system or entire deployment:

Creating a New Formula (Service Level)

If some of required services are missing, you can create a new service formula for Salt with the default model that describe the basic setup of the service. The process of creating new formula is streamlined by Using Cookiecutter and after the formula is created you can check Formula Authoring Guidelines chapter for further instructions.

If you download formula to salt master, you can point the formula metadata to the proper service level directory:

```
ln -s <service_name>/metadata/service /srv/salt/reclass/classes/service/<service_name>
```

And symlink of the formula content to the specific salt-master file root:

```
ln -s <service_name>/<service_name> /srv/salt/env/<env_name>/<service_name>
```

Creating New Business Units (System Level)

If some ‘system’ is missing, then you can create a new ‘system’ from the set of ‘services’ and extend the ‘services’ models with necessary settings for the system (additional ports for haproxy, additional network interfaces for linux, etc). Do not introduce too much of hard metadata on the system level, try to use class references as much as possible.

Creating New Deployments (Cluster Level)

Determine which products are being used in the selected deployment, you can have infrastructure services, applications, monitoring products defined at once for single deployment. You need to make sure that all necessary systems was already created and included into global system level, then it can be just referenced. Follow the guidelines further up in this text.

Making Changes to Existing Models

When you have decided to add or modify some options in the existing models, the right place of the modification should be considered depending of the impact of the change:
Updating Existing Formula (Service Level)

Change the model in salt-formula-<service-name> for some service-specific improvements. For example: if the change is related to the change in the new package version of this service; the change is fixing some bug or improve performance or security of the service and should be applied for every cluster. In most cases we introduce new resources or configuration parameters.

Example where the common changes can be applied to the service: https://github.com/openstack/salt-formula-horizon/tree/master/metadata/service/server/

Updating Business Unit (System Level)

Change the system level for a specific application, if the base services don’t provide required configurations for the application functionality. Example where the application-related change can be applied to the service,

Updating Deployment Configurations (Cluster Level)

Changes on the cluster level are related to the requirements that are specific for this particular cluster solution, for example: number and names of nodes; domain name; IP addresses; network interface names and configurations; locations of the specific ‘systems’ on the specific nodes; and other things that are used by formulas for services.

2.1.4 Service Ecosystem

Chapter 4. Service Ecosystem

Actual Service Ecosystem

The SaltStack-Formulas service are divided into several groups according to the target role in the system. All services/formulas share same structure and metadata definitions, expose vital information into the Salt Mine for further processing.

Infrastructure services  Core services needed for basic infrastructure operation.
Supplemental services  Support services as databases, proxies, application servers.
OpenStack services  All supported OpenStack cloud platform services.

Monitoring services  Monitoring, metering and log collecting tools implementing complete monitoring stack.

Integration services  Continuous integration services for automated integration and delivery pipelines.

Each of the service groups contain of several individual service formulas, listed in following tables.

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home  SaltStack-Formulas Project Introduction

Infrastructure Services

Core services needed for basic infrastructure operation.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>aptcacher</td>
<td><a href="https://github.com/salt-formulas/salt-formula-aptcacher">https://github.com/salt-formulas/salt-formula-aptcacher</a></td>
</tr>
<tr>
<td>backupninja</td>
<td><a href="https://github.com/salt-formulas/salt-formula-backupninja">https://github.com/salt-formulas/salt-formula-backupninja</a></td>
</tr>
<tr>
<td>ceph</td>
<td><a href="https://github.com/salt-formulas/salt-formula-ceph">https://github.com/salt-formulas/salt-formula-ceph</a></td>
</tr>
<tr>
<td>chrony</td>
<td><a href="https://github.com/salt-formulas/salt-formula-chrony">https://github.com/salt-formulas/salt-formula-chrony</a></td>
</tr>
<tr>
<td>freeipa</td>
<td><a href="https://github.com/salt-formulas/salt-formula-freeipa">https://github.com/salt-formulas/salt-formula-freeipa</a></td>
</tr>
<tr>
<td>git</td>
<td><a href="https://github.com/salt-formulas/salt-formula-git">https://github.com/salt-formulas/salt-formula-git</a></td>
</tr>
<tr>
<td>glusterfs</td>
<td><a href="https://github.com/salt-formulas/salt-formula-glusterfs">https://github.com/salt-formulas/salt-formula-glusterfs</a></td>
</tr>
<tr>
<td>iptables</td>
<td><a href="https://github.com/salt-formulas/salt-formula-iptables">https://github.com/salt-formulas/salt-formula-iptables</a></td>
</tr>
<tr>
<td>letsecrypt</td>
<td><a href="https://github.com/salt-formulas/salt-formula-letsecrypt">https://github.com/salt-formulas/salt-formula-letsecrypt</a></td>
</tr>
<tr>
<td>linux</td>
<td><a href="https://github.com/salt-formulas/salt-formula-linux">https://github.com/salt-formulas/salt-formula-linux</a></td>
</tr>
<tr>
<td>network</td>
<td><a href="https://github.com/salt-formulas/salt-formula-network">https://github.com/salt-formulas/salt-formula-network</a></td>
</tr>
<tr>
<td>nfs</td>
<td><a href="https://github.com/salt-formulas/salt-formula-nfs">https://github.com/salt-formulas/salt-formula-nfs</a></td>
</tr>
<tr>
<td>ntp</td>
<td><a href="https://github.com/salt-formulas/salt-formula-ntp">https://github.com/salt-formulas/salt-formula-ntp</a></td>
</tr>
<tr>
<td>openssh</td>
<td><a href="https://github.com/salt-formulas/salt-formula-openssh">https://github.com/salt-formulas/salt-formula-openssh</a></td>
</tr>
<tr>
<td>openvpn</td>
<td><a href="https://github.com/salt-formulas/salt-formula-openvpn">https://github.com/salt-formulas/salt-formula-openvpn</a></td>
</tr>
<tr>
<td>pritunl</td>
<td><a href="https://github.com/salt-formulas/salt-formula-pritunl">https://github.com/salt-formulas/salt-formula-pritunl</a></td>
</tr>
<tr>
<td>reclass</td>
<td><a href="https://github.com/salt-formulas/salt-formula-reclass">https://github.com/salt-formulas/salt-formula-reclass</a></td>
</tr>
<tr>
<td>salt</td>
<td><a href="https://github.com/salt-formulas/salt-formula-salt">https://github.com/salt-formulas/salt-formula-salt</a></td>
</tr>
<tr>
<td>sphinx</td>
<td><a href="https://github.com/salt-formulas/salt-formula-sphinx">https://github.com/salt-formulas/salt-formula-sphinx</a></td>
</tr>
<tr>
<td>squid</td>
<td><a href="https://github.com/salt-formulas/salt-formula-squid">https://github.com/salt-formulas/salt-formula-squid</a></td>
</tr>
</tbody>
</table>

aptcacher

Apt-Cacher NG is a caching HTTP proxy intended for use with download clients of system distribution’s package managers.

Sample pillars

Single apt-cacher service
aptcacher:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 3142

More advanced setup with Proxy and passthru patterns

aptcacher:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 3142
    proxy: 'http://proxy-user:proxy-pass@proxy-host:9999'
  passthruurl:
    - 'repos.influxdata.com'
    - 'packagecloud.io'
    - 'packagecloud-repositories.s3.dualstack.us-west-1.amazonaws.com'
    - 'launchpad.net'
    - 'apt.dockerproject.org'
  passthrupattern:
    - '.key$'
    - '.gpg$'
    - '.pub$'
    - '.jar$'

Read more

- https://www.unix-ag.uni-kl.de/~bloch/acng/

Backupninja formula

Backupninja allows you to coordinate system backup by dropping a few simple configuration files into /etc/backup.d/. Most programs you might use for making backups don’t have their own configuration file format.

Backupninja provides a centralized way to configure and schedule many different backup utilities. It allows for secure, remote, incremental filesystem backup (via rdiff-backup), compressed incremental data, backup system and hardware info, encrypted remote backups (via duplicity), safe backup of MySQL/PostgreSQL databases, subversion or trac repositories, burn CD/DVDs or create ISOs, incremental rsync with hardlinking.

Sample pillars

Backup client with ssh/rsync remote target

backupninja:
  client:
    enabled: true
  target:
    engine: rsync
    host: 10.10.10.208
    user: backupninja
Backup client with s3 remote target

```yaml
backupninja:
    client:
        enabled: true
        target:
            engine: dup
            url: s3+http://bucket-name/folder-name
            auth:
                awsaccesskeyid: awsaccesskeyid
                awssecretaccesskey: awssecretaccesskey
```

Backup client with webdav target

```yaml
backupninja:
    client:
        enabled: true
        target:
            engine: dup
            url: webdavs://backup.cloud.example.com/box.example.com/
            auth:
                gss:
                    principal: host/${linux:network:fqdn}
                    keytab: /etc/krb5.keytab
```

Backup server rsync/rdiff

```yaml
backupninja:
    server:
        enabled: true
        rdiff: true
        key:
            client1.domain.com:
                enabled: true
                key: ssh-key
```

Backup client with local storage

```yaml
backupninja:
    client:
        enabled: true
        target:
            engine: local
```

More information

- https://labs.riseup.net/code/projects/backupninja/wiki/Configuration
- http://www.debian-administration.org/articles/351
- https://github.com/riseuplabs/puppet-backupninja
- http://www.ushills.co.uk/2008/02/backup-with-backupninja.html
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-backupninja/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-backupninja

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Ceph formula

Ceph provides extraordinary data storage scalability. Thousands of client hosts or KVMs accessing petabytes to exabytes of data. Each one of your applications can use the object, block or file system interfaces to the same RADOS cluster simultaneously, which means your Ceph storage system serves as a flexible foundation for all of your data storage needs.

Use salt-formula-linux for initial disk partitioning.

Daemons

Ceph uses several daemons to handle data and cluster state. Each daemon type requires different computing capacity and hardware optimization.

These daemons are currently supported by formula:

- MON (ceph.mon)
- OSD (ceph.osd)
- RGW (ceph.radosgw)

Architecture decisions

Please refer to upstream architecture documents before designing your cluster. Solid understanding of Ceph principles is essential for making architecture decisions described below. http://docs.ceph.com/docs/master/architecture/

- Ceph version

There is 3 or 4 stable releases every year and many of nightly/dev release. You should decide which version will be used since the only stable releases are recommended for production. Some of the releases are marked LTS (Long Term Stable) and these releases receive bugfixed for longer period - usually until next LTS version is released.
- **Number of MON daemons**

Use 1 MON daemon for testing, 3 MONs for smaller production clusters and 5 MONs for very large production cluster. There is no need to have more than 5 MONs in normal environment because there isn’t any significant benefit in running more than 5 MONs. Ceph require MONs to form quorum so you need to have more than 50% of the MONs up and running to have fully operational cluster. Every I/O operation will stop once less than 50% MONs is available because they can’t form quorum.

- **Number of PGs**

Placement groups are providing mapping between stored data and OSDs. It is necessary to calculate number of PGs because there should be stored decent amount of PGs on each OSD. Please keep in mind decreasing number of PGs isn’t possible and increasing can affect cluster performance.


- **Daemon colocation**

It is recommended to dedicate nodes for MONs and RWG since colocation can have and influence on cluster operations. However, small clusters can be running MONs on OSD node but it is critical to have enough of resources for MON daemons because they are the most important part of the cluster.

Installing RGW on node with other daemons isn’t recommended because RGW daemon usually require a lot of bandwidth and it harm cluster health.

- **Store type (Bluestore/Filestore)**

Recent version of Ceph support Bluestore as storage backend and backend should be used if available.

http://docs.ceph.com/docs/master/rados/configuration/bluestore-config-ref/

- **Block.db location for Bluestore**

There are two ways to setup block.db:

  - **Colocated** block.db partition is created on the same disk as partition for the data. This setup is easier for installation and it doesn’t require any other disk to be used. However, colocated setup is significantly slower than dedicated

  - **Dedicate** block.db is placed on different disk than data (or into partition). This setup can deliver much higher performance than colocated but it require to have more disks in servers. Block.db drives should be carefully selected because high I/O and durability is required.

- **Block.wal location for Bluestore**

There are two ways to setup block.wal - stores just the internal journal (write-ahead log):

  - **Colocated** block.wal uses free space of the block.db device.

  - **Dedicate** block.wal is placed on different disk than data (better put into partition as the size can be small) and possibly block.db device. This setup can deliver much higher performance than colocated but it require to have more disks in servers. Block.wal drives should be carefully selected because high I/O and durability is required.

- **Journal location for Filestore**

There are two ways to setup journal:

  - **Colocated** journal is created on the same disk as partition for the data. This setup is easier for installation and it doesn’t require any other disk to be used. However, colocated setup is significantly slower than dedicated

  - **Dedicate** journal is placed on different disk than data (or into partition). This setup can deliver much higher performance than colocated but it require to have more disks in servers. Journal drives should be carefully selected because high I/O and durability is required.
• Cluster and public network

Ceph cluster is accessed using network and thus you need to have decent capacity to handle all the client. There are two networks required for cluster: public network and cluster network. Public network is used for client connections and MONs and OSDs are listening on this network. Second network is called cluster networks and this network is used for communication between OSDs.

Both networks should have dedicated interfaces, bonding interfaces and dedicating vlans on bonded interfaces isn’t allowed. Good practice is dedicate more throughput for the cluster network because cluster traffic is more important than client traffic.

• Pool parameters (size, min_size, type)

You should setup each pool according to it’s expected usage, at least min_size and size and pool type should be considered.

• Cluster monitoring

• Hardware

Please refer to upstream hardware recommendation guide for general information about hardware.

Ceph servers are required to fulfill special requirements because load generated by Ceph can be diametrically opposed to common load.

http://docs.ceph.com/docs/master/start/hardware-recommendations/

Basic management commands

Cluster

• ceph health - check if cluster is healthy (ceph health detail can provide more information)

root@c-01:~# ceph health
HEALTH_OK

• ceph status - shows basic information about cluster

root@c-01:~# ceph status
cluster e2dc51ae-c5e4-48f0-afc1-9e9e97dfd650
  health HEALTH_OK
  monmap e1: 3 mons at {1=192.168.31.201:6789/0,2=192.168.31.202:6789/0,3=192.168.31.203:6789/0}
    election epoch 38, quorum 0,1,2 1,2,3
  osdmap e226: 6 osds: 6 up, 6 in
    pgsmap v27916: 400 pgs, 2 pools, 21233 MB data, 5315 objects
    121 GB used, 10924 GB / 11058 GB avail
    400 active+clean
    client io 481 kB/s rd, 132 kB/s wr, 185 op/

MON

http://ceph.com/docs/master/rados/troubleshooting/troubleshooting-mon/
OSD

http://ceph.com/docs/master/rados/troubleshooting/troubleshooting-osd/

• ceph osd tree - show all OSDs and it's state

```
root@c-01:~# ceph osd tree
ID   WEIGHT   TYPE   NAME        UP/DOWN   REWEIGHT PRIMARY-AFFINITY
-4    0.00000  host  c-04
-1    10.79993 root  default
-2    3.59998  host  c-01
  0    1.79999  osd.0  up  1.00000  1.00000
  1    1.79999  osd.1  up  1.00000  1.00000
-3    3.59998  host  c-02
  2    1.79999  osd.2  up  1.00000  1.00000
  3    1.79999  osd.3  up  1.00000  1.00000
-5    3.59998  host  c-03
  4    1.79999  osd.4  up  1.00000  1.00000
  5    1.79999  osd.5  up  1.00000  1.00000
```

• ceph osd pools ls - list of pool

```
root@c-01:~# ceph osd lspools
0  rbd,1  test
```

PG

http://ceph.com/docs/master/rados/troubleshooting/troubleshooting-pg

• ceph pg ls - list placement groups

```
root@c-01:~# ceph pg ls | head -n 4
pg_stat     objects  mip  degr  misp  unf  bytes  log  disklog  state
          v  reported  up  up_primary  acting  acting_
          primary  last_scrub  scrub_stamp  last_deep_scrub  deep_scrub_stamp
0.0       11    0        0        0        0      46137344 3044 3044
          active+clean 2015-07-02 10:12:40.603692 226'10652 226:1798 [4, 2, 0] 4 [4,2,0] 4 0'0 2015-07-01 18:38:33.126953 0'0 2015-07-01 18:17:01.904194
0.1        7    0        0        0        0      25165936 3026 3026
          active+clean 2015-07-02 10:12:40.603692 226'10652 226:1798 [4, 2, 0] 4 [4,2,0] 4 0'0 2015-07-01 18:38:33.126953 0'0 2015-07-01 18:17:01.904194
0.2       18    0        0        0        0      75497472 3039 3039
          active+clean 2015-07-02 10:12:40.603692 226'10652 226:1798 [4, 2, 0] 4 [4,2,0] 4 0'0 2015-07-01 18:38:33.126953 0'0 2015-07-01 18:17:01.904194
```

• ceph pg map 1.1 - show mapping between PG and OSD

```
root@c-01:~# ceph pg map 1.1
osdmap e226 pg 1.1 (1.1) -> up [5,1,2] acting [5,1,2]
```
Sample pillars

Common metadata for all nodes/roles

```yaml
ceph:
  common:
    version: luminous
    config:
      global:
        param1: value1
        param2: value1
        param3: value1
      pool_section:
        param1: value2
        param2: value2
        param3: value2
    fsid: a619c5fc-c4ed-4f22-9ed2-66cf2feca23d
    members:
    - name: cmn01
      host: 10.0.0.1
    - name: cmn02
      host: 10.0.0.2
    - name: cmn03
      host: 10.0.0.3
    keyring:
      admin:
        caps:
          mds: "allow *"
          mgr: "allow *"
          mon: "allow *"
          osd: "allow *"
    bootstrap-osd:
      caps:
        mon: "allow profile bootstrap-osd"
```

Optional definition for cluster and public networks. Cluster network is used for replication. Public network for front-end communication.

```yaml
ceph:
  common:
    version: luminous
    fsid: a619c5fc-c4ed-4f22-9ed2-66cf2feca23d
    ....
    public_network: 10.0.0.0/24, 10.1.0.0/24
    cluster_network: 10.10.0.0/24, 10.11.0.0/24
```

Ceph mon (control) roles

Monitors: A Ceph Monitor maintains maps of the cluster state, including the monitor map, the OSD map, the Placement Group (PG) map, and the CRUSH map. Ceph maintains a history (called an “epoch”) of each state change in the Ceph Monitors, Ceph OSD Daemons, and PGs.
mon:
  key: value
mon:
  enabled: true
dashboard:
  enabled: true
  host: 10.103.255.252
  port: 7000

Ceph mgr roles

The Ceph Manager daemon (ceph-mgr) runs alongside monitor daemons, to provide additional monitoring and interfaces to external monitoring and management systems. Since the 12.x (luminous) Ceph release, the ceph-mgr daemon is required for normal operations. The ceph-mgr daemon is an optional component in the 11.x (kraken) Ceph release.

By default, the manager daemon requires no additional configuration, beyond ensuring it is running. If there is no mgr daemon running, you will see a health warning to that effect, and some of the other information in the output of ceph status will be missing or stale until a mgr is started.

ceph:
  mgr:
    enabled: true
    dashboard:
      enabled: true
      host: 10.103.255.252
      port: 7000

Ceph OSD (storage) roles

ceph:
  common:
    version: luminous
    fsid: a619c5fc-c4ed-4f22-9ed2-66cf2feca23d
    public_network: 10.0.0.0/24, 10.1.0.0/24
    cluster_network: 10.10.0.0/24, 10.11.0.0/24
  keyring:
    bootstrap-osd:
      caps:
        mon: "allow profile bootstrap-osd"
        ...
  osd:
    enabled: true
    crush_parent: rack01
    journal_size: 20480 (20G)
    bluestore_block_db_size: 10073741824 (10G)
    bluestore_block_wal_size: 10073741824 (10G)
    bluestore_block_size: 807374182400 (800G)
backend:
    filestore:
        disks:
            - dev: /dev/sdm
              enabled: false
              journal: /dev/ssd
              journal_partition: 5
              data_partition: 6
              lockbox_partition: 7
              data_partition_size: 12000 (MB)
              class: bestssd
              weight: 1.666
              dmcrypt: true
              journal_dmcrypt: false
            - dev: /dev/sdf
              journal: /dev/ssd
              journal_dmcrypt: true
              class: bestssd
              weight: 1.666
            - dev: /dev/sdl
              journal: /dev/ssd
              class: bestssd
              weight: 1.666
    bluestore:
        disks:
            - dev: /dev/sdb
            - dev: /dev/sdf
              block_db: /dev/ssd
              block_wal: /dev/ssd
              block_db_dmcrypt: true
              block_wal_dmcrypt: true
            - dev: /dev/sdc
              block_db: /dev/ssd
              block_wal: /dev/ssd
              data_partition: 1
              block_partition: 2
              lockbox_partition: 5
              block_db_partition: 3
              block_wal_partition: 4
              class: ssd
              weight: 1.666
              dmcrypt: true
              block_db_dmcrypt: false
              block_wal_dmcrypt: false
            - dev: /dev/sdd
              enabled: false

Ceph client roles - ... Deprecated - use ceph:common instead

Simple ceph client service

ceph:
    client:
        config:
            global:

(continues on next page)
At OpenStack control settings are usually located at cinder-volume or glance-registry services.

```yaml
ceph:
  client:
    config:
      global:
        fsid: 00000000-0000-0000-0000-000000000000
        mon initial members: ceph1,ceph2,ceph3
        mon host: 10.103.255.252:6789,10.103.255.253:6789,10.103.255.254:6789
        keyring:
          monitoring:
            key: 00000000000000000000000000000000000000==

Rados gateway with keystone v2 auth backend

```yaml
ceph:
  radosgw:
    enabled: true
    hostname: gw.ceph.lab
    bind:
      address: 10.10.10.1
      port: 8080
    identity:
      engine: keystone
      api_version: 2
      host: 10.10.10.100
      port: 5000
      user: admin
      password: password
      tenant: admin

Rados gateway with keystone v3 auth backend

```yaml
ceph:
  radosgw:
```
(continues on next page)
enabled: true
hostname: gw.ceph.lab
bind:
  address: 10.10.10.1
  port: 8080
identity:
  engine: keystone
  api_version: 3
  host: 10.10.10.100
  port: 5000
  user: admin
  password: password
  project: admin
  domain: default

Ceph setup role

Replicated ceph storage pool

```yaml
ceph:
  setup:
    pool:
      replicated_pool:
        pg_num: 256
        pgp_num: 256
        type: replicated
        crush_rule: sata
        application: rbd

.. note:: For Kraken and earlier releases please specify crush_rule as a ruleset.

  For Kraken and earlier releases application param is not needed.
```

Erasure ceph storage pool

```
ceph:
  setup:
    pool:
      erasure_pool:
        pg_num: 256
        pgp_num: 256
        type: erasure
        crush_rule: ssd
        application: rbd
```

Inline compression for Bluestore backend

```yaml
ceph:
  setup:
    pool:
      volumes:
        pg_num: 256
        pgp_num: 256
        type: replicated
        crush_rule: hdd
```
SaltStack-Formulas Documentation, Release master

(continued from previous page)

application: rbd
  compression_algorithm: snappy
  compression_mode: aggressive
  compression_required_ratio: .875
...

Ceph manage keyring keys

Keyrings are dynamically generated unless specified by the following pillar.

```yaml
ceph:
  common:
    manage_keyring: true
  keyring:
    glance:
      name: images
      key: AACf3ulZFFPNDxAAd2DWds3aEkHh4Ik1ZVgIaQ==
      caps:
        mon: "allow r"
        osd: "allow class-read object_prefix rdb_children, allow rwx pool=images"
```

Generate CRUSH map - Recommended way

It is required to define the type for crush buckets and these types must start with root (top) and end with host. OSD daemons will be assigned to hosts according to its hostname. Weight of the buckets will be calculated according to weight of its children.

If the pools that are in use have size of 3 it is best to have 3 children of a specific type in the root CRUSH tree to replicate objects across (Specified in rule steps by ‘type region’).

```yaml
ceph:
  setup:
    crush:
      enabled: True
      tunables:
        choose_total_tries: 50
        choose_local_tries: 0
        choose_local_fallback_tries: 0
        chooseleaf_descend_once: 1
        chooseleaf_vary_r: 1
        chooseleaf_stable: 1
        straw_calc_version: 1
        allowed_bucket_algs: 54
      type:
        - root
        - region
        - rack
        - host
        - osd
      root:
        - name: root-ssd
        - name: root-sata
      region:
```

(continues on next page)
- name: eu-1
  parent: root-sata
- name: eu-2
  parent: root-sata
- name: eu-3
  parent: root-ssd
- name: us-1
  parent: root-sata
rack:
  - name: rack01
    parent: eu-1
  - name: rack02
    parent: eu-2
  - name: rack03
    parent: us-1
rule:
  sata:
    ruleset: 0
    type: replicated
    min_size: 1
    max_size: 10
    steps:
      - take root-ssd
      - chooseleaf firstn 0 type region
      - emit
  ssd:
    ruleset: 1
    type: replicated
    min_size: 1
    max_size: 10
    steps:
      - take root-sata
      - chooseleaf firstn 0 type region
      - emit

Generate CRUSH map - Alternative way

It’s necessary to create per OSD pillar.

ceph:
  osd:
    crush:
      - type: root
        name: root1
      - type: region
        name: eu-1
      - type: rack
        name: rack01
      - type: host
        name: osd001

Add OSDs with specific weight

Add OSD device(s) with initial weight set specifically to certain value.
Apply CRUSH map

Before you apply CRUSH map please make sure that settings in generated file in /etc/ceph/crushmap are correct.

```
ceph:
  osd:
    crush_initial_weight: 0
```

Persist CRUSH map

After the CRUSH map is applied to Ceph it’s recommended to persist the same settings even after OSD reboots.

```
ceph:
  osd:
    crush_update: false
```

Ceph monitoring

By default monitoring is setup to collect information from MON and OSD nodes. To change the default values add the following pillar to MON nodes.

```
ceph:
  monitoring:
    space_used_warning_threshold: 0.75
    space_used_critical_threshold: 0.85
    apply_latency_threshold: 0.007
    commit_latency_threshold: 0.7
    pool_space_used_utilization_warning_threshold: 0.75
    pool_space_used_critical_threshold: 0.85
    pool_write_ops_threshold: 200
    pool_write_bytes_threshold: 70000000
    pool_read_bytes_threshold: 70000000
    pool_read_ops_threshold: 1000
```
Ceph monitor backups

Backup client with ssh/rsync remote host

```yaml
ceph:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
    target:
      host: cfg01
```

Backup client with local backup only

```yaml
ceph:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
```

Backup server rsync

```yaml
ceph:
  backup:
    server:
      enabled: true
      hours_before_full: 24
      full_backups_to_keep: 5
    key:
      ceph_pub_key:
        enabled: true
        key: ssh_rsa
```

- [https://github.com/cloud-ee/ceph-salt-formula](https://github.com/cloud-ee/ceph-salt-formula)
- [http://ceph.com/ceph-storage/](http://ceph.com/ceph-storage/)
- [http://ceph.com/docs/master/start/intro/](http://ceph.com/docs/master/start/intro/)

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:


For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

[https://github.com/salt-formulas/salt-formula-ceph](https://github.com/salt-formulas/salt-formula-ceph)
Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**chrony**

**WIP**

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-chrony/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-chrony

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**FreeIPA**

This formula installs and configured the FreeIPA Identity Management service and client.

**Sample pillars**

**Client**

```bash
freeipa:
  client:
    enabled: true
    server: ipa.example.com
    domain: {{ salt['grains.get']('domain', '') }}
    realm: {{ salt['grains.get']('domain', '').upper() }}
    hostname: {{ salt['grains.get']('fqdn', '') }}
```

To automatically register the client with FreeIPA, you will first need to create a Kerberos principal. Start by creating a service account in FreeIPA. You may wish to restrict that users permissions to only host creation (see https://www.freeipa.org/page/HowTos#Working_with_FreeIPA). Next, you will need to obtain a kerberos ticket as admin on the IPA server, then generate a service account principal.
kinit admin

ipa-getkeytab -p service-account@EXAMPLE.com -k ./principal.keytab -s freeipahost.example.com

scp ./principal.keytab user@saltmaster.example.com:/srv/salt/freeipa/files/principal.keytab

Then add to your pillar:

This will allow your client to use FreeIPA’s JSON interface to create a host entry with a One Time Password and then register to the FreeIPA server. For security purposes, the kerberos principal will only be pushed down to the client if the installer reports it is not registered to the FreeIPA server and will be removed from the client as soon as the endpoint has registered with the FreeIPA server.

Additionally, the openssh formula (see https://github.com/salt-formulas/salt-formula-openssh) is needed and is a dependency for this formula. Configure it thusly:

```
openssh:
  server:
    public_key_auth: true
    gssapi_auth: true
    kerberos_auth: false
    authorized_keys_command:
      command: /usr/bin/sss_ssh_authorizedkeys
      user: nobody
```

If you wish to update DNS records using nsupdate, add:

```
freeipa:
  client:
    nsupdate:
      - name: test.example.com
        ipv4:
          - 8.8.8.8
        ipv6:
          - 2a00:1450:4001:80a::1009
        ttl: 1800
        keytab: /etc/krb5.keytab
```

For requesting certificates using certmonger:

```
freeipa:
  client:
    cert:
      "HTTP/www.example.com":
        user: root
        group: www-data
        mode: 640
        cert: /etc/ssl/certs/http-www.example.com.crt
        key: /etc/ssl/private/http-www.example.com.key
```

**Server**

```
freeipa:
  server:
    realm: IPA.EXAMPLE.COM
```

(continues on next page)
Server definition for new version of freeipa (4.3+). Replicas don't require generation of gpg file on master. But principal user has to be defined with

```yaml
freeipa:
  server:
    realm: IPA.EXAMPLE.COM
    domain: ipa.example.com
    principal_user: admin
    admin:
      password: secretpassword
    servers:
      - idm01.ipa.example.com
      - idm02.ipa.example.com
      - idm03.ipa.example.com
```

Disable CA. Default is True.

```yaml
freeipa:
  server:
    ca: false
```

Disable LDAP access logs but enable audit

```yaml
freeipa:
  server:
    ldap:
      logging:
        access: false
        audit: true
```

Read more


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

[https://github.com/salt-formulas/salt-formula-freeipa/issues](https://github.com/salt-formulas/salt-formula-freeipa/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)
Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-freeipa

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Git formula**

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

**Sample pillars**

Simplest GIT setup

```yaml
git:
  client:
    enabled: true
```

GIT with user setup

```yaml
git:
  client:
    enabled: true
  user:
    name: jdoe
    email: j@doe.com
```

GIT with user and SSL setup

```yaml
git:
  client:
    disable_ssl_verification: True
    enabled: true
    user:
      name: jdoe
      email: j@doe.com
```

Reclass with GIT with user setup

```yaml
git:
  client:
    enabled: true
    user:
      - user: ${linux:system:user:jdoe}
```

Reclass with GIT with user and SSL setup

```yaml
git:
  client:
    disable_ssl_verification: True
```

(continues on next page)
enabled: true
user:
  - user: ${linux:system:user:jdoe}

Reclass with GIT over HTTP server setup. Requires web server.

```yaml
git:
  server:
    directory: /srv/git
    repos:
      - name: custom-repo-1
      - name: custom-repo-2
```

Reclass with GIT over HTTP server setup. Requires web server. Mirrored upstream repos example.

```yaml
git:
  server:
    directory: /srv/git
    repos:
      - name: gerritlib
        url: https://github.com/openstack-infra/gerritlib.git
      - name: jeepyb
        url: https://github.com/openstack-infra/jeepyb.git
```

Read more

- [https://github.com/nesi/puppet-git/tree/master/manifests](https://github.com/nesi/puppet-git/tree/master/manifests)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

[https://github.com/salt-formulas/salt-formula-git/issues](https://github.com/salt-formulas/salt-formula-git/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

[https://github.com/salt-formulas/salt-formula-git](https://github.com/salt-formulas/salt-formula-git)

Any questions or feedback is always welcome so feel free to join our IRC channel:

`#salt-formulas @ irc.freenode.net`
GlusterFS

Install and configure GlusterFS server and client.

Available states

- `glusterfs.server`
- `glusterfs.server.service`
- `glusterfs.server.setup`
- `glusterfs.client`

`glusterfs.server`

Setup GlusterFS server (including both service and setup)

`glusterfs.server.service`

Setup and start GlusterFS server service

`glusterfs.server.setup`

Setup GlusterFS peers and volumes

`glusterfs.client`

Setup GlusterFS client

Available metadata

- `metadata.glusterfs.server`
- `metadata.glusterfs.client`

`metadata.glusterfs.server`

Setup basic server

`metadata.glusterfs.client`

Setup client only
**Configuration parameters**

**Example reclass**

Example for distributed glance images storage where every control node is gluster peer.

```yaml
classes:
- service.glusterfs.server
- service.glusterfs.client

_param:
  cluster_node01_address: 192.168.1.21
  cluster_node02_address: 192.168.1.22
  cluster_node03_address: 192.168.1.23

parameters:
  glusterfs:
    server:
      peers:
      - $(_param:cluster_node01_address)
      - $(_param:cluster_node02_address)
      - $(_param:cluster_node03_address)
    volumes:
      glance:
        storage: /srv/glusterfs/glance
        replica: 3
        bricks:
        - $(_param:cluster_node01_address):/srv/glusterfs/glance
        - $(_param:cluster_node02_address):/srv/glusterfs/glance
        - $(_param:cluster_node03_address):/srv/glusterfs/glance
    options:
      cluster.readdir-optimize: On
      nfs.disable: On
      network.remote-dio: On
      diagnostics.client-log-level: WARNING
      diagnostics.brick-log-level: WARNING

client:
  volumes:
    glance:
      path: /var/lib/glance/images
      server: $(_param:cluster_node01_address)
      user: glance
      group: glance
```

**Example pillar**

**Server**

```yaml
clusterfs:
  server:
    peers:
      - 192.168.1.21
      - 192.168.1.22
      - 192.168.1.23
    volumes:
```

(continues on next page)
glance:
  storage: /srv/glusterfs/glance
  replica: 3
  bricks:
  - 172.168.1.21:/srv/glusterfs/glance
  - 172.168.1.21:/srv/glusterfs/glance
  - 172.168.1.21:/srv/glusterfs/glance
  enabled: true

Client

grusterfs:
  client:
    volumes:
      glance:
        path: /var/lib/glance/images
        server: 192.168.1.21
        user: glance
        group: glance
        enabled: true

Read more

- https://www.gluster.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-grusterfs/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-grusterfs

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
iptables formula

Iptables is used to set up, maintain, and inspect the tables of IPv4 packet filter rules in the Linux kernel. Several different tables may be defined. Each table contains a number of built-in chains and may also contain user-defined chains. Each chain is a list of rules which can match a set of packets. Each rule specifies what to do with a packet that matches. This is called a target, which may be a jump to a user-defined chain in the same table.

Sample pillars

Most common rules - allow traffic on localhost, accept related, established and ping

```
parameters:
ipables:
service:
    enabled: True
chain:
    INPUT:
rules:
    - in_interface: lo
      jump: ACCEPT
    - connection_state: RELATED,ESTABLISHED
      match: state
      jump: ACCEPT
    - protocol: icmp
      jump: ACCEPT
```

Accept connections on port 22

```
parameters:
ipables:
service:
    chain:
    INPUT:
rules:
    - destination_port: 22
      protocol: tcp
      jump: ACCEPT
```

Set drop policy on INPUT chain:

```
parameters:
ipables:
service:
    chain:
    INPUT:
policy: DROP
```

Redirect privileged port 443 to 8081

```
parameters:
ipables:
service:
    chain:
    PREROUTING:
      filter: nat
      destination_port: 443
```

(continues on next page)
Allow access from local network

```yaml
parameters:
  iptables:
    service:
      chain:
        INPUT:
          rules:
            - protocol: tcp
              destination_port: 22
              source_network: 192.168.1.0/24
              jump: ACCEPT
              comment: Blah
```

Support logging with custom prefix and log level

```yaml
parameters:
  iptables:
    service:
      chain:
        POSTROUTING:
          rules:
            - table: nat
              protocol: tcp
              match: multiport
              destination_ports:
                - 21
                - 80
                - 443
                - 2220
              source_network: '10.20.30.0/24'
              log_level: 7
              log_prefix: 'iptables-logging: '
              jump: LOG
```

IPv6 is supported as well

```yaml
parameters:
  iptables:
    service:
      enabled: True
      ipv6: True
    chain:
      INPUT:
        rules:
          - protocol: tcp
            family: ipv6
            destination_port: 22
            source_network: 2001:DB8::/32
            jump: ACCEPT
```

Read more

- https://help.ubuntu.com/community/IptablesHowTo
- http://wiki.centos.org/HowTos/Network/IPTables

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-iptables/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-iptables

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Let's Encrypt

Service letsencrypt description

Sample pillars

Installation

There are 3 installation methods available:

- package (default for Debian)

  For Debian Jessie, you need to use jessie-backports repository. For Ubuntu, use Launchpad PPA providing certbot package. You can use linux formula to manage these APT sources.

```python
letsencrypt:
  client:
    source:
      engine: pkg
```

If the certbot package doesn’t include Systemd .service and .timer files, you can set them to be installed by this formula by supplying install_units: True and cli.
letsencrypt:
  client:
    source:
      engine: pkg
      cli: /usr/bin/certbot
      install_units: true

- **URL to certbot-auto (default)**

  This is default installation method for systems with no available certbot package.

  ```yaml
  letsencrypt:
    client:
      source:
        engine: url
        url: "https://dl.eff.org/certbot-auto"
  ```

- **Docker container**

  Alternate installation method where Docker image is used to provide certbot tool and executed using wrapper script.

  ```yaml
  letsencrypt:
    client:
      source:
        engine: docker
        image: "deliverous/certbot"
  ```

**Usage**

Default authentication method using standalone server on specified port. But this won’t work without configuration of apache/nginx (read on) unless you don’t have webserver running so you can select port 80 or 443.

```yaml
letsencrypt:
  client:
    email: root@dummy.org
    auth:
      method: standalone
      type: http-01
      port: 9999
    domain:
      dummy.org:
        enabled: true
      www.dummy.org:
        enabled: true
    # Following will produce multidomain certificate:
    site.dummy.org:
      enabled: true
      names:
        - dummy.org
        - www.dummy.org
```

However ACME server always visits port 80 (or 443) where most likely Apache or Nginx is listening. This means that you need to configure `/.well-known/acme-challenge/` to proxy requests on localhost:9999. For example, ensure you have following configuration for Apache:
ProxyPass "/.well-known/acme-challenge/" "http://127.0.0.1:9999/.well-known/acme-challenge/" retry=1

<Location "/.well-known/acme-challenge/">
  ProxyPreserveHost On
  Order allow,deny
  Allow from all
  Require all granted
</Location>

You can also use apache or nginx auth methods and let certbot do what’s needed, this should be the simplest option.

```yaml
letsencrypt:
  client:
    auth: apache
```

Alternatively you can use webroot authentication (using eg. existing apache installation serving directory for all sites):

```yaml
letsencrypt:
  client:
    auth:
      method: webroot
      path: /var/www/html
      port: 80
    domain:
      dummy.org:
        enabled: true
      www.dummy.org:
        enabled: true
```

It’s also possible to override auth method or other options only for single domain:

```yaml
letsencrypt:
  client:
    email: root@dummy.org
    auth:
      method: standalone
      type: http-01
      port: 9999
    domain:
      dummy.org:
        enabled: true
      www.dummy.org:
        enabled: true
        auth:
          method: webroot
          path: /var/www/html/dummy.org
          port: 80
      www.dummy.org:
        enabled: true
```

You are able to use multidomain certificates:

```yaml
letsencrypt:
  client:
    email: sylvain@home
    staging: true
```

(continues on next page)
auth:
  method: apache
domain:
  keynotdomain:
    enabled: true
    name: ls.opensource-expert.com
    names:
    - www.ls.opensource-expert.com
    - vim22.opensource-expert.com
    - www.vim22.opensource-expert.com
rm.opensource-expert.com:
  enabled: true
  names:
  - www.rm.opensource-expert.com
vim7.opensource-expert.com:
  enabled: true
  names:
  - www.vim7.opensource-expert.com
vim88.opensource-expert.com:
  enabled: true
  names:
  - www.vim88.opensource-expert.com
  - awk.opensource-expert.com
  - www.awk.opensource-expert.com

Legacy configuration

Common metadata:

letsencrypt:
  client:
    enabled: true
    config:
      host = https://acme-v01.api.letsencrypt.org/directory
      email = webmaster@example.com
      authenticator = webroot
      webroot-path = /var/lib/www
      agree-tos = True
      renew-by-default = True
domainset:
  www:
  - example.com
  - www.example.com
  mail:
  - imap.example.com
  - smtp.example.com
  - mail.example.com
  intranet:
  - intranet.example.com

Example of authentication via another port without stopping nginx server:

location /well-known/acme-challenge/ { 
  proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for; 
  proxy_set_header Host $http_host; 
}
proxy_redirect off;
proxy_pass http://{{ site.host.name }}:9999/.well-known/acme-challenge/;

letsencrypt:
  client:
    enabled: true
    config: |
      ...
      renew-by-default = True
      http-01-port = 9999
      standalone-supported-challenges = http-01
    domainset:
      www:
        - example.com

Read more

- Certbot authentication plugins

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-letsencrypt/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-letsencrypt

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Linux Formula

Linux Operating Systems.

- Ubuntu
- CentOS
- RedHat
Sample Pillars

Linux System

Basic Linux box

```yaml
linux:
  system:
    enabled: true
    name: 'node1'
    domain: 'domain.com'
    cluster: 'system'
    environment: prod
    timezone: 'Europe/Prague'
    utc: true

Linux with system users, some with password set: .. WARNING:: If no 'password' variable has been passed - any predefined password will be removed.

```yaml
linux:
  system:
    ... user:
      jdoe:
        name: 'jdoe'
        enabled: true
        sudo: true
        shell: '/bin/bash
        full_name: 'Jonh Doe'
        home: '/home/jdoe'
        email: 'jonh@doe.com'
      jsmith:
        name: 'jsmith'
        enabled: true
        full_name: 'With clear password'
        home: '/home/jsmith'
        hash_password: true
        password: "userpassword"
      mark:
        name: 'mark'
        enabled: true
        full_name: 'unchange password'
        home: '/home/mark'
        password: false
      elizabeth:
        name: 'elizabeth'
        enabled: true
        full_name: 'With hased password'
        home: '/home/elizabeth'
        password: "$6$nUI7QEz3
˓→$dFYjzQqK5cJ6HQ38KqG4TWA9eJu3aKx6TRVDFh6BVJx3gFWg2akfAA7f1fCxcSUeOJ2arCO6EEI6XXnHXxG10 ...
""
```
Configure sudo for users and groups under `/etc/sudoers.d/`. This way `linux.system.sudo` pillar map to actual sudo attributes:

```
# simplified template:
Cmds_Alias {{ alias }}=({{ commands }})
{{ user }} {{ hosts }}=({{ runas }}) NOPASSWD: {{ commands }}
{{ group }} {{ hosts }}=({{ runas }}) NOPASSWD: {{ commands }}

# when rendered:
saltuser1 ALL=(ALL) NOPASSWD: ALL
```

```
linux:
    system:
        sudo:
            enabled: true
            aliases:
                host:
                    LOCAL:
                        - localhost
                    PRODUCTION:
                        - db1
                        - db2
                runas:
                    DBA:
                        - postgres
                        - mysql
                    SALT:
                        - root
            command:
                # Note: This is not 100% safe when ALL keyword is used, user still may modify configs and hide his actions.
                # Best practice is to specify full list of commands user is allowed to run.
                SUPPORT_RESTRICTED:
                    - /bin/vi /etc/sudoers*
                    - /bin/vim /etc/sudoers*
                    - /bin/nano /etc/sudoers*
                    - /bin/emacs /etc/sudoers*
                    - /bin/su - root
                    - /bin/su -
                    - /bin/su
                    - /usr/sbin/visudo
                SUPPORT_SHELLS:
                    - /bin/sh
                    - /bin/ksh
                    - /bin/bash
                    - /bin/rbash
                    - /bin/dash
                    - /bin/zsh
                    - /bin/csh
                    - /bin/fish
                    - /bin/tcsh
                    - /usr/bin/login
                    - /usr/bin/su
                    - /usr/su
                ALL_SALT_SAFE:
                    - /usr/bin/salt state*
                    - /usr/bin/salt service*
```
- /usr/bin/salt pillar*
- /usr/bin/salt grains*
- /usr/bin/salt saltutil*
- /usr/bin/salt-call state*
- /usr/bin/salt-call service*
- /usr/bin/salt-call pillar*
- /usr/bin/salt-call grains*
- /usr/bin/salt-call saltutil*
SALT_TRUSTED:
- /usr/bin/salt*

users:

  # saltuser1 with default values: saltuser1 ALL=(ALL) NOPASSWD: ALL
  saltuser1: {}
saltuser2:
    hosts:
    - LOCAL
  # User Alias DBA
  DBA:
    hosts:
    - ALL
    commands:
    - ALL_SALT_SAFE
groups:
  db-ops:
    hosts:
    - ALL
    - '!PRODUCTION'
    runas:
    - DBA
    commands:
    - /bin/cat *
    - /bin/less *
    - /bin/ls *
salt-ops:
    hosts:
    - 'ALL'
    runas:
    - SALT
    commands:
    - SUPPORT_SHELLS
salt-ops-2nd:
  name: salt-ops
  nopasswd: false
  setenv: true # Enable sudo -E option
  runas:
  - DBA
  commands:
  - ALL
  - '!SUPPORT_SHELLS'
  - '!SUPPORT_RESTRICTED'

Linux with package, latest version

linux:
  system:
  ...
  package:
Linux with package from certain repo, version with no upgrades

```yaml
tenant:
    system:
        ...
        package:
            package-name:
                version: 2132.323
                repo: 'custom-repo'
                hold: true
```

Linux with package from certain repo, version with no GPG verification

```yaml
tenant:
    system:
        ...
        package:
            package-name:
                version: 2132.323
                repo: 'custom-repo'
                verify: false
```

Linux with autoupdates (automatically install security package updates)

```yaml
tenant:
    system:
        ...
        autoupdates:
            enabled: true
            mail: root@localhost
            mail_only_on_error: true
            remove_unused_dependencies: false
            automatic_reboot: true
            automatic_reboot_time: "02:00"
```

Linux with cron jobs By default it will use name as an identifier, unless identifier key is explicitly set or False (then it will use Salt’s default behavior which is identifier same as command resulting in not being able to change it)

```yaml
tenant:
    system:
        ...
        job:
            cmd1:
                command: '/cmd/to/run'
                identifier: cmd1
                enabled: true
                user: 'root'
                hour: 2
                minute: 0
```

Linux security limits (limit sensu user memory usage to max 1GB):
Enable autologin on tty1 (may work only for Ubuntu 14.04):

```yaml
linux:
  system:
    console:
      tty1:
        autologin: root
        # Enable serial console
      ttyS0:
        autologin: root
        rate: 115200
        term: xterm
```

To disable set autologin to `false`.

Set policy-rc.d on Debian-based systems. Action can be any available command in `while true` loop and `case` context. Following will disallow dpkg to stop/start services for cassandra package automatically:

```yaml
linux:
  system:
    policyrcd:
      - package: cassandra
        action: exit 101
      - package: '*'
        action: switch
```

Set system locales:

```yaml
linux:
  system:
    locale:
      en_US.UTF-8:
        default: true
      "cs_CZ.UTF-8 UTF-8":
        enabled: true
```

Systemd settings:

```yaml
linux:
  system:
    systemd:
      systemd:
        Manager:
          DefaultLimitNOFILE: 307200
```
DefaultLimitNPROC: 307200
user:
  Manager:
    DefaultLimitCPU: 2
    DefaultLimitNPROC: 4

Ensure presence of directory:

```
linux:
  system:
    directory:
      /tmp/test:
        user: root
        group: root
        mode: 700
        makedirs: true
```

Ensure presence of file by specifying it’s source:

```
linux:
  system:
    file:
      /tmp/test.txt:
        source: http://example.com/test.txt
        user: root #optional
        group: root #optional
        mode: 700 #optional
        dir_mode: 700 #optional
        encoding: utf-8 #optional
        hash: <<hash>> or <<URI to hash>> #optional
        makedirs: true #optional

linux:
  system:
    file:
      test.txt:
        name: /tmp/test.txt
        source: http://example.com/test.txt
```

Ensure presence of file by specifying it’s contents:

```
linux:
  system:
    file:
      /tmp/test.txt:
        contents: |
          line1
          line2

linux:
  system:
    file:
      /tmp/test.txt:
        contents_pillar: linux:network:hostname
```

(continues on next page)
Kernel

Install always up to date LTS kernel and headers from Ubuntu trusty:

```bash
linux:
  system:
    kernel:
      type: generic
      lts: trusty
      headers: true
```

Load kernel modules and add them to `/etc/modules`:

```bash
linux:
  system:
    kernel:
      modules:
        - nf_conntrack
        - tp_smapi
        - 8021q
```

Configure or blacklist kernel modules with additional options to `/etc/modprobe.d` following example will add `/etc/modprobe.d/nf_conntrack.conf` file with line `options nf_conntrack hashsize=262144`:

```bash
linux:
  system:
    kernel:
      nf_conntrack:
        module:
          option:
            hashsize: 262144
```

Install specific kernel version and ensure all other kernel packages are not present. Also install extra modules and headers for this kernel:

```bash
linux:
  system:
    kernel:
      type: generic
      extra: true
      headers: true
      version: 4.2.0-22
```

Systcl kernel parameters

```bash
linux:
  system:
    kernel:
      sysctl:
        net.ipv4.tcp_keepalive_intvl: 3
```

(continues on next page)
net.ipv4.tcp_keepalive_time: 30
net.ipv4.tcp_keepalive_probes: 8

Configure kernel boot options:

```yaml
linux:
  system:
    kernel:
      boot_options:
        - elevator=deadline
        - spectre_v2=off
        - nopti
```

**CPU**

Enable cpufreq governor for every cpu:

```yaml
linux:
  system:
    cpu:
      governor: performance
```

**CGROUPS**

Setup linux cgroups:

```yaml
linux:
  system:
    cgroup:
      enabled: true
      group:
        ceph_group_1:
          controller:
            cpu:
              shares:
                value: 250
            cpuacct:
              usage:
                value: 0
            cpuset:
              cpus:
                value: 1,2,3
          memory:
            limit_in_bytes:
              value: 2G
            memsw.limit_in_bytes:
              value: 3G
      mapping:
        subjects:
          - '@ceph'
        generic_group_1:
          controller:
            cpu:
```

(continues on next page)
Shared Libraries

Set additional shared library to Linux system library path

```python
shared:
    value: 250
cpuacct:
    usage:
        value: 0
mapping:
    subjects:
        - '*:firefox'
        - 'student:cp'
```

Certificates

Add certificate authority into system trusted CA bundle

```bash
[shared]
value: 250
cpuacct:
usage:
    value: 0
mapping:
    subjects:
        - '*:firefox'
        - 'student:cp'

[Sysfs]
Install sysfsutil and set sysfs attributes:

```python
linux:
    system:
        ld:
            library:
                java:
                    - /usr/lib/jvm/jre-openjdk/lib/amd64/server
                    - /opt/java/jre/lib/amd64/server
```

```
-----BEGIN CERTIFICATE-----
MIICPDCCAAUCEH65B0Q2Sk0tjjKewPMur8wDQYJKoZIhvcNAQECBQAwZxELMAkG
A1UEBhMCMVVMxZAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMTcwNQYDVQQLEy5DbGFz
YXJ5IENlcnRpZmljYXRpb24gQXV0aG9yaXR5IENlcnRpZmljYXRpb24gQXV0aG9ya
R5MGFMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDJXFme8huKARS0EN8EQnvjV69q
RUCPhAwL0TP22RHP7gJHYx3KqhE
BARSAt9qf56Tu2oAqiN9lqyFomNFx3InzPRMxnvX0jnvTOLwdd8KkMaOIG+YD/is
I19wKTAxyBns2ogy101hec9vn2a/iRFM9x2Fe0PonFtTGuQWhFwpIDAQABMA0G
C5qG5i23DQEBAgUA4GBALtMEiPCLCYATxQT3ab7/AoRhIzzKEXnki98tsX63/Do
1bwgi2wqFHMc9ikwFPwTtYmxHYBV4GSX1X0bH/59AhWM1pF+NEHJwZRDmJXNyc
AA9WjQKZ7aKQRUxuxCkFpayAw7xrvjoyVGM5Kf5p/AfbdynMk20mufTqj/ZA1k
-----END CERTIFICATE-----
```
sysfs:
scheduler:
  block/sda/queue/scheduler: deadline
power:
  mode:
    power/state: 0660
owner:
  power/state: "root:power"
devices/system/cpu/cpu0/cpufreq/scaling_governor: powersave

Huge Pages

Huge Pages give a performance boost to applications that intensively deal with memory allocation/deallocation by decreasing memory fragmentation.

```
linux:
  system:
    kernel:
      hugepages:
        small:
          size: 2M
          count: 107520
          mount_point: /mnt/hugepages_2MB
          mount: false/true # default false
        large:
          default: true # default automatically mounted
          size: 1G
          count: 210
          mount_point: /mnt/hugepages_1GB
```

Note: not recommended to use both pagesizes in concurrently.

Intel SR-IOV

PCI-SIG Single Root I/O Virtualization and Sharing (SR-IOV) specification defines a standardized mechanism to virtualize PCIe devices. The mechanism can virtualize a single PCIe Ethernet controller to appear as multiple PCIe devices.

```
 linux:
   system:
     kernel:
       sriov: True
       unsafe_interrupts: False # Default is false. For older platforms and AMD we need to add interrupt remapping workaround
   rc:
     local: |
       #!/bin/sh -e
       # Enable 7 VF on eth1
       echo 7 > /sys/class/net/eth1/device/sriov_numvfs; sleep 2; ifup -a
       exit 0
```
Isolate CPU options

Remove the specified CPUs, as defined by the cpu_number values, from the general kernel SMP balancing and scheduler algorithms. The only way to move a process onto or off an “isolated” CPU is via the CPU affinity syscalls. cpu_number begins at 0, so the maximum value is 1 less than the number of CPUs on the system.

```yaml
linux:
  system:
    kernel:
      isolcpu: 1,2,3,4,5,6,7 # isolate first cpu 0
```

Repositories

RedHat based Linux with additional OpenStack repo

```yaml
linux:
  system:
    ... 
    repo:
      rdo-icehouse:
        enabled: true
        source: 'http://repos.fedorapeople.org/repos/openstack/openstack-icehouse/epel-6/'
        pgpcheck: 0

Ensure system repository to use czech Debian mirror (default: true) Also pin it’s packages with priority 900.

```yaml
linux:
  system:
    repo:
      debian:
        default: true
        source: "deb http://ftp.cz.debian.org/debian/ jessie main contrib non-free"
        # Import signing key from URL if needed
        key_url: "http://dummy.com/public.gpg"
        pin:
          - pin: 'origin "ftp.cz.debian.org"'
            priority: 900
            package: '*'
```

Package manager proxy setup globally:

```yaml
linux:
  system:
    ... 
    repo:
      apt-mk:
        source: "deb http://apt-mk.mirantis.com/ stable main salt"
        ...
      proxy:
        pkg:
          enabled: true
          ftp: ftp://ftp-proxy-for-apt.host.local:2121
        ...
        # NOTE: Global defaults for any other component that configure proxy on the
        system.
```
# If your environment has just one simple proxy, set it on

→ **linux:** system:proxy.

→ **# fall back system defaults if linux:system:proxy:pkg has no protocol specific entries**

→ **# as for https and http**

→ **ftp:** ftp://proxy.host.local:2121
→ **http:** http://proxy.host.local:3142
→ **https:** https://proxy.host.local:3143

Package manager proxy setup per repository:

```
linux:
  system:
    ...
    repo:
      debian:
        source: "deb http://apt-mk.mirantis.com/ stable main salt"
      ...
    apt-mk:
      source: "deb http://apt-mk.mirantis.com/ stable main salt"
      # per repository proxy
      proxy:
        enabled: true
        http: http://maas-01:8080
        https: http://maas-01:8080
    ...
  proxy:
    # package manager fallback defaults
    # used if linux:system:repo:apt-mk:proxy has no protocol specific entries
    pkg:
      enabled: true
      ftp: ftp://proxy.host.local:2121
      http: http://proxy.host.local:3142
      https: https://proxy.host.local:3143
    ...
    # global system fallback system defaults
    ftp: ftp://proxy.host.local:2121
    http: http://proxy.host.local:3142
    https: https://proxy.host.local:3143
```

Remove all repositories:

```
linux:
  system:
    purge_repos: true
```

Setup custom apt config options:

```
linux:
  system:
    apt:
      config:
        compression-workaround:
          "Acquire::CompressionTypes::Order": "gz"
        docker-clean:
          "DPkg::Post-Invoke":
```

(continues on next page)
- "rm -f /var/cache/apt/archives/*.deb /var/cache/apt/archives/partial/*.
   → deb /var/cache/apt/*.bin || true"

APT::Update::Post-Invoke:
- "rm -f /var/cache/apt/archives/*.deb /var/cache/apt/archives/partial/*.
   → deb /var/cache/apt/*.bin || true"

RC

rc.local example

```
linux:
  system:
    rc:
      local: |
        #!/bin/sh -e
        
        # rc.local
        
        # This script is executed at the end of each multiuser runlevel.
        # Make sure that the script will "exit 0" on success or any other
        # value on error.
        
        # In order to enable or disable this script just change the execution
        # bits.
        
        # By default this script does nothing.
        exit 0
```

Prompt

Setting prompt is implemented by creating /etc/profile.d/prompt.sh. Every user can have different prompt.

```
linux:
  system:
    prompt:
      root: \n\n\[\033[0;37m\]D\%y/\%m/\%d \%H:\%M:\%S \$(hostname -f)\n\n\[\e[1;31m\]
\[\u@\h:\w\]
[\e[0m]\n
default: \n\D\%y/\%m/\%d \%H:\%M:\%S \$(hostname -f)\n\n\[\u@\h:\w\]
```

On Debian systems to set prompt system-wide it’s necessary to remove setting PS1 in /etc/bash.bashrc and ~/.bashrc (which comes from /etc/skel/.bashrc). This formula will do this automatically, but will not touch existing user’s ~/.bashrc files except root.

Bash

Fix bash configuration to preserve history across sessions (like ZSH does by default).

```
linux:
  system:
    bash:
      preserve_history: true
```
Message of the day

`pam_motd` from package `update-motd` is used for dynamic messages of the day. Setting custom motd will cleanup existing ones.

```bash
linux:
  system:
    motd:
      - release: |
        #!/bin/sh
        [ -r /etc/lsb-release ] && . /etc/lsb-release

        if [ -z "DISTRIB_DESCRIPTION" ] && [ -x /usr/bin/lsb_release ]; then
          # Fall back to using the very slow lsb_release utility
          DISTRIB_DESCRIPTION=$(lsb_release -s -d)
        fi

        printf "Welcome to %s (%s %s %s)\n" "$DISTRIB_DESCRIPTION" "$(uname -o)" 
         "$(uname -r)" "$(uname -m)"
      - warning: |
        #!/bin/sh
        printf "This is [company name] network.\n"
        printf "Unauthorized access strictly prohibited.\n"
```

Services

Stop and disable linux service:

```bash
linux:
  system:
    service:
      apt-daily.timer:
        status: dead
```

Possible status is dead (disable service by default), running (enable service by default), enabled, disabled.

Linux with atop service:

```bash
linux:
  system:
    atop:
      enabled: true
      interval: 20
      logpath: "/var/log/atop"
      outfile: "/var/log/atop/daily.log"
```

RHEL / CentOS

Unfortunately `update-motd` is currently not available for RHEL so there's no native support for dynamic motd. You can still set static one, only pillar structure differs:

```bash
linux:
  system:
    motd: |
```

(continues on next page)
Haveged

If you are running headless server and are low on entropy, it may be a good idea to setup Haveged.

```
linux:
    system:
        haveged:
            enabled: true
```

Linux network

Linux with network manager

```
linux:
    network:
        enabled: true
        network_manager: true
```

Linux with default static network interfaces, default gateway interface and DNS servers

```
linux:
    network:
        enabled: true
        interface:
            eth0:
                enabled: true
                type: eth
                address: 192.168.0.102
                netmask: 255.255.255.0
                gateway: 192.168.0.1
                name_servers:
                    - 8.8.8.8
                    - 8.8.4.4
                mtu: 1500
```

Linux with bonded interfaces and disabled NetworkManager

```
linux:
    network:
        enabled: true
        interface:
            eth0:
                type: eth
            ...
            eth1:
                type: eth
            ...
            bond0:
                enabled: true
                type: bond
```
address: 192.168.0.102
netmask: 255.255.255.0
mtu: 1500
use_in:
  - interface: ${linux:interface:eth0}
  - interface: ${linux:interface:eth0}

network_manager:
  disable: true

Linux with vlan interface_params

linux:
  network:
    enabled: true
    interface:
      vlan69:
        type: vlan
        use_interfaces:
          - interface: ${linux:interface:bond0}

Linux with wireless interface parameters

linux:
  network:
    enabled: true
    gateway: 10.0.0.1
    default_interface: eth0
    interface:
      wlan0:
        type: eth
        wireless:
          essid: example
          key: example_key
          security: wpa
          priority: 1

Linux networks with routes defined

linux:
  network:
    enabled: true
    gateway: 10.0.0.1
    default_interface: eth0
    interface:
      eth0:
        type: eth
        route:
          default:
            address: 192.168.0.123
            netmask: 255.255.255.0
            gateway: 192.168.0.1

Native Linux Bridges

linux:
  network:
interface:
    eth1:
        enabled: true
        type: eth
        proto: manual
        up_cmds:
            - ip address add 0/0 dev $IFACE
            - ip link set $IFACE up
        down_cmds:
            - ip link set $IFACE down
    br-ex:
        enabled: true
        type: bridge
        address: ${linux:network:host:public_local:address}
        netmask: 255.255.255.0
        use_interfaces:
            - eth1

OpenVswitch Bridges

linux:
    network:
        bridge: openvswitch
    interface:
        eth1:
            enabled: true
            type: eth
            proto: manual
            up_cmds:
                - ip address add 0/0 dev $IFACE
                - ip link set $IFACE up
            down_cmds:
                - ip link set $IFACE down
        br-ex:
            enabled: true
            type: bridge
            address: ${linux:network:host:public_local:address}
            netmask: 255.255.255.0
            use_interfaces:
                - eth1
            br-prv:
                enabled: true
                type: ovs_bridge
                mtu: 65000
            br-ens7:
                enabled: true
                name: br-ens7
                type: ovs_bridge
                proto: manual
                mtu: 9000
                use_interfaces:
                    - ens7
            patch-br-ens7-br-prv:
                enabled: true
                name: ens7-prv
                ovs_type: ovs_port
                type: ovs_port

(continues on next page)
bridge: br-ens7
port_type: patch
peer: prv-ens7
mtu: 65000
patch-br-prv-br-ens7:
  enabled: true
  name: prv-ens7
  bridge: br-prv
  ovs_type: ovs_port
  type: ovs_port
  port_type: patch
  peer: ens7-prv
  mtu: 65000
ens7:
  enabled: true
  name: ens7
  proto: manual
  ovs_port_type: OVSPort
  type: ovs_port
  ovs_bridge: br-ens7
  bridge: br-ens7

Debian manual proto interfaces
When you are changing interface proto from static in up state to manual, you may need to flush ip addresses. For example, if you want to use the interface and the ip on the bridge. This can be done by setting the `ipflush_onchange` to true.

```
linux:
  network:
    interface:
      eth1:
        enabled: true
        type: eth
        proto: manual
        mtu: 9100
        ipflush_onchange: true
```

Debian static proto interfaces
When you are changing interface proto from dhcp in up state to static, you may need to flush ip addresses and restart interface to assign ip address from a managed file. For example, if you want to use the interface and the ip on the bridge. This can be done by setting the `ipflush_onchange` with combination `restart_on_ipflush` param set to `true`.

```
linux:
  network:
    interface:
      eth1:
        enabled: true
        type: eth
        proto: static
        address: 10.1.0.22
        netmask: 255.255.255.0
        ipflush_onchange: true
        restart_on_ipflush: true
```

Concatinating and removing interface files
Debian based distributions have `/etc/network/interfaces.d/` directory, where you can store configuration of network interfaces in separate files. You can concatenate the files to the defined destination when needed, this operation removes the file from the `/etc/network/interfaces.d/`. If you just need to remove iface files, you can use the `remove_iface_files` key.

```
linux:
  network:
    concat_iface_files:
      - src: '/etc/network/interfaces.d/50-cloud-init.cfg'
        dst: '/etc/network/interfaces'
    remove_iface_files:
      - '/etc/network/interfaces.d/90-custom.cfg'
```

DHCP client configuration

None of the keys is mandatory, include only those you really need. For full list of available options under send, supersede, prepend, append refer to dhcp-options(5)

```
linux:
  network:
    dhclient:
      enabled: true
      backoff_cutoff: 15
      initial_interval: 10
      retry: 60
      select_timeout: 0
      timeout: 120
      send:
        - option: host-name
          declaration: "= gethostname()"
      supersede:
        - option: host-name
          declaration: "spaceship"
        - option: domain-name
          declaration: "domain.home"
      # - option: arp-cache-timeout
      #   declaration: 20
      prepend:
        - option: domain-name-servers
          declaration:
            - 8.8.8.8
            - 8.8.4.4
        - option: domain-search
          declaration:
            - example.com
            - eng.example.com
      #append:
      #  - option: domain-name-servers
      #    declaration: 127.0.0.1
      #  # ip or subnet to reject dhcp offer from
      reject:
        - 192.33.137.209
        - 10.0.2.0/24
      request:
        - subnet-mask
        - broadcast-address
        - time-offset
```
- routers
- domain-name
- domain-name-servers
- domain-search
- host-name
- dhcp6.name-servers
- dhcp6.domain-search
- dhcp6.fqdn
- dhcp6.snmp-servers
- netbios-name-servers
- netbios-scope
- interface-mtu
- rfc3442-classless-static-routes
- ntp-servers
require:
- subnet-mask
- domain-name-servers
# if per interface configuration required add below
interface:
  ens2:
    initial_interval: 11
    reject:
      - 192.33.137.210
  ens3:
    initial_interval: 12
    reject:
      - 192.33.137.211

Linux network systemd settings:

```yaml
linux:
  network:
    ...
  systemd:
    link:
      10-iface-dmz:
        Match:
          MACAddress: c8:5b:67:fa:1a:af
          OriginalName: eth0
        Link:
          Name: dmz0
    netdev:
      20-bridge-dmz:
        match:
          name: dmz0
        network:
          description: bridge
          bridge: br-dmz0

# works with lowercase, keys are by default capitalized
40-dhcp:
  match:
    name: '*'
  network:
    DHCP: yes
```

Configure global environment variables

2.1. Project Introduction 95
Use `/etc/environment` for static system wide variable assignment after boot. Variable expansion is frequently not supported.

```
    linux:
      system:
        env:
          BOB_VARIABLE: Alice
          ...
          BOB_PATH:
            - /srv/alice/bin
            - /srv/bob/bin
          ...
          ftp_proxy: none
          https_proxy: ${linux:system:proxy:https}
          no_proxy:
            - 192.168.0.80
            - 192.168.1.80
            - .domain.com
            - .local
          ...
          # NOTE: global defaults proxy configuration.
          proxy:
            ftp: ftp://proxy.host.local:2121
            http: http://proxy.host.local:3142
            https: https://proxy.host.local:3143
            noproxy:
              - .domain.com
              - .local
```

Configure profile.d scripts

The profile.d scripts are being sourced during .sh execution and support variable expansion in opposite to `/etc/environment` global settings in `/etc/environment`.

```
    linux:
      system:
        profile:
          locales: |
            export LANG=C
            export LC_ALL=C
          ...
          vi_flavors.sh: |
            export PAGER=view
            export EDITOR=vim
            alias vi=vim
          shell_locales.sh: |
            export LANG=en_US
            export LC_ALL=en_US.UTF-8
          shell_proxies.sh: |
            export FTP_PROXY=ftp://127.0.3.3:2121
            export NO_PROXY='.'.local'
```

Linux with hosts

Parameter purge_hosts will enforce whole `/etc/hosts` file, removing entries that are not defined in model except defaults for both IPv4 and IPv6 localhost and hostname + fqdn.

It’s good to use this option if you want to ensure `/etc/hosts` is always in a clean state however it’s not enabled by default.
for safety.

```yaml
linux:
  network:
    purge_hosts: true
    host:
      # No need to define this one if purge_hosts is true
      hostname:
        address: 127.0.1.1
        names:
        - $(linux:network:fqdn)
        - $(linux:network:hostname)
      node1:
        address: 192.168.10.200
        names:
        - node2.domain.com
        - service2.domain.com
      node2:
        address: 192.168.10.201
        names:
        - node2.domain.com
        - service2.domain.com
```

Linux with hosts collected from mine

In this case all dns records defined within infrastructure will be passed to local hosts records or any DNS server. Only hosts with grain parameter to true will be propagated to the mine.

```yaml
linux:
  network:
    purge_hosts: true
    mine_dns_records: true
    host:
      node1:
        address: 192.168.10.200
        grain: true
        names:
        - node2.domain.com
        - service2.domain.com
```

Setup resolv.conf, nameservers, domain and search domains

```yaml
linux:
  network:
    resolv:
      dns:
      - 8.8.4.4
      - 8.8.8.8
      domain: my.example.com
      search:
      - my.example.com
      - example.com
      options:
      - ndots: 5
      - timeout: 2
      - attempts: 2
```

setting custom TX queue length for tap interfaces

2.1. Project Introduction 97
DPDK OVS interfaces

DPDK OVS NIC

```
linux:
  network:
    tap_custom_txqueue_len: 10000

DPDK OVS Bond

```

```text
linux:
  network:
    bridge: openvswitch
    dpdk:
      enabled: true
      driver: uio/vfio
    openvswitch:
      pmd_cpu_mask: "0x6"
      dpdk_socket_mem: "1024,1024"
      dpdk_lcore_mask: "0x400"
      memory_channels: 2
    interface:
      dpdk0:
        name: ${_param:dpdk_nic}
        pci: 0000:06:00.0
        driver: igb_uio/vfio-pci
        enabled: true
        type: dpdk_ovs_port
        n rxq: 2
        pmd_rxq_affinity: "0:1,1:2"
        bridge: br-prv
        mtu: 9000
      br-prv:
        enabled: true
        type: dpdk_ovs_bridge
```

(continues on next page)
dpdk_first_nic:
  name: ${_param:primary_first_nic}
  pci: 0000:05:00.0
  driver: igb_uio/vfio-pci
  bond: dpdkbond0
  enabled: true
  type: dpdk_ovs_port
  n_rxq: 2
  pmd_rxq_affinity: "0:1,1:2"
  mtu: 9000

dpdkbond0:
  enabled: true
  bridge: br-prv
  type: dpdk_ovs_bond
  mode: active-backup
  br-prv:
    enabled: true
    type: dpdk_ovs_bridge

DPDK OVS bridge for VXLAN
If VXLAN is used as tenant segmentation then ip address must be set on br-prv

linux:
  network:
    ...
  interface:
    br-prv:
      enabled: true
      type: dpdk_ovs_bridge
      address: 192.168.50.0
      netmask: 255.255.255.0
      tag: 101
      mtu: 9000

Linux storage

Linux with mounted Samba

linux:
  storage:
    enabled: true
  mount:
    samba1:
      enabled: true
      path: /media/myuser/public/
      device: //192.168.0.1/storage
      file_system: cifs
      options: guest,uid=myuser,iocharset=utf8,file_mode=0777,dir_mode=0777,noperm

NFS mount

linux:
  storage:
    enabled: true
mount:
    nfs_glance:
        enabled: true
        path: /var/lib/glance/images
        device: 172.16.10.110:/var/nfs/glance
        file_system: nfs
        opts: rw,sync

File swap configuration

    linux:
        storage:
            enabled: true
            swap:
                file:
                    enabled: true
                    engine: file
                    device: /swapfile
                    size: 1024

Partition swap configuration

    linux:
        storage:
            enabled: true
            swap:
                partition:
                    enabled: true
                    engine: partition
                    device: /dev/vg0/swap

LVM group vg1 with one device and data volume mounted into /mnt/data

    parameters:
        linux:
            storage:
                mount:
                    data:
                        enabled: true
                        device: /dev/vg1/data
                        file_system: ext4
                        path: /mnt/data
                    lvm:
                        vg1:
                            enabled: true
                            devices:
                            - /dev/sdb
                            volume:
                                data:
                                    size: 40G
                                    mount: ${linux:storage:mount:data}

Create partitions on disk. Specify size in MB. It expects empty disk without any existing partitions. (set startsector=1, if you want to start partitions from 2048)
linux:
  storage:
    disk:
      first_drive:
        startsector: 1
        name: /dev/loop1
        type: gpt
        partitions:
        - size: 200 #size in MB
type: fat32
        - size: 300 #size in MB
          mkfs: True
type: xfs
/dev/vda1:
  partitions:
  - size: 5
type: ext2
  - size: 10
type: ext4

Multipath with Fujitsu Eternus DXL

parameters:
  linux:
    storage:
      multipath:
        enabled: true
        blacklist_devices:
        - /dev/sda
        - /dev/sdb
        backends:
        - fujitsu_eternus_dxl

Multipath with Hitachi VSP 1000

parameters:
  linux:
    storage:
      multipath:
        enabled: true
        blacklist_devices:
        - /dev/sda
        - /dev/sdb
        backends:
        - hitachi_vsp1000

Multipath with IBM Storwize

parameters:
  linux:
    storage:
      multipath:
        enabled: true
        blacklist_devices:
        - /dev/sda
        - /dev/sdb
        backends:
        - ibm_storwize

2.1. Project Introduction
### Multipath with multiple backends

<table>
<thead>
<tr>
<th>parameters:</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>linux:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>storage:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>multipath:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>enabled:</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>blacklist_devices:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/dev/sda</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/dev/sdb</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/dev/sdc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>/dev/sdd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>backends:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ibm_storwize</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fujitsu_eternus_dxl</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hitachi_vsp1000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### PAM LDAP integration

<table>
<thead>
<tr>
<th>parameters:</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>linux:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>system:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>auth:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>enabled:</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>ldap:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>enabled:</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>binddn:</td>
<td>cn=bind,ou=service_users,dc=example,dc=com</td>
<td></td>
</tr>
<tr>
<td>bindpw:</td>
<td>secret</td>
<td></td>
</tr>
<tr>
<td>uri:</td>
<td>ldap://127.0.0.1</td>
<td></td>
</tr>
<tr>
<td>base:</td>
<td>ou=users,dc=example,dc=com</td>
<td></td>
</tr>
<tr>
<td>ldap_version:</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>pagesize:</td>
<td>65536</td>
<td></td>
</tr>
<tr>
<td>referrals:</td>
<td>off</td>
<td></td>
</tr>
<tr>
<td>filter:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>passwd:</td>
<td>(&amp;(&amp;(objectClass=person)(uidNumber=<em>))(unixHomeDirectory=</em>))</td>
<td></td>
</tr>
<tr>
<td>shadow:</td>
<td>(&amp;(&amp;(objectClass=person)(uidNumber=<em>))(unixHomeDirectory=</em>))</td>
<td></td>
</tr>
<tr>
<td>group:</td>
<td>(&amp;(objectClass=group)(gidNumber=*))</td>
<td></td>
</tr>
</tbody>
</table>

### Disabled multipath (the default setup)

<table>
<thead>
<tr>
<th>parameters:</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>linux:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>storage:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>multipath:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>enabled:</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

### Linux with local loopback device

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>linux:</td>
<td></td>
</tr>
<tr>
<td>storage:</td>
<td></td>
</tr>
<tr>
<td>loopback:</td>
<td></td>
</tr>
<tr>
<td>disk1:</td>
<td></td>
</tr>
<tr>
<td>file: /srv/disk1</td>
<td></td>
</tr>
<tr>
<td>size: 50G</td>
<td></td>
</tr>
</tbody>
</table>
External config generation

You are able to use config support metadata between formulas and only generate config files for external use, eg. docker, etc.

```yaml
parameters:
  linux:
    system:
      config:
        pillar:
          jenkins:
            master:
              home: /srv/volumes/jenkins
              approved_scripts:
                - method java.net.URL openConnection
              credentials:
                - type: username_password
                  scope: global
                  id: test
                  desc: Testing credentials
                  username: test
                  password: test
```

Netconsole Remote Kernel Logging

Netconsole logger could be configured for configfs-enabled kernels (`CONFIG_NETCONSOLE_DYNAMIC` should be enabled). Configuration applies both in runtime (if network is already configured), and on-boot after interface initialization. Notes:

- receiver could be located only in same L3 domain (or you need to configure gateway MAC manually)
- receiver’s MAC is detected only on configuration time
- using broadcast MAC is not recommended

```yaml
parameters:
  linux:
    system:
      netconsole:
        enabled: true
        port: 514 (optional)
        loglevel: debug (optional)
        target:
          192.168.0.1:
            interface: bond0
            mac: "ff:ff:ff:ff:ff:ff" (optional)
```

Usage

Set mtu of network interface eth0 to 1400

```
ip link set dev eth0 mtu 1400
```
Read more

- https://www.archlinux.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-linux/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-linux

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

network configuration formula

Sets up network devices.

Sample pillars

Single network config snippet

```
network:
  control:
    enabled: true
  config:
    switch_vlan:
      eth0-0-1:
        address: 10.0.0.1/24
      eth0-0-2:
        address: 10.0.0.2/24
      eth0-0-3:
        address: 10.0.0.3/24
  device:
    vsrx1:
      interface:
        eth0-0-1: $(network:control:config:switch_vlan)
```
JunOS VSRX device

```
network:
  control:
    enabled: true
    managed: true
  device:
    vsrx1:
      type: junos
      auth:
        password: $1$gpbfk/Jr$
      interface:
        eth0-0-1:
          address: 10.0.0.1/24
```

**Read more**

- links

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-network/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-network

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**NFS Formula**

**Sample Pillars**

NFS Server: Basic sharing

```
nfs:
  server:
    enabled: true
```

(continues on next page)
share:
  home_majklk:
    path: /home/majklk
    host:
      inter:
        host: 10.10.10.0/24
        params:
        - rw
        - no_root_squash
        - sync
      pub:
        host: 10.0.0.0/24
        params:
        - rw
        - no_root_squash
        - sync

NFS Client with mounted directory

nfs:
  client:
    enabled: true
  mount:
    samba1:
      path: /media/myuser/public/
      fstype: nfs
      device: 192.168.0.1:/home/majklk

NFS mount

linux:
  storage:
    mount:
      nfs:
        enabled: true
        path: /var/lib/glance
        file_system: nfs
        device: 10.0.103.152:/storage/glance/vpc20

More Information


NTP

Network time synchronisation services.

Sample pillars

NTP client
salt:\n  client:\n    enabled: true\n  strata:\n  \- ntp.cesnet.cz\n  \- ntp.nic.cz

Read more

\- https://collectd.org/wiki/index.php/Plugin:NTPd

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-ntp/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-ntp

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

OpenSSH

OpenSSH is a FREE version of the SSH connectivity tools that technical users of the Internet rely on. Users of telnet, rlogin, and ftp may not realize that their password is transmitted across the Internet unencrypted, but it is. OpenSSH encrypts all traffic (including passwords) to effectively eliminate eavesdropping, connection hijacking, and other attacks. Additionally, OpenSSH provides secure tunneling capabilities and several authentication methods, and supports all SSH protocol versions.

Sample pillar

OpenSSH client

OpenSSH client with shared private key
OpenSSH client with individual private key and known host

```yaml
openssh:
  client:
    enabled: true
  user:
    root:
      enabled: true
      private_key:
        type: rsa
        key: ${_param:root_private_key}
      user: ${linux:system:user:root}
    known_hosts:
      - name: repo.domain.com
        type: rsa
        fingerprint_hash_type: sha256|md5
```

Configure keep alive settings:

```yaml
openssh:
  client:
    alive:
      interval: 600
      count: 3
```

OpenSSH server

OpenSSH server with configuration parameters

```yaml
openssh:
  server:
    enabled: true
    permit_root_login: true
    public_key_auth: true
    password_auth: true
    host_auth: true
    banner: Welcome to server!
    bind:
      address: 0.0.0.0
      port: 22
```

OpenSSH server with auth keys for users. Parameter `purge` will ensure exact `authorized_keys` contents co undefined keys will be removed.
You can also bind openssh on multiple addresses and ports:

```yaml
openssh:
  server:
    enabled: true
    binds:
      - address: 127.0.0.1
        port: 22
      - address: 192.168.1.1
        port: 2222
```

**OpenSSH server for use with FreeIPA**

```yaml
openssh:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 22
    public_key_auth: true
    authorized_keys_command:
      command: /usr/bin/sss_ssh_authorizedkeys
      user: nobody
```

**Configure keep alive settings:**

```yaml
openssh:
  server:
    alive:
      keep: yes
      interval: 600
      count: 3
# will give you an timeout of 30 minutes (600 sec x 3)
```

**Enable DSA legacy keys:**
openssh:
  server:
    dss_enabled: true

Read more

- https://help.ubuntu.com/community/SSH/OpenSSH/Configuring
- http://www.zeitoun.net/articles/ssh-through-http-proxy/start

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-openssh/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-openssh

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

OpenVPN

OpenVPN can tunnel any IP subnetwork or virtual ethernet adapter over a single UDP or TCP port, configure a scalable, load-balanced VPN server farm using one or more machines which can handle thousands of dynamic connections from incoming VPN clients.

Sample pillars

Simple OpenVPN server

openvpn:
  server:
    enabled: true
    device: tun

(continues on next page)
ssl:
  authority: Domain_Service_CA
  certificate: server.domain.com

bind:
  address: 0.0.0.0
  port: 1194
  protocol: tcp

OpenVPN server with private subnet with DHCP and predefined clients

```
openvpn:
  server:
    ...
    interface:
      topology: subnet
      network: 10.0.8.0
      netmask: 255.255.255.0
      dhcp_pool:
        start: 10.0.8.100
        end: 10.0.8.199
      clients:
        - name: client1.domain.com
          address: 10.0.8.12
        - name: client2.domain.com
          address: 10.0.8.13
```

```
openvpn:
  server:
    ...
    topology: subnet
    interface:
      network: 10.0.8.0
      netmask: 255.255.255.0
    dhcp_pool:
      start: 10.0.8.100
      end: 10.0.8.199
    topology: gateway
    device: tun
    mode: p2p
    interface:
      network: 10.0.8.0
      netmask: 255.255.255.0
    endpoint:
      local: 10.8.0.1
      remote: 10.8.0.2
    dhcp_pool:
      start: 10.8.0.4
      end: 10.8.0.255
    routes:
      - network: 10.8.0.1
        netmask: 255.255.255.255
      - network: 10.0.110.0
        netmask: 255.255.255.0
      - network: 10.0.101.0
        netmask: 255.255.255.0
```

OpenVPN server with custom auth

2.1. Project Introduction
openvpn:
  server:
    ...
    interface:
      topology: subnet
      network: 10.0.8.0
      netmask: 255.255.255.0
    auth:
      engine: pam/google-authenticator
    ssl:
      authority: Domain_Service_CA
      certificate: server.domain.com

Single OpenVPN client with multiple servers

openvpn:
  client:
    enabled: true
    tunnel:
      tunnel_name:
        autostart: true
        servers:
          - host: 10.0.0.1
            port: 1194
          - host: 10.0.0.2
            port: 1194
        protocol: tcp
        device: tup
        compression: true
        ssl:
          authority: Domain_Service_CA
          certificate: client.domain.com

Multiple OpenVPN clients

openvpn:
  client:
    enabled: true
    tunnel:
      tunnel01:
        autostart: true
        server:
          host: 10.0.0.1
          port: 1194
          protocol: tcp
          device: tup
          compression: true
          ssl:
            engine: salt
            authority: Domain_Service_CA
            certificate: client.domain.com
      tunnel02:
        autostart: true
        server:
          host: 10.0.0.1
          port: 1194
          protocol: tcp
device: tup
  compression: true
  ssl:
    engine: salt
    authority: Domain_Service_CA
    certificate: client.domain.com

OpenVPN client auth

openvpn:
  client:
    enabled: true
  tunnel:
    tunnel01:
      auth:
        engine: pam/google-authenticator
      ssl:
        engine: salt
        authority: Domain_Service_CA
        certificate: client.domain.com

Read more

- https://github.com/luxflux/puppet-openvpn

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-openvpn/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-openvpn

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

2.1. Project Introduction
pritunl

Pritunl is a distributed enterprise vpn server built using the OpenVPN protocol.

Sample pillars

Single pritunl service

```yaml
pritunl:
  server:
    enabled: true
```

Read more

• https://github.com/pritunl/pritunl

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-pritunl/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-pritunl

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Reclass Formula

reclass is an “external node classifier” (ENC) as can be used with automation tools, such as Puppet, Salt, and Ansible. It is also a stand-alone tool for merging data sources recursively.

Sample Metadata

Install sources from [repository, git, pip]
salt:
  source:
    engine: pkg
...  
  source:
    engine: git
    repo: git+https://github.com/salt-formulas/reclass
    branch: master
...  
  source:
    engine: pip
...

If reclass is pre-installed, set the engine to None to avoid updates.

salt:
  source:
    engine: None

Reclass storage with data fetched from git

reclass:
  storage:
    enabled: true
    base_dir: /srv/reclass
    source:
      engine: git
      repo: git+https://github.com/salt-formulas/reclass
      branch: master

Reclass storage with local data source

reclass:
  storage:
    enabled: true
    base_dir: /srv/reclass
data_source:
      engine: local

Reclass storage with archive data source

reclass:
  storage:
    enabled: true
    base_dir: /srv/reclass
data_source:
      engine: archive
      address: salt://path/reclass-project.tar

Reclass storage with archive data source with content hash check

reclass:
  storage:
    enabled: true
    base_dir: /srv/reclass
data_source:
      engine: archive

(continues on next page)
Reclass model with single node definition

```yaml
reclass:
  storage:
    enabled: true
  node:
    service_node01:
      name: svc01
      domain: deployment.local
      classes:
        - cluster.deployment_name.service.role
    params:
      salt_master_host: <<salt-master-ip>>
      linux_system_codename: trusty
      single_address: <<node-ip>>
```

Reclass model with multiple node defined

```yaml
reclass:
  storage:
    enabled: true
    repeat_replace_symbol: <<count>>
  node:
    service_node01:
      name: node<<count>>
      domain: deployment.local
      classes:
        - cluster.deployment.service.role
    repeat:
      count: 2
      start: 5
      digits: 2
    params:
      single_address:
        value: 10.0.0.<<count>>
        start: 100
      deploy_address:
        value: part-<<count>>-whole
        start: 5
        digits: 3
      params:
        salt_master_host: <<salt-master-ip>>
        linux_system_codename: trusty
```

Reclass model with multiple node defined and interpolation enabled

```yaml
reclass:
  storage:
    enabled: true
    repeat_replace_symbol: <<count>>
  node:
    service_node01:
      name: node<<count>>
      domain: deployment.local
      classes:
        - cluster.deployment.service.role
    repeat:
      count: 2
      start: 5
      digits: 2
    params:
      single_address:
        value: 10.0.0.<<count>>
        start: 100
      deploy_address:
        value: part-<<count>>-whole
        start: 5
        digits: 3
      params:
        salt_master_host: <<salt-master-ip>>
        linux_system_codename: trusty
```
classes:
- cluster.deployment.service.role
repeat:
count: 2
start: 5
digits: 2
params:
  single_address:
  value: ceph_osd_node<<count>>_address
  start: 1
digits: 2
  interpolate: true
params:
  salt_master_host: <<salt-master-ip>>
linux_system_codename: trusty

Reclass storage with simple class mappings

reclass:
  storage:
    enabled: true
class_mappings:
- target: '\''
class: default
ignore_class_notfound: true

Reclass models with dynamic node classification

reclass:
  storage:
    enabled: true
class_mapping:
  common_node:
    expression: all
  node_param:
    single_address:
    value_template: <<node_ip>>
linux_system_codename:
    value_template: <<node_os>>
salt_master_host:
    value_template: <<node_master_ip>>
infra_config:
  expression: <<node_hostname>>__startswith__cfg
cluster_param:
  infra_config_address:
  value_template: <<node_ip>>
infra_config_deploy_address:
  value_template: <<node_ip>>
infra_proxy:
  expression: <<node_hostname>>__startswith__prx
node_class:
  value_template:
    - cluster.<<node_cluster>>.stacklight.proxy
kubernetes_control01:
  expression: <<node_hostname>>__equals__ctl01
cluster_param:
  kubernetes_control_node01_address:
Classify node after creation and unclassify on node deletion

```
salt:
   master:
      reactor:
         reclass/minion/classify:
            - salt://reclass/reactor/node_register.sls
         reclass/minion/declassify:
            - salt://reclass/reactor/node_unregister.sls
```

Event to trigger the node classification

```
salt-call event.send 'reclass/minion/classify' "{'node_master_ip': '$config_host',
    'node_ip': '${node_ip}', 'node_domain': '$node_domain', 'node_cluster': '$node_cluster', 'node_hostname': '${node_hostname}', 'node_os': '${node_os}')"
```

**Note:** You can send any parameters in the event payload, all will be checked against dynamic node classification conditions.

Both actions will use the minion ID as the node_name to be updated.

Event to trigger the node declassification

```
salt-call event.send 'reclass/minion/declassify'
```

**More Information**

- [http://reclass.pantsfullofunix.net/index.html](http://reclass.pantsfullofunix.net/index.html)
- [http://reclass.pantsfullofunix.net/operations.html](http://reclass.pantsfullofunix.net/operations.html)

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)
In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-reclass/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-reclass

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Salt Formula**

Salt is a new approach to infrastructure management. Easy enough to get running in minutes, scalable enough to manage tens of thousands of servers, and fast enough to communicate with them in seconds.

Salt delivers a dynamic communication bus for infrastructures that can be used for orchestration, remote execution, configuration management and much more.

**Sample Metadata**

**Salt Master**

Salt master with base formulas and pillar metadata backend

```yaml
git:
  client:
    enabled: true
linux:
  system:
    enabled: true
salt:
  master:
    enabled: true
    command_timeout: 5
    worker_threads: 2
    base_environment: prd
  environment:
    prd:
      formula:
        service01:
          source: git
          address: 'git@git.domain.com/service01-formula.git'
          revision: master
        service02:
          source: pkg
          name: salt-formula-service02
```

(continues on next page)
pillar:
    engine: salt
    source:
        engine: git
        address: 'git@repo.domain.com:salt/pillar-demo.git'
        branch: 'master'

Salt master with reclass ENC metadata backend

git:
    client:
        enabled: true

linux:
    system:
        enabled: true

reclass:
    storage:
        enabled: true

    data_source:
        engine: git
        address: 'git@git.domain.com'
        branch: master

salt:
    master:
        enabled: true

        command_timeout: 5
        worker_threads: 2

        base_environment: prd

    environment:
        prd:

            formula:
                service01:
                    source: git
                    address: 'git@git.domain.com/service01-formula.git'

                revision: master

            service02:
                source: pkg
                name: salt-formula-service02

pillar:
    engine: reclass

reclass:
    storage_type: yaml_fs

    inventory_base_uri: /srv/salt/reclass

    propagate_pillar_data_to_reclass: False

    reclass_source_path: /tmp/reclass

Salt master with Architect ENC metadata backend

salt:
    master:
        enabled: true

pillar:
    engine: architect

    project: project-name

    host: architect-api

    port: 8181

    username: salt
Salt Stack-Formulas Documentation, Release master

(continued from previous page)

password: password

Salt master with multiple ext_pillars

git:
  client:
    enabled: true
linux:
  system:
    enabled: true
reclass:
  storage:
    enabled: true
  data_source:
    engine: git
    branch: master
    address: 'https://github.com/salt-formulas/openstack-salt.git'
salt:
  master:
    enabled: true
    command_timeout: 5
    worker_threads: 2
    base_environment: prd
    pillar_safe_render_error: False

#environment:
  # prd:
  #  formula:
  #    python:
  #      source: git
  #      address: 'https://github.com/salt-formulas/salt-formula-python.git'
  #    revision: master
pillar:
  engine: composite
reclass:
  # index: 1 is default value
  index: 1
  storage_type: yaml_fs
  inventory_base_uri: /srv/salt/reclass_encrypted
  class_mappings:
    - target: '/^cfg\d+/'
      class: system.non-existing.class
  ignore_class_notfound: True
  ignore_class_regexp:
    - 'service.*'
    - '*.fluentd'
  propagate_pillar_data_to_reclass: False
stack: # not yet implemented
  # https://docs.saltstack.com/en/latest/ref/pillar/all/salt.pillar.stack.html
  # option 1
  # path:
  #   - /path/to/stack.cfg
  # option 2
  pillar:environment:
    dev: path/to/dev/stasck.cfg
    prod: path/to/prod/stasck.cfg
  grains:custom:grain:
    value:

(continues on next page)
Salt master with API

```yaml
- /path/to/stack1.cfg
- /path/to/stack2.cfg
saltclass:
  path: /srv/salt/saltclass
nacl:
  # if order is provided 99 is used to compose "99-nacl" key name which is
  # later used to order entries
  index: 99
  gpg: {}
vault-1: # not yet implemented
  name: vault
  path: secret/salt
vault-2: # not yet implemented
  name: vault
  path: secret/root
vault: # not yet implemented
  # https://docs.saltstack.com/en/latest/ref/modules/all/salt.modules.vault.html
  name: myvault
  url: https://vault.service.domain:8200
  auth:
    method: token
    token: 11111111-2222-3333-4444-555555555555
  policies:
    - saltstack/minions
    - saltstack/minion/{minion}
nacl:
  # https://docs.saltstack.com/en/develop/ref/modules/all/salt.modules.nacl.html
  box_type: sealedbox
  sk_file: /etc/salt/pki/master/nacl
  pk_file: /etc/salt/pki/master/nacl.pub
  #sk: None
  #pk: None
```

Salt master with API

```yaml
git:
  client:
    enabled: true
linux:
  system:
    enabled: true
salt:
  master:
    command_timeout: 5
    worker_threads: 2
    enabled: true
source:
  engine: pkg
pillar:
  engine: salt
  source:
    engine: local
environment:
  prd:
    formula: {}
api:
  enabled: true
```

(continues on next page)
ssl:
  engine: salt
bind:
  address: 0.0.0.0
  port: 8000

Salt master with defined user ACLs

git:
  client:
    enabled: true
linux:
  system:
    enabled: true
salt:
  master:
    command_timeout: 5
    worker_threads: 3
    enabled: true
    source:
      engine: pkg
pillar:
  engine: local
environment:
  prd:
    formula: {}
user:
  peter:
    enabled: true
    permissions:
      - 'fs.fs'
      - 'fs.\*'

Salt master with preset minions

salt:
  master:
    enabled: true
    minions:
      - name: 'node1.system.location.domain.com'

Salt master with pip based installation (optional)

salt:
  master:
    enabled: true
    ... 
    source:
      engine: pip
      version: 2016.3.0rc2

Install formula through system package management

salt:
  master:

(continues on next page)
enabled: true
...
environment:
  prd:
    keystone:
      source: pkg
      name: salt-formula-keystone
    nova:
      source: pkg
      name: salt-formula-keystone
      version: 0.1+0-20160818133412.24~1.gbp6e1ebb
  postresql:
    source: pkg
    name: salt-formula-postgresql
    version: purged

Formula keystone is installed latest version and the formulas without version are installed in one call to aptpkg module. If the version attribute is present sls iterates over formulas and take action to install specific version or remove it. The version attribute may have these values [latest|purged|removed|<VERSION>].

Clone master branch of keystone formula as local feature branch

salt:
  master:
    enabled: true
...
environment:
  dev:
    formula:
      keystone:
        source: git
        address: git@github.com:openstack/salt-formula-keystone.git
        revision: master
        branch: feature

Salt master with specified formula refs (for example for Gerrit review)

salt:
  master:
    enabled: true
...
environment:
  dev:
    formula:
      keystone:
        source: git
        address: https://git.openstack.org/openstack/salt-formula-keystone
        revision: refs/changes/56/123456/1

Salt master with logging handlers

salt:
  master:
    enabled: true
  handler:
    handler01:
      engine: udp
bind:
  host: 127.0.0.1
  port: 9999

minion:
  handler:
    handler01:
      engine: udp
      bind:
        host: 127.0.0.1
        port: 9999
    handler02:
      engine: zmq
      bind:
        host: 127.0.0.1
        port: 9999

Salt engine definition for saltgraph metadata collector

salt:
  master:
    engine:
      graph_metadata:
        engine: saltgraph
        host: 127.0.0.1
        port: 5432
        user: salt
        password: salt
        database: salt

Salt engine definition for Architect service

salt:
  master:
    engine:
      architect:
        engine: architect
        project: project-name
        host: architect-api
        port: 8181
        username: salt
        password: password

Salt engine definition for sending events from docker events

salt:
  master:
    engine:
      docker_events:
        docker_url: unix://var/run/docker.sock

Salt master peer setup for remote certificate signing

salt:
  master:
    peer:
      ".":
        - x509.sign_remote_certificate
Salt master backup configuration

```yaml
salt:
  master:
    backup: true
    initial_data:
      engine: backupninja
      source: backup-node-host
      host: original-salt-master-id
```

Configure verbosity of state output (used for `salt` command)

```yaml
salt:
  master:
    state_output: changes
```

Pass pillar render error to minion log

**Note:** When set to `False` this option is great for debugging. However it is not recommended for any production environment as it may contain templating data as passwords, etc., that minion should not expose.

```yaml
salt:
  master:
    pillar_safe_render_error: False
```

**Event/Reactor Systems**

Salt synchronize node pillar and modules after start

```yaml
salt:
  master:
    reactor:
      salt/minion/*/start:
        - salt://salt/reactor/node_start.sls
```

Trigger basic node install

```yaml
salt:
  master:
    reactor:
      salt/minion/install:
        - salt://salt/reactor/node_install.sls
```

Sample event to trigger the node installation

```bash
salt-call event.send 'salt/minion/install'
```

Run any defined orchestration pipeline

```yaml
salt:
  master:
    reactor:
      salt/orchestrate/start:
        - salt://salt/reactor/orchestrate_start.sls
```
Event to trigger the orchestration pipeline

```
salt-call event.send 'salt/orchestrate/start' "{'orchestrate': 'salt/orchestrate/infra_install.sls'}"
```

Synchronise modules and pillars on minion start.

```
salt:
  master:
    reactor:
      'salt/minion/*/start':
        - salt://salt/reactor/minion_start.sls
```

Add and/or remove the minion key

```
salt:
  master:
    reactor:
      salt/key/create:
        - salt://salt/reactor/key_create.sls
      salt/key/remove:
        - salt://salt/reactor/key_remove.sls
```

Event to trigger the key creation

```
salt-call event.send 'salt/key/create' \
> "{'node_id': 'id-of-minion', 'node_host': '172.16.10.100', 'orch_post_create': \
> 'kubernetes.orchestrate.compute_install', 'post_create_pillar': {'node_name': 'id-of-minion'}}"
```

**Note:** You can add pass additional `orch_pre_create`, `orch_post_create`, `orch_pre_remove` or `orch_post_remove` parameters to the event to call extra orchestrate files. This can be useful for example for registering/unregistering nodes from the monitoring alarms or dashboards.

The key creation event needs to be run from other machine than the one being registered.

Event to trigger the key removal

```
salt-call event.send 'salt/key/remove'
```

**Encrypted Pillars**

Note: NACL + below configuration will be available in Salt > 2017.7.

External resources:

- Tutorial to configure salt + reclass ext_pillar and nacl: http://apealive.net/post/2017-09-salt-nacl-ext-pillar/
- Saltstack documentation: https://docs.saltstack.com/en/latest/ref/modules/all/salt.modules.nacl.html

Configure salt NACL module:

```
pip install --upgrade libnacl==1.5.2
salt-call --local nacl.keygen /etc/salt/pki/master/nacl
```
local:
  saved sk_file:/etc/salt/pki/master/nacl  pk_file: /etc/salt/pki/master/nacl.pub

salt:
  master:
    pillar:
      reclass: *reclass
      nacl:
        index: 99
      nacl:
        box_type: sealedbox
        sk_file: /etc/salt/pki/master/nacl
        pk_file: /etc/salt/pki/master/nacl.pub
        #sk: None
        #pk: None

NACL encrypt secrets:

salt-call --local nacl.enc 'my_secret_value' pk_file=/etc/salt/pki/master/nacl.pub
hXTkJpC1hcKMS7yZVGESutWrkvzusXfETXkacSkIlxYjfWD1MjmR37Mlmthd1gjXpg4f2AlKb8tc9Woma7q
# or salt-run nacl.enc 'myotherpass'
ADDFD0Rav6p6+63sojl7Htfmcp5trDVyeE4BSPO7ipq8fZuLDIVAzQLf4PCbDqi+Fau5KD3/!E/Pw=

NACL encrypted values on pillar:

Use Boxed syntax NACL[EncryptedValue=] to encode value on pillar:

my_pillar:
  my_nacl:
    key0: unencrypted_value
    key1: 
    →NACL[hXTkJpC1hcKMS7yZVGESutWrkvzusXfETXkacSkIlxYjfWD1MjmR37Mlmthd1gjXpg4f2AlKb8tc9Woma7q]

NACL large files:

NACL within template/native pillars:

pillarexample: user: root password1: {{salt.nacl.dec('DRB7Q6/X5gGSRCtPZyxS6hIbWj0fUA+uaVytou3vJ4=')|json}}
cert_key: {{salt.nacl.dec_file('/srv/salt/env/dev/certs/example.com/cert.nacl')|json}}
cert_key2: {{salt.nacl.dec_file('salt:///certs/example.com/cert2.nacl')|json}}

Salt Syndic

The master of masters

salt:
  master:
    enabled: true
    order_masters: True

Lower syndicated master

salt:
  syndic:
    enabled: true

(continues on next page)
Syndicated master with multiple master of masters

```
master:
  host: master-of-master-host
  timeout: 5
```

Salt Minion

Simplest Salt minion setup with central configuration node

```
salt:
  minion:
    enabled: true
    master:
      host: config01.dc01.domain.com
```

Multi-master Salt minion setup

```
salt:
  minion:
    enabled: true
    masters:
      - host: config01.dc01.domain.com
      - host: config02.dc01.domain.com
```

Salt minion with salt mine options

```
salt:
  minion:
    enabled: true
    mine:
      interval: 60
      module:
        grains.items: []
        network.interfaces: []
```

Salt minion with graphing dependencies

```
salt:
  minion:
    enabled: true
    graph_states: true
```

Salt minion behind HTTP proxy

2.1. Project Introduction
Salt minion to specify non-default HTTP backend. The default tornado backend does not respect HTTP proxy settings set as environment variables. This is useful for cases where you need to set no_proxy lists.

Salt minion with PKI certificate authority (CA)

```yaml
salt:
  minion:
    backend: urllib2

salt-ca-default:
  enabled: true
  ca:
    salt-ca-default:
      common_name: Test CA Default
      country: Czech
      state: Prague
      locality: Zizkov
      days_valid:
        authority: 3650
        certificate: 90
      signing_policy:
        cert_server:
          type: v3_edge_cert_server
          minions: '*'
        cert_client:
          type: v3_edge_cert_client
          minions: '*'
    ca_edge:
      type: v3_edge_ca
      minions: '*'
    ca_intermediate:
      type: v3_intermediate_ca
      minions: '*'

salt-ca-test:
  common_name: Test CA Testing
  country: Czech
  state: Prague
  locality: Karlin
  days_valid:
    authority: 3650
    certificate: 90
  signing_policy:
    cert_server:
      type: v3_edge_cert_server
      minions: '*'
    cert_client:
      type: v3_edge_cert_client
      minions: '*'
    ca_edge:
      type: v3_edge_ca
```
minions: '*'
ca_intermediate:
  type: v3_intermediate_ca
  minions: '*'
salt-ca-alt:
  common_name: Alt CA Testing
  country: Czech
  state: Prague
  locality: Cesky Krumlov
  days_valid:
    authority: 3650
    certificate: 90
  signing_policy:
    cert_server:
      type: v3_edge_cert_server
      minions: '*'
    cert_client:
      type: v3_edge_cert_client
      minions: '*'
  ca_edge:
    type: v3_edge_ca
    minions: '*'
  ca_intermediate:
    type: v3_intermediate_ca
    minions: '*'
  ca_file: '/etc/test/ca.crt'
  ca_key_file: '/etc/test/ca.key'
user: test
group: test

Salt minion using PKI certificate

salt:
  #master:
  # enabled: true
  # accept_policy:
  # open_mode
  # peer:
  #   '*':
  #     - x509.sign_remote_certificate
minion:
  enabled: true
  trusted_ca_minions:
    - cfg01
  cert:
    ceph_cert:
      alternative_names:
        IP:127.0.0.1,DNS:salt.ci.local,DNS:ceph.ci.local,DNS:radosgw.ci.local,
        DNS:swift.ci.local
    cert_file: /srv/salt/pki/ci/ceph.ci.local.crt
    common_name:
      ceph_mon.ci.local
    key_file: /srv/salt/pki/ci/ceph.ci.local.key
    country: CZ
    state: Prague
locality: Karlin
signing_cert:
  /etc/pki/ca/salt-ca-test/ca.crt
signing_private_key:
  /etc/pki/ca/salt-ca-test/ca.key
# Kitchen-Salt CI trigger `salt-call --local`, below attributes
# can't be used as there is no required SaltMaster connectivity
authority:
  salt-ca-test
#host:
#  salt.ci.local
#signing_policy:
#  cert_server
proxy_cert:
  alternative_names:
    IP:127.0.0.1,DNS:salt.ci.local,DNS:proxy.ci.local
cert_file:
  /srv/salt/pki/ci/prx.ci.local.crt
common_name:
  prx.ci.local
key_file:
  /srv/salt/pki/ci/prx.ci.local.key
country: CZ
state: Prague
locality: Zizkov
signing_cert:
  /etc/pki/ca/salt-ca-default/ca.crt
signing_private_key:
  /etc/pki/ca/salt-ca-default/ca.key
# Kitchen-Salt CI trigger `salt-call --local`, below attributes
# can't be used as there is no required SaltMaster connectivity
authority:
  salt-ca-default
#host:
#  salt.ci.local
#signing_policy:
#  cert_server
test_cert:
  alternative_names:
    IP:127.0.0.1,DNS:salt.ci.local,DNS:test.ci.local
cert_file:
  /srv/salt/pki/ci/test.ci.local.crt
common_name:
  test.ci.local
key_file:
  /srv/salt/pki/ci/test.ci.local.key
country: CZ
state: Prague
locality: Cesky Krumlov
signing_cert:
  /etc/test/ca.crt
signing_private_key:
  /etc/test/ca.key
# Kitchen-Salt CI trigger `salt-call --local`, below attributes
# can't be used as there is no required SaltMaster connectivity
authority:
  salt-ca-alt
Salt minion trust CA certificates issued by salt CA on a specific host (ie: salt-master node)

```yaml
salt:
  minion:
    trusted_ca_minions:
      - cfg01
```

## Salt Minion Proxy

Salt proxy pillar

```yaml
salt:
  minion:
    proxy_minion:
      master: localhost
      device:
        vsrx01.mydomain.local:
          enabled: true
          engine: napalm
        csr1000v.mydomain.local:
          enabled: true
          engine: napalm
```

**Note:** This is pillar of the real salt-minion

Proxy pillar for IOS device

```yaml
proxy:
  proxytype: napalm
  driver: ios
  host: csr1000v.mydomain.local
  username: root
  passwd: r00tme
```

**Note:** This is pillar of the node that is not able to run salt-minion itself

Proxy pillar for JunOS device

```yaml
proxy:
  proxytype: napalm
  driver: junos
  host: vsrx01.mydomain.local
  username: root
  passwd: r00tme
  optional_args:
    config_format: set
```

**Note:** This is pillar of the node that is not able to run salt-minion itself

---

2.1. Project Introduction 133
Salt SSH

Salt SSH with sudoer using key

```yaml
git:
  client:
    enabled: true
linux:
  system:
    enabled: true
salt:
  master:
    command_timeout: 5
    worker_threads: 2
    enabled: true
  source:
    engine: pkg
pillar:
  engine: salt
  source:
    engine: local
environment:
  prd:
    formula: {}
ssh:
  minion:
    node01:
      host: 10.0.0.1
      user: saltssh
      sudo: true
      key_file: /path/to/the/key
      port: 22
```

Salt SSH with sudoer using password

```yaml
git:
  client:
    enabled: true
linux:
  system:
    enabled: true
salt:
  master:
    command_timeout: 5
    worker_threads: 2
    enabled: true
  source:
    engine: pkg
pillar:
  engine: salt
  source:
    engine: local
environment:
  prd:
    formula: {}
ssh:
  minion:
```

(continues on next page)
node01:
  host: 10.0.0.1
  user: saltssh
  sudo: true
  password: password
  port: 22

Salt SSH with root using password

git:
  client:
    enabled: true
linux:
  system:
    enabled: true
salt:
  master:
    command_timeout: 5
    worker_threads: 2
    enabled: true
    source:
      engine: pkg
pillar:
  engine: salt
  source:
    engine: local
environment:
  prd:
    formula: {}
ssh:
  minion:
    node01:
      host: 10.0.0.1
      user: root
      password: password
      port: 22

Salt control (cloud/kvm/docker)

Salt cloud with local OpenStack provider

salt:
  control:
    enabled: true
    cloud_enabled: true
  provider:
    openstack_account:
      engine: openstack
      insecure: true
      region: RegionOne
      identity_url: 'https://10.0.0.2:35357'
      tenant: project
      user: user
      password: 'password'
      fixed_networks:
floating_networks:
  - public
  ignore_cidr: 192.168.0.0/16
cluster:
  dc01_prd:
    domain: dc01.prd.domain.com
    engine: cloud
    config:
      engine: salt
      host: master.dc01.domain.com
    node:
      ubuntu1:
        provider: openstack_account
        image: Ubuntu14.04 x86_64
        size: m1.medium
      ubuntu2:
        provider: openstack_account
        image: Ubuntu14.04 x86_64
        size: m1.medium

Salt cloud with Digital Ocean provider

salt:
  control:
    enabled: true
  cloud_enabled: true
  provider:
    digitalocean_account:
      engine: digital_ocean
      region: New York 1
      client_key: xxxxxxx
      api_key: xxxxxxx
  cluster:
    dc01_prd:
      domain: dc01.prd.domain.com
      engine: cloud
      config:
        engine: salt
        host: master.dc01.domain.com
      node:
        ubuntu1:
          provider: digitalocean_account
          image: Ubuntu14.04 x86_64
          size: m1.medium
        ubuntu2:
          provider: digitalocean_account
          image: Ubuntu14.04 x86_64
          size: m1.medium

Salt virt with KVM cluster

virt:
  disk:
    three_disks:
      - system:
        size: 4096
image: ubuntu.qcow
- repository_snapshot:
  size: 8192
  image: snapshot.qcow
- cinder-volume:
  size: 2048

salt:
  minion:
    enabled: true
  master:
    host: config01.dc01.domain.com
  control:
    enabled: true
  virt_enabled: true
  size:
    small:
      cpu: 1
      ram: 1
    medium:
      cpu: 2
      ram: 4
    large:
      cpu: 4
      ram: 8
    medium_three_disks:
      cpu: 2
      ram: 4
      disk_profile: three_disks
  cluster:
    vpc20_infra:
      domain: neco.virt.domain.com
      engine: virt
      config:
        engine: salt
        host: master.domain.com
  node:
    ubuntu1:
      provider: node01.domain.com
      image: ubuntu.qcow
      size: medium
    ubuntu2:
      provider: node02.domain.com
      image: bubuntu.qcomw
      size: small
    ubuntu3:
      provider: node03.domain.com
      image: meowbuntu.qcom2
      size: medium_three_disks

salt virt with custom destination for image file

virt:
  disk:
    three_disks:
      system:
      size: 4096
      image: ubuntu.qcow
- repository_snapshot:
  size: 8192
  image: snapshot.qcow
- cinder-volume:
  size: 2048
salt:
  minion:
    enabled: true
  master:
    host: config01.dc01.domain.com
  control:
    enabled: true
  virt_enabled: true
  size:
    small:
      cpu: 1
      ram: 1
    medium:
      cpu: 2
      ram: 4
    large:
      cpu: 4
      ram: 8
  medium_three_disks:
    cpu: 2
    ram: 4
    disk_profile: three_disks
cluster:
  vpc20_infra:
    domain: neco.virt.domain.com
    engine: virt
    config:
      engine: salt
      host: master.domain.com
  node:
    ubuntu1:
      provider: node01.domain.com
      image: ubuntu.qcow
      size: medium
      img_dest: /var/lib/libvirt/ssdimages
    ubuntu2:
      provider: node02.domain.com
      image: bubuntu.qcow
      size: small
      img_dest: /var/lib/libvirt/hddimages
    ubuntu3:
      provider: node03.domain.com
      image: meowbuntu.qcow
      size: medium_three_disks

Usage

Working with salt-cloud
salt-cloud -m /path/to/map --assume-yes

Debug LIBCLOUD for salt-cloud connection

export LIBCLOUD_DEBUG=/dev/stderr; salt-cloud --list-sizes provider_name --log-level all

References

- https://github.com/saltstack-formulas/salt-formula

salt-cloud

- http://docs.saltstack.com/topics/cloud/digitalocean.html

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-salt/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.
Sphinx

Sphinx is a tool that makes it easy to create intelligent and beautiful documentation, written by Georg Brandl and licensed under the BSD license. It was originally created for the new Python documentation, and it has excellent facilities for the documentation of Python projects, but C/C++ is already supported as well, and it is planned to add special support for other languages as well.

Sample pillars

Simple documentation with local source

```yaml
sphinx:
  server:
    enabled: true
doc:
  board:
    builder: 'html'
  source:
    engine: local
    path: '/path/to/sphinx/documentation'
```

Simple documentation with Git source

```yaml
sphinx:
  server:
    enabled: true
doc:
  board:
    builder: 'html'
  source:
    engine: git
    address: 'git@repo1.domain.com/repo.git'
    revision: master
```

Simple documentation with reclass source

```yaml
sphinx:
  server:
    enabled: true
doc:
  board:
    builder: 'html'
    source:
      engine: reclass
```

Read more

- [http://sphinx-doc.org/tutorial.html](http://sphinx-doc.org/tutorial.html)
**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-sphinx/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-sphinx

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Squid Formula**

**Sample Pillars**

Squid as proxy

```
squid:
  proxy:
    enabled: true
    admin:
      user: manager
      password: passwd
    deny:
    - 192.168.2.30
    allow:
    - localnet
```

**More Information**

- https://raw.githubusercontent.com/saltstack-formulas/squid-formula

- Documentation Home
- Project Introduction
- Installation and Operations Manual
Supplemental Services

Support services as databases, proxies, application servers.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>apache</td>
<td><a href="https://github.com/salt-formulas/salt-formula-apache">https://github.com/salt-formulas/salt-formula-apache</a></td>
</tr>
<tr>
<td>bind</td>
<td><a href="https://github.com/salt-formulas/salt-formula-bind">https://github.com/salt-formulas/salt-formula-bind</a></td>
</tr>
<tr>
<td>bird</td>
<td><a href="https://github.com/salt-formulas/salt-formula-bird">https://github.com/salt-formulas/salt-formula-bird</a></td>
</tr>
<tr>
<td>cadf</td>
<td><a href="https://github.com/salt-formulas/salt-formula-cadf">https://github.com/salt-formulas/salt-formula-cadf</a></td>
</tr>
<tr>
<td>cassandra</td>
<td><a href="https://github.com/salt-formulas/salt-formula-cassandra">https://github.com/salt-formulas/salt-formula-cassandra</a></td>
</tr>
<tr>
<td>dovecot</td>
<td><a href="https://github.com/salt-formulas/salt-formula-dovecot">https://github.com/salt-formulas/salt-formula-dovecot</a></td>
</tr>
<tr>
<td>elasticsearch</td>
<td><a href="https://github.com/salt-formulas/salt-formula-elasticsearch">https://github.com/salt-formulas/salt-formula-elasticsearch</a></td>
</tr>
<tr>
<td>etcd</td>
<td><a href="https://github.com/salt-formulas/salt-formula-etcd">https://github.com/salt-formulas/salt-formula-etcd</a></td>
</tr>
<tr>
<td>galera</td>
<td><a href="https://github.com/salt-formulas/salt-formula-galera">https://github.com/salt-formulas/salt-formula-galera</a></td>
</tr>
<tr>
<td>haproxy</td>
<td><a href="https://github.com/salt-formulas/salt-formula-haproxy">https://github.com/salt-formulas/salt-formula-haproxy</a></td>
</tr>
<tr>
<td>keepalived</td>
<td><a href="https://github.com/salt-formulas/salt-formula-keepalived">https://github.com/salt-formulas/salt-formula-keepalived</a></td>
</tr>
<tr>
<td>knot</td>
<td><a href="https://github.com/salt-formulas/salt-formula-knot">https://github.com/salt-formulas/salt-formula-knot</a></td>
</tr>
<tr>
<td>letsencrypt</td>
<td><a href="https://github.com/salt-formulas/salt-formula-letsencrypt">https://github.com/salt-formulas/salt-formula-letsencrypt</a></td>
</tr>
<tr>
<td>logrotate</td>
<td><a href="https://github.com/salt-formulas/salt-formula-logrotate">https://github.com/salt-formulas/salt-formula-logrotate</a></td>
</tr>
<tr>
<td>memcached</td>
<td><a href="https://github.com/salt-formulas/salt-formula-memcached">https://github.com/salt-formulas/salt-formula-memcached</a></td>
</tr>
<tr>
<td>mosquitto</td>
<td><a href="https://github.com/salt-formulas/salt-formula-mosquitto">https://github.com/salt-formulas/salt-formula-mosquitto</a></td>
</tr>
<tr>
<td>mongodb</td>
<td><a href="https://github.com/salt-formulas/salt-formula-mongodb">https://github.com/salt-formulas/salt-formula-mongodb</a></td>
</tr>
<tr>
<td>mysql</td>
<td><a href="https://github.com/salt-formulas/salt-formula-mysql">https://github.com/salt-formulas/salt-formula-mysql</a></td>
</tr>
<tr>
<td>nginx</td>
<td><a href="https://github.com/salt-formulas/salt-formula-nginx">https://github.com/salt-formulas/salt-formula-nginx</a></td>
</tr>
<tr>
<td>openidap</td>
<td><a href="https://github.com/salt-formulas/salt-formula-openidap">https://github.com/salt-formulas/salt-formula-openidap</a></td>
</tr>
<tr>
<td>postfix</td>
<td><a href="https://github.com/salt-formulas/salt-formula-postfix">https://github.com/salt-formulas/salt-formula-postfix</a></td>
</tr>
<tr>
<td>postgresql</td>
<td><a href="https://github.com/salt-formulas/salt-formula-postgresql">https://github.com/salt-formulas/salt-formula-postgresql</a></td>
</tr>
<tr>
<td>powerdns</td>
<td><a href="https://github.com/salt-formulas/salt-formula-powerdns">https://github.com/salt-formulas/salt-formula-powerdns</a></td>
</tr>
<tr>
<td>rabbitmq</td>
<td><a href="https://github.com/salt-formulas/salt-formula-rabbitmq">https://github.com/salt-formulas/salt-formula-rabbitmq</a></td>
</tr>
<tr>
<td>redis</td>
<td><a href="https://github.com/salt-formulas/salt-formula-redis">https://github.com/salt-formulas/salt-formula-redis</a></td>
</tr>
<tr>
<td>rsync</td>
<td><a href="https://github.com/salt-formulas/salt-formula-rsync">https://github.com/salt-formulas/salt-formula-rsync</a></td>
</tr>
<tr>
<td>supervisor</td>
<td><a href="https://github.com/salt-formulas/salt-formula-supervisor">https://github.com/salt-formulas/salt-formula-supervisor</a></td>
</tr>
<tr>
<td>varnish</td>
<td><a href="https://github.com/salt-formulas/salt-formula-varnish">https://github.com/salt-formulas/salt-formula-varnish</a></td>
</tr>
<tr>
<td>zookeeper</td>
<td><a href="https://github.com/salt-formulas/salt-formula-zookeeper">https://github.com/salt-formulas/salt-formula-zookeeper</a></td>
</tr>
</tbody>
</table>

Apache Formula

Install and configure Apache webserver

Sample Pillars

Simple Apache proxy

```yaml
apache:
  server:
    enabled: true
```

(continues on next page)
bind:
  address: '0.0.0.0'
  ports: - 80
modules:
  - proxy
  - proxy_http
  - proxy_balancer

Apache plain static sites (eg. sphinx generated, from git/hg sources)

apache:
  server:
    enabled: true
  bind:
    address: '0.0.0.0'
    ports: - 80
  modules:
    - rewrite
    - status
  site:
    enabled: true
    name: 'sphinxdoc'
    type: 'static'
    host:
      name: 'doc.domain.com'
      port: 80
      source:
        engine: local
    - enabled: true
      name: 'impressjs'
      type: 'static'
      host:
        name: 'pres.domain.com'
        port: 80
        source:
          engine: git
          address: 'git@repol.domain.cz:impress/billometer.git'
          revision: 'master'

Tune settings of mpm_prefork

parameters:
  apache:
    mpm:
      prefork:
        max_clients: 250
        servers:
          min: 32
          max: 64
          max_requests: 4000

Apache kerberos authentication:

parameters
  apache:
server:
  site:
    auth:
      engine: kerberos
      name: "Kerberos Authentication"
      require:
        - "ldap-attribute memberOf='cn=somegroup,cn=groups,cn=accounts,dc=example,˓
dc=com'"

      kerberos:
        realms:
          - EXAMPLE.COM

        # Below is optional
        keytab: /etc/apache2/ipa.keytab
        service: HTTP
        method:
          negotiate: true
          k5passwd: true

      ldap:
        url: "ldaps://idm01.example.com/dc=example,dc=com?krbPrincipalName"

        # mech is optional
        mech: GSSAPI

Tune security settings (these are default):

parameters:
  apache:
    server:
      # ServerTokens
      tokens: Prod
      # ServerSignature, can be also set per-site
      signature: false
      # TraceEnable, can be also set per-site
      trace: false
      # Deny access to .git, .svn, .hg directories
      secure_scm: true
      # Required for settings bellow
      modules:
        - headers
          # Set X-Content-Type-Options
          content_type_options: nosniff
          # Set X-Frame-Options
          frame_options: sameorigin

Tuned up log configuration:

parameters:
  apache:
    server:
      site:
        foo:
          enabled: true
          type: static
          log:
            custom:
              enabled: true
(continued from previous page)

```
file: /var/log/apache2/mylittleponysitecustom.log
format: >-
  %{X-Forwarded-For}i %l %u %t "%r" %>s %b %D "%{Referer}i" "%
  →{User-Agent}i"
error:
  enabled: false
file: /var/log/apache2/foo.error.log
level: notice
```

Apache wsgi application.

```
apache:
  server:
    enabled: true
  default_mpm: event
  site:
    manila:
      enabled: false
      available: true
      type: wsgi
      name: manila
    wsgi:
      daemon_process: manila-api
      threads: 2
      user: manila
      group: manila
      display_name: '%{GROUP}'
      script_alias: '/usr/bin/manila-wsgi'
      application_group: '%{GLOBAL}'
      authorization: 'On'
    limits:
      request_body: 114688
```

Roundcube webmail, postfixadmin and mailman

```
classes:
- service.apache.server.single
parameters:
  apache:
    server:
      enabled: true
      modules:
        - cgi
        - php
    site:
      roundcube:
        enabled: true
        type: static
        name: roundcube
        root: /usr/share/roundcube
        locations:
          - uri: /admin
            path: /usr/share/postfixadmin
          - uri: /mailman
            path: /usr/lib/cgi-bin/mailman
            script: true
          - uri: /pipermail
```

(continues on next page)
More Information

- https://httpd.apache.org/docs/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-apache/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-apache

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Bind formula

BIND is open source software that enables you to publish your Domain Name System (DNS) information on the Internet, and to resolve DNS queries for your users. The name BIND stands for “Berkeley Internet Name Domain”, because the software originated in the early 1980s at the University of California at Berkeley.
Sample pillars

Server

```yaml
bind:
  server:
    enabled: true
    key:
      keyname:
        secret: xyz
        algorithm: hmac-sha512
    8.8.8.8:
      keys:
        - keyname
  control:
    local:
      enabled: true
      bind:
        address: 127.0.0.1
        port: 953
        allow:
          - 127.0.0.1
        keys:
          - xyz
  zone:
    sub.domain.com:
      ttl: 86400
      root: "hostmaster@domain.com"
      type: master
      records:
        - name: @
          type: A
          ttl: 7200
          value: 192.168.0.5
    1.168.192.in-addr.arpa:
      type: master
      notify: false
    slave.domain.com:
      type: slave
      notify: true
      masters:
        # Masters must be specified by IP address
        - 8.8.8.8
        - 8.8.4.4
    dnssec:
      enabled: true
      # Don't hide version
      version: true
      # Allow recursion, better don't on public dns servers
      recursion:
        hosts:
          - localhost
```

You can use following command to generate key:
dnssec-keygen -a HMAC-SHA512 -b 512 -n HOST -r /dev/urandom mykey

Client

```yaml
bind:
  client:
    enabled: true
    option:
      default:
        server: localhost
        port: 953
        key: keyname
    key:
      keyname:
        secret: xyz
        algorithm: hmac-sha512
    server:
      8.8.8.8:
        keys:
          - keyname
```

Read more

- [https://github.com/theforeman/puppet-dns](https://github.com/theforeman/puppet-dns)
- [https://help.ubuntu.com/community/BIND9ServerHowto](https://help.ubuntu.com/community/BIND9ServerHowto)
- [https://www.isc.org/downloads/bind/](https://www.isc.org/downloads/bind/)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:


For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

[https://github.com/salt-formulas/salt-formula-bind](https://github.com/salt-formulas/salt-formula-bind)

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
BIRD Formula

The BIRD project aims to develop a fully functional dynamic IP routing daemon primarily targeted on (but not limited to) Linux, FreeBSD and other UNIX-like systems and distributed under the GNU General Public License.

Sample Pillars

```yaml
bird:
  server:
    enabled: True
  logging:
    engine: syslog
  protocol:
    my_ospf:
      type: ospf
      tick: 2
      rfc1583compat: True
      ecmp: True
      area:
        0.0.0.0:
          interface:
            p3p1:
              type: ptp
              paramX: xxx
            p3p2:
              type: ptp
              paramX: xxx
            tap0: {}
            vhost0:
              hello: 9
              type: broadcast
              paramX: xxx
```

More Information


CADF Formula

The Cloud Auditing Data Federation (CADF) standard defines a full event model anyone can use to fill in the essential data needed to certify, self-manage and self-audit application security in cloud environments.

Sample Pillars

Single cadf service

```yaml
cadf:
  distpather:
    enabled: true
```

(continues on next page)
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-cadf/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-cadf

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

cassandra

Service cassandra description

Sample pillars

Single cassandra service

```yaml
[cassandra]
  server:
    enabled: true
    version: icehouse
```

Backup client with ssh/rsync remote host

```yaml
[cassandra]
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
    target:
      host: cfg01
```

(continues on next page)
 Backup client with local backup only

```
cassandra:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
```

.. note:: full_backups_to_keep param states how many backup will be stored locally on `-->cassandra client`.
          
          More options to relocate local backups can be done using salt-formula-`-->backuninja`.

 Backup server rsync

```
cassandra:
  backup:
    server:
      enabled: true
      hours_before_full: 24
      full_backups_to_keep: 5
      key:
        cassandra_pub_key:
          enabled: true
          key: ssh_rsa
```

 Client restore from local backup:

```
cassandra:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
      target:
        host: cfg01
        restore_latest: 1
        restore_from: local
```

.. note:: restore_latest param with a value of 1 means to restore db from the last`-->full backup`. 2 would mean to restore second latest full backup.

 Client restore from remote backup:

```
cassandra:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
      target:
        host: cfg01
```
**Read more**

- links

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-cassandra/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-cassandra

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

dovecot

Install and configure dovecot.

**Available states**

- **dovecot.server**

**dovecot.server**

Setup dovecot server
Available metadata

- `metadata.dovecot.server`

**metadata.dovecot.server**

Setup dovecot server

**Requirements**

- linux
- mysql (for mysql backend)

**Optional**

- glusterfs (to serve as mail storage backend)
- postfix
- roundcube

**Configuration parameters**

For complete list of parameters, please check `metadata/service/server.yml`

**Example reclass**

**Server**

```yaml
classes:
  - service.dovecot.server
parameters:
  __param:
    dovecot_origin: mail.eru
    mysql_mailserver_password: Peixeilaephahoosa2daihoh4yiaThe
  dovecot:
    server:
      origin: ${__param:dovecot_origin}
  mysql:
    server:
      database:
        mailserver:
          encoding: UTF8
          locale: cs_CZ
          users:
            - name: mailserver
              password: ${__param:mysql_mailserver_password}
```

(continues on next page)
host: 127.0.0.1
rights: all privileges

apache:
server:
site:
    dovecotadmin:
        enabled: true
        type: static
        name: dovecotadmin
        root: /usr/share/dovecotadmin
        host:
            name: ${_param:dovecot_origin}
        aliases:
            - ${linux:system:name}.${linux:system:domain}
            - ${linux:system:name}

LDAP and GSSAPI

parameters:
    dovecot:
        server:
            gssapi:
                host: imap01.example.com
                keytab: /etc/dovecot/krb5.keytab
                realms:
                    - example.com
                default_realm: example.com
            userdb:
                driver: ldap
            passdb:
                driver: ldap
            ldap:
                servers:
                    - ldaps://idm01.example.com
                    - ldaps://idm02.example.com
                basedn: dc=example,dc=com
                bind:
                    dn: uid=dovecot,cn=users,cn=accounts,dc=example,dc=com
                    password: password
                auth_bind:
                    enabled: true
                    userdn: "mail=%u,cn=users,cn=accounts,dc=example,dc=com"
                    user_filter: "(&(objectClass=posixAccount)(mail=%u))"

Director

Dovecot Director is used to ensure connection affinity to specific backends. This seems to be a must-have for shared storage such as NFS, GlusterFS, etc. otherwise you are going to meet split-brains, corrupted files and other issues.

Unfortunately director for LMTP can’t be used when director and backend servers are the same.

See http://wiki2.dovecot.org/Director for more informations.
dovecot:
  server:
    admin: postmaster@${_param:postfix_origin}
    # GlusterFS storage is used
    nfs: true
  service:
    director:
      enabled: true
      port: 9090
    backends:
      - ${_param:cluster_node01_address}
      - ${_param:cluster_node02_address}
    directors:
      - ${_param:cluster_node01_address}
      - ${_param:cluster_node02_address}
  lmtp:
    inet_enabled: true
    port: 24
postfix:
  server:
    dovecot_lmtp:
      enabled: true
      type: inet
      address: "localhost:24"

Example pillar

Server

dovecot:
  server:
    origin: ${_param:dovecot_origin}
    admin:
      enabled: false

Read more

- http://wiki2.dovecot.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-dovecot/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

2.1. Project Introduction
Elasticsearch

Elasticsearch provides a distributed, multitenant-capable full-text search engine with a HTTP web interface and schema-free JSON documents.

Sample pillars

Single-node elasticsearch with clustering disabled:

```
elasticsearch:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 9200
    cluster:
      multicast: false
    index:
      shards: 1
      replicas: 0
```

Setup shared repository for snapshots:

```
elasticsearch:
  server:
    snapshot:
      reponame:
        path: /var/lib/glusterfs/repo
        compress: true
```

Cluster with manually defined members:

```
elasticsearch:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 9200
    cluster:
      multicast: false
    members:
      - host: elastic01
        port: 9300
      - host: elastic02
```

(continues on next page)
port: 9300
  - host: elastic03
    port: 9300

index:
  shards: 5
  replicas: 1

Common definition for curator:

```
elasticsearch:
  server:
    curator:
      timeout: 900
      logfile: /var/log/elasticsearch/curator.log
      logformat: json
      master_only: true
      actions:
        - action: delete_indices
          description: >-
            Delete indices older than 45 days (based on index name).
            Ignore the error if the filter does not result in an actionable
            list of indices (ignore_empty_list) and exit cleanly.
          options:
            ignore_empty_list: True
            continue_if_exception: False
            disable_action: False
          filters:
            - filtertype: pattern
              kind: regex
              value: '.*\-\d\d\d\d\..*\-\d\d\d\d\..*\-\d\d\d\d\..*\-\d\d\d\d\..*\-\d\d\d\d$'
            - filtertype: age
              source: name
              direction: older
              timestring: '%Y.%m.%d'
              unit: days
              unit_count: 90
        - action: replicas
          description: >-
            Reduce the replica count to 0 for indices older than 30 days
            (based on index creation_date)
          options:
            count: 0
            wait_for_completion: False
            continue_if_exception: False
            disable_action: False
          filters:
            - filtertype: pattern
              kind: regex
              value: '.*\-\d\d\d\d\..*\-\d\d\d\d\..*\-\d\d\d\d\..*\-\d\d\d\d$'
            - filtertype: age
              source: creation_date
              direction: older
              unit: days
              unit_count: 30
        - action: forcemerge
          description: >-
            forceMerge indices older than 2 days (based on index
(continues on next page)
(continued from previous page)

creation_date) to 2 segments per shard. Delay 120 seconds between each forceMerge operation to allow the cluster to quiesce. This action will ignore indices already forceMerged to the same or fewer number of segments per shard, so the 'forcemerged' filter is unneeded.

options:
  max_num_segments: 2
delay: 120
  continue_if_exception: False
disable_action: True

filters:
- filtertype: pattern
  kind: regex
  value: '.*\d\d\d\d\d\d\d\d\d\d\d$'
- filtertype: age
  source: creation_date
  direction: older
  unit: days
  unit_count: 2

Client setup

Client with host and port:

```yaml
elasticsearch:
  client:
    enabled: true
  server:
    host: elasticsearch.host
    port: 9200
```

Client where you download an index template that is stored in the directory `files/`:

```yaml
elasticsearch:
  client:
    enabled: true
  server:
    host: elasticsearch.host
    port: 9200
  index:
    my_index:
      enabled: true
      template: elasticsearch/files/my_index_template.json
```

Client where you download an index template from the metadata definition and force index creation:

```yaml
elasticsearch:
  client:
    enabled: true
  server:
    host: elasticsearch.host
    port: 9200
  index:
    my_index:
```

(continues on next page)
enabled: true
force_operation: true
definition:
  template: notifications
settings:
  number_of_shards: 5
  number_of_replicas: 1
mappings:
  notification:
    properties:
      applicationId:
        type: long
      content:
        type: text
      fields:
        keyword:
          type: keyword
          ignore_above: 256

Read more

- https://www.elastic.co/
- https://gist.github.com/wingdspur/2026107

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-elasticsearch/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-elasticsearch

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

2.1. Project Introduction
etcd Formula

Service etcd description

Possible `source.engine`:

- `pkg` - install etcd package (default)
- `docker_hybrid` - copy binaries from docker image (specified in `server.image`)

Sample pillars

Certificates

Use certificate authentication (for peers and clients). Certificates must be prepared in advance.

```
etcd:
  server:
    enabled: true
    ssl:
      enabled: true
      bind:
        host: 10.0.175.101
        token: $(uuidgen)
        members:
          - host: 10.0.175.101
            name: etcd01
            port: 4001
```

Single etcd service

```
etcd:
  server:
    enabled: true
    bind:
      host: 10.0.175.101
      token: $(uuidgen)
      members:
        - host: 10.0.175.101
          name: etcd01
          port: 4001
```

Cluster etcd service

```
etcd:
  server:
    enabled: true
    bind:
      host: 10.0.175.101
      token: $(uuidgen)
      members:
        - host: 10.0.175.101
```

(continues on next page)
name: etcd01
  port: 4001
- host: 10.0.175.102
  name: etcd02
  port: 4001
- host: 10.0.175.103
  name: etcd03
  port: 4001

etcd proxy

etcd:
  server:
    enabled: true
    bind:
      host: 10.0.175.101
    proxy: true
  members:
  - host: 10.0.175.101
    name: etcd01
  - host: 10.0.175.102
    name: etcd02
  - host: 10.0.175.103
    name: etcd03

Run etcd on k8s

etcd:
  server:
    engine: kubernetes
    image: etcd:latest

Copy etcd binary from container

etcd:
  server:
    image: quay.io/coreos/etcd:latest

Read more

- https://github.com/coreos/etcd

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/
In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-etcd/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-etcd

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Galera

Galera Cluster for MySQL is a true Multimaster Cluster based on synchronous replication. Galera Cluster is an easy-to-use, high-availability solution, which provides high system uptime, no data loss and scalability for future growth.

Sample pillars

Galera cluster master node

galera:
  version:
    mysql: 5.6
    galera: 3
  master:
    enabled: true
    name: openstack
    bind:
      address: 192.168.0.1
      port: 3306
    members:
      - host: 192.168.0.1
        port: 4567
      - host: 192.168.0.2
        port: 4567
    admin:
      user: root
      password: pass
  database:
    name:
      encoding: 'utf8'
    users:
      - name: 'username'
        password: 'password'
        host: 'localhost'
        rights: 'all privileges'

Galera cluster slave node
galera:
  slave:
    enabled: true
    name: openstack
    bind:
      address: 192.168.0.2
      port: 3306
    members:
      - host: 192.168.0.1
        port: 4567
      - host: 192.168.0.2
        port: 4567
    admin:
      user: root
      password: pass

Enable TLS support:

galera:
  slave or master:
    ssl:
      enabled: True

      # path
      cert_file: /etc/mysql/ssl/cert.pem
      key_file: /etc/mysql/ssl/key.pem
      ca_file: /etc/mysql/ssl/ca.pem

      # content (not required if files already exists)
      key: << body of key >>
      cert: << body of cert >>
      cacert_chain: << body of ca certs chain >>

Additional mysql users:

mysql:
  server:
    users:
      - name: clustercheck
        password: clustercheck
        database: '*.*'
        grants: PROCESS
      - name: inspector
        host: 127.0.0.1
        password: password
        databases:
          mydb:
            - database: mydb
            - table: mytable
            - grant_option: True
            - grants:
              - all privileges

Additional mysql SSL grants:

mysql:
  server:

(continues on next page)
users:
  - name: clustercheck
    password: clustercheck
    database: '*.*'
    grants: PROCESS
    ssl_option:
      - SSL: True
      - X509: True
      - SUBJECT: <subject>
      - ISSUER: <issuer>
      - CIPHER: <cipher>

Additional check params:

galera:
  clustercheck:
    - enabled: True
    - user: clustercheck
    - password: clustercheck
    - available_when_donor: 0
    - available_when_readonly: 1
    - port 9200

Configurable soft parameters

- **galera_innodb_buffer_pool_size** - the default value is 3138M
- **galera_max_connections** - the default value is 20000

Usage: .. code-block:: yaml

```yaml
  _param:
    galera_innodb_buffer_pool_size: 1024M
    galera_max_connections: 200
```

**Usage**

MySQL Galera check scripts

```bash
mysql> SHOW STATUS LIKE 'wsrep*';
```

Galera monitoring command, performed from extra server

```bash
garbd -a gcomm://ipaddrofone:4567 -g my_wsrep_cluster -l /tmp/1.out -d
```

1. salt-call state.sls mysql
2. Comment everything starting wsrep* (wsrep_provider, wsrep_cluster, wsrep_sst)
3. service mysql start
4. run on each node mysql_secure_install and filling root password.
Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MySQL root user without the proper authorisation.

Set root password? [Y/n] y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
... Success!

By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] y
... Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
... skipping.

By default, MySQL comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] y
... Success!

Cleaning up...

5. service mysql stop
6. uncomment all wsrep* lines except first server, where leave only in my.cnf wsrep_cluster_address='gcomm://';
7. start first node
8. Start third node which is connected to first one
9. Start second node which is connected to third one
10. After starting cluster, it must be change cluster address at first starting node without restart database and change config my.cnf.

2.1. Project Introduction
mysql> SET GLOBAL wsrep_cluster_address='gcomm://10.0.0.2';

Read more

- https://github.com/CaptTofu/ansible-galera
- **Best one:** http://www.sebastien-han.fr/blog/2012/04/01/mysql-multi-master-replication-with-galera/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-galera/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-galera

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**HAproxy**

The Reliable, High Performance TCP/HTTP Load Balancer.

**Sample pillars**

Simple admin listener

```python
haproxy:
  proxy:
    enabled: True
    listen:
      admin_page:
        type: admin
```
binder:
- address: 0.0.0.0
  port: 8801
user: dsfdsfd
password: dsfd

Simple stats listener

haproxy:
  proxy:
    enabled: True
    listen:
      admin_page:
        type: stats
        binds:
          - address: 0.0.0.0
            port: 8801

Sample pillar with admin

haproxy:
  proxy:
    enabled: True
    mode: http/tcp
    logging: syslog
    maxconn: 1024
    timeout:
      connect: 5000
      client: 50000
      server: 50000
    listen:
      https-in:
        binds:
          - address: 0.0.0.0
            port: 443
        servers:
          - name: server1
            host: 10.0.0.1
            port: 8443
          - name: server2
            host: 10.0.0.2
            port: 8443
            params: 'maxconn 256'

Sample pillar with custom logging

haproxy:
  proxy:
    enabled: True
    mode: http/tcp
    logging: syslog
    maxconn: 1024
    timeout:
      connect: 5000
      client: 50000
      server: 50000
    listen:
https-in:
  binds:
    - address: 0.0.0.0
      port: 443
  servers:
    - name: server1
      host: 10.0.0.1
      port: 8443
    - name: server2
      host: 10.0.0.2
      port: 8443
      params: 'maxconn 256'

haproxy:
  proxy:
    enabled: true
    mode: tcp
    logging: syslog
    max_connections: 1024
  listen:
    mysql:
      type: mysql
      binds:
        - address: 10.0.88.70
          port: 3306
      servers:
        - name: node1
          host: 10.0.88.13
          port: 3306
          params: check inter 15s fastinter 2s downinter 1s rise 5 fall 3
        - name: node2
          host: 10.0.88.14
          port: 3306
          params: check inter 15s fastinter 2s downinter 1s rise 5 fall 3 backup
        - name: node3
          host: 10.0.88.15
          port: 3306
          params: check inter 15s fastinter 2s downinter 1s rise 5 fall 3 backup
    rabbitmq:
      type: rabbitmq
      binds:
        - address: 10.0.88.70
          port: 5672
      servers:
        - name: node1
          host: 10.0.88.13
          port: 5673
          params: check inter 5000 rise 2 fall 3
        - name: node2
          host: 10.0.88.14
          port: 5673
          params: check inter 5000 rise 2 fall 3 backup
        - name: node3
          host: 10.0.88.15
          port: 5673
          params: check inter 5000 rise 2 fall 3 backup
keystone-1:
   type: general-service
   binds:
      - address: 10.0.106.170
        port: 5000
   servers:
      - name: node1
        host: 10.0.88.13
        port: 5000
        params: check
haproxy:
   proxy:
      enabled: true
      mode: tcp
      logging: syslog
      max_connections: 1024
      listen:
         mysql:
            type: mysql
            binds:
               - address: 10.0.88.70
                 port: 3306
            servers:
               - name: node1
                 host: 10.0.88.13
                 port: 3306
                 params: check inter 15s fastinter 2s downinter 1s rise 5 fall 3
               - name: node2
                 host: 10.0.88.14
                 port: 3306
                 params: check inter 15s fastinter 2s downinter 1s rise 5 fall 3 backup
               - name: node3
                 host: 10.0.88.15
                 port: 3306
                 params: check inter 15s fastinter 2s downinter 1s rise 5 fall 3 backup
rabbitmq:
   type: rabbitmq
   binds:
      - address: 10.0.88.70
        port: 5672
   servers:
      - name: node1
        host: 10.0.88.13
        port: 5673
        params: check inter 5000 rise 2 fall 3
      - name: node2
        host: 10.0.88.14
        port: 5673
        params: check inter 5000 rise 2 fall 3 backup
      - name: node3
        host: 10.0.88.15
        port: 5673
        params: check inter 5000 rise 2 fall 3 backup
keystone-1:
   type: general-service

2.1. Project Introduction
Custom more complex listener (for Artifactory and subdomains for docker registries)

```yaml
haproxy:
  proxy:
    listen:
      artifactory:
        mode: http
        options:
          - forwardfor
          - forwardfor header X-Real-IP
          - httpchk
          - httpclose
          - httplog
        sticks:
          - stick on src
          - stick-table type ip size 200k expire 2m
        acl:
          is_docker: "path_reg ^/v[12]/.*"
        http_request:
          - action: "set-path /artifactory/api/docker/%[req.hdr(host),lower,field(1,'.'→')]%[path]"
            condition: "if is_docker"
        balance: source
        binds:
          - address: ${_param:cluster_vip_address}
            port: 8082
            ssl:
              enabled: true
              # This PEM file needs to contain key, cert, CA and possibly intermediate certificates
              pem_file: /etc/haproxy/ssl/server.pem
        servers:
          - name: ${_param:cluster_node01_name}
            host: ${_param:cluster_node01_address}
            port: 8082
            params: check
          - name: ${_param:cluster_node02_name}
            host: ${_param:cluster_node02_address}
            port: 8082
            params: backup check
```

It’s also possible to use multiple certificates for one listener (eg. when it’s bind on multiple interfaces):

```yaml
haproxy:
  proxy:
    listen:
      dummy_site:
        mode: http
```

(continues on next page)
binds:
- address: 127.0.0.1
  port: 8080
  ssl:
    enabled: true
    key: |
      my super secret key follows
    cert: |
      certificate
    chain: |
      CA chain (if any)
- address: 127.0.1.1
  port: 8081
  ssl:
    enabled: true
    key: |
      my super secret key follows
    cert: |
      certificate
    chain: |
      CA chain (if any)

Definition above will result in creation of `/etc/haproxy/ssl/dummy_site` directory with files 1-all.pem and 2-all.pem (per binds).

Custom listener with tcp-check options specified (for Redis cluster with Sentinel)

```
haproxy:
  proxy:
    listen:
      redis_cluster:
        service_name: redis
        health-check:
          tcp:
            enabled: True
            options:
              - send PING\r\n              - expect string +PONG
              - send info\ replication\r\n              - expect string role:master
              - send QUIT\r\n              - expect string +OK
        binds:
          - address: ${_param:cluster_address}
            port: 6379
        servers:
          - name: ${_param:cluster_node01_name}
            host: ${_param:cluster_node01_address}
            port: 6379
            params: check inter 1s
          - name: ${_param:cluster_node02_name}
            host: ${_param:cluster_node02_address}
            port: 6379
            params: check inter 1s
          - name: ${_param:cluster_node03_name}
            host: ${_param:cluster_node03_address}
            port: 6379
```

(continues on next page)
Frontend for routing between exists listeners via URL with SSL and redirects. You can use one backend for several URLs.

```
haproxy:
  proxy:
    listen:
      service_proxy:
        mode: http
        balance: source
        format: end
        binds:
          - address: ${_param:haproxy_bind_address}
            port: 80
            ssl: ${_param:haproxy_frontend_ssl}
            ssl_port: 443
        redirects:
          - code: 301
            location: domain.com/images
        conditions:
          - type: hdr_dom(host)
            condition: images.domain.com
    acls:
      - name: gerrit
        conditions:
          - type: hdr_dom(host)
            condition: gerrit.domain.com
      - name: jenkins
        conditions:
          - type: hdr_dom(host)
            condition: jenkins.domain.com
      - name: docker
        backend: artifactroy
        conditions:
          - type: hdr_dom(host)
            condition: docker.domain.com
```

Enable customisable `forwardfor` option in `defaults` section.

```
ahaproxy:
  proxy:
    enabled: true
    mode: tcp
    logging: syslog
    max_connections: 1024
    forwardfor:
      enabled: true
      except:
        header:
          if-none: false
```

```
haproxy:
  proxy:
    enabled: true
    mode: tcp
```

(continues on next page)
logging: syslog
max_connections: 1024
forwardfor:
  enabled: true
  except: 127.0.0.1
  header: X-Real-IP
  if-none: false

Sample pillar with multiprocess multicore configuration

haproxy:
  proxy:
    enabled: True
    nbproc: 4
    cpu_map:
      1: 0
      2: 1
      3: 2
      4: 3
    stats_bind_process: "1 2"
  mode: http/tcp
  logging: syslog
  maxconn: 1024
  timeout:
    connect: 5000
    client: 50000
    server: 50000
  listen:
    https-in:
      bind_process: "1 2 3 4"
      binds:
        - address: 0.0.0.0
          port: 443
      servers:
        - name: server1
          host: 10.0.0.1
          port: 8443
        - name: server2
          host: 10.0.0.2
          port: 8443
      params: 'maxconn 256'

Read more

- https://gist.github.com/tomeduarte/6340205 - example on how to use peer from within a config file (using jinja)
- http://youtu.be/jJJ8cfDjcTc?t=8m58s - from 9:00 on, a good overview of peer vs mine
- https://github.com/russki/cluster-agents

2.1. Project Introduction
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-haproxy/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-haproxy

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Keepalived

Keepalived is a routing software written in C. The main goal of this project is to provide simple and robust facilities for loadbalancing and high-availability to Linux system and Linux based infrastructures. Loadbalancing framework relies on well-known and widely used Linux Virtual Server (IPVS) kernel module providing Layer4 loadbalancing. Keepalived implements a set of checkers to dynamically and adaptively maintain and manage loadbalanced server pool according their health. On the other hand high-availability is achieved by VRRP protocol. VRRP is a fundamental brick for router failover. In addition, Keepalived implements a set of hooks to the VRRP finite state machine providing low-level and high-speed protocol interactions. Keepalived frameworks can be used independently or all together to provide resilient infrastructures.

Sample pillar

Simple virtual IP on an interface

```yaml
keepalived:
  cluster:
    enabled: True
    instance:
      VIP1:
        nopreempt: True
        priority: 100 (highest priority must be on primary server, different for ...
        virtual_router_id: 51
        auth_type: AH
        password: pass
        address: 192.168.10.1
        interface: eth0
      VIP2:
        nopreempt: True

(continues on next page)```
priority: 150 (highest priority must be on primary server, different for cluster members)
  virtual_router_id: 52
  auth_type: PASS
  password: pass
  address: 10.0.0.5
  interface: eth1

### Multiple virtual IPs on single interface

```yaml
keepalived:
  cluster:
    enabled: True
    instance:
      VIP1:
        nopreempt: True
        priority: 100 (highest priority must be on primary server, different for cluster members)
        virtual_router_id: 51
        password: pass
        addresses:
          - 192.168.10.1
          - 192.168.10.2
        interface: eth0

Use unicast

```yaml
keepalived:
  cluster:
    enabled: True
    instance:
      VIP1:
        nopreempt: True
        priority: 100 (highest priority must be on primary server, different for cluster members)
        virtual_router_id: 51
        password: pass
        address: 192.168.10.1
        interface: eth0
        unicast_src_ip: 172.16.10.1
        unicast_peer:
          172.16.10.2
          172.16.10.3

Disable nopreempt mode to have Master. Highest priority is taken in all cases.

```yaml
keepalived:
  cluster:
    enabled: True
    instance:
      VIP1:
        nopreempt: False
        priority: 100 (highest priority must be on primary server, different for cluster members)
        virtual_router_id: 51
        password: pass
        addresses:
```

(continues on next page)
- 192.168.10.1
- 192.168.10.2
interface: eth0

Notify action in keepalived.

```
keepalived:
    cluster:
        enabled: True
        instance:
            VIP1:
                nopreempt: True
                notify_action:
                    master:
                        - /usr/bin/docker start jenkins
                        - /usr/bin/docker start gerrit
                    backup:
                        - /usr/bin/docker stop jenkins
                        - /usr/bin/docker stop gerrit
                    fault:
                        - /usr/bin/docker stop jenkins
                        - /usr/bin/docker stop gerrit
                priority: 100 # highest priority must be on primary server, different for
                virtual_router_id: 51
                password: pass
                addresses:
                    - 192.168.10.1
                    - 192.168.10.2
                interface: eth0
```

Track/vrrp scripts for keepalived instance:

```
keepalived:
    cluster:
        enabled: True
        instance:
            VIP2:
                priority: 100
                virtual_router_id: 10
                password: pass
                addresses:
                    - 192.168.11.1
                    - 192.168.11.2
                interface: eth0
                track_script: check_haproxy
            VIP3:
                priority: 100
                virtual_router_id: 11
                password: pass
                addresses:
                    - 192.168.10.1
                    - 192.168.10.2
                interface: eth0
                track_script:
                    check_random_exit:
                        interval: 10
```

(continues on next page)
check_port:
  weight: 50
vrrp_scripts:
  check_haproxy:
    name: check_pidof
    args:
      - haproxy
  check_mysql_port:
    name: check_port
    args:
      - 3306
      - TCP
      - 4
check_ssh:
  name: check_port
  args: "22"
check_mysql_cluster:
  args:
    # github: olafz/percona-clustercheck
    # <user> <pass> <available_when_donor=0|1> <log_file> <available_when_readonly=0|1> <defaults_extra_file>
    - clustercheck
    - clustercheck
    - available_when_donor=0
    - available_when_readonly=0
check_random_exit:
  interval: 10
  content: |
    #!/bin/bash
    exit $(($RANDOM%2))
  weight: 50

Read more

- https://raymii.org/s/tutorials/Keepalived-Simple-IP-failover-on-Ubuntu.html

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-keepalived/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

2.1. Project Introduction
Knot

Knot DNS is a high-performance authoritative-only DNS server which supports all key features of the modern domain name system.

Sample pillars

Simple server

```yaml
knot:
  server:
    enabled: true
```

Server dns templates

```yaml
knot:
  server:
    enabled: true
    template:
      default:
        storage: /var/lib/knot/master
      signed:
        storage: /var/lib/knot/signed
      slave:
        storage: /var/lib/knot/slave
```

Server dns zones

```yaml
knot:
  server:
    enabled: true
  zone:
    example1.com: {}
    example2.com:
      semantic-checks: False
      template: default
```

Read more

- https://www.knot-dns.cz/

Salt Logrotate Formula

Logrotate is designed to ease administration of systems that generate large numbers of log files. It allows automatic rotation, compression, removal, and mailing of log files. Each log file may be handled daily, weekly, monthly, or when it grows too large.
Example pillar

Configuration for syslog from Ubuntu 14.04 (trusty):

```
logrotate:
  server:
    enabled: true
  job:
    rsyslog:
      files:
        - /var/log/mail.info
        - /var/log/mail.warn
        - /var/log/mail.err
        - /var/log/mail.log
        - /var/log/daemon.log
        - /var/log/kern.log
        - /var/log/auth.log
        - /var/log/user.log
        - /var/log/lpr.log
        - /var/log/cron.log
        - /var/log/debug
        - /var/log/messages
      options:
        - rotate: 4
        - weekly
        - missingok
        -notifempty
        - compress
        - delaycompress
        - sharedscripts
        - postrotate: "reload rsyslog >/dev/null 2>&1 || true"
    files:
      - /var/log/syslog
      options:
        - rotate: 7
        - daily
        - missingok
        - notifempty
        - delaycompress
        - compress
        - postrotate: "reload rsyslog >/dev/null 2>&1 || true"
```

Change parameters in main logrotate.conf file:

```
logrotate:
  server:
    enabled: true
  global_conf:
    compress: true
    rotate: daily
    keep_rotate: 6
    dateext: true
```

Cross-formula relationship

It’s possible to use support meta to define logrotate rules from within other formula.
Example meta/logrotate.yml for horizon formula:

```yaml
job:
  horizon:
    files:
    - /var/log/horizon/*.log
    options:
      - compress
      - delaycompress
      - missingok
      - notifempty
      - rotate: 10
      - daily
      - minsize: 20M
      - maxsize: 500M
      - postrotate: "if /etc/init.d/apache2 status > /dev/null; then /etc/init.d/apache2 reload > /dev/null; fi"
```

**Reference**


**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:


For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

[https://github.com/salt-formulas/salt-formula-logrotate](https://github.com/salt-formulas/salt-formula-logrotate)

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Memcached Formula**

Memcached is an in-memory key-value store for small chunks of arbitrary data (strings, objects) from results of database calls, API calls, or page rendering.
Sample Metadata

memcached:
  server:
    enabled: true
    cache_size: 64
    bind:
      address: 0.0.0.0
      port: 11211
      protocol: tcp

References

- http://memcached.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-memcached/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-memcached

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

MongoDB

MongoDB (from “humongous”) is an open-source document database, and the leading NoSQL database. Written in C++.

Available states

- mongodb.server
mongodb.server

Setup MongoDB server

Available metadata

- metadata.mongodb.server.single
- metadata.mongodb.server.cluster

metadata.mongodb.server.single

Single-node MongoDB setup

metadata.mongodb.server.cluster

Clustered MongoDB setup

Configuration parameters

Example reclass

Setup MongoDB with database for ceilometer.

```plaintext
classes:
  - service.mongodb.server.cluster
parameters:
  _param:
    mongodb_server_replica_set: ceilometer
    mongodb_ceilometer_password: clouldab
    mongodb_admin_password: clouldab
    mongodb_shared_key: xxx
  mongodb:
    server:
      database:
        ceilometer:
          enabled: true
          password: ${_param:mongodb_ceilometer_password}
          users:
            - name: ceilometer
              password: ${_param:mongodb_ceilometer_password}
```

Sample pillars

Simple single server
mongodb:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 27017
    admin:
      username: admin
      password: magicunicorn
    database:
      dbname:
        enabled: true
        encoding: 'utf8'
      users:
      - name: 'username'
        password: 'password'

Cluster of 3 nodes

mongodb:
  server:
    enabled: true
    logging:
      verbose: false
      logLevel: 1
      oplogLevel: 0
    admin:
      user: admin
      password: magicunicorn
    master: mongo01
    members:
    - host: 192.168.1.11
      priority: 2
    - host: 192.168.1.12
    - host: 192.168.1.13
    replica_set: default
    shared_key: magicunicorn

It’s possible that first Salt run on master node won’t pass correctly before all slave nodes are up and ready. Simply run salt again on master node to setup cluster, databases and users.

To check cluster status, execute following:

```bash
mongo 127.0.0.1:27017/admin -u admin -p magicunicorn --eval "rs.status()"
```

Read more

- [http://docs.mongodb.org/manual/](http://docs.mongodb.org/manual/)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

2.1. Project Introduction
Mosquitto formula

Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol versions 3.1 and 3.1.1. MQTT provides a lightweight method of carrying out messaging using a publish/subscribe model.

Sample metadata

Single mosquitto service

mosquitto:
  server:
    enabled: true

References

- http://mosquito.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-mongodb/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users
Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-mosquitto

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**MySQL Formula**

MySQL is the world’s second most widely used open-source relational database management system (RDBMS).

**Sample Metadata**

**Standalone setups**

Standalone MySQL server

```yaml
mysql:
  server:
    enabled: true
    version: '5.5'
  admin:
    user: root
    password: pass
  bind:
    address: '127.0.0.1'
    port: 3306
  database:
    name:
      encoding: 'utf8'
      users:
        - name: 'username'
          password: 'password'
          host: 'localhost'
          rights: 'all privileges'
```

MySQL replication master with SSL

```yaml
mysql:
  server:
    enabled: true
    version: 5.5
  replication:
    role: master
  ssl:
    enabled: true
    authority: Org_CA
    certificate: name_of_service
  admin:
    user: root
    password: pass
  bind:
```

(continues on next page)
MySQL replication slave with SSL

```bash
def mysql:
    server:
        enabled: true
        version: '5.5'
        replication:
            role: slave
            master: master.salt.id
        ssl:
            enabled: true
            authority: Org_CA
            certificate: name_of_service
            client_certificate: name_of_client_cert
        admin:
            user: root
            password: pass
        bind:
            address: '127.0.0.1'
            port: 3306
```

Tuned up MySQL server

```bash
def mysql:
    server:
        enabled: true
        version: '5.5'
        admin:
            user: root
            password: pass
        bind:
            address: '127.0.0.1'
            port: 3306
        key_buffer: 250M
        max_allowed_packet: 32M
        max_connections: 1000
        thread_stack: 512K
        thread_cache_size: 64
        query_cache_limit: 16M
        query_cache_size: 96M
        force_encoding: utf8
        sql_mode: "ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION"
    database:
        name: 'utf8'
        users:
            - name: 'username'
              password: 'password'
              host: 'localhost'
              rights: 'all privileges'
MySQL Galera cluster

MySQL Galera cluster is configured for ring connection between 3 nodes. Each node should have just one member.

Galera initial server (master)

```yaml
mysql:
  cluster:
    enabled: true
    name: openstack
    role: master
    bind:
      address: 192.168.0.1
    members:
      - host: 192.168.0.1
        port: 4567
    user:
      name: wsrep_sst
      password: password
  server:
    enabled: true
    version: 5.5
    admin:
      user: root
      password: pass
    bind:
      address: 192.168.0.1
    database:
      name:
        encoding: 'utf8'
        users:
          - name: 'username'
            password: 'password'
            host: 'localhost'
            rights: 'all privileges'
```

MySQL client

Database with initial data (Restore DB)

```yaml
mysql:
  client:
    server:
      database:
        admin:
          host: localhost
          port: 3306
          user: ${_param:mysql_admin_user}
          password: ${_param:mysql_admin_password}
          encoding: utf8
        database:
          neutron_upgrade:
            encoding: utf8
            users:
              - name: neutron
                password: ${_param:mysql_neutron_password}
```

(continues on next page)
host: '%'
    rights: all
- name: neutron
  password: ${_param:mysql_neutron_password}
  host: ${_param:single_address}
  rights: all
initial_data:
  engine: backupninja
  source: ${_param:backupninja_backup_host}
  host: ${linux:network:fqdn}
  database: neutron

Note: This client role needs to be put directly on dbs node. The provided setup restores db named neutron_upgrade with data from db called neutron.

Database management on remote MySQL server

```yaml
mysql:
  client:
    enabled: true
  server:
    server01:
      admin:
        host: database.host
        port: 3306
        user: root
        password: password
        encoding: utf8
      database:
        database01:
          encoding: utf8
          users:
          - name: username
            password: 'password'
            host: 'localhost'
            rights: 'all privileges'
```

User management on remote MySQL server

```
mysql:
  client:
    enabled: true
  server:
    server01:
      admin:
        host: database.host
        port: 3306
        user: root
        password: password
        encoding: utf8
      users:
      - name: user01
        host: "*"
        password: 'sdgdsgdsgd'
        - name: user02
(continues on next page)```
Sample Usage

MySQL Galera check scripts

```bash
mysql> SHOW STATUS LIKE 'wsrep%';
mysql> SHOW STATUS LIKE 'wsrep_cluster_size' ;
```

Galera monitoring command, performed from extra server

```bash
garbd -a gcomm://ipaddrofone:4567 -g my_wsrep_cluster -l /tmp/1.out -d
```

1. salt-call state.sls mysql
2. Comment everything starting wsrep* (wsrep_provider, wsrep_cluster, wsrep_sst)
3. service mysql start
4. run on each node mysql_secure_install and filling root password.

```
Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MySQL root user without the proper authorisation.

Set root password? [Y/n] y
New password: 
Re-enter new password: 
Password updated successfully!
Reloading privilege tables..
... Success!

By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without having to have a user account created for them. This is intended only for testing, and to make the installation go a bit smoother. You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] y
... Success!

Normally, root should only be allowed to connect from 'localhost'. This ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
... skipping.

By default, MySQL comes with a database named 'test' that anyone can access. This is also intended only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] y
- Dropping test database...
```
.. Success!
- Removing privileges on test database...
  .. Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

**Reload privilege tables now? [Y/n]** y
... Success!

Cleaning up...

5. service mysql stop
6. uncomment all wsrep* lines except first server, where leave only in my.cnf wsrep_cluster_address='gcomm://';
7. start first node
8. Start third node which is connected to first one
9. Start second node which is connected to third one
10. After starting cluster, it must be change cluster address at first starting node without restart database and change config my.cnf.

```
mysql> SET GLOBAL wsrep_cluster_address='gcomm://10.0.0.2';
```

**More Information**

- [http://dev.mysql.com/doc/](http://dev.mysql.com/doc/)
- [http://www.slideshare.net/osscube/mysql-performance-tuning-top-10-tips](http://www.slideshare.net/osscube/mysql-performance-tuning-top-10-tips)
- [http://sourceforge.net/projects/automysqlbackup/](http://sourceforge.net/projects/automysqlbackup/)
- [https://labs.riseup.net/code/projects/backupninja/wiki](https://labs.riseup.net/code/projects/backupninja/wiki)

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

[https://github.com/salt-formulas/salt-formula-mysql/issues](https://github.com/salt-formulas/salt-formula-mysql/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.
Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Nginx Formula**

Nginx is an open source reverse proxy server for HTTP, HTTPS, SMTP, POP3, and IMAP protocols, as well as a load balancer, HTTP cache, and a web server (origin server). The nginx project started with a strong focus on high concurrency, high performance and low memory usage.

**Sample Pillars**

Gitlab server setup

```yaml
nginx:
  server:
    enabled: true
    bind:
      address: '0.0.0.0'
      ports:
      - 80
    site:
      gitlab_domain:
        enabled: true
        type: gitlab
        name: domain
        ssl:
          enabled: true
          key:
          cert: |
          chain: |
          host:
            name: gitlab.domain.com
            port: 80
```

Simple static HTTP site

```yaml
nginx:
  server:
    site:
      nginx_static_site01:
        enabled: true
        type: nginx_static
        name: site01
        host:
          name: gitlab.domain.com
          port: 80
```

Simple load balancer
ngxin:
server:
  upstream:
    horizon-upstream:
      backend1:
        address: 10.10.10.113
        port: 8078
        opts: weight=3
      backend2:
        address: 10.10.10.114
    site:
      nginx_proxy_openstack_web:
        enabled: true
        type: nginx_proxy
        name: openstack_web
        proxy:
          upstream_proxy_pass: http://horizon-upstream
        host:
          name: 192.168.0.1
          port: 31337

Static site with access policy

ngxin:
server:
  site:
    nginx_static_site01:
      enabled: true
      type: nginx_static
      name: site01
      access_policy:
        allow:
          - 192.168.1.1/24
          - 127.0.0.1
        deny:
          - 192.168.1.2
          - all
      host:
        name: gitlab.domain.com
        port: 80

Simple TCP/UDP proxy

ngxin:
server:
  stream:
    rabbitmq:
      host:
        port: 5672
      backend:
        server1:
          address: 10.10.10.113
          port: 5672
          least_conn: true
          hash: "$remote_addr consistent"
  unbound:
    host:
bind: 127.0.0.1
port: 53
protocol: udp
backend:
  server1:
    address: 10.10.10.113
    port: 5353

Simple HTTP proxy

```yaml
nginx:
  server:
    site:
      nginx_proxy_site01:
        enabled: true
        type: nginx_proxy
        name: site01
        proxy:
          host: local.domain.com
          port: 80
          protocol: http
        host:
          name: gitlab.domain.com
          port: 80
```

Simple Websocket proxy

```yaml
nginx:
  server:
    site:
      nginx_proxy_site02:
        enabled: true
        type: nginx_proxy
        name: site02
        proxy:
          websocket: true
          host: local.domain.com
          port: 80
          protocol: http
        host:
          name: gitlab.domain.com
          port: 80
```

Content filtering proxy

```yaml
nginx:
  server:
    enabled: true
    site:
      nginx_proxy_site03:
        enabled: true
        type: nginx_proxy
        name: site03
        proxy:
          host: local.domain.com
          port: 80
          protocol: http
```

(continues on next page)
filter:
  search: https://www.domain.com
  replace: http://10.10.10.10
host:
  name: gitlab.domain.com
  port: 80

Proxy with access policy

nginx:
  server:
    site:
      nginx_proxy_site01:
        enabled: true
        type: nginx_proxy
        name: site01
        access_policy:
          allow:
            - 192.168.1.1/24
            - 127.0.0.1
          deny:
            - 192.168.1.2
            - all
        proxy:
          host: local.domain.com
          port: 80
          protocol: http
          host:
            name: gitlab.domain.com
            port: 80

Gitlab server with user for basic auth

nginx:
  server:
    enabled: true
  user:
    username1:
      enabled: true
      password: magicunicorn
      htpasswd: htpasswd-site1
    username2:
      enabled: true
      password: magicunicorn

Proxy buffering

nginx:
  server:
    enabled: true
    bind:
      address: '0.0.0.0'
    ports:
    - 80
  site:
    gitlab_proxy:
      enabled: true
Let’s Encrypt

```
nginx:
  server:
    enabled: true
    bind:
      address: '0.0.0.0'
      ports:
        - 443
    site:
      gitlab_domain:
        enabled: true
        type: gitlab
        name: domain
        ssl:
          enabled: true
          engine: letsencrypt
          host:
            name: gitlab.domain.com
            port: 443
```

SSL using already deployed key and cert file. Note that cert file should already contain CA cert and complete chain.

```
nginx:
  server:
    enabled: true
  site:
    mysite:
      ssl:
        enabled: true
        key_file: /etc/ssl/private/mykey.key
        cert_file: /etc/ssl/cert/mycert.crt
```

Nginx stats server (required by collectd nginx plugin)

```
nginx:
  server:
    enabled: true
  site:
    nginx_stats_server:
      enabled: true
      type: nginx_stats
      name: server
      host:
        name: 127.0.0.1
        port: 8888
```
Change nginx server ssl protocol options in openstack/proxy.yml

More Information

- http://wiki.nginx.org/Main
- https://mozilla.github.io/server-side-tls/ssl-config-generator/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-nginx/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-nginx

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

openldap

Sample pillars

Client

```yaml
openldap:
  client:
    server:
      basedn: dc=example,dc=local
      host: ldap.example.local
      tls: true
      port: 389
      auth:
        user: cn=admin,dc=example,dc=local
        password: dummypass
      entry:
```

(continues on next page)
people:
  type: ou
  classes:
    - top
    - organizationalUnit
  entry:
    jdoe:
      type: cn
      # Change attributes that already exists with different content
      action: replace
      # Delete all other attributes
      purge: true
      attr:
        uid: jdoe
        uidNumber: 20001
        gidNumber: 20001
        gecos: John Doe
        givenName: John
        sn: Doe
        homeDirectory: /home/jdoe
        loginShell: /bin/bash
      classes:
        - posixAccount
        - inetOrgPerson
        - top
        - ldapPublicKey
        - shadowAccount
    karel:
      # Simply remove cn=karel
      type: cn
      enabled: false

Read more


Postfix

Install and configure Postfix.

Available states

- `postfix.server`
- `postfix.relay`
- `postfix.backupmx`
postfix.server

Setup postfix server

postfix.relay

Setup postfix relay

postfix.backupmx

Setup postfix backup MX

Available metadata

- metadata.postfix.server
- postfix.relay
- postfix.backupmx

metadata.postfix.server

Setup postfix server

postfix.relay

Setup postfix relay

postfix.backupmx

Setup postfix backup MX

Requirements

- linux
- mysql (for mysql backend and postfixadmin)
- apache (for postfixadmin)

Optional

- glusterfs (to serve as mail storage backend)
- dovecot
- roundcube
Configuration parameters

For complete list of parameters, please check `metadata/service/server.yml`

Example reclass

Server

```
classes:
  - service.postfix.server
parameters:
  _param:
    postfix_origin: mail.eru
    mysql_mailserver_password: Peixeilaephahmoosa2daihoh4yiaThe
postfix:
  server:
    origin: ${_param:postfix_origin}
  ssl:
    enabled: true
    key: ${_secret:ssl_domain_wild_key}
    cert: ${_secret:ssl_domain_wild_cert}
    chain: ${_secret:ssl_domain_wild_chain}
    # Set smtpd_tls_security_level to encrypt and require TLS encryption
    required: true
mysql:
  server:
    database:
      mailserver:
        encoding: UTF8
        locale: cs_CZ
        users:
          - name: mailserver
            password: ${_param:mysql_mailserver_password}
            host: 127.0.0.1
            rights: all privileges
apache:
  server:
    site:
      postfixadmin:
        enabled: true
        type: static
        name: postfixadmin
        root: /usr/share/postfixadmin
        host:
          name: ${_param:postfix_origin}
          aliases:
            - ${linux:system:name}.${linux:system:domain}
            - ${linux:system:name}
```

Example pillar

Server

Setup without postfixadmin:
postfix:
  server:
    origin: ${_param:postfix_origin}
  admin:
    enabled: false

DKIM

postfix:
  server:
    dkim:
      enabled: true
      domains:
      - name: example.com
        selector: mail
        key: |
          super_secret_private_key

First you need to generate private and public key, eg.:

opendkim-genkey -r -s mail -d example.com

And set public key in your DNS records, see mail.txt for public key.

Mailman

postfix:
  server:
    mailman:
      enabled: true
      admin_password: SaiS0kai
      distributed: true
      use_https: false
      lists:
      - name: support
        admin: test@lxc.eru
        password: test
        domain: lxc.eru
        domainweb: lists.lxc.eru
        members:
          - test@lxc.eru

It’s also good idea to mount GlusterFS volume on /var/lib/mailman for multi-master setup. In that case distributed has to be true to bind-mount qfiles directory which must not be shared.

Parameter use_https needs to be set before setting up any lists, otherwise you need to fix lists urls manually using:

withlist -l -a -r fix_url

You can also set per-list parameters. For example you can setup private mailing list with these options:

lists:
  - name: support
admin: test@lxc.eri
password: test
domain: lxc.eri
domainweb: lists.lxc.eri
members:
- test@lxc.eri
parameters:
  real_name: support
description: "Support mailing list"
  # Don’t be advertised
  advertised: 0
  # Require admin to confirm subscription
  subscribe_policy: 2
  # Show members only to admins
  private_roster: 2
  # Archive only for members
  archive_private: 1

To list all available configuration options for given list, see output of following command:

`config_list -o - <list_name>`

**Warning:** If you want to have list on your domain, eg. `support@example.com` instead of `support@lists.example.com`, you may need to set up aliases like this, depending on your setup:

support-owner@example.com -> support-owner@lists.example.com
support-admin@example.com -> support-admin@lists.example.com
support-request@example.com -> support-request@lists.example.com
support-confirm@example.com -> support-confirm@lists.example.com
support-join@example.com -> support-join@lists.example.com
support-leave@example.com -> support-leave@lists.example.com
support-unsubscribe@example.com -> support-unsubscribe@lists.example.com
support-bounces@example.com -> support-bounces@lists.example.com
support@example.com -> support@lists.example.com

**Relay**

`postfix:
relay:
  # Postfix will listen only on localhost
  interfaces: loopback-only
  host: mail.cloudlab.cz
  domain: cloudlab.cz
  sasl:
    user: test
    password: changeme`
Backup MX

```
postfix:
  backupmx:
    domains:
    - cloudlab.cz
    - lists.cloudlab.cz
```

Development and testing

Development and test workflow with Test Kitchen and kitchen-salt provisioner plugin.

Test Kitchen is a test harness tool to execute your configured code on one or more platforms in isolation. There is a `.kitchen.yml` in main directory that defines `platforms` to be tested and `suites` to execute on them.

Kitchen CI can spin instances locally or remote, based on used `driver`. For local development `.kitchen.yml` defines a `vagrant` or `docker` driver.

To use backend drivers or implement your CI follow the section `INTEGRATION.rst#Continuous Integration`.

A listing of scenarios to be executed:

```
$ kitchen list

<table>
<thead>
<tr>
<th>Instance</th>
<th>Driver</th>
<th>Provisioner</th>
<th>Verifier</th>
<th>Transport</th>
<th>Last Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>server-bento-ubuntu-1404</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>server-bento-ubuntu-1604</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>Created</td>
</tr>
<tr>
<td>server-bento-centos-71</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>relay-bento-ubuntu-1404</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>relay-bento-ubuntu-1604</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>relay-bento-centos-71</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>backupmx-bento-ubuntu-1404</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>backupmx-bento-ubuntu-1604</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
<tr>
<td>backupmx-bento-centos-71</td>
<td>Vagrant</td>
<td>SaltSolo</td>
<td>Busser</td>
<td>Ssh</td>
<td>&lt;Not Created&gt;</td>
</tr>
</tbody>
</table>
```

The `Busser Verifier` is used to setup and run tests implemented in `<repo>/test/integration`. It installs the particular driver to tested instance (`Serverspec`, `InSpec`, `Shell`, `Bats`, ...) prior the verification is executed.

Usage:

```
# manually
kitchen [test || [create|converge|verify|exec|login|destroy|...]] -t tests/integration

# or with provided Makefile within CI pipeline
make kitchen
```

Read more

- [http://fog.ccsf.edu/~msapiro/scripts/](http://fog.ccsf.edu/~msapiro/scripts/)
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-postfix/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-postfix

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

PostgreSQL Formula

PostgreSQL, often simply Postgres, is an object-relational database management system available for many platforms including Linux, FreeBSD, Solaris, Microsoft Windows and Mac OS X. It is released under the PostgreSQL License, which is an MIT-style license, and is thus free and open source software. PostgreSQL is developed by the PostgreSQL Global Development Group, consisting of a handful of volunteers employed and supervised by companies such as Red Hat and EnterpriseDB.

Sample pillars

Single deployment

Single database server with empty database

```
postgresql:
  server:
    enabled: true
    version: 9.1
    bind:
      address: 127.0.0.1
      port: 5432
      protocol: tcp
    clients:
      - 127.0.0.1
  database:
    databasename:
      encoding: 'UTF8'
      locale: 'cs_CZ'
      users:
```

(continues on next page)
- name: 'username'
  password: 'password'
  host: 'localhost'
  rights: 'all privileges'

Single database server with initial data

```yaml
postgresql:
  server:
    enabled: true
    version: 9.1
    bind:
      - address: 127.0.0.1
        port: 5432
        protocol: tcp
    clients:
      - 127.0.0.1
  database:
    databasename:
      encoding: 'UTF8'
      locale: 'cs_CZ'
    initial_data:
      engine: backupninja
      source: backup.host
      host: original-host-name
      database: original-database-name
    users:
      - name: 'username'
        password: 'password'
        host: 'localhost'
        rights: 'all privileges'
```

User with createdb privileges

```yaml
postgresql:
  server:
    enabled: true
    version: 9.1
    bind:
      address: 127.0.0.1
      port: 5432
      protocol: tcp
    clients:
      - 127.0.0.1
  database:
    databasename:
      encoding: 'UTF8'
      locale: 'cs_CZ'
    users:
      - name: 'username'
        password: 'password'
        host: 'localhost'
        createdb: true
        rights: 'all privileges'
```

Database extensions
postgresql:
  server:
    enabled: true
    version: 9.1
    bind:
      address: 127.0.0.1
      port: 5432
      protocol: tcp
    clients:
    - 127.0.0.1
    database:
      databasename:
      encoding: 'UTF8'
      locale: 'cs_CZ'
      users:
      - name: 'username'
        password: 'password'
        host: 'localhost'
        createdb: true
        rights: 'all privileges'
    extension:
      postgis_topology:
        enabled: true
      fuzzystrmatch:
        enabled: true
      postgis_tiger_geocoder:
        enabled: true
      postgis:
        enabled: true
      pkgs:
      - postgresql-9.1-postgis-2.1

Master-slave cluster

Master node

postgresql:
  server:
    enabled: true
    version: 9.6
    bind:
      address: 0.0.0.0
    database:
      mydb: ...
  cluster:
    enabled: true
    role: master
    mode: hot_standby
    members:
    - host: "172.16.10.101"
    - host: "172.16.10.102"
    - host: "172.16.10.103"
    replication_user:
      name: repuser
      password: password

(continues on next page)
multi-master cluster

Multi-master cluster with 2ndQuadrant bi-directional replication plugin

Master node
postgresql:
    server:
        enabled: true
        version: 9.4
        bind:
            address: 0.0.0.0
        database:
            mydb:
                extension:
                    bdr:
                        enabled: true
                    btree_gist:
                        enabled: true
    cluster:
        enabled: true
        mode: bdr
        role: master
        members:
            - host: "172.16.10.101"
            - host: "172.16.10.102"
            - host: "172.16.10.101"
            local: "172.16.10.101"
        replication_user:
            name: repuser
            password: password

Slave node

postgresql:
    server:
        enabled: true
        version: 9.4
        bind:
            address: 0.0.0.0
        database:
            mydb:
                extension:
                    bdr:
                        enabled: true
                    btree_gist:
                        enabled: true
    cluster:
        enabled: true
        mode: bdr
        role: master
        members:
            - host: "172.16.10.101"
            - host: "172.16.10.102"
            - host: "172.16.10.101"
            local: "172.16.10.102"
            master: "172.16.10.101"
        replication_user:
            name: repuser
            password: password

2.1. Project Introduction
Client

```
postgresql:
    client:
        server:
            server01:
                admin:
                    host: database.host
                    port: 5432
                    user: root
                    password: password
        database:
            mydb:
                enabled: true
                encoding: 'UTF8'
                locale: 'en_US'
                users:
                    - name: test
                      password: test
                      host: localhost
                      createdb: true
                      rights: all privileges
        init:
            maintenance_db: mydb
            queries:
                - INSERT INTO login VALUES (11, 1) ;
                - INSERT INTO device VALUES (1, 11, 42);
```

Sample usage

Init database cluster with given locale

```
```

Convert PostgreSQL cluster from 9.1 to 9.3

```
```

Ubuntu on 14.04 on some machines won’t create default cluster

```
sudo pg_createcluster 9.3 main --start
```

More information

- [http://www.postgresql.org/](http://www.postgresql.org/)
- [https://gist.github.com/ibussieres/11262268 - upgrade instructions for ubuntu](https://gist.github.com/ibussieres/11262268 - upgrade instructions for ubuntu)
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-postgresql/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-postgresql

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

PowerDNS

Sample pillar:

PowerDNS server with MySQL backend
PowerDNS server with sqlite backend

Read more

RabbitMQ messaging system

RabbitMQ is a complete and highly reliable enterprise messaging system based on the emerging AMQP standard.

Sample pillars

Standalone Broker

RabbitMQ as AMQP broker with admin user and vhosts

```
rabbitmq:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 5672
    secret_key: rabbit_master_cookie
    admin:
      name: adminuser
```
RabbitMQ as a Stomp broker

```yaml
rabbitmq:
server:
    enabled: true
    secret_key: rabbit_master_cookie
    bind:
        address: 0.0.0.0
        port: 5672
    host:
        '/monitor':
            enabled: true
            user: 'monitor'
            password: 'password'
    plugins:
        - rabbitmq_stomp
```

RabbitMQ cluster

RabbitMQ as base cluster node

```yaml
rabbitmq:
server:
    enabled: true
    bind:
        address: 0.0.0.0
        port: 5672
    secret_key: rabbit_master_cookie
    admin:
        name: adminuser
        password: pwd
    cluster:
        enabled: true
        role: master
        mode: disc
        members:
            - name: openstack1
              host: 10.10.10.212
            - name: openstack2
              host: 10.10.10.213
```

HA Queues definition

```yaml
rabbitmq:
server:
```

(continues on next page)
Enable TLS support

To enable support of TLS for rabbitmq-server you need to provide a path to cacert, server cert and private key:

```
rabbitmq:
  server:
    enabled: true
    ...
  ssl:
    enabled: True
    key_file: /etc/rabbitmq/ssl/key.pem
    cert_file: /etc/rabbitmq/ssl/cert.pem
    ca_file: /etc/rabbitmq/ssl/ca.pem
```

To manage content of these files you can either use the following options:

```
rabbitmq:
  server:
    enabled: true
    ...
  ssl:
    enabled: True
    key_file: /etc/rabbitmq/ssl/key.pem
    key: |
    -----BEGIN RSA PRIVATE KEY-----
    ...
    -----END RSA PRIVATE KEY-----
    ca_file: /etc/rabbitmq/ssl/ca.pem
    cacert_chain: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
    cert_file: /etc/rabbitmq/ssl/cert.pem
    cert: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
```

Or you can use the `salt.minion.cert` salt state which creates all required files according to defined reclass model [1]. In this case you need just to enable ssl and nothing more:

2.1. Project Introduction
rabbitmq:
  server:
    enabled: true
    ...
  ssl:
    enabled: True

Default port for TLS is 5671:

rabbitmq:
  server:
    bind:
      ssl:
        port: 5671


Usage

Check cluster status, example shows running cluster with 3 nodes: ctl-1, ctl-2, ctl-3

```bash
> rabbitmqctl cluster_status
Cluster status of node 'rabbit@ctl-1' ...[nodes,[[disc,['rabbit@ctl-1','rabbit@ctl-2','rabbit@ctl-3']]]],
  [running_nodes,['rabbit@ctl-3','rabbit@ctl-2','rabbit@ctl-1']],
  [partitions,[]]
...done.
```

Setup management user.

```bash
> rabbitmqctl add_vhost vhost
> rabbitmqctl add_user user alive
> rabbitmqctl set_permissions -p vhost user ".*." ".*" ".*."
> rabbitmqctl set_user_tags user management
```

EPD process is Erlang Port Mapper Daemon. It’s a feature of the Erlang runtime that helps Erlang nodes to find each other. It’s a pretty tiny thing and doesn’t contain much state (other than “what Erlang nodes are running on this system?”) so it’s not a huge deal for it to still be running. Although it’s running as user rabbitmq, it was started automatically by the Erlang VM when we started. We’ve considered adding “epmd -kill” to our shutdown script - but that would break any other Erlang apps running on the system; it’s more “global” than RabbitMQ.

Read more

- http://docs.saltstack.com/ref/states/all/salt.states.rabbitmq_user.html
- http://stackoverflow.com/questions/14699873/how-to-reset-user-for-rabbitmq-management
Clustering

- http://www.rabbitmq.com/clustering.html#auto-config

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-rabbitmq/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula:

https://github.com/salt-formulas/salt-formula-rabbitmq

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Redis formula

key value storage

Sample pillars

Redis localhost server

```
redis:
  server:
    enabled: true
    bind:
      address: 127.0.0.1
      port: 6379
      protocol: tcp
```

Redis world open

2.1. Project Introduction
Redis modes

redis:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 6379
      protocol: tcp

appendfsync: no | everysec | always

Redis cluster master

redis:
  cluster:
    enabled: True
    master:
      host: 192.168.1.100
      port: 6379
      mode: sentinel
      quorum: 2
      role: master
  supervisor:
    server:
      service:
        redis_sentinel:
          name: sentinel
          type: redis

Redis cluster slave

redis:
  cluster:
    enabled: True
    master:
      host: 192.168.1.100
      port: 6379
      mode: sentinel
      quorum: 2
      role: slave
  supervisor:
    server:
      service:
        redis_sentinel:
          name: sentinel
          type: redis

Command usage

Removes data from your connection’s CURRENT database.

> redis-cli FLUSHDB
Removes data from ALL databases.

```bash
> redis-cli FLUSHALL
```

More information

- http://redis.io/topics/admin
- http://redis.io/topics/quickstart
- http://redis.io/topics/persistence

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-redis/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-redis

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

rsync Formula

rsync is an open source utility that provides fast incremental file transfer.

Sample Metadata

```
rsync:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
  module:
    name:
      max_connections: 2
      path: /srv/rsync
      read_only: False
    timeout: 300
```
**References**

- http://rsync.samba.org/

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-rsync/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-rsync

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Supervisor Formula**

Supervisor is a client/server system that allows its users to monitor and control a number of processes on UNIX-like operating systems.

It shares some of the same goals of programs like launchd, daemontools, and runit. Unlike some of these programs, it is not meant to be run as a substitute for init as “process id 1”. Instead it is meant to be used to control processes related to a project or a customer, and is meant to start like any other program at boot time.

**Sample Pillars**

Robotice services

```python
supervisor:
    server:
        enabled: true
        service:
            robotice_planner:
                name: planner
                type: robotice
            robotice_monitor:
                name: monitor
                type: robotice
```
OctoPrint services

```
supervisor:
    server:
        enabled: true
        service:
            octoprint_server:
                name: server
                type: octoprint
```

Sentry services

```
supervisor:
    server:
        enabled: true
        service:
            sentry_web:
                name: web
                type: sentry
            sentry_worker:
                name: worker
                type: sentry
```

More Information

- http://supervisord.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-supervisor/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-supervisor

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

varnish

Varnish cache.

2.1. Project Introduction
Sample pillars

Single varnish service

```yaml
varnish:
    server:
        enabled: true
        version: 4.0
        lookup:
            varnish_leonardo_majklk:
                type: leonardo
                name: leonardo_majklk
                bind:
                    port: 7000
                    host: 0.0.0.0
                backend:
                    gunicorn1:
                        host: localhost
                        port: 80
```

And Supervisor like this:: yaml

```
supervisor:
    server:
        service:
            varnish_leonardo_majklk: name: leonardo_majklk type: varnish
```

Note: This formulas runs varnish processes under supervisor instead of init script.

Using nginx type:: yaml

```
nginx:
    server:
        site:
            leonardo_majklk: enabled: true type: varnish name: varnish_leonardo_majklk host:
```

Read more

- links

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-varnish/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:
You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-varnish

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

## zookeeper

Service zookeeper description

### Sample pillars

#### Single zookeeper service

```yaml
zookeeper:
  server:
    enabled: true
    members:
- host: ${_param:single_address}
  id: 1
```

#### Cluster zookeeper service

```yaml
zookeeper:
  server:
    enabled: true
    members:
- host: ${_param:cluster_node01_address}
  id: 1
- host: ${_param:cluster_node02_address}
  id: 2
- host: ${_param:cluster_node03_address}
  id: 3
```

#### Backup client with ssh/rsync remote host

```yaml
zookeeper:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
      target:
        host: cfg01

.. note:: full_backups_to_keep param states how many backup will be stored locally on zookeeper client.

  More options to relocate local backups can be done using salt-formula-backupninja.
```
Backup client with local backup only

```yaml
zookeeper:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24

.. note:: full_backups_to_keep param states how many backup will be stored locally on →zookeeper client
```

Backup server rsync

```yaml
zookeeper:
  backup:
    server:
      enabled: true
      hours_before_full: 24
      full_backups_to_keep: 5
      key:
        zookeeper_pub_key:
          enabled: true
          key: ssh_rsa
```

Client restore from local backup:

```yaml
zookeeper:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
      target:
        host: cfg01
        restore_latest: 1
        restore_from: local

.. note:: restore_latest param with a value of 1 means to restore db from the last →full backup. 2 would mean to restore second latest full backup.
```

Client restore from remote backup:

```yaml
zookeeper:
  backup:
    client:
      enabled: true
      full_backups_to_keep: 3
      hours_before_full: 24
      target:
        host: cfg01
        restore_latest: 1
        restore_from: remote

.. note:: restore_latest param with a value of 1 means to restore db from the last →full backup. 2 would mean to restore second latest full backup.
```
Read more

- links

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-zookeeper/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula:

https://github.com/salt-formulas/salt-formula-zookeeper

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Deployment Services

Deployment services for automated delivery pipelines.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>gateone</td>
<td><a href="https://github.com/salt-formulas/salt-formula-gateone">https://github.com/salt-formulas/salt-formula-gateone</a></td>
</tr>
<tr>
<td>foreman</td>
<td><a href="https://github.com/salt-formulas/salt-formula-foreman">https://github.com/salt-formulas/salt-formula-foreman</a></td>
</tr>
<tr>
<td>isc-dhcp</td>
<td><a href="https://github.com/salt-formulas/salt-formula-isc-dhcp">https://github.com/salt-formulas/salt-formula-isc-dhcp</a></td>
</tr>
<tr>
<td>libvirt</td>
<td><a href="https://github.com/salt-formulas/salt-formula-libvirt">https://github.com/salt-formulas/salt-formula-libvirt</a></td>
</tr>
<tr>
<td>maas</td>
<td><a href="https://github.com/salt-formulas/salt-formula-maas">https://github.com/salt-formulas/salt-formula-maas</a></td>
</tr>
<tr>
<td>stackstorm</td>
<td><a href="https://github.com/salt-formulas/salt-formula-stackstorm">https://github.com/salt-formulas/salt-formula-stackstorm</a></td>
</tr>
<tr>
<td>tftp-d-hpa</td>
<td><a href="https://github.com/salt-formulas/salt-formula-tftp-d-hpa">https://github.com/salt-formulas/salt-formula-tftp-d-hpa</a></td>
</tr>
<tr>
<td>vagrant</td>
<td><a href="https://github.com/salt-formulas/salt-formula-vagrant">https://github.com/salt-formulas/salt-formula-vagrant</a></td>
</tr>
<tr>
<td>virtualbox</td>
<td><a href="https://github.com/salt-formulas/salt-formula-virtualbox">https://github.com/salt-formulas/salt-formula-virtualbox</a></td>
</tr>
</tbody>
</table>

2.1. Project Introduction
GateOne Formula

Gate One is an open source, web-based terminal emulator with a powerful plugin system. It comes bundled with a plugin that turns Gate One into an amazing SSH client but Gate One can actually be used to run any terminal application. You can even embed Gate One into other applications to provide an interface into serial consoles, virtual servers, or anything you like. It’s a great supplement to any web-based administration interface.

Sample Pillars

```yaml
gateone:
  server:
    enabled: true
  bind:
    address: '0.0.0.0'
    port: 8888
    protocol: 'tcp'
  auth:
    engine: pam
    realm: local
```

More Information

- [http://liftoff.github.io/GateOne/](http://liftoff.github.io/GateOne/)
- [https://github.com/liftoff/GateOne](https://github.com/liftoff/GateOne)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

**http://salt-formulas.readthedocs.io/**

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

**https://github.com/salt-formulas/salt-formula-gateone/issues**

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

**https://launchpad.net/salt-formulas**

You can also join salt-formulas-users team and subscribe to mailing list:

**https://launchpad.net/~salt-formulas-users**

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

**https://github.com/salt-formulas/salt-formula-gateone**

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Foreman

Foreman is aimed to be a Single Address For All Machines Life Cycle Management.

- Foreman integrates with Puppet (and acts as web front end to it).
- Foreman takes care of provisioning until the point puppet is running, allowing Puppet to do what it does best.
- Foreman shows you Systems Inventory (based on Facter) and provides real time information about hosts status based on Puppet reports.
- Foreman creates everything you need when adding a new machine to your network. It’s goal being automatically managing everything you would normally manage manually - this include DNS, DHCP, TFTP, Virtual Machines, PuppetCA, CMDB etc.

Sample pillar

Foreman server to use with apache

```yaml
foreman:
  server:
    enabled: true
    domain: domain.com
    fqdn: foreman.domain.com
  database:
    engine: 'postgresql'
    host: '127.0.0.1'
    name: 'foreman'
    password: 'password'
    user: 'foreman'
  mail:
    host: mail.domain.com
    password: passwd
    user: robot@domain.com
    domain: domain.com
```

Foreman smart proxy

```yaml
foreman:
  smart_proxy:
    enabled: true
```

Usage

Generated user:password is in database seed and printed to the output during db:seed process.

Read more

- http://theforeman.org/manuals/1.5/index.html#3.InstallingForeman
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-foreman/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-foreman

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

ISC DHCP formula

Sample pillars

ISC DHCP server with defined host and subnet (client must use the same key)

```yaml
isc_dhcp:
  server:
    enabled: true
    omapi_port: 7911
    omapi_key: iFdQ0kvpUo+3gzXGJTpk7/
    d19DI5uDoqMzasDUbRGEg6VfVNYUX+MAUl4WoJZDQbrvC4Pgsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdfsdf
More information


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-isc-dhcp/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula:

https://github.com/salt-formulas/salt-formula-isc-dhcp

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Libvirt

Sample pillars

simple libvirt server

libvirt:
  server:
    enabled: true
    unix_sock_group: libvirt
    virtualizations:
    - kvm
    network:
      default:
        ensure: absent

libvirt:
  server:
    enabled: true
    network:
      default:
        ensure: absent  #present, running, stopped, absent
    mydefault:
      xml: |
      <network>

(continues on next page)
<network>
  <name>mydefault</name>
  <bridge name="virbr0"/>
  <forward/>
  <ip address="192.168.122.1" netmask="255.255.255.0">
    <dhcp>
      <range start="192.168.122.2" end="192.168.122.254"/>
    </dhcp>
  </ip>
</network>

<network>
  <name>ovs-net</name>
  <forward mode='bridge'/>
  <bridge name='ovsbr0'/>
  <virtualport type='openvswitch'>
    <parameters interfaceid='09b11c53-8b5c-4eeb-8f00-d84eaa0aa4f'/>
  </virtualport>
</network>

libvirt:
server:
  enabled: true
pool:
  virtimages:
    type: dir
    path: /var/lib/libvirt/images
    xml: |
      <pool type="dir">
        <name>virtimages</name>
        <target>
          <path>/var/lib/libvirt/images</path>
        </target>
      </pool>
  virtimages2:
    ensure: absent
    type: dir
    path: /var/lib/libvirt/images2
    xml: |
      <pool type="dir">
        <name>virtimages2</name>
        <target>
          <path>/var/lib/libvirt/images2</path>
        </target>
      </pool>

Read more

- https://github.com/bechtoldt/saltstack-libvirt-formula

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:
In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-libvirt/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-libvirt

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

### Metal as a Service

Service maas description

### Sample pillars

Single maas service

```
maas:
  server:
    enabled: true
```

Single MAAS region service [single UI/API]

```
maas:
  salt_master_ip: 192.168.0.10
  region:
    upstream_proxy:
      address: 10.0.0.1
      port: 8080
      user: username
      password: password
    theme: mirantis
  bind:
    host: 192.168.0.10:5240
    port: 5240
  admin:
    username: exampleuser
    password: examplepassword
    email: email@example.com
  database:
    engine: null
    host: localhost
    name: maasdb
```

(continues on next page)
password: qwqwqzw
username: maas
enabled: true
user: mirantis
token: "89EgtWkX45ddjMYpuL:SqVjxFG87Dr6kVf4Wp:5WLfbUgmm9XQtJxm3V2LUUy7bpCmqmnk"
fabrics:
  test-fabric1:
    description: "Test fabric"
  test-fabric2:
    description: "Test fabric2"
subnets:
  subnet1:
    fabric: test-fabric1
    cidr: 2.2.3.0/24
    gateway_ip: 2.2.3.2
    iprange: # reserved range for DHCP\auto mapping
      start: 2.2.3.20
      end: 2.2.3.250

dhcp_snippets:
  test-snippet:
    value: option bootfile-name "tftp://192.168.0.10/snippet";
    description: Test snippet
    enabled: true
    subnet: subnet1
boot_resources:
  bootscript1:
    title: bootscript
    architecture: amd64/generic
    filetype: tgz
    content: /srv/salt/reclass/nodes/path_to_file
package_repositories:
  Saltstack:
    distributions:
      - trusty
    components:
      - main
    arches: amd64
    enabled: true
machines:
  machine1_new_schema:
    pxe_interface_mac: "11:22:33:44:55:66" # Node will be identified by those mac
    interfaces:
      nic01: # could be any, used for iterate only
        type: eth # NotImplemented
        name: eth0 # Override default nic name. Interface to rename will be identified by mac
mode: "static"
ip: "2.2.3.19"  # ip should be out of reserved subnet range, but still in subnet range
subnet: "subnet1"
gateway: "2.2.3.2"  # override default gateway from subnet
nic02:
type: eth  # Not-implemented
subnet: "subnet2"
mode: "dhcp"
power_parameters:
power_type: ipmi
power_address: '192.168.10.10'
power_user: bmc_user
power_password: bmc_password
#Optional (for legacy HW)
power_driver: LAN
distro_series: xenial
hwe_kernel: hwe-16.04
machine1_old_schema:
interface:
mac: "11:22:33:44:55:88"  # Node will be identified by those mac
mode: "static"
ip: "2.2.3.15"
subnet: "subnet1"
gateway: "2.2.3.2"
power_parameters:
power_type: ipmi
power_address: '192.168.10.10'
power_user: bmc_user
power_password: bmc_password
#Optional (for legacy HW)
power_driver: LAN
# FIXME: that's should be moved into another,livirt example.
# Used in case of power_type: virsh
power_id: my_libvirt_vm_name
distro_series: xenial
hwe_kernel: hwe-16.04
devices:
machine1-ipmi:
interface:
ip_address: 192.168.10.10
subnet: cidr:192.168.10.0/24
commissioning_scripts:
00-maas-05-simplify-network-interfaces: /etc/maas/files/commisioning_scripts/00-
maas-05-simplify-network-interfaces
maas_config:
domain: mydomain.local
http_proxy: http://192.168.0.10:3142
commissioning_distro_series: xenial
default_distro_series: xenial
default_osystem: 'ubuntu'
default_storage_layout: lvm
disk_erase_with_secure_erase: true
dnssec_validation: 'no'
enable_third_party_drivers: true
Usage of local repos

```
maas:
  cluster:
    enabled: true
    region:
      port: 80
      host: localhost
  saltstack_repo_key: |
    -----BEGIN PGP PUBLIC KEY BLOCK-----
    Version: GnuPG v2
    mQENBFOpvpgBCADkP656H41i8fpp1EEB8IeLhugyC2rTEwwSc1b8tQNYtUiGdna9
    ....
    fuBmScum8uQTnEF5+Um5zkwC7EXTdH1co/+V/fp0tx1g4XO4kg2efVm5ERfVS
    MA==
    =dtMN
    -----END PGP PUBLIC KEY BLOCK-----
  saltstack_repo_xenial: "http://${_param:local_repo_url}/ubuntu-xenial stable salt"
  saltstack_repo_trusty: "http://${_param:local_repo_url}/ubuntu-trusty stable salt"
```

Single MAAS cluster service [multiple racks]

```
maas:
  cluster:
    enabled: true
    role: master/slave
```

```
maas:
  cluster:
    enabled: true
    role: master/slave
```

MAAS region service with backup data

**Module function’s example:**

- Wait for status of selected machine’s:

```
> cat maas/machines/wait_for_machines_ready.sls
...
wait_for_machines_ready:
```
module.run:
- name: maas.wait_for_machine_status
- kwargs:
  machines:
  - kvm01
  - kvm02
  timeout: 1200 # in seconds
  req_status: "Ready"
- require:
  - cmd: maas_login_admin
...

If module run w/o any extra paremeters - `wait_for_machines_ready` will wait for defined in salt machines. In those case, will be useful to skip some machines:

```bash
> cat maas/machines/wait_for_machines_deployed.sls
...
wait_for_machines_ready:
  module.run:
  - name: maas.wait_for_machine_status
  - kwargs:
    timeout: 1200 # in seconds
    req_status: "Deployed"
    ignore_machines:
    - kvm01 # in case it's broken or whatever
  - require:
    - cmd: maas_login_admin
...
```

List of available `req_status` defined in global variable:

```python
STATUS_NAME_DICT = dict(
    (0, 'New'), (1, 'Commissioning'), (2, 'Failed commissioning'),
    (3, 'Missing'), (4, 'Ready'), (5, 'Reserved'), (10, 'Allocated'),
    (9, 'Deploying'), (6, 'Deployed'), (7, 'Retired'), (8, 'Broken'),
    (11, 'Failed deployment'), (12, 'Releasing'),
    (13, 'Releasing failed'), (14, 'Disk erasing'),
    (15, 'Failed disk erasing'))
```

**Read more**

- https://maas.io/

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-maas/issues
For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-maas

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

---

**stackstorm**

Service stackstorm description

**Sample pillars**

Single stackstorm service

```yaml
stackstorm:
  server:
    enabled: true
    version: icehouse
```

**Read more**

- links

**TFTPD HPA formula**

A TFTP server is mainly required for booting operating systems or configurations over the network.

**Sample pillars**

TFTPD HPA server

```yaml
tftpd_hpa:
  server:
    enabled: true
```

**More information**

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-tftpd-hpa/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-tftpd-hpa

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Vagrant formula

Vagrant provides easy to configure, reproducible, and portable work environments built on top of industry-standard technology and controlled by a single consistent workflow to help maximize the productivity and flexibility of you and your team.

To achieve its magic, Vagrant stands on the shoulders of giants. Machines are provisioned on top of VirtualBox, VMware, AWS, or any other provider. Then, industry-standard provisioning tools such as shell scripts, Chef, or Puppet, can be used to automatically install and configure software on the machine.

Sample pillars

Vagrant with VirtualBox cluster

```yaml
vagrant:
  control:
    enabled: true
    cluster:
      clustername:
        provider: virtualbox
        domain: local.domain.com
        control:
          engine: salt
          host: salt.domain.com
          version: '2016.3'
        node:
          box1:
            status: suspended
            image: ubuntu1204
            memory: 512
```

(continues on next page)
Vagrant with Windows plugin

```yaml
vagrant:
  control:
    enabled: true
    plugin:
      vagrant-windows:
        version: 1.2.3
```

Vagrant with presseded images

```yaml
vagrant:
  control:
    enabled: true
    image:
      ubuntu1204:
        source: http://files.vagrantup.com/precise64.box
```

Sample usage

Start and connect machine

```bash
cd /srv/vagrant/<cluster_name>
vagrant up <node_name>
vagrant ssh <node_name>
```

More information

- http://www.vagrantup.com/
- http://docs.vagrantup.com/v2/
- http://docs.vagrantup.com/v2/synced-folders/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-vagrant/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:
Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**VirtualBox**

VirtualBox is a general-purpose full virtualizer for x86 hardware, targeted at server, desktop and embedded use.

**Sample pillars**

**VirtualBox version 4.3**

```yaml
virtualbox:
  host:
    enabled: true
    version: 4.3
    extensions: false
```

**VirtualBox version 5.0**

```yaml
virtualbox:
  host:
    enabled: true
    version: 5.0
```

**Read more**

- [https://www.virtualbox.org/wiki/Technical_documentation](https://www.virtualbox.org/wiki/Technical_documentation)
- [https://www.virtualbox.org/wiki/Linux_Downloads](https://www.virtualbox.org/wiki/Linux_Downloads)

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

[https://github.com/salt-formulas/salt-formula-virtualbox/issues](https://github.com/salt-formulas/salt-formula-virtualbox/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:
Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

---

### Integration Services

Continuous integration services for automated integration and delivery pipelines.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>aptly</td>
<td><a href="https://github.com/salt-formulas/salt-formula-aptly">https://github.com/salt-formulas/salt-formula-aptly</a></td>
</tr>
<tr>
<td>artifactory</td>
<td><a href="https://github.com/salt-formulas/salt-formula-artifactory">https://github.com/salt-formulas/salt-formula-artifactory</a></td>
</tr>
<tr>
<td>gerrit</td>
<td><a href="https://github.com/salt-formulas/salt-formula-gerrit">https://github.com/salt-formulas/salt-formula-gerrit</a></td>
</tr>
<tr>
<td>gitlab</td>
<td><a href="https://github.com/salt-formulas/salt-formula-gitlab">https://github.com/salt-formulas/salt-formula-gitlab</a></td>
</tr>
<tr>
<td>gource</td>
<td><a href="https://github.com/salt-formulas/salt-formula-gource">https://github.com/salt-formulas/salt-formula-gource</a></td>
</tr>
<tr>
<td>jenkins</td>
<td><a href="https://github.com/salt-formulas/salt-formula-jenkins">https://github.com/salt-formulas/salt-formula-jenkins</a></td>
</tr>
<tr>
<td>owncloud</td>
<td><a href="https://github.com/salt-formulas/salt-formula-owncloud">https://github.com/salt-formulas/salt-formula-owncloud</a></td>
</tr>
<tr>
<td>packer</td>
<td><a href="https://github.com/salt-formulas/salt-formula-packer">https://github.com/salt-formulas/salt-formula-packer</a></td>
</tr>
<tr>
<td>roundcube</td>
<td><a href="https://github.com/salt-formulas/salt-formula-roundcube">https://github.com/salt-formulas/salt-formula-roundcube</a></td>
</tr>
</tbody>
</table>

**Aptly**

Install and configure Aptly server and client.

**Available states**

- `aptly.server`
- `aptly.publisher`

**aptly.server**

Setup aptly server
aptly.publisher

Setup aptly publisher

Available metadata

- metadata.aptly.server.single
- metadata.aptly.client.publisher

metadata.aptly.server.single

Setup basic server

metadata.aptly.client.publisher

Setup aptly publisher client

Configuration parameters

Example reclass

Basic Apty server with no repos or mirrors.

```
classes:
  - service.aptly.server.single
parameters:
  aptly:
    server:
      enabled: true
      secure: true
      gpg_keypair_id: A76882D3
      gpg_passphrase:
      gpg_public_key: |
        -----BEGIN PGP PUBLIC KEY BLOCK-----
        Version: GnuPG v1
        ...
      gpg_private_key: |
        -----BEGIN PGP PRIVATE KEY BLOCK-----
        Version: GnuPG v1
        ...

Define s3 endpoint:

```
parameters:
  aptly:
    server:
      endpoint:
        mys3endpoint:
```

(continues on next page)
engine: s3
awsAccessKeyID: xxxx
awsSecretAccessKey: xxxx
bucket: test

Example pillar

aptly:
  server:
    enabled: true
    repo:
      myrepo:
        distribution: trusty
        component: main
        architectures: amd64
        comment: "Custom components"
        sources: false
        publisher:
          component: mycomponent
          distributions:
            - nightly/trusty

Basic Aply server mirrors

aptly:
  server:
    mirror:
      mirror_name:
        source: http://example.com/debian
        distribution: xenial
        components: main
        architectures: amd64
        gpgkeys: 460F3999
        filter: "!(Name (% *-dbg))"
        filter_with_deps: true
        publisher:
          component: example
          distributions:
            - xenial/repo/nightly
            - "s3:aptcdn:xenial/repo/nightly"

Proxy environment variables (optional) in cron job for mirroring script

aptly:
  server:
    enabled: true
    ...
    mirror_update:
      enabled: true
      http_proxy: "http://1.2.3.4:8000"
      https_proxy: "http://1.2.3.4:8000"
    ...

Read more

- http://www.aptly.info/doc/configuration/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-aptly/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-aptly

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Artifactory

JFrog Artifactory is the only Universal Repository Manager supporting all major packaging formats, build tools and CI servers.

Sample pillars

Server

Single artifactory OSS edition from OS package

```yaml
artifactory:
  server:
    enabled: true
    edition: oss
    version: 4
    source:
      engine: pkg
```

Single artifactory pro edition from OS package

```yaml
artifactory:
  server:
    enabled: true
```

(continues on next page)
Single artifactory with PostgreSQL database

```yaml
artifactory:
  server:
    database:
      engine: postgresql
      host: localhost
      port: 5432
      name: artifactory
      user: artifactory
      password: pass
```

Client

Basic client setup

```yaml
artifactory:
  client:
    enabled: true
  server:
    host: 10.10.10.148
    port: 8081
    user: admin
    password: password
```

Artifactory repository definition

```yaml
artifactory:
  client:
    enabled: true
  repo:
    local_artifactory_repo:
      name: local_artifactory_repo
      package_type: docker
      repo_type: local
    remote_artifactory_repo:
      name: remote_artifactory_repo
      package_type: generic
      repo_type: remote
      url: "http://totheremoterepo:80/"
```

Repository configuration

Sample pillar above shows basic repository configuration, but you can use any parameters described in [https://www.jfrog.com/confluence/display/RTF/Repository+Configuration+JSON](https://www.jfrog.com/confluence/display/RTF/Repository+Configuration+JSON)

This module does direct map from pillar parameters to repository JSON description with two aliases for compatibility:

- `repo_type` -> `rclass`
• package_type -> packageType

Read more

• https://www.jfrog.com/confluence/display/RTF/Debian+Repositories
• https://www.jfrog.com/confluence/display/RTF/PostgreSQL
• https://www.jfrog.com/confluence/display/RTF/Artifactory+REST+API#ArtifactoryRESTAPI-REPOSITORIES
• https://www.jfrog.com/confluence/display/RTF/Repository+Configuration+JSON

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-artifactory/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-artifactory

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Gerrit

Gerrit provides web based code review and repository management for the Git version control system.

Sample pillars

Simple gerrit service

gerrit:
  server:
    enabled: true
    source:
      engine: http
    hash: 2e17064b8742c4622815593ec496c571

Full service setup

2.1. Project Introduction
**gerrit:**

server:
- canonical_web_url: http://10.10.10.148:8082/
- email_private_key: ""
- token_private_key: ""

initial_user:
- full_name: John Doe
- email: 'mail@jdoe.com'
- username: jdoe

plugin:
- download-commands:
  - engine: gerrit

replication:
  - engine: gerrit

reviewnotes:
- engine: gerrit

singleusergroup:
- engine: gerrit

ssh_rsa_key:
```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAi4OZfX5NWR4_bDQ8jGQ8vCVW3WE/F9iUJ3XzWmCfwm8rJmW01
j7OlE3d6GK8j7vZ1vRo6jZDZP07zR4Q6JnS7KZHhRQ2Ww1nM8o1cKu5T4cK5S

... ...

-----END RSA PRIVATE KEY-----
```

ssh_rsa_key_pub: ssh-rsa
```
AAAAB3NzaC1yc2EAAAADAQABAAABAQC9e1LCG7RJGlYiCB+yA4TQtHjWDf8o95

...

-----END RSA PRIVATE KEY-----
```

email: mail@domain.com

auth:
- engine: HTTP

source:
- engine: http
  address: https://gerrit-releases.storage.googleapis.com/gerrit-2.12.4.war
  hash: sha256=45786a920a929c6258de6461bcf03ddec8925577bd485905f102ceb6e5e1e47c
  receive_timeout: 5min

sshd:
- threads: 64
- batch_threads: 16
- max_connections_per_user: 64

database:
- engine: postgresql
- host: localhost
- port: 5432
- name: gerrit
- user: gerrit
- password: $(_param:postgresql_gerrit_password)
- pool_limit: 250
- pool_max_idle: 16

---

Gerrit change auto abandon

---

242 Chapter 2. Contents
Gerrit client enforcing groups

```yaml
gerrit:
  client:
    group:
      Admin001:
        description: admin 01
      Admin002:
        description: admin 02
```

Gerrit client enforcing users, install using pip

```yaml
gerrit:
  client:
    source:
      engine: pip
    user:
      jdoe:
        fullname: John Doe
        email: "jdoe@domain.com"
        ssh_key: ssh-rsa
        http_password: password
        groups:
          - Admin001
```

Gerrit client enforcing projects

```yaml
gerrit:
  client:
    enabled: True
  server:
    host: 10.10.10.148
    user: newt
    key: |
      -----BEGIN RSA PRIVATE KEY-----
      MIIEowIBAAKCAQEAs0Y8mxS3dfs5zG8Du5vdBkfOCOnglIEUmFZIrJ8oB0d54
      QgmkDFB7oP9eTCgz9k/rix1uJWihVCMBzrWZh51ODG+tyy/tK66pv2WtVfTDhBA
      ...
      l1UruxQkBgEkIBTuEiDRibKQXQbwIAYvK2Hve9hWpqpt9/DFe16s4AlbbTWbHyOP
      jvMxsm60ID/A5OpG33LWHNNzQBP486xG75LB+Xs5sp5j2/b7VF5LJLhpGjJv9Mk
      ydbuy8iuvvali2uF133kAlLqnrWfVTYQQ110fW5g1Ov1L6kv94dU
      -----END RSA PRIVATE KEY-----
    email: "Project Creator <infra@lists.domain.com>"
  project:
    test_salt_project:
      enabled: true
```

Gerrit client enforcing project, full project example

```yaml
gerrit:
  client:
    enabled: True
```

(continues on next page)
project:
  test_salt_project:
    enabled: true
    access:
      "refs/heads/**":
        actions:
          - name: abandon
            group: openstack-salt-core
          - name: create
            group: openstack-salt-release
        labels:
          - name: Code-Review
            group: openstack-salt-core
            score: -2..+2
          - name: Workflow
            group: openstack-salt-core
            score: -1..+1
      "refs/tags/**":
        actions:
          - name: pushSignedTag
            group: openstack-salt-release
    inherit_access: All-Projects
    require_change_id: true
    require_agreement: true
    merge_content: true
    action: "fast forward only"

gerrit:
  client:
    enabled: True
  group:
    groupname:
      enabled: true
      members:
        - username
  account:
    username:
      enabled: true
      full_name: hovno
      email: mail@newt.cz
      public_key: rsa
      http_password: passwd

Sample project access

[access "refs/*"]
  read = group Administrators
  read = group Anonymous Users
[access "refs/for/refs/*"]
  push = group Registered Users
  pushMerge = group Registered Users
[access "refs/heads/*"]
  create = group Administrators
  create = group Project Owners
  forgeAuthor = group Registered Users
  forgeCommitter = group Administrators

(continues on next page)
forgeCommitter = group Project Owners
push = group Administrators
push = group Project Owners
label-Code-Review = -2..+2 group Administrators
label-Code-Review = -2..+2 group Project Owners
label-Code-Review = -1..+1 group Registered Users
label-Verified = -1..+1 group Non-Interactive Users
submit = group Administrators
submit = group Project Owners
editTopicName = +force group Administrators
editTopicName = +force group Project Owners

[access "refs/meta/config"]
exclusiveGroupPermissions = read
read = group Administrators
read = group Project Owners
push = group Administrators
push = group Project Owners
label-Code-Review = -2..+2 group Administrators
label-Code-Review = -2..+2 group Project Owners
submit = group Administrators
submit = group Project Owners

[access "refs/tags/*"]
pushTag = group Administrators
pushTag = group Project Owners
pushSignedTag = group Administrators
pushSignedTag = group Project Owners

[label "Code-Review"]
function = MaxWithBlock
copyMinScore = true
value = -2 This shall not be merged
value = -1 I would prefer this is not merged as is
value = 0 No score
value = +1 Looks good to me, but someone else must approve
value = +2 Looks good to me, approved

[label "Verified"]
function = MaxWithBlock
copyMinScore = true
value = -1 Fails
value = 0 No score
value = +1 Verified

Read more

- https://www.gerritcodereview.com/
- https://gerrit-review.googlesource.com/Documentation/
- https://github.com/openstack-infra/puppet-gerrit/
- https://gerrit-ci.gerritforge.com/
- https://github.com/morucci/exzuul

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:
In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-gerrit/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-gerrit

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Gitlab formula

Gitlab is a free git repository management application based on Ruby on Rails. It is distributed under the MIT License and its source code can be found on Github. It is a very active project with a monthly release cycle and ideal for businesses that want to keep their code private. Consider it as a self hosted Github but open source.

Sample metadata

Server role

Gitlab server with local MTA and PostgreSQL database

```
gitlab:
  server:
    enabled: true
    server_name: 'repol.domain.com'
    source:
      engine: pkg
    database:
      engine: 'postgresql'
      host: 'locslhost'
      name: 'gitlab'
      password: 'LfTno5mYdZmRf0PV'
      user: 'gitlab'
    mail:
      engine: 'smtp'
      host: 'localhost'
      port: 25
      domain: 'domain.com'
      use_tls: false
      from: 'gitlab@domain.com'
      no_reply: 'no-reply@domain.com'
```

Gitlab server from custom source code repository
gitlab:
  server:
    enabled: true
    engine: git
    host: git://git.domain.com
    server_name: 'repo.domain.com'

Gitlab server with LDAP authentication

```yaml
gitlab:
  server:
    enabled: true
    version: '6.2'
    server_name: 'repol.domain.com'
  identity:
    engine: ldap
    host: lda.domain.com
    base: OU=ou,DC=domain,DC=com
    port: 389
    uid: sAMAccountName
    method: plain
    bind_dn: uid=ldap,ou=Users,dc=domain,dc=com
    password: pwd
```

Client role

Gitlab groups/namespaces

```yaml
gitlab:
  client:
    enabled: true
    server:
      url: http://repo.domain.com/
      token: fdsfdsfdsfdsfds
  group:
    hovno53:
      enabled: true
      description: some tex2
```

Gitlab repository enforcement with import url repository and deploy keys and hooks.

```yaml
gitlab:
  client:
    enabled: true
    server:
      url: http://repo.domain.com/
      token: fdsfdsfdsfdsfds
  repository:
    name-space/repo-name:
      enabled: true
      import_url: https://repo01.domain.com/namespace/repo.git
      description: Repo description
      deploy_key:
        keyname:
```

(continues on next page)
enabled: true
key: public_part_of_ssh_key
hook:
  hookname:
    enabled: true
    address: http://ci-tool/

More information

- https://wiki.archlinux.org/index.php/gitlab
- https://github.com/gitlabhq/gitlabhq/issues/6687

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-gitlab/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-gitlab

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Gource

OpenGL-based 3D visualisation tool for source control repositories. This formulas helps to generate video from multiple git repositories.
Sample pillars

Single gource service

```
gource:
  client:
    enabled: true
    workspace: /media/majklk/9ECC42B6CC42890B
  video:
    leonardo:
      resolution: 1920x1080
      convert: true
    source:
      core:
        address: https://github.com/django-leonardo/django-leonardo.git
      package_index:
        address: https://github.com/leonardo-modules/leonardo-package-index.git
    blog:
      address: 'git@repo1.robotice.cz:leonardo-modules/leonardo-module-blog.git'
```

Read more

- links

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-gource/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-gource

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Jenkins formula

Jenkins CI is an open source automation server written in Java. Jenkins helps to automate the non-human part of software development process, with continuous integration and facilitating technical aspects of continuous delivery.

(Source: Wikipedia)
More information can be found at https://jenkins.io/
Setup jenkins client, works with Salt 2016.3+, supports pipeline workflow projects only now.

Dependencies

To install on Ubuntu, you will need to add the jenkins debian repository to the target server. You can do this with the salt-formula-linux formula, with the following pillar data:

```yaml
linux:
  system:
    enabled: true
  repo:
    jenkins:
      enabled: true
      source: "deb http://pkg.jenkins.io/debian-stable binary/
      key_url: "https://pkg.jenkins.io/debian/jenkins-ci.org.key"
```

This state will need to be applied before the jenkins state.

Using this formula

To use this formula, you must install the formula to your salt master as documented in saltstack formula docs
This formula is driven by pillar data, and can be used to install either a Jenkins Master or Client. See pillar data below for examples.

Sample pillars

Master role

Simple master with reverse proxy

```yaml
nginx:
  server:
    site:
      jenkins:
        enabled: true
        type: nginx_proxy
        name: jenkins
        proxy:
          host: 127.0.0.1
          port: 8080
          protocol: http
          host:
            name: jenkins.example.com
            port: 80
      jenkins:
        master:
          mode: EXCLUSIVE
          # Do not manage config.xml from Salt, use UI instead
          no_config: true
        slaves:
          - name: slave01
```

(continues on next page)
label: pbuilder
executors: 2
- name: slave02
  label: image_builder
  mode: EXCLUSIVE
  executors: 2
views:
- name: "Package builds"
  regex: "debian-build-.*"
- name: "Contrail builds"
  regex: "contrail-build-.*"
- name: "Aptly"
  regex: "aptly-.*"
plugins:
- name: slack
- name: extended-choice-parameter
- name: rebuild
- name: test-stability

Jenkins master with experimental plugin source support

jenkins:
  master:
    enabled: true

SMTP server settings

jenkins:
  master:
    email:
      engine: "smtp"
      host: "smtp.domain.com"
      user: "user@domain.cz"
      password: "smtp-password"
      port: 25

Script approvals from client

jenkins:
  client:
    approved_scripts:
      - method groovy.json.JsonSlurperClassic.parseText java.lang.String

Script approvals

jenkins:
  master:
    approved_scripts:
      - method groovy.json.JsonSlurperClassic.parseText java.lang.String

User enforcement

jenkins:
  master:
    user:

(continues on next page)
Agent (slave) role

jenkins:
  slave:
    master:
      host: jenkins.example.com
      port: 80
      protocol: http
    user:
      name: jenkins_slave
      password: dexieeh6AepohthaiHook2iesh7015ook40v3leid3yek6daid2ooNg3Ee2oKeYo
  gpg:
    keypair_id: A76882D3
    public_key: |
      -----BEGIN PGP PUBLIC KEY BLOCK-----
      ...
    private_key: |
      -----BEGIN PGP PRIVATE KEY BLOCK-----
      ...

Client role

Simple client with workflow job definition

jenkins:
  client:
    master:
      host: jenkins.example.com
      port: 80
      protocol: http
    job:
      jobname:
        type: workflow
        param:
          bool_param:
            type: boolean
            description: true/false
            default: true
          string_param:
            type: string
            description: 1 liner
            default: default_string
          text_param:
            type: text
description: multi-liner
default: default_text

jobname_scm:
type: workflow-scm
concurrent: false

scm:
type: git
url: https://github.com/jenkinsci/docker.git
branch: master
script: Jenkinsfile
github:
  url: https://github.com/jenkinsci/docker
  name: "Jenkins Docker Image"

trigger:
timer:
  spec: "H H * * *"
github:
pollscm:
  spec: "H/15 * * * *
reverse:
  projects:
    - test1
    - test2
  state: SUCCESS

param:
  bool_param:
type: boolean
description: true/false
default: true

  string_param:
type: string
description: 1 liner
default: default_string

  text_param:
type: text
description: multi-liner
default: default_text

Inline Groovy scripts

jenkins:
  client:
  job:
    test_workflow_jenkins_simple:
      type: workflow
display_name: Test jenkins simple workflow
script:
  content: |
    node {
      stage 'Stage 1'
echo 'Hello World 1'
      stage 'Stage 2'
echo 'Hello World 2'
    }

test_workflow_jenkins_input:
  type: workflow
display_name: Test jenkins workflow inputs

(continues on next page)
script:
    content: |
    node {
        stage 'Enter string'
            input message: 'Enter job parameters', ok: 'OK', parameters: [ 
                string(defaultValue: 'default', description: 'Enter a string.', name: 'string'),
        ]
        stage 'Enter boolean'
            input message: 'Enter job parameters', ok: 'OK', parameters: [ 
                booleanParam(defaultValue: false, description: 'Select boolean.', name: 'Bool'),
        ]
        stage 'Enter text'
            input message: 'Enter job parameters', ok: 'OK', parameters: [ 
                text(defaultValue: '', description: 'Enter multiline', name: 'Multiline')
        ]
    }

GIT controlled groovy scripts

jenkins:
    client:
        source:
            base:
                engine: git
                    address: repo_url
                    branch: branch
            domain:
                engine: git
                    address: domain_url
                    branch: branch
    job:
        test_workflow_jenkins_simple:
            type: workflow
            display_name: Test jenkins simple workflow
            param:
                bool_param:
                    type: boolean
                    description: true/false
                    default: true
            script:
                repository: base
                    file: workflows/test_workflow_jenkins_simple.groovy
        test_workflow_jenkins_input:
            type: workflow
            display_name: Test jenkins workflow inputs
            script:
                repository: domain
                    file: workflows/test_workflow_jenkins_input.groovy
        test_workflow_jenkins_input_jenkinsfile:
            type: workflow
            display_name: Test jenkins workflow inputs (jenkinsfile)
            script:
                repository: domain
                    file: workflows/test_workflow_jenkins_input/Jenkinsfile
GIT controlled groovy script with shared libraries

```groovy
jenkins:
  client:
    source:
      base:
        engine: git
        address: repo_url
        branch: branch
      domain:
        engine: git
        address: domain_url
        branch: branch
    job:
      test_workflow_jenkins_simple:
        display_name: Test jenkins simple workflow
        param:
          bool_param:
            type: boolean
            description: true/false
            default: true
        script:
          repository: base
          file: workflows/test_workflow_jenkins_simple.groovy
        libs:
          - repository: base
            file: macros/cookiecutter.groovy
          - repository: base
            file: macros/git.groovy
```

Setting job max builds to keep (amount of last builds stored on Jenkins master)

```groovy
jenkins:
  client:
    job:
      my-amazing-job:
        type: workflow
        discard:
          build:
            keep_num: 5
            keep_days: 5
        artifact:
            keep_num: 6
            keep_days: 6
```

Using job templates in similar way as in jjb. For now just 1 defined param is supported.

```groovy
jenkins:
  client:
    job_template:
      test_workflow_template:
        name: test-{{formula}}-workflow
        template:
          type: workflow
        display_name: Test jenkins {{name}} workflow
        param:
          repo_param:
```

(continues on next page)
Interpolating parameters for job templates.

```yaml
_param:
  salt_formulas:
    - aodh
    - git
    - nova
    - xorg
jenkins:
  client:
    job_template:
      test_workflow_template:
        name: test-{{formula}}-workflow
        template:
          ...
        param:
          formula: ${_param:salt_formulas}

Or simply define multiple jobs and it’s parameters to replace from template:

```yaml
jenkins:
  client:
    job_template:
      test_workflow_template:
        name: test-{{name}}-{{myparam}}
        template:
          ...
        jobs:
          - name: firstjob
            myparam: dummy
          - name: secondjob
            myparam: dummyaswell

Purging undefined jobs from Jenkins

```yaml
jenkins:
  client:
    purge_jobs: true
    job:
      my-amazing-job:
        type: workflow

Plugins management from client

```yaml
jenkins:
  client:

(continues on next page)
Adding plugin params to job

```
jenkins:
  client:
    job:
      my_plugin_parametrized_job:
        plugin_properties:
          throttleconcurrents:
            enabled: True
            max_concurrent_per_node: 3
            max_concurrent_total: 1
            throttle_option: category #one of project (default or category)
            categories:
              - my_throttle_category

  plugin:
    swarm:
      restart: false
    hipchat:
      enabled: false
      restart: true
```

LDAP configuration (depends on LDAP plugin)

```
jenkins:
  client:
    security:
      ldap:
        server: 1.2.3.4
        root_dn: dc=foo,dc=com
        user_search_base: cn=users,cn=accounts
        manager_dn: ""
        manager_password: password
        user_search: ""
        group_search_base: ""
        inhibit_infer_root_dn: false
```

Matrix configuration (depends on auth-matrix plugin)

```
jenkins:
  client:
    security:
      matrix:
        # set true for use ProjectMatrixAuthStrategy instead of GlobalMatrixAuthStrategy
        project_based: false
        permissions:
          Jenkins:
            # administrator access
            ADMINISTER:
              - admin
```

2.1. Project Introduction
# read access (anonymous too)
READ:
  - anonymous
  - user1
  - user2

# agents permissions
MasterComputer:
  BUILD:
    - user3

# jobs permissions
hudson:
  model:
    Item:
      BUILD:
        - user4

Common matrix strategies

Views enforcing from client

jenkins:
  client:
    view:
      my-list-view:
        enabled: true
        type: ListView
        include_regex: ".*"
      my-view:
        # set false to disable
        enabled: true
        type: MyView

View specific params:

- include_regex for ListView and CategorizedJobsView
- categories for CategorizedJobsView

Categorized views

jenkins:
  client:
    view:
      my-categorized-view:
        enabled: true
        type: CategorizedJobsView
        include_regex: ".*"
        categories:
          - group_regex: "aptly-.*-nightly-testing"
            naming_rule: "Nightly -> Testing"
          - group_regex: "aptly-.*-nightly-production"
            naming_rule: "Nightly -> Production"

Credentials enforcing from client

jenkins:
  client:
    credential:
### Users enforcing from client

```yaml
jenkins:
  client:
    user:
      admin:
        password: admin_password
        admin: true
      user01:
        password: user_password
```

### Node enforcing from client using JNLP launcher

```yaml
jenkins:
  client:
    node:
      node01:
        remote_home: /remote/home/path
        desc: node-description
        num_executors: 1
        node_mode: Normal
        ret_strategy: Always
        labels:
          - example
          - label
        launcher:
          type: jnlp
```

### Node enforcing from client using SSH launcher

```yaml
jenkins:
  client:
    node:
      node01:
        remote_home: /remote/home/path
        desc: node-description
        num_executors: 1
        node_mode: Normal
        ret_strategy: Always
        labels:
          - example
          - label
        launcher:
          type: ssh
          host: test-launcher
          port: 22
```

(continues on next page)
Configure Jenkins master

```yaml
jenkins:
  client:
    node:
      master:
        num_executors: 1
        node_mode: Normal # or Exclusive
      labels:
        - example
        - label
```

Setting node labels

```yaml
jenkins:
  client:
    label:
      node-name:
        lbl_text: label-offline
        append: false # set true for label append instead of replace
```

SMTP server settings from client

```yaml
jenkins:
  client:
    smtp:
      host: "smtp.domain.com"
      username: "user@domain.cz"
      password: "smtp-password"
      port: 25
      ssl: false
      reply_to: reply_to@address.com
```

Jenkins admin user email enforcement from client

```yaml
jenkins:
  client:
    smtp:
      admin_email: "My Jenkins <jenkins@myserver.com>"
```

Slack plugin configuration

```yaml
jenkins:
  client:
    slack:
      team_domain: example.com
      token: slack-token
      room: slack-room
      token_credential_id: cred_id
      send_as: Some slack user
```

Pipeline global libraries setup
jenkins:
  client:
    lib:
      my-pipeline-library:
        enabled: true
        url: https://path-to-my-library
        credential_id: github
        branch: master # optional, default master
        implicit: true # optional default true

Artifactory server enforcing

jenkins:
  client:
    artifactory:
      my-artifactory-server:
        enabled: true
        url: https://path-to-my-library
        credential_id: github

Jenkins Global env properties enforcing

Usage

Generate password hash:

```bash
$ echo -n "salt{plainpassword}" | openssl dgst -sha256
```

Place in the configuration `salt:hashpassword`.

External links

- [https://wiki.jenkins-ci.org/display/JENKINS/Use+Jenkins](https://wiki.jenkins-ci.org/display/JENKINS/Use+Jenkins)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

- [http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

- [https://github.com/salt-formulas/salt-formula-jenkins/issues](https://github.com/salt-formulas/salt-formula-jenkins/issues)
For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-jenkins

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

owncloud

Install and configure owncloud.

Available states

- owncloud.server

owncloud.server

Setup owncloud server

Available metadata

- metadata.owncloud.server

metadata.owncloud.server

Setup owncloud server

Requirements

- linux
- mysql (for mysql backend)
- apache
Optional

- memcached
- glusterfs (for clustered setup)

Configuration parameters

For complete list of parameters, please check metadata/service/server.yml

Example reclass

```yaml
classes:
- system.linux.system.single
- service.memcached.server.local
- service.apache.server.single
- service.mysql.server.single
- service.owncloud.server
params:
  salt_master_host: ${_param:reclass_config_master}
  mysql_admin_user: root
  mysql_admin_password: cloudlab
parameters:
  owncloud:
    server:
      version: 8.1.5.2
      # pwgen -A 12 | head -1
      instanceid: iy5opia6chaae
      # pwgen 31 | head -1
      passwordsalt: Een7riefohSahchaigh9ohcho6xoFe
      # pwgen -y 49 | head -1
      secret: |
        "guth9keelfe9hoo\g6oowei6er9aigohK=ieM4uvojaicha4a"
    url: "http://owncloud.lxc.eu"
    trusted_domains:
    - owncloud.lxc.eu
    mail:
      domain: lxc.eu
    database:
      password: eikaithiuka2iex1ChieYaGeiguqu0iw
    cache:
      enabled: true
    servers:
    - address: localhost
    admin:
      username: admin
      password: cloudlab
    users:
    - test:
      enabled: true
      group: Users
      password: test
      name: Test user
    appstore:
      experimental: true
(continues on next page)
mysql:
    server:
    ssl:
        enabled: false
    database:
        owncloud:
            encoding: UTF8
            locale: cs_CZ
            users:
                - name: owncloud
                  password: eikaithiu2aix1ChieYaGeiguqu0iw
                  host: localhost
                  rights: all privileges
apache:
    server:
    site:
        owncloud:
            enabled: true
            type: owncloud
            name: owncloud
            host:
                Name: owncloud.lxc.eru

Read more

- https://doc.owncloud.org/
- http://sabre.io/dav/service-discovery/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-owncloud/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-owncloud

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Packer

Packer is an open source tool for creating identical machine images for multiple platforms from a single source configuration. Packer is lightweight, runs on every major operating system, and is highly performant, creating machine images for multiple platforms in parallel.

Sample pillar

Basic linux distros

```yaml
packer:
  build:
    system:
      ubuntu:
        source:
          engine: git
          address: https://github.com/boxcutter/ubuntu.git
          revision: master
        template:
          ubuntu1404-salt:
            file: ubuntu1404.json
            provisioner: salt
            builders:
              - vmware
              - virtualbox
          ubuntu1504-desktop-salt:
            file: ubuntu1504-desktop.json
            provisioner: salt
            builders:
              - vmware
              - virtualbox
```

Usage

Openstack image prepare guide

- Install cloud-init - add epel - package epel-centos 6, yum cloud-init
- Set network to DHCP
- `/etc/udev.rules/70netrules` - remove MAC address records

Read more

- [http://www.packer.io/docs/installation.html](http://www.packer.io/docs/installation.html)
- [https://github.com/mitchellh/packer-ubuntu-12.04-docker](https://github.com/mitchellh/packer-ubuntu-12.04-docker)
- [https://github.com/boxcutter](https://github.com/boxcutter)
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-packer/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-packer

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

roundcube

Install and configure roundcube.

Available states

- roundcube.server

roundcube.server

Setup roundcube server

Available metadata

- metadata.roundcube.server

metadata.roundcube.server

Setup roundcube server
**Requirements**

- linux
- mysql (for mysql backend)
- dovecot

**Configuration parameters**

For complete list of parameters, please check metadata/service/server.yml

**Example reclass**

**Server**

```yaml
classes:
  - service.roundcube.server
parameters:
  _param:
  postfix_origin: mail.eru
  mysql_roundcube_password: changeme
roundcube:
  force_https: false
mail:
  host: ${_param:postfix_origin}
mysql:
  server:
    database:
      roundcube:
        encoding: UTF8
        locale: cs_CZ
        users:
          - name: roundcube
            password: ${_param:mysql_roundcube_password}
            host: 127.0.0.1
            rights: all privileges
apache:
  server:
    site:
      roundcube:
        enabled: true
        type: static
        name: roundcube
        root: /usr/share/roundcube
        host:
          name: ${_param:postfix_origin}
          aliases:
          - ${linux:system:name}.${linux:system:domain}
          - ${linux:system:name}
```
Example pillar

Server

```python
roundcube:
    server:
        mail:
            host: mail.cloudlab.cz
        session:
            # 24 random characters
            des_key: 'Ckhuv6W6iUdbxpvKzhbepk'
            # 30 minutes
            lifetime: 30
        plugins:
            - archive
            - zipdownload
            - newmail_notifier
```

Read more

- https://roundcube.net/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-roundcube/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-roundcube

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Monitoring Services

Monitoring, metering and log collecting tools implementing complete monitoring stack.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>collectd</td>
<td><a href="https://github.com/salt-formulas/salt-formula-collectd">https://github.com/salt-formulas/salt-formula-collectd</a></td>
</tr>
<tr>
<td>fluentd</td>
<td><a href="https://github.com/salt-formulas/salt-formula-fluentd">https://github.com/salt-formulas/salt-formula-fluentd</a></td>
</tr>
<tr>
<td>grafana</td>
<td><a href="https://github.com/salt-formulas/salt-formula-grafana">https://github.com/salt-formulas/salt-formula-grafana</a></td>
</tr>
<tr>
<td>graphite</td>
<td><a href="https://github.com/salt-formulas/salt-formula-graphite">https://github.com/salt-formulas/salt-formula-graphite</a></td>
</tr>
<tr>
<td>heka</td>
<td><a href="https://github.com/salt-formulas/salt-formula-heka">https://github.com/salt-formulas/salt-formula-heka</a></td>
</tr>
<tr>
<td>influxdb</td>
<td><a href="https://github.com/salt-formulas/salt-formula-influxdb">https://github.com/salt-formulas/salt-formula-influxdb</a></td>
</tr>
<tr>
<td>kedb</td>
<td><a href="https://github.com/salt-formulas/salt-formula-kedb">https://github.com/salt-formulas/salt-formula-kedb</a></td>
</tr>
<tr>
<td>kibana</td>
<td><a href="https://github.com/salt-formulas/salt-formula-kibana">https://github.com/salt-formulas/salt-formula-kibana</a></td>
</tr>
<tr>
<td>nagios</td>
<td><a href="https://github.com/salt-formulas/salt-formula-nagios">https://github.com/salt-formulas/salt-formula-nagios</a></td>
</tr>
<tr>
<td>rsyslog</td>
<td><a href="https://github.com/salt-formulas/salt-formula-rsyslog">https://github.com/salt-formulas/salt-formula-rsyslog</a></td>
</tr>
<tr>
<td>sensu</td>
<td><a href="https://github.com/salt-formulas/salt-formula-sensu">https://github.com/salt-formulas/salt-formula-sensu</a></td>
</tr>
<tr>
<td>statsd</td>
<td><a href="https://github.com/salt-formulas/salt-formula-statsd">https://github.com/salt-formulas/salt-formula-statsd</a></td>
</tr>
</tbody>
</table>

Collectd formula

Collectd is a daemon which collects system performance statistics periodically and provides mechanisms to store the values in a variety of ways, for example in RRD files.

Sample pillars

Data writers

Send data over TCP to Graphite Carbon

collectd:
  client:
    enabled: true
    read_interval: 60
  backend:
    carbon_service:
      engine: carbon
      host: carbon1.comain.com
      port: 2003

Send data over AMQP

collectd:
  client:
    enabled: true
    read_interval: 60
  backend:
    amqp_broker:
      engine: amqp
      host: broker1.comain.com
      port: 5672
      user: monitor

(continues on next page)
password: amqp-pwd
virtual_host: '/monitor'

Send data over HTTP

```yaml
collectd:
  client:
    enabled: true
    read_interval: 60
  backend:
    http_service:
      engine: http
      host: service.comain.com
      port: 8123
```

Data collectors

Monitor network devices, defined in `external` dictionary

```yaml
external:
  network_device:
    MX80-01:
      community: test
      model: Juniper_MX80
    management:
      address: 10.0.0.254
      port: fxp01
      engine: snmp/ssh
    interface:
      xe-0/0/0:
        description: MEMBER-OF-LACP-TO-QFX
        type: 802.3ad
        subinterface:
          xe-0/0/0.0:
            description: MEMBER-OF-LACP-TO-QFX
  collectd:
    client:
      enabled: true
      ...
```

Collecting the SNMP metrics

```yaml
collectd:
  client:
    data:
      connected_devices:
        type: devices
        values:
        - IF-MIB::ifNumber.0
  host:
    ubiquity:
      address: 10.0.0.1
      community: public
      version: 2
```

(continues on next page)
Collecting the cURL response times and codes

```yaml
collectd:
  client:
    check:
      curl:
        service1:
          url: "https://service.domain.com:443/
        service2:
          url: "https://service.domain.com:443/
```

Collecting the ping response times

```yaml
collectd:
  client:
    check:
      ping:
        host_label1:
          host: "172.10.31.14"
        host_label2:
          host: "172.10.31.12"
```

External links

- [http://collectd.org/documentation.shtml](http://collectd.org/documentation.shtml)
- [http://www.canopsis.org/2013/02/collectd-graphite/](http://www.canopsis.org/2013/02/collectd-graphite/)
- [http://collectd.org/documentation/manpages/collectd.conf.5.shtml#plugin_libvirt](http://collectd.org/documentation/manpages/collectd.conf.5.shtml#plugin_libvirt)
- [http://libvirt.org/uri.html#URI_qemu](http://libvirt.org/uri.html#URI_qemu)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

[https://github.com/salt-formulas/salt-formula-collectd/issues](https://github.com/salt-formulas/salt-formula-collectd/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

[https://github.com/salt-formulas/salt-formula-collectd](https://github.com/salt-formulas/salt-formula-collectd)
Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Fluentd Formula**

Many web/mobile applications generate huge amount of event logs (c.f. login, logout, purchase, follow, etc). Analyzing these event logs can be quite valuable for improving services. However, collecting these logs easily and reliably is a challenging task.

Fluentd solves the problem by having: easy installation, small footprint, plugins reliable buffering, log forwarding, etc.

**NOTE: WORK IN PROGRESS**
**NOTE: DESIGN OF THIS FORMULA IS NOT YET STABLE AND MAY CHANGE**
**NOTE: FORMULA NOT COMPATIBLE WITH OLD VERSION**

**Sample Pillars**

**General pillar structure**

```yaml
fluentd:
  config:
    label:
      filename:
        input:
          input_name:
            params
        filter:
          filter_name:
            params
          filter_name2:
            params
        match:
          match_name:
            params
      input:
        filename:
          input_name:
            params
          input_name2:
            params
        filename2:
          input_name3:
            params
      filter:
        filename:
          filter_name:
            params
          filter_name2:
            params
        filename2:
          filter_name3:
            params
      match:
        filename:
          ... (continues on next page)
```
Example pillar

```yaml
fluentd:
   enabled: true
config:
   monitoring:
      filter:
         parse_log:
            tag: 'docker.monitoring.{alertmanager,remote_storage_adapter,prometheus}.*
                  →'
            type: parser
            reserve_data: true
            key_name: log
            parser:
               type: regexp
               format: >-
                  /^time="(?<time>[^ "]*)" level=(?<severity>[a-zA-Z]*)(?<message>.+)
               →+?"/
               time_format: '%FT%TZ'
         remove_log_key:
            tag: 'docker.monitoring.{alertmanager,remote_storage_adapter,prometheus}.*
                  →'
            type: record_transformer
            remove_keys: log
match:
   docker_log:
      tag: 'docker.*'
      type: file
      path: /tmp/flow-docker.log
grok_example:
input:
   test_log:
      type: tail
      path: /var/log/test
      tag: test.test
      parser:
         type: grok
         custom_pattern_path: /etc/td-agent/config.d/global.grok
         rule:
            - pattern: >-
               %{KEYSTONEACCESS}
syslog:
filter:
   add_severity:
      tag: 'syslog.*'
      type: record_transformer
      enable_ruby: true
      record:
         - name: severity
            value: 'record["pri"][0] - (record["pri"][0] / 8).floor * 8'
```
severity_to_string:
tag: 'syslog.*'
type: record_transformer
enable_ruby: true
record:
  - name: severity
    value: "({'debug'=>7,"info'=>6,"notice'=>5,"warning'=>4,"error'=>3,
      "critical'=>2,"alert'=>1,"emerg'=>0}.key(record['severity']))'

severity_for_telegraf:
tag: 'syslog.*.telegraf'
type: parser
reserve_data: true
key_name: message
parser:
  type: regexp
  format: >-
    /^\<(time>[^ \]*\)(?<severity>[A-Z])! (?<message>.*)/
  time_format: '%FT%TZ'

severity_for_telegraf_string:
tag: 'syslog.*.telegraf'
type: record_transformer
enable_ruby: true
record:
  - name: severity
    value: "({'debug'=>'D","info'=>'I","notice'=>'N","warning'=>'W","error
      =>'E","critical'=>'C","alert'=>'A","emerg'=>'E'}.key(record['severity']))'

prometheus_metric:
tag: 'syslog.*.*'
type: prometheus
label:
  - name: ident
    type: variable
    value: ident
  - name: severity
    type: variable
    value: severity
metric:
  - name: log_messages
    type: counter
    desc: The total number of log messages.

match:
  rewrite_tag_key:
tag: 'syslog.*'
type: rewrite_tag_filter
rule:
  - name: ident
    regexp: '^(.*$)
    result: '__TAG__.$1'

syslog_log:
tag: 'syslog.*.*'
type: file
path: /tmp/syslog
input:
  syslog:
    syslog_log:
      type: tail
      label: syslog

(continues on next page)
path: /var/log/syslog
tag: syslog.syslog
parser:
type: regexp
format: >-
  '/\<(?<pri>[0-9]+)\>(?<time>[^ \*]) (?<host>[^ \*]) (?<ident>[a-zA-Z0-9_/\.-]*)\{(?:\[(?<pid>[0-9]+)\])?(?:[^\:*]::)? *(?<message>.*$)'/
time_format: '%FT%T.%L%:z'
auth_log:
type: tail
label: syslog
path: /var/log/auth.log
tag: syslog.auth
parser:
type: regexp
format: >-
  '/\<(?<pri>[0-9]+)\>(?<time>[^ \*]) (?<host>[^ \*]) (?<ident>[a-zA-Z0-9_/\.-]*)\{(?:\[(?<pid>[0-9]+)\])?(?:[^\:*]::)? *(?<message>.*$)'/
time_format: '%FT%T.%L%:z'
prometheus:
prometheus:
prometheus_monitor:
type: prometheus_monitor
prometheus_output_monitor:
type: prometheus_output_monitor
forward:
forward_listen:
type: forward
port: 24224
bind: 0.0.0.0
match:
docker_monitoring:
docker_monitoring:
tag: 'docker.monitoring.*'(alertmanager,remote_storage_adapter,prometheus).*'
type: relabel
label: monitoring

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-nova/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

2.1. Project Introduction
Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-nova

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Grafana**

A beautiful, easy to use and feature rich Graphite dashboard replacement and graph editor.

**Sample pillars**

**Server deployments**

Server installed from system package and listening on 1.2.3.4:3000 (the default is 0.0.0.0:3000)

```yaml
grafana:
  server:
    enabled: true
    bind:
      address: 1.2.3.4
      port: 3000
    admin:
      user: admin
      password: passwd
    database:
      engine: sqlite
```

Server installed with PostgreSQL database

```yaml
grafana:
  server:
    enabled: true
    admin:
      user: admin
      password: passwd
    database:
      engine: postgresql
      host: localhost
      port: 5432
      name: grafana
      user: grafana
      password: passwd
```

Server installed with LDAP authentication and all authenticated users are administrators

```yaml
grafana:
  server:
    enabled: true
    admin:
      user: admin
      password: passwd
```

(continues on next page)
Server installed with LDAP and basic authentication

```
grafana:
  server:
    enabled: true
  admin:
    user: admin
    password: passwd
  auth:
    basic:
      enabled: true
    ldap:
      enabled: true
      host: '127.0.0.1'
      port: 389
      use_ssl: false
      bind_dn: "cn=admin,dc=grafana,dc=org"
      bind_password: "grafana"
      user_search_filter: "(cn=%s)"
      user_search_base_dns:
        - "dc=grafana,dc=org"
```

Server installed with LDAP for authentication and authorization

```
grafana:
  server:
    enabled: true
  admin:
    user: admin
    password: passwd
  auth:
    ldap:
      enabled: true
      host: '127.0.0.1'
      port: 389
      use_ssl: false
      bind_dn: "cn=admin,dc=grafana,dc=org"
      bind_password: "grafana"
      user_search_filter: "(cn=%s)"
      user_search_base_dns:
        - "dc=grafana,dc=org"
      group_search_filter: "(&(objectClass=posixGroup)(memberUid=%s))"
      group_search_base_dns:
        - "ou=groups,dc=grafana,dc=org"
```
enabled: true
admin_group: "admins"
editor_group: "editors"
viewer_group: "viewers"

Server installed with default StackLight JSON dashboards. This will be replaced by the possibility for a service to provide its own dashboard using salt-mine.

```
grafana:
  server:
    enabled: true
    dashboards:
      enabled: true
      path: /var/lib/grafana/dashboards
```

Server with theme overrides

```
grafana:
  server:
    enabled: true
    theme:
      light:
        css_override:
          source: http://path.to.theme
          source_hash: sha256=xyz
          build: xyz
      dark:
        css_override:
          source: salt://path.to.theme
```

Server with two additionals plugins. It requires to have access to the Internet.

```
grafana:
  server:
    enabled: true
    plugins:
      grafana-piechart-panel:
        enabled: true
      grafana-example-app:
        enabled: true
```

**Collector setup**

Used to aggregate dashboards from monitoring node.

```
grafana:
  collector:
    enabled: true
```

**Client setups**

Client with token based auth
grafana:
  client:
    enabled: true
    server:
      protocol: https
      host: grafana.host
      port: 3000
      token: token

Client with base auth

grafana:
  client:
    enabled: true
    server:
      protocol: https
      host: grafana.host
      port: 3000
      user: admin
      password: password

Client enforcing graphite data source

grafana:
  client:
    enabled: true
    datasource:
      graphite:
        type: graphite
        host: mtr01.domain.com
        protocol: https
        port: 443

Client enforcing elasticsearch data source

grafana:
  client:
    enabled: true
    datasource:
      elasticsearch:
        type: elasticsearch
        host: log01.domain.com
        port: 80
        index: grafana-dash

Client defined and enforced dashboard

grafana:
  client:
    enabled: true
    server:
      host: grafana.host
      port: 3000
      token: token
    dashboard:
      system_metrics:
        title: "Generic system metrics"
Client enforced dashboards defined in salt-mine

```json
grafana:
    client:
        enabled: true
        remote_data:
            engine: salt_mine
            server:
                host: grafana.host
                port: 3000
                token: token
```

**Usage**

There’s a difference between JSON dashboard representation and models we use. The lists used in JSON format [for rows, panels and target] were replaced by dictionaries. This form of serialization allows better merging and overrides of hierarchical data structures that dashboard models are.

The default format of Grafana dashboards with lists for rows, panels and targets.

```json
system_metrics:
    title: graph
    editable: true
    hideControls: false
    rows:
        - title: Usage
          height: 250px
          panels:
            - title: Panel Title
              span: 6
              editable: false
              type: graph
              targets:
                - refId: A
                  target: "support_prd.cfg01_iot_tcpcloud_eu.cpu.0.idle"
                  datasource: graphite01
                  renderer: flot
                  showTitle: true
```

The modified version of Grafana dashboard format with dictionary declarations. Please note that dictionary keys are only for logical separation and are not displayed in generated dashboards.

```json
system_metrics:
    system_metrics2:
        title: graph
        editable: true
        hideControls: false
        row:
            usage:
```

(continues on next page)
Read more

- http://grafana.org/
- http://docs.grafana.org/reference/export_import/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-grafana/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-grafana

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Graphite

Graphite is an enterprise-scale monitoring tool that runs well on cheap hardware.
Sample pillars

Single Graphite web server

```yaml
graphite:
  server:
    enabled: true
    debug: true
    timezone: 'Europe/Prague'
  cache:
    engine: 'memcached'
    host: '127.0.0.1'
    prefix: 'GRAPHITE'
  database:
    engine: 'postgresql'
    host: '127.0.0.1'
    name: 'graphite'
    password: 'password'
    user: 'graphite'
  mail:
    host: mail1.domain.com
    password: pwd
    user: username
```

Graphite web server cluster

```yaml
graphite:
  server:
    enabled: true
    timezone: 'Europe/Prague'
  database:
  mail:
  carbon_links:
  - host: 10.0.0.1
    port: 7002
  - host: 10.0.0.2
    port: 7002
  cache:
    engine: 'memcached'
  members:
  - host: 10.0.0.1
    port: 11211
  - host: 10.0.0.2
    port: 11211
```

Complete single Carbon collector

```yaml
carbon:
  relay:
    enabled: true
    method: consistent-hashing
  aggregator:
    enabled: false
  cache:
    storage_schema:
      default:
        pattern: '.*'
```

(continues on next page)
Clustered Carbon with AMQP and aggregation

carbon:
  relay:
    enabled: true
    method: rules
    message_queue:
      host: broker1.domain.com
      port: 5672
      user: monitor
      password: pwd
      virtual_host: '/monitor'
      exchange: 'metrics'
    destinations:
      - host: 10.0.0.1
        port: 2024
      - host: 10.0.0.2
        port: 2024
  aggregator:
    enabled: true
    destinations:
      - host: 10.0.0.1
        port: 2004
      - host: 10.0.0.2
        port: 2004
  cache:
    storage_schema:
      default:
        pattern: '.*'
        retentions:
          - 60s:1d
          - 600s:90d

Read more

http://graphite.readthedocs.org/en/latest/
http://www.canopsis.org/2013/02/collectd-graphite/
https://github.com/obfuscurity/graphite-scripts/blob/master/init.d/carbon-relay

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-graphite/issues
For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-graphite

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

### Heka Formula

Heka is an open source stream processing software system developed by Mozilla. Heka is a Swiss Army Knife type tool for data processing.

#### Sample pillars

**Log collector service**

```yaml
heka:
  log_collector:
    automatic_starting: true
    elasticsearch_host: 172.16.10.253
    elasticsearch_port: 9200
    enabled: true
    metric_collector_host: 127.0.0.1
    metric_collector_port: 5567
    poolsize: 100
    max_message_size: 262144
```

Default values:

- automatic_starting: true
- elasticsearch_port: 9200
- enabled: false
- metric_collector_host: 127.0.0.1
- metric_collector_port: 5567
- poolsize: 100
- max_message_size: 262144

**Local Metric collector service**

```yaml
heka:
  metric_collector:
    aggregator_host: 172.16.20.253
    aggregator_port: 5565
    automatic_starting: true
```
enabled: true
influxdb_database: lma
influxdb_host: 172.16.10.101
influxdb_password: lmapass
influxdb_port: 8086
influxdb_time_precision: ms
influxdb_timeout: 500
influxdb_username: lma
nagios_host: 172.16.20.253
nagios_host_dimension_key: nagios_host
nagios_password: secret
nagios_port: 5601
nagios_username: nagiosadmin
poolsize: 100
max_message_size: 262144

Default values:

- aggregator_port: 5565
- automatic_starting: true
- enabled: false
- influxdb_port: 8086
- influxdb_time_precision: ms
- influxdb_timeout: 5000
- nagios_port: 8001
- poolsize: 100
- max_message_size: 262144

Remote Metric Collector service

heka:
  remote_collector:
    aggregator_host: 172.16.20.253
    aggregator_port: 5565
    amqp_exchange: nova
    amqp_host: 172.16.10.254
    amqp_password: workshop
    amqp_port: 5672
    amqp_user: openstack
    amqp_vhost: /openstack
    automatic_starting: false
    elasticsearch_host: 172.16.10.253
    elasticsearch_port: 9200
    enabled: true
    influxdb_database: lma
    influxdb_host: 172.16.10.101
    influxdb_password: lmapass
    influxdb_port: 8086
    influxdb_time_precision: ms
    influxdb_username: lma
    poolsize: 100
    max_message_size: 262144

2.1. Project Introduction 285
Default values:

- aggregator_port: 5565
- amqp_exchange: nova
- amqp_port: 5672
- amqp_vhost: ''
- automatic_starting: true
- elasticsearch_port: 9200
- enabled: false
- influxdb_port: 8086
- influxdb_time_precision: ms
- influxdb_timeout: 5000
- poolsize: 100
- max_message_size: 262144

**Aggregator service**

```
heka:
  aggregator:
    automatic_starting: false
    enabled: true
    influxdb_database: lma
    influxdb_host: 172.16.10.101
    influxdb_password: imapass
    influxdb_port: 8086
    influxdb_time_precision: ms
    influxdb_username: lma
    nagios_default_host_alarm_clusters: 00-clusters
    nagios_host: 172.16.20.253
    nagios_host_dimension_key: nagios_host
    nagios_password: secret
    nagios_port: 5601
    nagios_username: nagiosadmin
    poolsize: 100
    max_message_size: 262144
```

Default values:

- automatic_starting: true
- enabled: false
- influxdb_port: 8086
- influxdb_time_precision: ms
- influxdb_timeout: 5000
- nagios_port: 8001
- nagios_default_host_alarm_clusters: 00-clusters
- poolsize: 100
- max_message_size: 262144
Ceilometer service

```yaml
heka:
  ceilometer_collector:
    elasticsearch_host: 172.16.10.253
    elasticsearch_port: 9200
    enabled: true
    influxdb_database: lma
    influxdb_host: 172.16.10.101
    influxdb_password: imapass
    influxdb_port: 8086
    influxdb_time_precision: ms
    influxdb_username: lma
    resource_decoding: false
    amqp_exchange: ceilometer
    amqp_host: 172.16.10.253
    amqp_port: 5672
    amqp_queue: metering.sample
    amqp_vhost: /openstack
```

Default values:

- automatic_starting: true
- elasticsearch_port: 9200
- enabled: false
- influxdb_port: 8086
- influxdb_time_precision: ms
- influxdb_timeout: 5000
- poolsize: 100
- amqp_exchange: ceilometer
- amqp_port: 5672
- amqp_queue: metering.sample
- amqp_vhost: /openstack
- resource_decoding: false

Read more


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-heka/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:
https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-heka

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**InfluxDB**

InfluxData is based on the TICK stack, the first open source platform for managing IoT time-series data at scale.

**Sample pillars**

Single-node influxdb, enabled http frontend and admin web interface:

```yaml
influxdb:
  server:
    enabled: true
  http:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8086
  admin:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8083
```

Single-node influxdb, SSL for http frontend:

```yaml
influxdb:
  server:
    enabled: true
  http:
    bind:
      ssl:
        enabled: true
        key_file: /etc/influxdb/ssl/key.pem
        cert_file: /etc/influxdb/ssl/cert.pem
```

Single-node influxdb where you specify paths for data and metastore directories. Custom directories are created by this formula:

```yaml
influxdb:
  server:
    enabled: true
  data:
    dir: '/opt/influxdb/data'
```

(continues on next page)
InfluxDB server with customized parameters for the data service:

```
influxdb:
  server:
    enabled: true
    data:
      max_series_per_database: 20000000
      cache_max_memory_size: 524288000
      cache_snapshot_memory_size: 26214400
      cache_snapshot_write_cold_duration: "5m"
      compact_full_write_cold_duration: "2h"
      max_values_per_tag: 5000
```

Single-node influxdb with an admin user:

```
influxdb:
  server:
    enabled: true
  http:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8086
  admin:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8083
  user:
    enabled: true
    name: root
    password: secret
```

Single-node influxdb with new users:

```
influxdb:
  server:
    user:
      user1:
        enabled: true
        admin: true
        name: username1
        password: keepsecret1
      user2:
        enabled: true
        admin: false
        name: username2
        password: keepsecret2
```

Single-node influxdb with new databases:

```
influxdb:
  server:
```
Manage the retention policies for a database:

```yaml
influxdb:
  server:
    database:
      mydb1:
        enabled: true
        name: mydb1
      mydb2:
        enabled: true
        name: mydb2
      retention_policy:
        - name: rp_db1
          duration: 30d
          replication: 1
          is_default: true
```

Where default values are:
- name = autogen
- duration = INF
- replication = 1
- is_default: false

Here is how to manage grants on database:

```yaml
influxdb:
  server:
    grant:
      username1_mydb1:
        enabled: true
        user: username1
        database: mydb1
        privilege: all
      username2_mydb1:
        enabled: true
        user: username2
        database: mydb1
        privilege: read
      username2_mydb2:
        enabled: true
        user: username2
        database: mydb2
        privilege: write
```

InfluxDB relay:

```yaml
influxdb:
  server:
    enabled: true
```
[continued from previous page]

```yaml
http:
  enabled: true
output:
  idb01:
    location: http://idb01.local:8086/write
    timeout: 10
  idb02:
    location: http://idb02.local:8086/write
    timeout: 10
udp:
  enabled: true
output:
  idb01:
    location: idb01.local:9096
  idb02:
    location: idb02.local:9096
```

InfluxDB cluster:

```yaml
influxdb:
  server:
    enabled: true
  meta:
    bind:
      address: 0.0.0.0
      port: 8088
      http_address: 0.0.0.0
      http_port: 8091
  cluster:
    members:
      - host: idb01.local
        port: 8091
      - host: idb02.local
        port: 8091
      - host: idb03.local
        port: 8091
```

Deploy influxdb apt repository (using linux formula):

```yaml
linux:
  system:
    os: ubuntu
    dist: xenial
    repo:
      influxdb:
        enabled: true
        key_url: 'https://repos.influxdata.com/influxdb.key'
```

InfluxDB client for configuring databases, users and retention policies:

```yaml
influxdb:
  client:
    enabled: true
  server:
    protocol: http
```

(continues on next page)
host: 127.0.0.1
port: 8086
user: admin
password: foobar
user:
  user1:
    enabled: true
    admin: true
    name: username1
database:
  mydb1:
    enabled: true
    name: mydb
    retention_policy:
      - name: rp_db1
        duration: 30d
        replication: 1
        is_default: true
grant:
  username1_mydb1:
    enabled: true
    user: username1
    database: mydb1
    privilege: all

InfluxDB client state’s that uses curl can be forced to retry query if curl call fails:

```yaml
influxdb:
  client:
    enabled: true
    retry:
      count: 3
      delay: 3
```

Create an continuous queries:

```yaml
influxdb:
  client:
    database:
      mydb:
        continuous_query:
          cq_avg_bus_passengers:
            SELECT mean("passengers") INTO "transportation"."three_weeks"."average_passengers" FROM "bus_data" GROUP BY time(1h)
```

Prunning data and data management:

Intended to use in scheduled jobs, executed to maintain data life cycle above retention policy. These states are executed by query.sls and you are expected to trigger sls_id individually.

```yaml
influxdb:
  client:
    database:
      mydb:
        query:
          drop_measurement_h2o:
            DROP MEASUREMENT h2o_quality
```

(continues on next page)
drop_shard_h2o: >-
    DROP SHARD h2o_quality
drop_series_h2o_feet: >-
    DROP SERIES FROM "h2o_feet"
drop_series_h2o_feet_loc_smonica: >-
    DROP SERIES FROM "h2o_feet" WHERE "location" = 'santa_monica'
delete_h2o_quality_rt3: >-
    DELETE FROM "h2o_quality" WHERE "randtag" = '3'
delete_h2o_quality: >-
    DELETE FROM "h2o_quality"

```bash
salt '*' state.sls_id influxdb_query_delete_h2o_quality influxdb.query
```

InfluxDB relay with HTTP outputs:

```yaml
influxdb:
    relay:
        enabled: true
    telemetry:
        enabled: true
    bind:
        address: 127.0.0.1
        port: 9196
    listen:
        http_backend:
            type: http
            bind:
                address: 127.0.0.1
                port: 9096
            output:
                server1:
                    location: http://server1:8086/write
                    timeout: 20s
                    buffer_size_mb: 512
                    max_batch_kb: 1024
                    max_delay_interval: 30s
                server2:
                    location: http://server2:8086/write
```

Read more

- https://influxdata.com/time-series-platform/influxdb/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-influxdb/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:
https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-influxdb

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Known Error Database**

**Sample pillar**

```
kedb:

  server: enabled: true workers: 3 secret_key: secret_token bind:
            address: 0.0.0.0 port: 9753 protocol: tcp
  source: type: ‘git’ address: ‘git@repo1.robotice.cz:django/django-kedb.git’ rev: ‘master’
  cache: engine: ‘memcached’ host: ‘127.0.0.1’ prefix: ‘CACHE_KEDB’
  mail: host: ‘mail.domain.com’ password: ‘mail-pwd’ user: ‘mail-user’
  logger_handler: engine: raven dsn: http://public:private@host/project
```

**Read more**


**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-kedb/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.
Kibana

Kibana is an open source (Apache Licensed), browser based analytics and search interface to Logstash and other timestamped data sets stored in ElasticSearch. With those in place Kibana is a snap to setup and start using (seriously). Kibana strives to be easy to get started with, while also being flexible and powerful

Sample pillar

```yaml
kibana:
  server:
    addrepo: true
    enabled: true
    bind:
      address: 0.0.0.0
      port: 5601
    database:
      engine: elasticsearch
      host: localhost
      port: 9200

Or without adding elasticsearch kibana repository, but with modified path to config file

```yaml
kibana:
  server:
    configpath: /usr/share/kibana/config/kibana.yml
    enabled: true
    bind:
      address: 0.0.0.0
      port: 5601
    database:
      engine: elasticsearch
      host: localhost
      port: 9200

Client setup

Client with host and port (Kibana use Elasticsearch to store its data):

```yaml
kibana:
  client:
    enabled: true
    server:
      host: elasticsearch.host
      port: 9200

Client where you download a Kibana object that is stored in the directory files/:

```
kibana:
  client:
    enabled: true
  server:
    host: elasticsearch.host
    port: 9200
  object:
    logs:
      enabled: true
      name: Logs
      template: kibana/files/objects/dashboard_logs.json
      type: 'dashboard'

Read more

- https://github.com/elasticsearch/kibana/blob/master/src/config.js

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-kibana/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-kibana

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

nagios

Salt formula to set up and manage nagios

Available states

nagios.server
Set up Nagios server
## Sample pillars

### Single nagios service

```yaml
nagios:
  server:
    enabled: true
```

### All Nagios configurations can be configured

```yaml
nagios:
  server:
    enabled: true
  accept_passive_service_checks: 1
  process_performance_data: 0
  check_service_freshness: 1
  check_host_freshness: 0
```

Nagios UI configuration with HTTP basic authentication (use “readonly” flag to specify readonly users)

```yaml
nagios:
  server:
    enabled: true
  ui:
    enabled: true
  auth:
    basic:
      # this is the main admin, it cannot have a 'readonly' flag.
      username: nagiosadmin
      password: secret
      # 'users' section is optional, allows defining additional users.
      users:
        - username: nagios_admin_2
          password: secret2
        - username: nagios_user
          password: secret3
          readonly: true
```

Nagios UI configuration with LDAP authentication/authorization:

```yaml
nagios:
  server:
    enabled: true
  ui:
    enabled: true
  auth:
    basic:
      # this is the main admin, it cannot have a 'readonly' flag.
      username: nagiosadmin
      password: secret
    ldap:
      enabled: true
      # Url format is described here
      url: ldaps://ldap.domain.ltd:<port>/cn=users,dc=domain,dc=local?uid?sub?
      bind_dn: cn=admin,dc=domain,dc=local
      bind_password: secret
```

(continues on next page)
# Optionally, restrict access to members of a group:

```bash
ldap_group_dn: cn=admins,ou=groups,dc=domain,dc=local
ldap_group_attribute: memberUid
```

Nagios objects can be defined in pillar:

```bash
nagios:
  server:
    enabled: true
  objects:
    contactgroups:
      group1:
        contactgroup_name: Operator
    contacts:
      contact1:
        alias: 'root_at_localhost'
        contact_name: Me
        contactgroups:
          - Operator
        email: 'root@localhost'
        host_notifications_enabled: 1
        host_notification_period: 24x7
        host_notification_options: 'd,r'
        host_notification_commands: notify-host-by-smtp
        service_notifications_enabled: 1
        service_notification_period: 24x7
        service_notification_options: 'w,u,c,r'
        service_notification_commands: notify-service-by-smtp
    commands:
      check_http_basic_auth:
        command_line: "check_http -4 -I '$ARG1$' -w 2 -c 3 -t 5 -p $ARG2$ -u '/' -e 401 Unauthorized"
    services:
      generic_service_tpl:
        register: 0
        contact_groups: Operator
        process_perf_data: 0
        max_check_attempts: 3
      hosts:
        generic_host_tpl:
          notifications_enabled: 1
          event_handler_enabled: 1
          flap_detection_enabled: 1
          failure_prediction_enabled: 1
          process_perf_data: 0
          retain_status_information: 1
          retain_nonstatus_information: 1
          max_check_attempts: 10
          notification_interval: 0
          notification_period: 24x7
          notification_options: d,u,r
          contact_groups: Operator
          register: 0
```

Also, `hostgroups`, `hosts` and `services` can be created dynamically using `mine`:
SaltStack-Formulas Documentation, Release master

nagios:
  server:
    enabled: true
dynamic:
    enabled: true
grain_hostname: 'host'
grain_interfaces: 'ip4_interfaces' # the default
#hostname_suffix: .prod # optionally suffix hostnames
hostgroups:
  - target: '
    name: All
    expr_from: glob
  - target: 'G@roles:nova.controller'
    name: Nova Controller
  - target: 'G@roles:nova.compute'
    name: Nova Compute
  - target: 'G@roles:keystone.server'
    name: Keystone server
  - target: 'G@roles:influxdb.server'
    name: InfluxDB server
  - target: 'G@roles:elasticsearch.server'
    name: Elasticsearch server
hosts:
  - target: 'G@services:openssh'
    contact_groups: Operator
    use: generic_host_tpl
    network: 10.0.0.0/8
services:
  - target: 'G@roles:openssh.server'
    name: SSH
    use: generic_service_tpl
    check_command: check_ssh
  - target: 'G@roles:nagios.server'
    name: HTTP Nagios
    use: generic_service_tpl
    check_command: check_http_basic_auth!localhost!${nagios:server:ui:port}

Note about dynamic hosts IP addresses configuration:

There are 2 different ways to configure the Host IP addresses, the preferred way is to define the network of the nodes to pickup the first IP address found belonging to this network.

nagios:
  server:
    enabled: true
dynamic:
    enabled: true
hosts:
  - target: '*'
    contact_groups: Operator
    network: 10.0.0.0/8

The alternative way is to define the interface list, to pickup the first IP address of the first interface found.

nagios:
  server:
    enabled: true

(continues on next page)
dynamic:
    enabled: true
hosts:
  - target: '.*'
    contact_groups: Operator
    interface:
      - eth0
      - ens0

If both properties are defined, the `network` option wins and the `interface` is ignored.

**StackLight Alarms**

StackLight alarms are configured dynamically using `mine` data which are exposed by the Heka formula, respectively `heka:metric_collector:alarm` and `heka:aggregator:alarm_cluster`.

To configure StackLight alarms per nodes (known as AFD):

```yaml
nagios:
  server:
    enabled: true
dynamic:
    enabled: true
hosts:
  - target: 'G@services:openssh'
    contact_groups: Operator
    use: generic_host_tpl
    interface:
      - eth0
      - ens3
  stacklight_alarms:
    enabled: true
    service_template: generic_service_tpl # optional
```

To configure StackLight alarm clusters (known as GSE):

```yaml
nagios:
  server:
    enabled: true
dynamic:
    enabled: true
  stacklight_alarm_clusters:
    enabled: true
    service_template: generic_service_tpl # optional
    host_template: generic_host_tpl # optional
dimension_key: nagios_host # optional
default_host: clusters # optional
```

**Nagios Notification Handlers**

You can configure notification handlers. Currently supported handlers are SMTP, Slack, Salesforce, and Pagerduty.
nagios:
  server:
    enabled: true
  notification:
    slack:
      enabled: true
      webhook_url: https://hooks.slack.com/services/abcdef/12345
    pagerduty:
      enabled: true
      key: abcdef12345
    sfdc:
      enabled: true
      client_id: abcdef12345
      client_secret: abcdef12345
      username: abcdef
      password: abcdef
      auth_url: https://abcedf.my.salesforce.com
      environment: abcedf
      organization_id: abcedf

# SMTP without auth
nagios:
  server:
    enabled: true
  notification:
    smtp:
      auth: false
      url: smtp://127.0.0.1:25
      from: nagios@localhost

  # Notification email subject can be defined, must be one line
  # default subjects are:
  # host_subject: ** $NOTIFICATIONTYPE$ Host Alert: $HOSTNAME$ is $HOSTSTATE$ **
  # service_subject: ** $NOTIFICATIONTYPE$ Service Alert: $HOSTNAME$/"SERVICEDESC" is $SERVICESTATE$ **

  # An example using a Gmail account as a SMTP relay
nagios:
  server:
    enabled: true
  notification:
    smtp:
      auth: login
      url: smtp://smtp.gmail.com:587
      from: <you>@gmail.com
      starttls: true
      username: foo
      password: secret

Each handler adds two commands, `notify-host-by-<HANDLER>`, and `notify-service-by-<HANDLER>`, that you can reference in a contact.

nagios:
  server:
  objects:
    contact:

(continues on next page)
By default in Stacklight, notifications are only enabled for 00-top-clusters and individual host and SSH checks. If you want to enable notifications for all checks you can enable this value:

```
nagios:
  server:
    enabled: true
    notification:
      alarm_enabled_override: true
```

The notification interval defaults to zero, which will only send one notification when the alert triggers. You can override the interval if you want notifications to repeat. For example, to have them repeat every 30 minutes:

```
nagios:
  server:
    enabled: true
    objects:
      hosts:
        generic_host_tpl:
          notification_interval: 30
      services:
        generic_service_tpl:
          notification_interval: 30
```

---

**Read more**

- [https://www.nagios.org](https://www.nagios.org)

**Platform support**

This formula has been tested on Ubuntu Xenial only.

**TODO**

- Configure Apache using salt-formula-apache (using service metadata) or alternatively using Nginx.
**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-nagios/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-nagios

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**rsyslog**

In computing, syslog is a widely used standard for message logging. It permits separation of the software that generates messages, the system that stores them, and the software that reports and analyzes them.

**Sample pillars**

Rsyslog service with default logging template

```yaml
rsyslog:
  client:
    enabled: true
```

Rsyslog service with precise timestamps, severity, facility.

```yaml
rsyslog:
  client:
    enabled: true
    format:
      name: TraditionalFormatWithPRI
      template: '"%syslogpriority% %syslogfacility% %timestamp:::date-rfc3339% %HOSTNAME% %syslogtag% msg:::sp-if-no-1st-sp%msg:::drop-last-lf
''
  output:
    file:
      filter: '.*;auth,authpriv.none
      owner: syslog
group: adm
createmode: 0640
umask: 0022
```

(continues on next page)
/var/log/auth.log:
  filter: auth,authpriv.*
  owner: syslog
  group: adm
  createmode: 0640
  umask: 0022
/var/log/kern.log:
  filter: kern.*
  owner: syslog
  group: adm
  createmode: 0640
  umask: 0022
/var/log/mail.log:
  filter: mail.*
  owner: syslog
  group: adm
  createmode: 0640
  umask: 0022
/var/log/mail.err:
  filter: mail.err
  owner: syslog
  group: adm
  createmode: 0640
  umask: 0022
*:omusrmsg:*:
  filter: *.emerg
|/dev/xconsole:
  filter: "daemon.*;mail.*; news.err; *.=debug;*.=info;*.=notice;*.=warn":
/var/log/your-app.log:
  filter: "if $programname startswith 'your-app' then"
  owner: syslog
  group: adm
  createmode: 0640
  umask: 0022
  stop_processing: true

Rslog service with RainerScript (module, ruleset, template, input).

rsyslog:
  client:
    run_user: syslog
    run_group: adm
    enabled: true
  rainerscript:
    module:
      imfile: {}
    input:
      imfile:
        nginx:
          File: "/var/log/nginx/*.log"
          Tag: "nginx__"
          Severity: "notice"
          Facility: "local0"
          PersistStateInterval: "0"
          Ruleset: "myapp_logs"
        apache2:
          File: "/var/log/apache2/*.log"
Tag: "apache2__"
Severity: "notice"
Facility: "local0"
Ruleset: "myapp_logs"
PersistStateInterval: "0"
rabbitmq:
  File: "/var/log/rabbitmq/*.log"
  Tag: "rabbitmq__"
  Severity: "notice"
  Facility: "local0"
  PersistStateInterval: "0"
  Ruleset: "myapp_logs"
  template:
    ImfileFilePath:
      parameter:
        type: string
        string: "<%PRI>%TIMESTAMP:::date-rfc3339% %HOSTNAME% %syslogtag:1:32%%.$.suffix%%msg:::sp-if-no-1st-sp%%msg\n"
  ruleset:
    remote_logs:
      description: 'action(type="omfwd" Target="172.16.10.92" Port="10514" Protocol="udp" Template="ImfileFilePath")'
    myapp_logs:
      description: 'set $.suffix=re_extract($!metadata!filename, "(.*)/([^/])", 0, 2, "all.log"); call remote_logs'

Custom templates

It is possible to define a specific syslog template per output file instead of using the default one.

rsyslog:
  output:
    file:
      /var/log/your-app.log:
        template: "%syslogtag:1:32%%msg:::sp-if-no-1st-sp%%msg\n"
        filter: "if $programname startswith 'your-app' then"

Remote rsyslog server

It is possible to have rsyslog act as remote server, collecting, storing or forwarding logs. This functionality is provided via rsyslog input/output modules, rulesets and templates.

rsyslog:
  server:
    enabled: true
    module:
      imudp: {}
    template:
      RemoteFilePath:
        parameter:
          type: string
          string: /var/log/%HOSTNAME%/%programname%.log
    ruleset:
remote10514:
    description: action(type="omfile" dynaFile="RemoteFilePath")
input:
    imudp:
        port: 10514
        ruleset: remote10514

Support metadata

If the *heka* support metadata is enabled, all output files are automatically parsed by the *log_collector* service. To skip the *log_collector* configuration, set the *skip_log_collector* to true.

rsyslog:
    output:
        file: /var/log/your-app.log:
            filter: "if $programname startswith 'your-app' then"
            skip_log_collector: true

Read more


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-rsyslog/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-rsyslog

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Sensu

Sample pillars

Sensu Server with API

```yaml
sensu:
  server:
    enabled: true
    keepalive_warning: 20
    keepalive_critical: 60
    mine_checks: true
  database:
    engine: redis
    host: localhost
    port: 6379
  message_queue:
    engine: rabbitmq
    host: rabbitmq
    port: 5672
    user: monitor
    password: pwd
    virtual_host: '/monitor'
  bind:
    address: 0.0.0.0
    port: 4567
  handler:
    default:
      enabled: true
    set:
      - mail
      - pipe
    stdout:
      enabled: true
  mail:
    mail_to: 'mail@domain.cz'
    host: smtp1.domain.cz
    port: 465
    user: 'mail@domain.cz'
    password: 'pwd'
    authentication: cram_md5
    encryption: ssl
    domain: 'domain.cz'
  pipe:
    enabled: true
    command: /usr/bin/tee /tmp/debug
```

Sensu Dashboard (now uchiwa)

```yaml
sensu:
  dashboard:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8080
  admin:
    username: admin
```

(continues on next page)
Sensu Client

```yaml
sensu:
  client:
    enabled: true
    message_queue:
      engine: rabbitmq
      host: rabbitmq
      port: 5672
      user: monitor
      password: pwd
      virtual_host: '/monitor'
```

Sensu Client with check explicitly disabled

```yaml
sensu:
  client:
    enabled: true
    message_queue:
      engine: rabbitmq
      host: rabbitmq
      port: 5672
      user: monitor
      password: pwd
      virtual_host: '/monitor'
    check:
      local_linux_storage_swap_usage:
        enabled: False
```

Sensu Client with subscriptions explicitly disabled

```yaml
sensu:
  client:
    enabled: true
    message_queue:
      engine: rabbitmq
      host: rabbitmq
      port: 5672
      user: monitor
      password: pwd
      virtual_host: '/monitor'
  unsubscribe:
    - collectd.client
    - git.client
```

Sensu Client with community plugins

```yaml
sensu:
  client:
    enabled: true
    plugin:
      sensu_community_plugins:
        enabled: true
        monitoring_for_openstack:
```

(continues on next page)
enabled: true
ruby_gems:
  enabled: True
name:
  bunny:
message_queue:
  engine: rabbitmq
  host: rabbitmq
  port: 5672
  user: monitor
  password: pwd
  virtual_host: '/monitor'

Sensu SalesForce handler

sensu:
  server:
    enabled: true
  handler:
    default:
      enabled: true
      set:
        - sfdc
      stdout:
        enabled: true
      sfdc:
        enabled: true
        sfdc_client_id: "3MV90e7T3010ea4MKj"
        sfdc_client_secret: 11482216293059
        sfdc_username: test@test1.test
        sfdc_password: passTemp
        sfdc_auth_url: https://mysite--scloudqa.cs12.my.salesforce.com
        environment: a2XX0000001
        sfdc_organization_id: 00DV00000
        sfdc_http_proxy: 'http://10.10.10.10:8888'
        token_cache_file: "/path/to/cache/token"

Sensu Slack handler

sensu:
  server:
    enabled: true
  handler:
    default:
      enabled: true
      set:
        - slack
      stdout:
        enabled: true
      slack:
        enabled: True
        channel: '#channel_name'
        webhook_url: 'https://hooks.slack.com/services/kastan12T/B57X3SDQA/
        fasfsaf0632hjk13dsccln9v'
        proxy_address: '10.10.10.10'
        proxy_port: '8888'
Read more

- http://docs.sensuapp.org/0.9/installing_sensu.html
- https://github.com/fridim/nagios-plugin-check_galera_cluster
- https://github.com/opinkerfi/nagios-plugins/tree/master/check_ibm_bladecenter
- https://github.com/opinkerfi/nagios-plugins/tree/master/check_storwize
- https://github.com/ehazlett/sensu-py/
- https://github.com/Level-Up/Supervisord-Nagios-Plugin/blob/master/check_supv.py

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-sensu/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-sensu

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Statsd formula

Simple daemon for easy stats aggregation.
Sample pillars

Standalone Statsd server with Graphite/carbon backend

```yaml
statsd:
  server:
    enabled: true
    bind:
      port: 8125
      address: 0.0.0.0
    backend:
      engine: carbon
      host: metrics1.domain.com
      port: 2003
```

Standalone Statsd server with Graphite/AMQP backend

```yaml
statsd:
  server:
    enabled: true
    bind:
      port: 8125
      address: 0.0.0.0
    backend:
      engine: amqp
      host: metrics1.domain.com
      port: 5672
```

Standalone Statsd server with OpenTSDB backend

```yaml
statsd:
  server:
    enabled: true
    bind:
      port: 8125
      address: 0.0.0.0
    backend:
      engine: amqp
      host: metrics1.domain.com
      port: 2003
```

More information

- https://github.com/etsy/statsd/
- https://github.com/mrtazz/statsd-amqp-backend
- https://github.com/danslimmon/statsd-opentsdb-backend

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/
In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-statsd/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-statsd

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

---

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home SaltStack-Formulas Project Introduction

## Container Services

Container services for automated container management.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>calico</td>
<td><a href="https://github.com/salt-formulas/salt-formula-calico">https://github.com/salt-formulas/salt-formula-calico</a></td>
</tr>
<tr>
<td>dekapod</td>
<td><a href="https://github.com/salt-formulas/salt-formula-dekapod">https://github.com/salt-formulas/salt-formula-dekapod</a></td>
</tr>
<tr>
<td>docker</td>
<td><a href="https://github.com/salt-formulas/salt-formula-docker">https://github.com/salt-formulas/salt-formula-docker</a></td>
</tr>
<tr>
<td>kubernetes</td>
<td><a href="https://github.com/salt-formulas/salt-formula-kubernetes">https://github.com/salt-formulas/salt-formula-kubernetes</a></td>
</tr>
</tbody>
</table>

### Salt Formula Calico

Salt formula for calico deployment.

### Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-calico/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:
You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-calico

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

### Decapod formula

Decapod is intended to simplify deployment and lifecycle management of Ceph.

#### Sample pillars

Single decapod service

```
decapod:
  server:
    enabled: true
```

#### Read more


### Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-letsencrypt/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-letsencrypt

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Docker Formula

Docker is a platform for developers and sysadmins to develop, ship, and run applications. Docker lets you quickly assemble applications from components and eliminates the friction that can come when shipping code. Docker lets you get your code tested and deployed into production as fast as possible.

Sample Pillars

Docker Host

```yaml
docker:
  host:
    enabled: true
    options:
      bip: 172.31.255.1/16
      insecure-registries:
        - 127.0.0.1
        - 10.0.0.1
      log-driver: json-file
      log-opts:
        max-size: 50m
```

Configure proxy for docker host

```yaml
docker:
  host:
    proxy:
      enabled: true
      http: http://user:pass@proxy:3128
      https: http://user:pass@proxy:3128
      no_proxy:
        - localhost
        - 127.0.0.1
        - docker-registry
```

Docker Swarm

Role can be master, manager or worker. Where master is the first manager that will initialize the swarm.

Metadata for manager (first node):

```yaml
docker:
  host:
    enabled: true
  swarm:
    role: manager
    advertise_addr: 192.168.1.5
    bind:
      address: 192.168.1.5
      port: 2377
```

Metadata for worker.
Token to join to master node is obtained from grains using salt.mine. In case of any join_token undefined issues, ensure you have docker_swarm_grains available.

**Docker Client**

**Container**

docker:
   client:
      container:
         jenkins:
            # Don’t start automatically
            start: false
            restart: unless-stopped
            image: jenkins:2.7.1
            ports:
               - 8081:8080
               - 50000:50000
            environment:
               JAVA_OPTS: "-Dhudson.footerURL=https://www.example.com"
            volumes:
               - /srv/volumes/jenkins:/var/jenkins_home

**Using Docker Compose**

There are two states that provides this functionality:

- docker.client.stack
- docker.client.compose

Stack is new and works with Docker Swarm Mode. Compose is legacy and works only if node isn’t member of Swarm. Metadata for both states are similar and differs only in implementation.

**Stack**

docker:
   client:
      stack:
         django_web:
            enabled: true
            update: true
            environment:
Compose

There are three options how to install docker-compose:

- distribution package (default)
- using Pip
- using Docker container

Install docker-compose using Docker (default is distribution package)

docker:
  client:
    compose:
      source:
        engine: docker
        image: docker/compose:1.8.0
  django_web:
    # Run up action, any positional argument to docker-compose CLI
    # If not defined, only docker-compose.yml is generated
    status: up
    # Run image pull every time state is run triggering container
    # restart in case it's changed
    pull: true
    environment:
      SOMEVAR: somevalue
    service:
      db:
        image: postgres
      web:
        image: djangoapp
        volumes:
          - /srv/volumes/django:/srv/django
        ports:
          - 8000:8000
        depends_on:
          - db
Registry

docker:
  client:
    registry:
      target_registry: apt:5000
      image:
        - registry: docker
          name: compose:1.8.0
        - registry: tcpcloud
          name: jenkins:latest
        - registry: ""
          name: registry:2
          target_registry: myregistry

Service

To deploy service in Swarm mode, you can use docker.client.service:

parameters:
docker:
  client:
    service:
      postgresql:
        environment:
          POSTGRES_USER: user
          POSTGRES_PASSWORD: password
          POSTGRES_DB: mydb
        restart:
          condition: on-failure
        image: "postgres:9.5"
        ports:
          - 5432:5432
        volume:
          data:
            type: bind
            source: /srv/volumes/postgresql/maas
            destination: /var/lib/postgresql/data

Docker Registry

docker:
  registry:
    log:
      level: debug
    formatter: json
    cache:
      engine: redis
      host: localhost
    storage:
      engine: filesystem
      root: /srv/docker/registry
    bind:
host: 0.0.0.0
port: 5000
hook:
  mail:
    levels:
      - panic
    # Options are rendered as yaml as is so use hook-specific options here
    options:
      smtp:
        addr: smtp.sendhost.com:25
        username: sendername
        password: password
        insecure: true
        from: name@sendhost.com
        to:
          - name@receivehost.com

Docker login to private registry

docker:
  host:
    enabled: true
  registry:
    first:
      address: private.docker.com
      user: username
      password: password
    second:
      address: private2.docker.com
      user: username2
      password: password2

Docker container service management

Enforce the service in container is started

contrail_control_started:
  dockerng_service.start:
  - container: f020d0d3efa8
  - service: contrail-control

or

contrail_control_started:
  dockerng_service.start:
  - container: contrail_controller
  - service: contrail-control

Enforce the service in container is stopped

contrail_control_stoped:
  dockerng_service.stop:
  - container: f020d0d3efa8
  - service: contrail-control

Enforce the service in container will be restarted
contrail_control_restart:
dockerng_service.restart:
  - container: f020d0d3efa8
  - service: contrail-control

Enforce the service in container is enabled

contrail_control_enable:
dockerng_service.enable:
  - container: f020d0d3efa8
  - service: contrail-control

contrail_control_disable:
dockerng_service.disable:
  - container: f020d0d3efa8
  - service: contrail-control

More Information

- https://docs.docker.com/installation/ubuntu-linux/
- https://github.com/salt-stack-formulas/docker-formula

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-docker/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-docker

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Kubernetes Formula

Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications. This formula deploys production ready Kubernetes and generate Kubernetes manifests as well.
You can download `kubectl` configuration and connect to your cluster. However, keep in mind `kubernetes_control_address` needs to be accessible from your computer:

```bash
mkdir -p ~/.kube
[ -f ~/.kube/config ] && cp -v ~/.kube/config ~/.kube/config-backup
ssh cfg01 "sudo ssh ctl01 /etc/kubernetes/kubeconfig.sh" > ~/.kube/config
kubectl get no
```

`cfg01` is Salt master node and `ctl01` is one of Kubernetes masters.

## Sample Pillars

**REQUIRED:** Define image to use for hyperkube, CNIs and calicoctl image

```yaml
parameters:
  kubernetes:
    common:
      hyperkube:
        image: gcr.io/google_containers/hyperkube:v1.6.5
      pool:
        network:
          calicoctl:
            image: calico/ctl
          cni:
            image: calico/cni
```

Enable helm-tiller addon

```yaml
parameters:
  kubernetes:
    common:
      addons:
        helm:
          enabled: true
```

Enable calico-policy addon

```yaml
parameters:
  kubernetes:
    common:
      addons:
        calico_policy:
          enabled: true
```

Enable virtlet addon

```yaml
parameters:
  kubernetes:
    common:
      addons:
        virtlet:
          enabled: true
          namespace: kube-system
          image: mirantis/virtlet:v0.8.0
          hosts:
            - cmp01
            - cmp02
```
Enable netchecker addon

```yaml
parameters:
  kubernetes:
    common:
      addons:
        netchecker:
          enabled: true
    master:
      namespace:
        netchecker:
          enabled: true
```

Enable Kubernetes Federation control plane

```yaml
parameters:
  kubernetes:
    master:
      federation:
        enabled: True
        name: federation
        namespace: federation-system
        source: https://dl.k8s.io/v1.6.6/kubernetes-client-linux-amd64.tar.gz
        hash: 94b2c9cd29981a8e150c187193bab0d8c0b6e906260f837367feff99860a6376
        service_type: NodePort
        dns_provider: coredns
        childclusters:
          - secondcluster.mydomain
          - thirdcluster.mydomain
```

Enable external DNS addon with CoreDNS provider

```yaml
parameters:
  kubernetes:
    common:
      addons:
        coredns:
          enabled: True
        externaldns:
          enabled: True
          domain: company.mydomain
          provider: coredns
```

Enable external DNS addon with Designate provider

```yaml
parameters:
  kubernetes:
    common:
      addons:
        externaldns:
          enabled: True
          domain: company.mydomain
          provider: designate
          designate_os_options:
            OS_AUTH_URL: https://keystone_auth_endpoint:5000
            OS_PROJECT_DOMAIN_NAME: default
            OS_USER_DOMAIN_NAME: default
            OS_PROJECT_NAME: admin
```

(continues on next page)
Enable external DNS addon with AWS provider

```yaml
parameters:
  kubernetes:
    common:
      addons:
        externaldns:
          enabled: True
          domain: company.mydomain
          provider: aws
          aws_options:
            AWS_ACCESS_KEY_ID: XXXXXXXXXXXXXXXXXXXXXXXX
            AWS_SECRET_ACCESS_KEY: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX
```

Enable external DNS addon with Google CloudDNS provider

```yaml
parameters:
  kubernetes:
    common:
      addons:
        externaldns:
          enabled: True
          domain: company.mydomain
          provider: google
          google_options:
            key: ''
            project: default-123
```

key should be exported from google console and processed as `cat key.json | tr -d 'n'`

Enable OpenStack cloud provider

```yaml
parameters:
  kubernetes:
    common:
      cloudprovider:
        enabled: True
        provider: openstack
        params:
          auth_url: https://openstack.mydomain:5000/v3
          username: nova
          password: nova
          region: RegionOne
          tenant_id: 4bce4162d8744c599e350099cfa22a0a
          domain_name: default
          subnet_id: 72407854-aca6-4cf1-b873-e9affb09484b
          lb_version: v2
```

Configure service verbosity

```yaml
parameters:
  kubernetes:
    master:
```

(continues on next page)
verbosity: 2
pool:
  verbosity: 2

Set cluster name and domain

parameters:
  kubernetes:
    common:
      kubernetes_cluster_domain: mycluster.domain
      cluster_name: mycluster

Enable autoscaler for dns addon. Poll period can be skipped.

kubernetes:
  common:
    addons:
      dns:
        domain: cluster.local
        enabled: true
        replicas: 1
        server: 10.254.0.10
        autoscaler:
          enabled: true
          poll-period-seconds: 60

Pass additional parameters to daemons:

parameters:
  kubernetes:
    master:
      apiserver:
        daemon_opts:
          storage-backend: pigeon
      controller_manager:
        daemon_opts:
          log-dir: /dev/null
    pool:
      kubelet:
        daemon_opts:
          max-pods: "$\""

Containers on pool definitions in pool.service.local

parameters:
  kubernetes:
    pool:
      service:
        local:
          enabled: False
          service: libvirt
          cluster: openstack-compute
          namespace: default
          role: "$\{linux:system:name\}"
          type: LoadBalancer
          kind: Deployment
Master definition

```yaml
apiVersion: extensions/v1beta1
replicas: 1
host_pid: True
nodeSelector:
  - key: openstack
    value: ${linux:system:name}
hostNetwork: True
container:
  libvirt-compute:
    privileged: True
    image: ${_param:docker_repository}/libvirt-compute
    tag: ${_param:openstack_container_tag}
```

```
Master definition

kubernetes:
  common:
    cluster_name: cluster
  addons:
    dns:
      domain: cluster.local
      enabled: true
      replicas: 1
      server: 10.254.0.10
  master:
    admin:
      password: password
      username: admin
    apiserver:
      address: 10.0.175.100
      secure_port: 443
      insecure_address: 127.0.0.1
      insecure_port: 8080
    ca: kubernetes
    enabled: true
  etcd:
    host: 127.0.0.1
    members:
      - host: 10.0.175.100
        name: node040
        name: node040
        token: ca939ec9c2a17b0786f6d411fe019e9b
    kubelet:
      allow_privileged: true
    network:
      engine: calico
      mtu: 1500
      hash: fb5e30ebe6154911a66ec3fb5f1195b2
      private_ip_range: 10.150.0.0/16
      version: v0.19.0
    service_addresses: 10.254.0.0/16
  storage:
    engine: glusterfs
    members:
      - host: 10.0.175.101
        port: 24007
      - host: 10.0.175.102
```

(continues on next page)
Enable basic, token and http authentication, disable ssl auth, create some static users:

```
kubernetes:
  master:
    auth:
      basic:
        enabled: true
        user:
          jdoe:
            password: dummy
            groups:
```

(continues on next page)
- system:admin
  http:
    enabled: true
  header:
    user: X-Remote-User
    group: X-Remote-Group
  ssl:
    enabled: false
  token:
    enabled: true
  user:
    jdoe:
      token: dummytoken
    groups:
      - system:admin

Kubernetes with OpenContrail network plugin

On Master:

```yaml
kubernetes:
  common:
    addons:
      contrail_network_controller:
        enabled: true
        namespace: kube-system
        image: yashulyak/contrail-controller:latest
  master:
    network:
      engine: opencontrail
      default_domain: default-domain
      default_project: default-domain:default-project
      public_network: default-domain:default-project:Public
      public_ip_range: 185.22.97.128/26
      private_ip_range: 10.150.0.0/16
      service_cluster_ip_range: 10.254.0.0/16
      network_label: name
      service_label: uses
      cluster_service: kube-system/default
      config:
        api:
          host: 10.0.170.70
```

On pools:

```yaml
kubernetes:
  pool:
    network:
      engine: opencontrail
```

Dashboard public IP must be configured when Contrail network is used:

```yaml
kubernetes:
  common:
```

(continues on next page)
addons:
  public_ip: 1.1.1.1

Kubernetes control plane running in systemd

By default kube-apiserver, kube-scheduler, kube-controllermanager, kube-proxy, etcd running in docker containers through manifests. For stable production environment this should be run in systemd.

```yaml
kubernetes:
  master:
    container: false

kubernetes:
  pool:
    container: false
```

Because k8s services run under kube user without root privileges, there is need to change secure port for apiserver.

```yaml
kubernetes:
  master:
    apiserver:
      secure_port: 8081
```

Kubernetes with Flannel

On Master:

```yaml
kubernetes:
  master:
    network:
      engine: flannel

# If you don't register master as node:
  etcd:
    members:
      - host: 10.0.175.101
        port: 4001
      - host: 10.0.175.102
        port: 4001
      - host: 10.0.175.103
        port: 4001

common:
  network:
    engine: flannel
```

On pools:

```yaml
kubernetes:
  pool:
    network:
      engine: flannel

  etcd:
    members:
      - host: 10.0.175.101
```
port: 4001
- host: 10.0.175.102
  port: 4001
- host: 10.0.175.103
  port: 4001

common:
  network:
    engine: flannel

Kubernetes with Calico

On Master:

```
kubernetes:
  master:
    network:
      engine: calico
      mtu: 1500
# If you don't register master as node:
  etcd:
    members:
      - host: 10.0.175.101
        port: 4001
      - host: 10.0.175.102
        port: 4001
      - host: 10.0.175.103
        port: 4001
```

On pools:

```
kubernetes:
  pool:
    network:
      engine: calico
      mtu: 1500
    etcd:
      members:
        - host: 10.0.175.101
          port: 4001
        - host: 10.0.175.102
          port: 4001
        - host: 10.0.175.103
          port: 4001
```

Running with secured etcd:

```
kubernetes:
  pool:
    network:
      engine: calico
      mtu: 1500
    etcd:
      ssl:
        enabled: true
      master:
```

Running with calico-policy controller:

```yaml
kubernetes:
  pool:
    network:
      engine: calico
      mtu: 1500
      addons:
        calico_policy:
          enabled: true
    master:
      network:
        engine: calico
        mtu: 1500
        addons:
          calico_policy:
            enabled: true
```

Enable Prometheus metrics in Felix

```yaml
kubernetes:
  pool:
    network:
      prometheus:
        enabled: true
    master:
      network:
        prometheus:
          enabled: true
```

Post deployment configuration

```bash
# set ETCD
export ETCD_AUTHORITY=10.0.111.201:4001

# Set NAT for pods subnet
calicoctl pool add 192.168.0.0/16 --nat-outgoing

# Status commands
calicoctl status
calicoctl node show
```

Kubernetes with GlusterFS for storage

```yaml
kubernetes:
  master:
    ...
```
Kubernetes Storage Class

AWS EBS storageclass integration. It also requires to create IAM policy and profiles for instances and tag all resources by KubernetesCluster in EC2.

```
kubernetes:
  common:
    addons:
      storageclass:
        aws_slow:
          enabled: True
          default: True
          provisioner: aws-ebs
          name: slow
          type: gp2
          iopspergb: "10"
          zones: xxx
        nfs_shared:
          name: elasti01
          enabled: True
          provisioner: nfs
          spec:
            name: elastic_data
            nfs:
              server: 10.0.0.1
              path: /exported_path
```
Kubernetes labels

Label node:

```yaml
kubernetes:
  master:
    label:
      label01:
        value: value01
        node: node01
        enabled: true
        key: key01
    ...
```

Pull images from private registries

```yaml
kubernetes:
  master:
    ...
    registry:
      secret:
        registry01:
          enabled: True
          key: (get from `cat /root/.docker/config.json | base64`)
          namespace: default
    ...
```  

Kubernetes Service Definitions in pillars

Following samples show how to generate kubernetes manifest as well and provide single tool for complete infrastructure management.

Deployment manifest

```yaml
salt:
  control:
    enabled: True
    hostNetwork: True
    service:
      ...
```
memcached:
  privileged: True
  service: memcached
  role: server
  type: LoadBalancer
  replicas: 3
  kind: Deployment
  apiVersion: extensions/v1beta1
  ports:
    - port: 8774
      name: nova-api
    - port: 8775
      name: nova-metadata
  volume:
    volume_name:
      type: hostPath
      mount: /certs
      path: /etc/certs
  container:
    memcached:
      image: memcached
      tag: 2
      ports:
        - port: 8774
          name: nova-api
        - port: 8775
          name: nova-metadata
  variables:
    - name: HTTP_TLS_CERTIFICATE:
        value: /certs/domain.crt
    - name: HTTP_TLS_KEY
        value: /certs/domain.key
  volumes:
    - name: /etc/certs
      type: hostPath
      mount: /certs
      path: /etc/certs

**PetSet manifest**

```yaml
service:
  memcached:
    apiVersion: apps/v1alpha1
    kind: PetSet
    service_name: 'memcached'
    container:
      memcached:
```

**Configmap**

You are able to create configmaps using support layer between formulas. It works simple, eg. in nova formula there’s file `meta/config.yml` which defines config files used by that service and roles.
Kubernetes formula is able to generate these files using custom pillar and grains structure. This way you are able to run docker images built by any way while still re-using your configuration management.

Example pillar:

```yaml
kubernetes:
  control:
    config_type: default|kubernetes  # Output is yaml k8s or default single files
    configmap:
      nova-control:
        grains:
          # Alternate grains as OS running in container may differ from
          # salt minion OS. Needed only if grains matters for config
          # generation.
          os_family: Debian
        pillar:
          # Generic pillar for nova controller
          nova:
            controller:
              enabled: true
              version: liberty
              ...
```

To tell which services supports config generation, you need to ensure pillar structure like this to determine support:

```yaml
nova:
  _support:
    config:
      enabled: true
```

**initContainers**

Example pillar:

```yaml
kubernetes:
  control:
    service:
      memcached:
        init_containers:
        - name: test-mysql
          image: busybox
          command:
            - sleep
            - 3600
          volumes:
            - name: config
              mount: /test
```

2.1. Project Introduction
Affinity

**podAffinity**

Example pillar:

```yaml
kubernetes:
  control:
  service:
    memcached:
      affinity:
        pod_affinity:
          name: podAffinity
          expression:
            label_selector:
              name: labelSelector
              selectors:
                - key: app
                  value: memcached
                  topology_key: kubernetes.io/hostname
```

**podAntiAffinity**

Example pillar:

```yaml
kubernetes:
  control:
  service:
    memcached:
      affinity:
        anti_affinity:
          name: podAntiAffinity
          expression:
            label_selector:
              name: labelSelector
              selectors:
                - key: app
                  value: opencontrail-control
                  topology_key: kubernetes.io/hostname
```

**nodeAffinity**

Example pillar:

```yaml
kubernetes:
  control:
  service:
    memcached:
      affinity:
        node_affinity:
          name: nodeAffinity
          expression:
            match_expressions:
```

(continues on next page)
name: matchExpressions
selectors:
- key: key
  operator: In
  values:
  - value1
  - value2

Volumes

hostPath

```
service:
  memcached:
    container:
      memcached:
        volumes:
          - name: volume1
            mountPath: /volume
            readOnly: True

... volume:
  volume1:
    name: /etc/certs
    type: hostPath
    path: /etc/certs
```

emptyDir

```
service:
  memcached:
    container:
      memcached:
        volumes:
          - name: volume1
            mountPath: /volume
            readOnly: True

... volume:
  volume1:
    name: /etc/certs
    type: emptyDir
```

configMap

```
service:
  memcached:
    container:
      memcached:
```

(continues on next page)
To mount single configuration file instead of whole directory:

```yaml
service:
  memcached:
    container:
      memcached:
        volumes:
          - name: volumel
            mountPath: /volume/config.conf
            sub_path: config.conf
```

### Generating Jobs

**Example pillar:**

```yaml
kubernetes:
  control:
    job:
      sleep:
        job: sleep
        restart_policy: Never
    container:
      sleep:
        image: busybox
        tag: latest
        command:
          - sleep
          - "3600"
```

Volumes and Variables can be used as the same way as during Deployment generation.

**Custom params:**

```yaml
kubernetes:
  control:
    job:
      host_network: True
      host_pid: True
      container:
```
Role-based access control

To enable RBAC, you need to set following option on your apiserver:

```
kubernetes:
  master:
    auth:
      mode: Node,RBAC
```

Then you can use `kubernetes.control.role` state to orchestrate role and rolebindings. Following example shows how to create brand new role and binding for service account:

```
control:
  role:
    etcd-operator:
      kind: ClusterRole
      rules:
        - apiGroups:
            - etcd.coreos.com
          resources:
            - clusters
          verbs:
            - "*
        - apiGroups:
            - extensions
          resources:
            - thirdpartyresources
          verbs:
            - create
        - apiGroups:
            - storage.k8s.io
          resources:
            - storageclasses
          verbs:
            - create
  binding:
    etcd-operator:
      kind: ClusterRoleBinding
      namespace: test # <-- if no namespace, then it's clusterrolebinding
      subject:
        etcd-operator:
          kind: ServiceAccount
```
Simplest possible use-case, add user test edit permissions on it’s test namespace:

```
kubernetes:
  control:
    role:
      edit:
        kind: ClusterRole
        # No rules defined, so only binding will be created assuming role
        # already exists
    binding:
      test:
        namespace: test
        subject:
          test:
            kind: User
```

**More Information**

- [https://github.com/kubernetes/kubernetes/tree/master/cluster/saltbase](https://github.com/kubernetes/kubernetes/tree/master/cluster/saltbase)

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

- [http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

- [https://github.com/salt-formulas/salt-formula-kubernetes/issues](https://github.com/salt-formulas/salt-formula-kubernetes/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

- [https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join [salt-formulas-users](https://launchpad.net/~salt-formulas-users) team and subscribe to mailing list:

- [https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula:

- [https://github.com/salt-formulas/salt-formula-kubernetes](https://github.com/salt-formulas/salt-formula-kubernetes)

Any questions or feedback is always welcome so feel free to join our IRC channel:

- #salt-formulas @ irc.freenode.net
OpenStack Services

All supported OpenStack cloud platform services.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>aodh</td>
<td><a href="https://github.com/salt-formulas/salt-formula-aodh">https://github.com/salt-formulas/salt-formula-aodh</a></td>
</tr>
<tr>
<td>avinetworks</td>
<td><a href="https://github.com/salt-formulas/salt-formula-avinetworks">https://github.com/salt-formulas/salt-formula-avinetworks</a></td>
</tr>
<tr>
<td>billometer</td>
<td><a href="https://github.com/salt-formulas/salt-formula-billometer">https://github.com/salt-formulas/salt-formula-billometer</a></td>
</tr>
<tr>
<td>ceilometer</td>
<td><a href="https://github.com/salt-formulas/salt-formula-ceilometer">https://github.com/salt-formulas/salt-formula-ceilometer</a></td>
</tr>
<tr>
<td>cinder</td>
<td><a href="https://github.com/salt-formulas/salt-formula-cinder">https://github.com/salt-formulas/salt-formula-cinder</a></td>
</tr>
<tr>
<td>designate</td>
<td><a href="https://github.com/salt-formulas/salt-formula-designate">https://github.com/salt-formulas/salt-formula-designate</a></td>
</tr>
<tr>
<td>glance</td>
<td><a href="https://github.com/salt-formulas/salt-formula-glance">https://github.com/salt-formulas/salt-formula-glance</a></td>
</tr>
<tr>
<td>heat</td>
<td><a href="https://github.com/salt-formulas/salt-formula-heat">https://github.com/salt-formulas/salt-formula-heat</a></td>
</tr>
<tr>
<td>horizon</td>
<td><a href="https://github.com/salt-formulas/salt-formula-horizon">https://github.com/salt-formulas/salt-formula-horizon</a></td>
</tr>
<tr>
<td>keystone</td>
<td><a href="https://github.com/salt-formulas/salt-formula-keystone">https://github.com/salt-formulas/salt-formula-keystone</a></td>
</tr>
<tr>
<td>magnum</td>
<td><a href="https://github.com/salt-formulas/salt-formula-magnum">https://github.com/salt-formulas/salt-formula-magnum</a></td>
</tr>
<tr>
<td>midonet</td>
<td><a href="https://github.com/salt-formulas/salt-formula-midonet">https://github.com/salt-formulas/salt-formula-midonet</a></td>
</tr>
<tr>
<td>murano</td>
<td><a href="https://github.com/salt-formulas/salt-formula-murano">https://github.com/salt-formulas/salt-formula-murano</a></td>
</tr>
<tr>
<td>neutron</td>
<td><a href="https://github.com/salt-formulas/salt-formula-neutron">https://github.com/salt-formulas/salt-formula-neutron</a></td>
</tr>
<tr>
<td>nova</td>
<td><a href="https://github.com/salt-formulas/salt-formula-nova">https://github.com/salt-formulas/salt-formula-nova</a></td>
</tr>
<tr>
<td>opencontrail</td>
<td><a href="https://github.com/salt-formulas/salt-formula-opencontrail">https://github.com/salt-formulas/salt-formula-opencontrail</a></td>
</tr>
<tr>
<td>openvstorage</td>
<td><a href="https://github.com/salt-formulas/salt-formula-openvstorage">https://github.com/salt-formulas/salt-formula-openvstorage</a></td>
</tr>
<tr>
<td>rally</td>
<td><a href="https://github.com/salt-formulas/salt-formula-rally">https://github.com/salt-formulas/salt-formula-rally</a></td>
</tr>
<tr>
<td>sahara</td>
<td><a href="https://github.com/salt-formulas/salt-formula-sahara">https://github.com/salt-formulas/salt-formula-sahara</a></td>
</tr>
<tr>
<td>swift</td>
<td><a href="https://github.com/salt-formulas/salt-formula-swift">https://github.com/salt-formulas/salt-formula-swift</a></td>
</tr>
<tr>
<td>tempest</td>
<td><a href="https://github.com/salt-formulas/salt-formula-tempest">https://github.com/salt-formulas/salt-formula-tempest</a></td>
</tr>
</tbody>
</table>

aodh

Aodh is an alarming service for OpenStack. It used to be a part of Ceilometer, but starting from Mitaka it is a separate project. Aodh supports several types of alarms like threshold, event, composite and gnocchi-specific. In cluster mode, coordination is enabled via tooz with Redis backend. MySQL is used as a data backend for alarms and alarm history.

Sample pillars

Cluster aodh service

```yaml
aodh:
  server:
    enabled: true
    version: mitaka
ttl: 86400
    cluster: true
database:
  engine: "mysql+pymysql"
  host: 10.0.106.20
  port: 3306
  name: aodh
  user: aodh
  password: password
bind:
```

(continues on next page)
host: 10.0.106.20
port: 8042
identity:
  engine: keystone
  host: 10.0.106.20
  port: 35357
  tenant: service
  user: aodh
  password: password
message_queue:
  engine: rabbitmq
  port: 5672
  user: openstack
  password: password
  virtual_host: '/openstack'
cache:
  members:
  - host: 10.10.10.10
    port: 11211
  - host: 10.10.10.11
    port: 11211
  - host: 10.10.10.12
    port: 11211

Enhanced logging with logging.conf

By default logging.conf is disabled.

That is possible to enable per-binary logging.conf with new variables:

- openstack_log_appender - set it to true to enable log_config_append for all OpenStack services;
- openstack_fluentd_handler_enabled - set to true to enable FluentHandler for all Openstack services.

Only WatchedFileHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

aodh:
  server:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
          enabled: true

Development and testing

Development and test workflow with Test Kitchen and kitchen-salt provisioner plugin.

Test Kitchen is a test harness tool to execute your configured code on one or more platforms in isolation. There is a .kitchen.yml file in the main directory that defines platforms to be tested and suites to execute on them.
SaltStack-Formulas Documentation, Release master

Kitchen CI can spin instances locally or remote, based on used driver. For local development .kitchen.yml defines a vagrant or docker driver.

To use backend drivers or implement your CI follow the section ‘INTEGRATION.rst#Continuous Integration‘.

The Busser Verifier is used to setup and run tests implemented in <repo>/test/integration. It installs the particular driver to tested instance (Serverspec, InSpec, Shell, Bats, …) prior the verification is executed.

Usage:

```bash
# list instances and status
kitchen list

# manually execute integration tests
kitchen [test || [create|converge|verify|exec|login|destroy|...]] [instance] -t tests/integration

# use with provided Makefile (ie: within CI pipeline)
make kitchen
```

Read more

- https://docs.openstack.org/cli-reference/aodh.html
- https://docs.openstack.org/developer/aodh/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-aodh/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-aodh

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Avinetworks formula

Sample pillars

Salt formula to setup Avi Networks LBaaS

2.1. Project Introduction
avinetworks:
    server:
        enabled: true
        identity: cloud1
        image_location: http://...
        disk_format: qcow2
        public_network: INET1
        saltmaster_ip: 10.0.0.90

avinetworks:
    client:
        enabled: true

External links


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-avinetworks/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-avinetworks

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Billometer

Sample pillar

billometer:
    server:
        enabled: true
        workers: 3
        secret_key: secret_token
        sync_time: 600

(continues on next page)
collect_time: 1800
metric:
in:
    engine: graphite
    host: 10.10.10.180
    port: 80
out:
    engine: statsd
    host: 10.10.10.180
    prefix: foo
    port: 81
bind:
    address: 0.0.0.0
    port: 9753
    protocol: tcp
source:
    type: 'git'
    address: 'git@repo1.robotice.cz:python-apps/billometer.git'
    rev: 'master'
cache:
    engine: 'memcached'
    host: '127.0.0.1'
    prefix: 'CACHE_DJANGO_ENC'
database:
    engine: 'postgresql'
    host: '127.0.0.1'
    name: 'django_billometer'
    password: 'db-pwd'
    user: 'django_billometer'
identity:
    engine: 'keystone'
    region: 'regionOne'
    token: 'token'
    host: '127.0.0.1'
    port: 5000
    api_version: 2
mail:
    host: 'mail.domain.com'
    password: 'mail-pwd'
    user: 'mail-user'
logging:
    engine: sentry
    dsn: pub@sec:dsn.cz/12
    extra_resource:

Extra Resources

billometer:
    server:
        enabled: true
        workers: 3
        secret_key: secret_token
        sync_time: 600
        collect_time: 1800
        extra_resource:
network.rx:
  label: Network RX
  resource: network.rx
  price_rate: 0.0002
  threshold: 150000
7k2_SAS
  price_rate: 0.008205
  resource: cinder.volume
  name: 7k2_SAS
  label: 7k2 SA
10k_SAS
  price_rate: 0.027383
  resource: cinder.volume
  label: 10k2 SAS
  name: 10k_SAS
15k_SAS
  price_rate: 0.034232
  resource: cinder.volume
  label: 15k2 SAS
  name: 15k_SAS
EasyTier
  price_rate: 0.041082
  resource: cinder.volume
  label: Easy Tier
  name: 'EasyTier

Read more


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-billometer/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-billometer

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Ceilometer Formula

The ceilometer project aims to deliver a unique point of contact for billing systems to acquire all of the measurements they need to establish customer billing, across all current OpenStack components with work underway to support future OpenStack components. This formula provides different backends for Ceilometer data: MongoDB, InfluxDB. Also, Graphite and direct (to Elasticsearch) publishers are available. If InfluxDB is used as a backend, heka is configured to consume messages from RabbitMQ and write in to InfluxDB, i.e. ceilometer collector service is not used in this configuration.

Sample Pillars

Ceilometer API/controller node

```
ceilometer:
  server:
    enabled: true
    version: mitaka
    cluster: true
    secret: pwd
    bind:
      host: 127.0.0.1
      port: 8777
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      tenant: service
      user: ceilometer
      password: pwd
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
```

Enable CORS parameters

```
ceilometer:
  server:
    cors:
      allowed_origin: https:localhost.local,http:localhost.local
      expose_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_methods: GET,PUT,POST,DELETE,PATCH
      allow_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_credentials: True
      max_age: 86400
```
Configuration of policy.json file

```yaml
ceilometer:
  server:
    ....
    policy:
      segregation: 'rule:context_is_admin'
# Add key without value to remove line from policy.json
'telemetry:get_resource':
```

Databases configuration

**MongoDB example:**

```yaml
ceilometer:
  server:
    database:
      engine: mongodb
      members:
      - host: 10.0.106.10
        port: 27017
      - host: 10.0.106.20
        port: 27017
      - host: 10.0.106.30
        port: 27017
      name: ceilometer
      user: ceilometer
      password: password
```

**InfluxDB/Elasticsearch example:**

```yaml
ceilometer:
  server:
    database:
      influxdb:
        host: 10.0.106.10
        port: 8086
        user: ceilometer
        password: password
      database: ceilometer
      elasticsearch:
        enabled: true
        host: 10.0.106.10
        port: 9200
```

**Client-side RabbitMQ HA setup**

```yaml
ceilometer:
  server:
    ....
```
message_queue:
  engine: rabbitmq
  members:
  - host: 10.0.106.10
  - host: 10.0.106.20
  - host: 10.0.106.30
  user: openstack
  password: pwd
  virtual_host: '/openstack'
  ....

Ceilometer Graphite publisher

ceilometer:
  server:
    enabled: true
  publisher:
    graphite:
      enabled: true
      host: 10.0.0.1
      port: 2003

Ceilometer compute agent

ceilometer:
  agent:
    enabled: true
    version: mitaka
    secret: pwd
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      tenant: service
      user: ceilometer
      password: pwd
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
      rabbit_ha_queues: true

Ceilometer instance discovery method

ceilometer:
  agent:
    ...
    discovery_method: naive
Keystone auth caching

```yaml
ceilometer:
  server:
    cache:
      members:
        - host: 10.10.10.10
          port: 11211
        - host: 10.10.10.11
          port: 11211
        - host: 10.10.10.12
          port: 11211
  agent:
    cache:
      members:
        - host: 10.10.10.10
          port: 11211
        - host: 10.10.10.11
          port: 11211
        - host: 10.10.10.12
          port: 11211
```

Enhanced logging with logging.conf

By default logging.conf is disabled.

**That is possible to enable per-binary logging.conf with new variables:**

- openstack_log_appender - set it to true to enable log_config_append for all OpenStack services;
- openstack_fluentd_handler_enabled - set to true to enable FluentHandler for all Openstack services.
- openstack_ossyslog_handler_enabled - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

```yaml
ceilometer:
  server:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
          enabled: true
        ossyslog:
          enabled: true
  agent:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
```

(continues on next page)
More Information

- https://wiki.openstack.org/wiki/Ceilometer
- http://docs.openstack.org/developer/ceilometer/
- https://github.com/spilgames/ceilometer_graphite_publisher

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-ceilometer/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-ceilometer

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Openstack Cinder Block Storage

Cinder provides an infrastructure for managing volumes in OpenStack. It was originally a Nova component called nova-volume, but has become an independent project since the Folsom release.

Sample pillars

New structure divides cinder-api,cinder-scheduler to role controller and cinder-volume to role volume.
cinder:
   controller:
      enabled: true
      version: juno
      cinder_uid: 304
      cinder_gid: 304
      nas_secure_file_permissions: false
      nas_secure_file_operations: false
      cinder_internal_tenant_user_id: f46924c112a14c80ab0a24a613d95eef
      cinder_internal_tenant_project_id: b7455b8974bb4064ad247c8f375eae6c
      default_volume_type: 7k2SaS
      enable_force_upload: true
      availability_zone_fallback: True
      database:
         engine: mysql
         host: 127.0.0.1
         port: 3306
         name: cinder
         user: cinder
         password: pwd
      identity:
         engine: keystone
         host: 127.0.0.1
         port: 35357
         tenant: service
         user: cinder
         password: pwd
      message_queue:
         engine: rabbitmq
         host: 127.0.0.1
         port: 5672
         user: openstack
         password: pwd
         virtual_host: '/openstack'
      backend:
         7k2_SAS:
            engine: storwize
            type_name: slow-disks
            host: 192.168.0.1
            port: 22
            user: username
            password: pass
            connection: FC/iSCSI
            multipath: true
            multipath: true
            pool: SAS7K2
         audit:
            enabled: false
         osapi_max_limit: 500
         barbican:
            enabled: true
   volume:
      enabled: true
      version: juno
      cinder_uid: 304
cinder_gid: 304
nas_secure_file_permissions: false
nas_secure_file_operations: false
cinder_internal_tenant_user_id: f46924c112a14c80ab0a24a613d95eeef
cinder_internal_tenant_project_id: b7455b8974bb4064ad247c8f375eae6c
default_volume_type: 7k2SAS
enable_force_upload: true
database:
  engine: mysql
  host: 127.0.0.1
  port: 3306
  name: cinder
  user: cinder
  password: pwd
identity:
  engine: keystone
  host: 127.0.0.1
  port: 35357
  tenant: service
  user: cinder
  password: pwd
message_queue:
  engine: rabbitmq
  host: 127.0.0.1
  port: 5672
  user: openstack
  password: pwd
  virtual_host: '/openstack'
backend:
  7k2_SAS:
    engine: storwize
    type_name: 7k2 SAS disk
    host: 192.168.0.1
    port: 22
    user: username
    password: pass
    connection: FC/iSCSI
    multipath: true
    pool: SAS7K2
audit:
  enabled: false
barbican:
  enabled: true

Enable CORS parameters

cinder:
  controller:
    cors:
      expose_headers: X-Auth-Token, X-Openstack-Request-Id, X-Subject-Token
      allow_methods: GET, PUT, POST, DELETE, PATCH
      allow_headers: X-Auth-Token, X-Openstack-Request-Id, X-Subject-Token
      allow_credentials: True
      max_age: 86400

2.1. Project Introduction
Client-side RabbitMQ HA setup for controller

```
cinder:
    controller:
        ....
        message_queue:
            engine: rabbitmq
            members:
                - host: 10.0.16.1
                - host: 10.0.16.2
                - host: 10.0.16.3
            user: openstack
            password: pwd
            virtual_host: '/openstack'
        ....
```

Client-side RabbitMQ HA setup for volume component

```
cinder:
    volume:
        ....
        message_queue:
            engine: rabbitmq
            members:
                - host: 10.0.16.1
                - host: 10.0.16.2
                - host: 10.0.16.3
            user: openstack
            password: pwd
            virtual_host: '/openstack'
        ....
```

Configuring TLS communications

**Note:** by default system wide installed CA certs are used, so `cacert_file` param is optional, as well as `cacert`.

- **RabbitMQ TLS**

```
cinder:
    controller, volume:
        message_queue:
            port: 5671
            ssl:
                enabled: True
                (optional) cacert: cert body if the cacert_file does not exists
                (optional) cacert_file: /etc/openstack/rabbitmq-ca.pem
                (optional) version: TLSv1_2
```

- **MySQL TLS**

```
cinder:
    controller:
        database:
            ssl:
                enabled: True
                (optional) cacert: cert body if the cacert_file does not exists
                (optional) cacert_file: /etc/openstack/mysql-ca.pem
```
• Openstack HTTPS API

```yaml
Cinder setup with zeroing deleted volumes
```

```yaml
Cinder setup with shredding deleted volumes
```

```yaml
Configuration of policy.json file
```

```yaml
Default Cinder setup with iSCSI target
```

(continues on next page)
message_queue:
    engine: rabbitmq
    host: 127.0.0.1
    port: 5672
    user: openstack
    password: pwd
    virtual_host: '/openstack'
backend:
    lvmdriver-1:
    engine: lvm
    type_name: lvmdriver-1
    volume_group: cinder-volume

Cinder setup for IBM Storwize

cinder:
    volume:
        enabled: true
    backend:
        7k2_SAS:
            engine: storwize
            type_name: 7k2 SAS disk
            host: 192.168.0.1
            port: 22
            user: username
            password: pass
            connection: FC/iSCSI
            multipath: true
            multihost: true
            pool: SAS7K2
        10k_SAS:
            engine: storwize
            type_name: 10k SAS disk
            host: 192.168.0.1
            port: 22
            user: username
            password: pass
            connection: FC/iSCSI
            multipath: true
            multihost: true
            pool: SAS10K
        15k_SAS:
            engine: storwize
            type_name: 15k SAS disk
            host: 192.168.0.1
            port: 22
            user: username
            password: pass
            connection: FC/iSCSI
            multipath: true
            multihost: true
            pool: SAS15K

Cinder setup with NFS

cinder:
    controller:
enabled: true

default_volume_type: nfs-driver

backend:
  nfs-driver:
    engine: nfs
    type_name: nfs-driver
    volume_group: cinder-volume
    path: /var/lib/cinder/nfs
    devices:
    - 172.16.10.110:/var/nfs/cinder
    options: rw,sync

Cinder setup with NetApp

cinder:
  controller:
    backend:
      netapp:
        engine: netapp
        type_name: netapp
        user: openstack
        vserver: vml
        server_hostname: 172.18.2.3
        password: password
        storage_protocol: nfs
        transport_type: https
        lun_space_reservation: enabled
        use_multipath_for_image_xfer: True
        nas_secure_file_operations: false
        nas_secure_file_permissions: false
        devices:
        - 172.18.1.2:/vol_1
        - 172.18.1.2:/vol_2
        - 172.18.1.2:/vol_3
        - 172.18.1.2:/vol_4

linux:
  system:
    package:
      nfs-common:
        version: latest

Cinder setup with Hitachi VPS

cinder:
  controller:
    enabled: true
    backend:
      hus100_backend:
        type_name: HUS100
        backend: hus100_backend
        engine: hitachi_vsp
        connection: FC

Cinder setup with Hitachi VPS with defined ldev range

cinder:
  controller:

(continues on next page)
Cinder setup with CEPH

```python
cinder:
    controller:
        enabled: true
    backend:
        ceph_backend:
            type_name: standard-iops
            backend: ceph_backend
            pool: volumes
            engine: ceph
            user: cinder
            secret_uuid: da74cc7-aa59-1721-a172-0006b1aa4e3e
            client_cinder_key: AQDOavlU6BsSJhAAnpFR906mvdgdfRqLHwu0Uw==
            report_discard_supported: True
```

http://ceph.com/docs/master/rbd/rbd-openstack/

Cinder setup with HP3par

```python
cinder:
    controller:
        enabled: true
    backend:
        hp3par_backend:
            type_name: hp3par
            backend: hp3par_backend
            user: hp3paruser
            password: something
            url: http://10.10.10.10/api/v1
            cpg: OpenStackCPG
            host: 10.10.10.10
            login: hp3paradmin
            sanpassword: something
            debug: True
            snapcpg: OpenStackSNAPCPG
```

Cinder setup with Fujitsu Eternus

```python
cinder:
    volume:
        enabled: true
    backend:
        10kThinPro:
            type_name: 10kThinPro
            engine: fujitsu
            pool: 10kThinPro
            host: 192.168.0.1
```

(continues on next page)
port: 5988
user: username
password: pass
connection: FC/iSCSI
name: 10kThinPro

10k_SAS:
  type_name: 10k_SAS
  pool: SAS10K
  engine: fujitsu
  host: 192.168.0.1
  port: 5988
  user: username
  password: pass
  connection: FC/iSCSI
  name: 10k_SAS

Cinder setup with IBM GPFS filesystem

cinder:
  volume:
    enabled: true
  backend:
    GPFS-GOLD:
      type_name: GPFS-GOLD
      engine: gpfs
      mount_point: '/mnt/gpfs-openstack/cinder/gold'
    GPFS-SILVER:
      type_name: GPFS-SILVER
      engine: gpfs
      mount_point: '/mnt/gpfs-openstack/cinder/silver'

Cinder setup with HP LeftHand

cinder:
  volume:
    enabled: true
  backend:
    HP-LeftHand:
      type_name: normal-storage
      engine: hp_lefthand
      api_url: 'https://10.10.10.10:8081/lhos'
      username: user
      password: password
      clustername: cluster1
      iscsi_chap_enabled: false

Extra parameters for HP LeftHand

cinder type-key normal-storage set hplh:data_pl=r-10-2 hplh:provisioning=full

Cinder setup with Solidfire

cinder:
  volume:
    enabled: true
  backend:
solidfire:
  type_name: normal-storage
  engine: solidfire
  san_ip: 10.10.10.10
  san_login: user
  san_password: password
  clustername: cluster1
  sf_emulate_512: false

Cinder setup with Block Device driver

cinder:
  volume:
    enabled: true
    backend:
      bdd:
        engine: bdd
        enabled: true
        type_name: bdd
        devices:
          - sdb
          - sdc
          - sdd

Enable cinder-backup service for ceph

cinder:
  controller:
    enabled: true
    version: mitaka
  backup:
    engine: ceph
    ceph_conf: "/etc/ceph/ceph.conf"
    ceph_pool: backup
    ceph_stripe_count: 0
    ceph_stripe_unit: 0
    ceph_user: cinder
    ceph_chunk_size: 134217728
    restore_discard_excess_bytes: false
  volume:
    enabled: true
    version: mitaka
    backup:
      engine: ceph
      ceph_conf: "/etc/ceph/ceph.conf"
      ceph_pool: backup
      ceph_stripe_count: 0
      ceph_stripe_unit: 0
      ceph_user: cinder
      ceph_chunk_size: 134217728
      restore_discard_excess_bytes: false

Enable auditing filter, ie: CADF

cinder:
  controller:
    audit:
Cinder setup with custom availability zones:

cinder:
controller:
default_availability_zone: my-default-zone
storage_availability_zone: my-custom-zone-name
cinder:
volume:
default_availability_zone: my-default-zone
storage_availability_zone: my-custom-zone-name

Cinder setup with custom non-admin volume query filters:

cinder:
controller:
query_volume_filters:
- name
- status
- metadata
- availability_zone
- bootable

public_endpoint and osapi_volume_base_url parameters: “public_endpoint” is used for configuring versions endpoint, “osapi_volume_base_URL” is used to present Cinder URL to users. They are useful when running Cinder under load balancer in SSL.

cinder:
controller:
public_endpoint_address: https://${_param:cluster_domain}:8776

The default availability zone is used when a volume has been created, without specifying a zone in the create request. (this zone must exist in your configuration obviously) The storage availability zone is the actual zone where the node belongs to. Make sure to specify this per node. Check the documentation of OpenStack for more information

Client role

cinder:
client:
enabled: true
identity:
  host: 127.0.0.1
  port: 35357
  project: service
  user: cinder
Enable Barbican integration

cinder:
    controller:
        barbican:
            enabled: true

Enhanced logging with logging.conf

By default logging.conf is disabled.

That is possible to enable per-binary logging.conf with new variables:

- openstack_log_appender - set it to true to enable log_config_append for all OpenStack services;
- openstack_fluentd_handler_enabled - set to true to enable FluentHandler for all Openstack services.
- openstack_ossyslog_handler_enabled - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

cinder:
    controller:
        logging:
            log_appender: true
            log_handlers:
                watchedfile:
                    enabled: true
                fluentd:
                    enabled: true
                ossyslog:
                    enabled: true

volume:
    logging:
        log_appender: true
        log_handlers:
            watchedfile:
                enabled: true
            fluentd:
                enabled: true
            ossyslog:
                enabled: true
**Documentation and Bugs**

To learn how to deploy OpenStack Salt, consult the documentation available online at:

https://wiki.openstack.org/wiki/OpenStackSalt

In the unfortunate event that bugs are discovered, they should be reported to the appropriate bug tracker. If you obtained the software from a 3rd party operating system vendor, it is often wise to use their own bug tracker for reporting problems. In all other cases use the master OpenStack bug tracker, available at:

http://bugs.launchpad.net/openstack-salt

Developers wishing to work on the OpenStack Salt project should always base their work on the latest formulas code, available from the master GIT repository at:

https://git.openstack.org/cgit/openstack/salt-formula-cinder

Developers should also join the discussion on the IRC list, at:

https://wiki.openstack.org/wiki/Meetings/openstack-salt

**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-cinder/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-cinder

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Designate formula**

Designate provides DNSaaS services for OpenStack.

**Sample pillars**

For Designate with BIND9 local backend:
designate:
    server:
        enabled: true
        region: RegionOne
        domain_id: 5186883b-91fb-4891-bd49-e6769234a8fc
        version: ocata
        backend:
            bind9:
                rndc_key: 4pc+X4PDqb2q+5o72dISm72LM1Ds9X2EY2jqq+nms7FhdTzwFFY81/icyEDmHxnyjkA33EQC8H+z0fLLBunoitw==
                rndc_algorithm: hmac-sha512
            bind:
                api:
                    address: 127.0.0.1
                database:
                    engine: mysql
                    host: 127.0.0.1
                    port: 3306
                    name:
                        main_database: designate
                        pool_manager: designate_pool_manager
                    user: designate
                    password: passw0rd
                identity:
                    engine: keystone
                    host: 127.0.0.1
                    port: 35357
                    tenant: service
                    user: designate
                    password: passw0rd
            message_queue:
                engine: rabbitmq
                members:
                    - host: 127.0.0.1
                    user: openstack
                    password: password
                    virtual_host: '/openstack'
    pools:
        default:
            description: 'default pool'
            attributes:
                service_tier: GOLD
            ns_records:
                - hostname: 'ns1.example.org.'
                priority: 10
            nameservers:
                - host: 127.0.0.1
                port: 53
            targets:
                default_target:
                    type: bind9
                    description: 'default target'
                    masters:
                        - host: 127.0.0.1
                        port: 5354
                    options:
                        host: 127.0.0.1
port: 53
rndc_host: 127.0.0.1
rndc_port: 953
rndc_key_file: /etc/designate/rndc.key

**Note:** *domain_id* parameter is UUID of DNS zone managed by designate-sink service. This zone will be populated by A records for fixed and floating ip addresses of spawned VMs. After designate is deployed and zone is created, this parameter should be updated accordingly to UUID of newly created zone. Then designate state should be reapplied.

**Usage**

Create server

```bash
designate server-create --name ns.example.com.
```

Create domain

```bash
designate domain-create --name example.com. --email mail@example.com
```

Create record
designate record-create example.com. --name test.example.com. --type A --data 10.2.14.15

Test it

dig @127.0.0.1 test.example.com.

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-letsencrypt/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-letsencrypt

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Glance formula

The Glance project provides services for discovering, registering, and retrieving virtual machine images. Glance has a RESTful API that allows querying of VM image metadata as well as retrieval of the actual image.

Sample pillars

glance:
server:
    enabled: true
    version: juno
    workers: 8
    glance_uid: 302
    glance_gid: 302
    policy:
        publicize_image:
            - "role:admin"
            - "role:image_manager"
    database:
        engine: mysql
        host: 127.0.0.1

(continues on next page)
- **port**: 3306
- **name**: glance
- **user**: glance
- **password**: pwd

**identity:**
- **engine**: keystone
- **host**: 127.0.0.1
- **port**: 35357
- **tenant**: service
- **user**: glance
- **password**: pwd

**message_queue:**
- **engine**: rabbitmq
- **host**: 127.0.0.1
- **port**: 5672
- **user**: openstack
- **password**: pwd
  - **virtual_host**: '/openstack'

**storage:**
- **engine**: file
- **images:**
  - **name**: "CirrOS 0.3.1"
    - **format**: qcow2
    - **file**: cirros-0.3.1-x86_64-disk.img
    - **source**: http://cdn.download.cirros-cloud.net/0.3.1/cirros-0.3.1-x86_64-disk.img
    - **public**: true

**audit:**
- **enabled**: false
- **api_limit_max**: 100
- **limit_param_default**: 50

**barbican:**
- **enabled**: true

The pagination is controlled by the **api_limit_max** and **limit_param_default** parameters as shown above:

- **api_limit_max** defines the maximum number of records that the server will return.
- **limit_param_default** is the default limit parameter that applies if the request didn’t defined it explicitly.

Configuration of policy.json file

```json
glance:
  server:
    ....
  policy:
    publicize_image: "role:admin"
    # Add key without value to remove line from policy.json
    add_member:
```

Keystone and cinder region

```json
glance:
  server:
    enabled: true
    version: kilo
  identity:
    engine: keystone
```

(continues on next page)
host: 127.0.0.1
region: RegionTwo
...

Ceph integration glance

```yaml
glance:
  server:
    enabled: true
    version: juno
    storage:
      engine: rbd,http
      user: glance
      pool: images
      chunk_size: 8
      client_glance_key: AQDOav1U6BsSjAAnpFR906mvdgdfRqLHwu0Uw==
```

RabbitMQ HA setup

```yaml
glance:
  server:
    ....
    message_queue:
      engine: rabbitmq
      members:
        - host: 10.0.16.1
        - host: 10.0.16.2
        - host: 10.0.16.3
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    ....
```

Quota Options

```yaml
glance:
  server:
    ....
    quota:
      image_member: -1
      image_property: 256
      image_tag: 256
      image_location: 15
      user_storage: 0
    ....
```

Configuring TLS communications

**Note:** by default system wide installed CA certs are used, so `cacert_file` param is optional, as well as `cacert`.

- RabbitMQ TLS

```yaml
glance:
  server:
    message_queue:
```

(continues on next page)
port: 5671
ssl:
  enabled: True
  (optional) cacert: cert body if the cacert_file does not exists
  (optional) cacert_file: /etc/openstack/rabbitmq-ca.pem
  (optional) version: TLSv1_2

• **MySQL TLS**

glance:
  server:
    database:
      ssl:
        enabled: True
        (optional) cacert: cert body if the cacert_file does not exists
        (optional) cacert_file: /etc/openstack/mysql-ca.pem

• **Openstack HTTPS API**

Set the `https` as protocol at `glance:server` sections:

glance:
  server:
    identity:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    registry:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    storage:
      engine: cinder, swift
      cinder:
        protocol: https
        (optional) cacert_file: /etc/openstack/proxy.pem
      swift:
        store:
          (optional) cafile: /etc/openstack/proxy.pem

Enable Glance Image Cache:

```
glance:
  server:
    image_cache:
      enabled: true
      enable_management: true
      directory: /var/lib/glance/image-cache/
      max_size: 21474836480
....
```

Enable auditing filter (CADF):

```
glance:
  server:
    audit:
      enabled: true
....
  filter_factory: 'keystonemiddleware.audit:filter_factory'
```
map_file: '/etc/pycadf/glance_api_audit_map.conf'

Swift integration glance

```yaml
glance:
  server:
    enabled: true
    version: mitaka
  storage:
    engine: swift,http
    swift:
      store:
        auth:
          address: http://keystone.example.com:5000/v2.0
          version: 2
          endpoint_type: publicURL
          container: glance
          create_container_on_put: true
          retry_get_count: 5
          user: 2ec7966596504f59acc3a76b3b9d9291:glance-user
          key: someRandomPassword
```

Another way, which also supports multiple swift backends, can be configured like this:

```yaml
glance:
  server:
    enabled: true
    version: mitaka
  storage:
    engine: swift,http
    swift:
      store:
        endpoint_type: publicURL
        container: glance
        create_container_on_put: true
        retry_get_count: 5
        references:
          my_objectstore_reference_1:
            auth:
              address: http://keystone.example.com:5000/v2.0
              version: 2
              user: 2ec7966596504f59acc3a76b3b9d9291:glance-user
              key: someRandomPassword
```

Enable CORS parameters

```yaml
glance:
  server:
    cors:
      allowed_origin: https:localhost.local,http:localhost.local
      expose_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_methods: GET,PUT,POST,DELETE,PATCH
      allow_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_credentials: True
      max_age: 86400
```
Enable Viewing Multiple Locations

If you want to expose all locations available (for example when you have multiple backends configured), then you can configure this like so:

```
glance:
  server:
    show_multiple_locations: True
    location_strategy: store_type
    store_type_preference: rbd,swift,file
```

Please note: the `show_multiple_locations` option is deprecated since Newton and is planned to be handled by policy files _only_ starting with the Pike release.

This feature is convenient in a scenario when you have swift and rbd configured and want to benefit from rbd enhancements.

Barbican integration glance

```
glance:
  server:
    barbican:
      enabled: true
```

Client role

Glance images

```
glance:
  client:
    enabled: true
  server:
    profile_admin:
      image:
        cirros-test:
          visibility: public
          protected: false
          location: http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-i386-disk.
```

Enhanced logging with logging.conf

By default logging.conf is disabled.

**That is possible to enable per-binary logging.conf with new variables:**

- `openstack_log_appender` - set it to true to enable `log_config_append` for all OpenStack services;
- `openstack_fluentd_handler_enabled` - set to true to enable FluentHandler for all Openstack services.
- `openstack_ossyslog_handler_enabled` - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:
Usage

Import new public image

```
$ glance image-create --name 'Windows 7 x86_64' --is-public true --container-format=bare --disk-format qcow2 < ./win7.qcow2
```

Change new image’s disk properties

```
$ glance image-update "Windows 7 x86_64" --property hw_disk_bus=ide
```

Change new image’s NIC properties

```
$ glance image-update "Windows 7 x86_64" --property hw_vif_model=rtl8139
```

External links

- http://ceph.com/docs/master/rbd/rbd-openstack/

Documentation and Bugs

To learn how to deploy OpenStack Salt, consult the documentation available online at:

https://wiki.openstack.org/wiki/OpenStackSalt

In the unfortunate event that bugs are discovered, they should be reported to the appropriate bug tracker. If you obtained the software from a 3rd party operating system vendor, it is often wise to use their own bug tracker for reporting problems. In all other cases use the master OpenStack bug tracker, available at:

http://bugs.launchpad.net/openstack-salt

Developers wishing to work on the OpenStack Salt project should always base their work on the latest formulas code, available from the master GIT repository at:

https://git.openstack.org/cgit/openstack/salt-formula-glance

Developers should also join the discussion on the IRC list, at:

https://wiki.openstack.org/wiki/Meetings/openstack-salt
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-glance/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-glance

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Heat Formula

Heat is the main project in the OpenStack Orchestration program. It implements an orchestration engine to launch multiple composite cloud applications based on templates in the form of text files that can be treated like code. A native Heat template format is evolving, but Heat also endeavours to provide compatibility with the AWS CloudFormation template format, so that many existing CloudFormation templates can be launched on OpenStack. Heat provides both an OpenStack-native ReST API and a CloudFormation-compatible Query API.

Sample Pillars

Single Heat services on the controller node

```yaml
heat:
  server:
    enabled: true
    version: icehouse
    region: RegionOne
    bind:
      metadata:
        address: 10.0.106.10
        port: 8000
        protocol: http
      waitcondition:
        address: 10.0.106.10
        port: 8000
        protocol: http
    watch:
      address: 10.0.106.10
      port: 8003
      protocol: http
```

(continues on next page)
Define server clients keystone parameter

```yaml
heat:
  server:
    clients:
      keystone:
        protocol: https
        host: 10.0.106.10
        port: 5000
        insecure: false
```

Enable CORS parameters

```yaml
heat:
  server:
    cors:
      expose_headers: X-Auth-Token, X-Openstack-Request-Id, X-Subject-Token
      allow_methods: GET, PUT, POST, DELETE, PATCH
      allow_headers: X-Auth-Token, X-Openstack-Request-Id, X-Subject-Token
      allow_credentials: True
      max_age: 86400
```

Heat client with specified git templates
heat:
  client:
    enabled: true
  template:
    admin:
      domain: default
      source:
        engine: git
        address: git@repo.domain.com/admin-templates.git
        revision: master
    default:
      domain: default
      source:
        engine: git
        address: git@repo.domain.com/default-templates.git
        revision: master

Ceilometer notification

heat:
  server:
    enabled: true
    version: icehouse
    notification: true

Configuration of policy.json file

heat:
  server:
    ....
    policy:
      deny_stack_user: 'not role:heat_stack_user'
      'cloudformation:ValidateTemplate': 'rule:deny_stack_user'
    # Add key without value to remove line from policy.json
    'cloudformation:DescribeStackResource':

Client-side RabbitMQ HA setup

heat:
  server:
    ....
    message_queue:
      engine: rabbitmq
      members:
      - host: 10.0.16.1
      - host: 10.0.16.2
      - host: 10.0.16.3
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    ....

Configuring TLS communications

Note: by default system wide installed CA certs are used, so cacert_file param is optional, as well as cacert.
• **RabbitMQ TLS**

```yaml
heat:
  server:
    message_queue:
      port: 5671
      ssl:
        enabled: True
        (optional) cacert: cert body if the cacert_file does not exists
        (optional) cacert_file: /etc/openstack/rabbitmq-ca.pem
        (optional) version: TLSv1_2
```

• **MySQL TLS**

```yaml
heat:
  server:
    database:
      ssl:
        enabled: True
        (optional) cacert: cert body if the cacert_file does not exists
        (optional) cacert_file: /etc/openstack/mysql-ca.pem
```

• **Openstack HTTPS API**

```yaml
heat:
  server:
    identity:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    clients:
      keystone:
        protocol: https
        (optional) cacert_file: /etc/openstack/proxy.pem
```

**Enhanced logging with logging.conf**

By default logging.conf is disabled.

That is possible to enable per-binary logging.conf with new variables:

- openstack_log_appender - set it to true to enable log_config_append for all OpenStack services;
- openstack_fluentd_handler_enabled - set to true to enable FluentHandler for all Openstack services.
- openstack_ossyslog_handler_enabled - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

```yaml
heat:
  server:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
      fluentd:
```

(continues on next page)
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-heat/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-heat

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Horizon Formula

Horizon is the canonical implementation of OpenStack’s Dashboard, which provides a web based user interface to OpenStack services including Nova, Swift, Keystone, etc.

Sample Pillars

Simplest horizon setup

```yaml
horizon:
  server:
    enabled: true
    secret_key: secret
  host:
    name: cloud.lab.cz
  cache:
    engine: 'memcached'
    host: '127.0.0.1'
    port: 11211
    prefix: 'CACHE_HORIZON'
  api_versions:
    identity: 2
    identity:
      engine: 'keystone'
```

(continues on next page)
Multidomain setup for horizon

```
horizon:
    server:
        enabled: true
        default_domain: MYDOMAIN
        multidomain: True
```

Simple branded horizon

```
horizon:
    server:
        enabled: true
        branding: 'OpenStack Company Dashboard'
        default_dashboard: 'admin'
        help_url: 'http://doc.domain.com'
```

Horizon with policy files metadata. With source mine you can obtain real time policy file state from targeted node (OpenStack control node), provided you have policy file published to specified grain key. Source file will obtain static policy definition from formula files directory.

```
horizon:
    server:
        enabled: true
        policy:
            identity:
                source: mine
                host: ctl01.my-domain.local
                name: keystone_policy.json
                grain_name: keystone_policy
                enabled: true
            compute:
                source: file
                name: nova_policy.json
                enabled: true
            network:
                source: file
                name: neutron_policy.json
                enabled: true
            image:
                source: file
                name: glance_policy.json
                enabled: true
            volume:
                source: file
                name: cinder_policy.json
                enabled: true
            telemetry:
                source: file
                name: ceilometer_policy.json
                enabled: true
```
Horizon with enabled SSL security (when SSL is realised by proxy)

```yaml
horizon:
  server:
    enabled: True
    secure: True
```

Horizon package setup with SSL

```yaml
horizon:
  server:
    enabled: true
    secret_key: MEGASECRET
    version: juno
    ssl:
      enabled: true
      authority: CA_Authority
    host:
      name: cloud.lab.cz
    cache:
      engine: 'memcached'
      host: '127.0.0.1'
      port: 11211
      prefix: 'CACHE_HORIZON'
    api_versions:
      identity: 2
      identity:
        engine: 'keystone'
        host: '127.0.0.1'
        port: 5000
    mail:
      host: '127.0.0.1'
```

Horizon with custom SESSION_ENGINE (default is “signed_cookies”, valid options are: “signed_cookies”, “cache”, “file”) and SESSION_TIMEOUT

```yaml
horizon:
  server:
    enabled: True
    secure: True
    session:
      engine: 'cache'
      timeout: 43200
```

Multi-regional horizon setup

```yaml
horizon:
  server:
    enabled: true
    version: juno
    secret_key: MEGASECRET
```

(continues on next page)
cache:
  engine: 'memcached'
  host: '127.0.0.1'
  port: 11211
  prefix: 'CACHE_HORIZON'
api_versions:
  identity: 2
  identity:
    engine: 'keystone'
    host: '127.0.0.1'
    port: 5000
mail:
  host: '127.0.0.1'
regions:
- name: cluster1
  address: http://cluster1.example.com:5000/v2.0
- name: cluster2
  address: http://cluster2.example.com:5000/v2.0

Horizon setup with sensu plugin

horizon:
  server:
    enabled: true
    version: juno
  sensu_api:
    host: localhost
    port: 4567
  plugin:
    monitoring:
      app: horizon_monitoring
      source:
        type: git
        address: git@repo1.robotice.cz:django/horizon-monitoring.git
      rev: develop

Sensu multi API

horizon:
  server:
    enabled: true
    version: juno
  sensu_api:
    dc1:
      host: localhost
      port: 4567
    dc2:
      host: anotherhost
      port: 4567

Horizon setup with jenkins plugin

horizon:
  server:
    enabled: true
    version: juno
  jenkins_api:
Horizon setup with billometer plugin

```yaml
horizon:
  server:
    enabled: true
    version: juno
  billometer_api:
    host: localhost
    port: 9753
    api_version: 1
  plugin:
    billing:
      app: horizon_billing
      source:
        type: git
        address: git@repo1.robotice.cz:django/horizon-billing.git
        rev: develop
```

Horizon setup with contrail plugin

```yaml
horizon:
  server:
    enabled: true
    version: icehouse
  plugin:
    contrail:
      app: contrail_openstack_dashboard
      override: true
      source:
        type: git
        address: git@repo1.robotice.cz:django/horizon-contrail.git
        rev: develop
```

Horizon setup with sentry log handler

```yaml
horizon:
  server:
    enabled: true
    version: juno
  ...'
  logging:
    engine: raven
    dsn: http://pub:private@sentry1.test.cz/2
```

**Multisite with Git source**

Simple Horizon setup from git repository

2.1. Project Introduction
Themed multisite setup

horizon:
  server:
    enabled: true
  app:
    openstack1c:
      secret_key: MEGASECRET1
      source:
        engine: git
        address: https://github.com/openstack/horizon.git
        rev: stable/havana
      plugin:
        contrail:
          app: contrail_openstack_dashboard
          override: true
          source:
            type: git
            address: git@repo1.robotice.cz:django/horizon-contrail.git
            rev: develop
      theme:
        app: site1_theme
        source:
          type: git
          address: git@repo1.domain.com:django/horizon-site1-theme.git
  cache:
    engine: 'memcached'
    host: '127.0.0.1'
    port: 11211
    prefix: 'CACHE_SITE1'
  api_versions:
    identity: 2
    identity:
      engine: 'keystone'
      host: '127.0.0.1'
      port: 5000
    mail:
      host: '127.0.0.1'

(continues on next page)
engine: 'keystone'
host: '127.0.0.1'
port: 5000
mail:
  host: '127.0.0.1'

openstack2:
  secret_key: MEGASECRET2
  source:
    engine: git
    address: https://repo1.domain.com/openstack/horizon.git
    rev: stable/icehouse
  plugin:
    contrail:
      app: contrail_openstack_dashboard
      override: true
      source:
        type: git
        address: git@repo1.domain.com:django/horizon-contrail.git
        rev: develop
  monitoring:
    app: horizon_monitoring
    source:
      type: git
      address: git@domain.com:django/horizon-monitoring.git
      rev: develop
  theme:
    app: bootswatch_theme
    source:
      type: git
      address: git@repo1.robotice.cz:django/horizon-bootswatch-theme.git
      rev: develop

cache:
  engine: 'memcached'
  host: '127.0.0.1'
  port: 11211
  prefix: 'CACHE_SITE2'

api_versions:
  identity: 3

2.1. Project Introduction
Control dashboard behaviour

```
horizon:
  server:
    enabled: true
  app:
    openstack_dashboard_override:
      secret_key: password
    dashboards:
      enabled: true
    settings:
      project:
        enabled: false
        order: 10
      admin:
        enabled: false
        order: 20
    source:
      engine: git
      address: https://github.com/openstack/horizon.git
      rev: stable/juno
```

Enable WebSSO feature

```
horizon:
  server:
    enabled: true
  websso:
    login_url: "WEBROOT + 'auth/login/'"
    logout_url: "WEBROOT + 'auth/logout/'"
    websso_choices:
      - saml2
      - oidc
```

More Information

- https://github.com/openstack/horizon

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-horizon/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:
You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-horizon

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**OpenStack Keystone**

Keystone provides authentication, authorization and service discovery mechanisms via HTTP primarily for use by projects in the OpenStack family. It is most commonly deployed as an HTTP interface to existing identity systems, such as LDAP.

From Kilo release Keystone v3 endpoint has definition without version in url

<table>
<thead>
<tr>
<th>id</th>
<th>region</th>
<th>publicurl</th>
<th>adminurl</th>
<th>service_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>91663a8db11c487c9253c8c456863494</td>
<td>RegionOne</td>
<td><a href="http://10.0.150.37:5000/">http://10.0.150.37:5000/</a></td>
<td><a href="http://10.0.150.37:5000/">http://10.0.150.37:5000/</a></td>
<td>0fd2dba3153d45a1ba7f709cfc2d69c9</td>
</tr>
</tbody>
</table>

**Sample pillars**

**Caution:** When you use localhost as your database host (keystone:server: atabase:host), sqlalchemy will try to connect to /var/run/mysql/ mysqld.sock, may cause issues if you located your mysql socket elsewhere

Full stacked keystone

```yaml
keystone:
  server:
    enabled: true
    version: juno
    service_token: 'service_token'
    service_tenant: service
    service_password: 'servicepwd'
    admin_tenant: admin
    admin_name: admin
    admin_password: 'adminpwd'
    admin_email: stackmaster@domain.com
    roles:
      - admin
      - Member
```

(continues on next page)
- image_manager
  bind:
    address: 0.0.0.0
    private_address: 127.0.0.1
    private_port: 35357
    public_address: 127.0.0.1
    public_port: 5000
  api_version: 2.0
  region: RegionOne
  database:
    engine: mysql
    host: '127.0.0.1'
    name: 'keystone'
    password: 'LfTno5mYdZmRfoPV'
    user: 'keystone'

Keystone public HTTPS API

keystone:
  server:
    enabled: true
    version: juno
    ... 
  services:
    - name: nova
      type: compute
      description: OpenStack Compute Service
      user:
        name: nova
        password: password
      bind:
        public_address: cloud.domain.com
        public_protocol: https
        public_port: 8774
        internal_address: 10.0.0.20
        internal_port: 8774
        admin_address: 10.0.0.20
        admin_port: 8774

Keystone with custom policies. Keys with specified rules are created or set to this value if they already exist. Keys with no value (like our “existing_rule”) are deleted from the policy file.

keystone:
  server:
    enabled: true
    policy:
      new_rule: "rule:admin_required"
      existing_rule:

Keystone memcached storage for tokens

keystone:
  server:
    enabled: true
    version: juno
    ...
token_store: cache
cache:
  engine: memcached
  host: 127.0.0.1
  port: 11211
services:
...

Keystone clustered memcached storage for tokens

keystone:
  server:
    enabled: true
    version: juno
    ...
  token_store: cache
cache:
  engine: memcached
  members:
  - host: 192.160.0.1
    port: 11211
  - host: 192.160.0.2
    port: 11211
services:
...

Keystone client

keystone:
  client:
    enabled: true
    server:
      host: 10.0.0.2
      public_port: 5000
      private_port: 35357
      service_token: 'token'
      admin_tenant: admin
      admin_name: admin
      admin_password: 'passwd'

Keystone cluster

keystone:
  control:
    enabled: true
    provider:
      os15_token:
        host: 10.0.0.2
        port: 35357
        token: token
      os15_tcp_core_stg:
        host: 10.0.0.5
        port: 5000
        tenant: admin
        name: admin
        password: password
Keystone fernet tokens for OpenStack Kilo release

```yaml
keystone:
  server:
    ...  
    tokens:
      engine: fernet
      max_active_keys: 3
    ...
```

Keystone auth methods

```yaml
keystone:
  server:
    ...
    auth_methods:
      - external
      - password
      - token
      - oauth1
    ...
```

Keystone domain with LDAP backend, using SQL for role/project assignment

```yaml
keystone:
  server:
    domain:
      external:
        description: "Testing domain"
        backend: ldap
        assignment:
          backend: sql
          ldap:
            url: "ldaps://idm.domain.com"
            suffix: "dc=cloud,dc=domain,dc=com"
            # Will bind as uid=keystone,cn=users,cn=accounts,dc=cloud,dc=domain,dc=com
            uid: keystone
            password: password
```

Using LDAP backend for default domain

```yaml
keystone:
  server:
    backend: ldap
    assignment:
      backend: sql
    ldap:
      url: "ldaps://idm.domain.com"
      suffix: "dc=cloud,dc=domain,dc=com"
      # Will bind as uid=keystone,cn=users,cn=accounts,dc=cloud,dc=domain,dc=com
      uid: keystone
      password: password
```

Using LDAP backend for default domain with “user_enabled” field emulation

```yaml
keystone:
  server:
    backend: ldap
```
assignment:
  backend: sql
ldap:
  url: "ldap://idm.domain.com"
  suffix: "ou=Openstack Service Users,o=domain.com"
  bind_user: keystone
  password: password

# Define LDAP "group" object class and "membership" attribute
  group_objectclass: groupOfUniqueNames
  group_member_attribute: uniqueMember

# User will receive "enabled" attribute basing on membership in "os-user-enabled" group
  user_enabled_emulation: True
  user_enabled_emulation_dn: "cn=os-user-enabled,ou=Openstack,o=domain.com"
  user_enabled_emulation_use_group_config: True

Simple service endpoint definition (defaults to RegionOne)

```yaml
keystone:
  server:
    service:
      ceilometer:
        type: metering
        description: OpenStack Telemetry Service
        user:
          name: ceilometer
          password: password
        bind:
          ...
```

Region-aware service endpoints definition

```yaml
keystone:
  server:
    service:
      ceilometer_region01:
        service: ceilometer
        type: metering
        region: region01
        description: OpenStack Telemetry Service
        user:
          name: ceilometer
          password: password
        bind:
          ...
      ceilometer_region02:
        service: ceilometer
        type: metering
        region: region02
        description: OpenStack Telemetry Service
        bind:
          ...
```

Enable ceilometer notifications

```yaml
keystone:
  server:
    ...
```

(continues on next page)
notification: true
message_queue:
  engine: rabbitmq
  host: 127.0.0.1
  port: 5672
  user: openstack
  password: password
  virtual_host: '/openstack'
  ha_queues: true

Client-side RabbitMQ HA setup

```
keystone:
  server:
    ....
    message_queue:
      engine: rabbitmq
      members:
        - host: 10.0.16.1
        - host: 10.0.16.2
        - host: 10.0.16.3
        user: openstack
        password: pwd
        virtual_host: '/openstack'
    ....
```

Client-side RabbitMQ TLS configuration:

By default system-wide CA certs are used. Nothing should be specified except ssl.enabled.

```
keystone:
  server:
    ....
    message_queue:
      ssl:
        enabled: True
```

Use cacert_file option to specify the CA-cert file path explicitly:

```
keystone:
  server:
    ....
    message_queue:
      ssl:
        enabled: True
        cacert_file: /etc/ssl/rabbitmq-ca.pem
```

To manage content of the cacert_file use the cert option:

```
keystone:
  server:
    ....
```

(continued on next page)
message_queue:
  ssl:
    enabled: True
    cacert: |
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE------
  cacert_file: /etc/openstack/rabbitmq-ca.pem

Notice:

- The message_queue.port is set to **5671** (AMQPS) by default if ssl.enabled=True.
- Use message_queue.ssl.version if you need to specify protocol version. By default is TLSv1 for python < 2.7.9 and TLSv1_2 for version above.

Enable CADF audit notification

```yaml
keystone:
  server:
    notification: true
    notification_format: cadf
```

Run keystone under Apache

```yaml
keystone:
  server:
    service_name: apache2
apache:
  server:
    enabled: true
    default_mpm: event
  site:
    keystone:
      enabled: true
      type: keystone
      name: wsgi
    host:
      name: ${linux:network:fqdn}
  modules:
    - wsgi
```

Enable SAML2 Federated keystone

```yaml
keystone:
  server:
    auth_methods:
      - password
      - token
      - saml2
  federation:
    saml2:
      protocol: saml2
      remote_id_attribute: Shib-Identity-Provider
      shib_url_scheme: https
```

(continues on next page)
Enable OIDC Federated keystone

```yaml
shib_compat_valid_user: 'on'
federation_driver: keystone.contrib.federation.backends.sql.Federation
federated_domain_name: Federated
trusted_dashboard:
  - https://${_param:cluster_public_host}/horizon/auth/websso/
apache:
  server:
    pkgs:
      - apache2
      - libapache2-mod-shib2
    modules:
      - wsgi
      - shib2
```

```
keystone:
  server:
    auth_methods:
      - password
      - token
      - oidc
    federation:
      oidc:
        protocol: oidc
        remote_id_attribute: HTTP_OIDC_ISS
        remote_id_attribute_value: https://accounts.google.com
        oidc_claim_prefix: "OIDC-"
        oidc_response_type: id_token
        oidc_scope: "openid email profile"
        oidc_provider_metadata_url: https://accounts.google.com/.well-known/openid-configuration
        oidc_client_id: <openid_client_id>
        oidc_client_secret: <openid_client_secret>
        oidc_crypto_passphrase: openstack
        oidc_redirect_uri: https://key.example.com:5000/v3/auth/OS-FEDERATION/websso/
        oidc/oauth_introspection_endpoint: https://www.googleapis.com/oauth2/v1/tokeninfo
        oidc_oauth_remote_user_claim: user_id
        oidc_ssl_validate_server: 'off'
    federated_domain_name: Federated
    federation_driver: keystone.contrib.federation.backends.sql.Federation
    trusted_dashboard:
      - https://${_param:cluster_public_host}/auth/websso/
apache:
  server:
    pkgs:
      - apache2
      - libapache2-mod-auth-openidc
    modules:
      - wsgi
      - auth_openidc
```

Notes: Ubuntu Trusty repository doesn’t contain libapache2-mod-auth-openidc package. Additional repository should be added to source list.
Use a custom identity driver with custom options

```yaml
keystone:
  server:
    backend: k2k
  k2k:
    auth_url: 'https://keystone.example.com/v2.0'
    read_user: 'example_user'
    read_pass: 'password'
    read_tenant_id: 'admin'
    identity_driver: 'sql'
    id_prefix: 'k2k:'
    domain: 'default'
    caching: true
    cache_time: 600
```

Enable CORS parameters

```yaml
keystone:
  server:
    cors:
      allowed_origin: https:localhost.local,http:localhost.local
      expose_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_methods: GET,PUT,POST,DELETE,PATCH
      allow_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_credentials: True
      max_age: 86400
```

**Keystone client**

Service endpoints enforcement with service token

```yaml
keystone:
  client:
    enabled: true
  server:
    keystone01:
      admin:
        host: 10.0.0.2
        port: 35357
        token: 'service_token'
      service:
        nova:
          type: compute
          description: OpenStack Compute Service
          endpoints:
            region: region01
              public_address: 172.16.10.1
              public_port: 8773
              public_path: '/v2'
              internal_address: 172.16.10.1
              internal_port: 8773
              internal_path: '/v2'
              admin_address: 172.16.10.1
              admin_port: 8773
              admin_path: '/v2'
```
Project, users, roles enforcement with admin user

```yaml
keystone:
  client:
    enabled: true
  server:
    keystone01:
      admin:
        host: 10.0.0.2
        port: 5000
        project: admin
        user: admin
        password: 'passwd'
        region_name: RegionOne
        protocol: https
        roles:
          - admin
          - member
  project:
    tenant01:
      description: "test env"
      quota:
        instances: 100
        cores: 24
        ram: 151200
        floating_ips: 50
        fixed_ips: -1
        metadata_items: 128
        injected_files: 5
        injected_file_content_bytes: 10240
        injected_file_path_bytes: 255
        key_pairs: 100
        security_groups: 20
        security_group_rules: 40
        server_groups: 20
        server_group_members: 20
  user:
    user01:
      email: jdoe@domain.com
      is_admin: true
      password: some
    user02:
      email: jdoe2@domain.com
      password: some
      roles:
        - custom-roles
```

Multiple servers example

```yaml
keystone:
  client:
    enabled: true
  server:
    keystone01:
      admin:
        host: 10.0.0.2
        port: 5000
        project: 'admin'
```

(continues on next page)
user: admin
password: 'workshop'
region_name: RegionOne
protocol: https

keystone02:
    admin:
        host: 10.0.0.3
        port: 5000
        project: 'admin'
        user: admin
        password: 'workshop'
        region_name: RegionOne

Tenant quotas

keystone:
    client:
        enabled: true
    server:
        keystone01:
            admin:
                host: 10.0.0.2
                port: 5000
                project: admin
                user: admin
                password: 'passwd'
                region_name: RegionOne
                protocol: https
                roles:
                    - admin
                    - member
    project:
        tenant01:
            description: "test env"
            quota:
                instances: 100
                cores: 24
                ram: 151200
                floating_ips: 50
                fixed_ips: -1
                metadata_items: 128
                injected_files: 5
                injected_file_content_bytes: 10240
                injected_file_path_bytes: 255
                key_pairs: 100
                security_groups: 20
                security_group_rules: 40
                server_groups: 20
                server_group_members: 20

Extra config params in keystone.conf (since Mitaka release)

keystone:
    server:
        ....
        extra_config:
            ini_section1:
param1: value
param2: value
ini_section2:
  param1: value
  param2: value
...

Configuration of policy.json file

keystone:
  server:
    ...
    policy:
      admin_or_token_subject: 'rule:admin_required or rule:token_subject'

Setting up default admin project name and domain

keystone:
  server:
    ...
    admin_project:
      name: "admin"
      domain: "default"

Enhanced logging with logging.conf

By default logging.conf is disabled.

That is possible to enable per-binary logging.conf with new variables:

- openstack_log_appender - set it to true to enable log_config_append for all OpenStack services;
- openstack_fluentd_handler_enabled - set to true to enable FluentHandler for all Openstack services.
- openstack_ossyslog_handler_enabled - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

keystone:
  server:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
          enabled: true
        ossyslog:
          enabled: true

Usage

Apply state `keystone.client.service` first and then `keystone.client` state.
Documentation and Bugs

To learn how to deploy OpenStack Salt, consult the documentation available online at:

https://wiki.openstack.org/wiki/OpenStackSalt

In the unfortunate event that bugs are discovered, they should be reported to the appropriate bug tracker. If you obtained the software from a 3rd party operating system vendor, it is often wise to use their own bug tracker for reporting problems. In all other cases use the master OpenStack bug tracker, available at:

http://bugs.launchpad.net/openstack-salt

Developers wishing to work on the OpenStack Salt project should always base their work on the latest formulas code, available from the master GIT repository at:

https://git.openstack.org/cgit/openstack/salt-formula-keystone

Developers should also join the discussion on the IRC list, at:

https://wiki.openstack.org/wiki/Meetings/openstack-salt

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-keystone/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-keystone

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

magnum

Service magnum description

Sample pillars

Single magnum service
Read more

• links

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-magnum/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-magnum

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Midonet

MidoNet is an advanced Software Defined Networking (SDN) solution, which provides network virtualization for public and private cloud environments.

Sample pillars

Cluster Control

midonet:
   control:
      version: v5.0
      enterprise:
         enabled: true
         enabled: true
         host: 127.0.0.1
         nova:
         (continues on next page)
```yaml
control:
  host: 127.0.0.1
database:
  members:
  - host: 127.0.0.1
    port: 9160
  - host: 127.0.0.1
    port: 9160
  - host: 127.0.0.1
    port: 9160
zookeeper:
  members:
  - host: 127.0.0.1
  - host: 127.0.0.1
  - host: 127.0.0.1

identity:
  user: midonet
  password: passwd

admin:
  host: 127.0.0.1
  admin:
    token: tokenpass
    password: passwd
```

### Analytics

```yaml
midonet:
  analytics:
    version: v5.0
    enterprise:
      enabled: true
      enabled: true
      host: 127.0.0.1
```

### Gateway

```yaml
midonet:
  gateway:
    version: v5.0
    enterprise:
      enabled: true
      enabled: true
    zookeeper:
      members:
      - host: 127.0.0.1
      - host: 127.0.0.1
      - host: 127.0.0.1
    template: medium
```
Compute

```
midonet:
compute:
    version: v5.0
    enterprise:
        enabled: true
        enabled: true
    zookeeper:
        members:
            - host: 127.0.0.1
            - host: 127.0.0.1
            - host: 127.0.0.1
    template: medium
```

Web

```
midonet:
    web:
        version: v5.0
        enabled: true
        api:
            host: 127.0.0.1
        analytics:
            host: 127.0.0.1
```

Read More

- http://www.midokura.com/midonet/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-midonet/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-midonet

Any questions or feedback is always welcome so feel free to join our IRC channel:
Murano Project introduces an application catalog, which allows application developers and cloud administrators to publish various cloud-ready applications in a browsable categorised catalog, which may be used by the cloud users (including the inexperienced ones) to pick-up the needed applications and services and composes the reliable environments out of them in a “push-the-button” manner.

Sample pillars

Single murano services on the controller node

```
murano:
    server:
        enabled: true
        version: liberty
        insecure: false
        database:
            engine: mysql
            host: 10.10.20.20
            port: 3306
            name: murano
            user: murano
            password: password
        identity:
            engine: keystone
            host: 10.10.20.20
            port: 35357
            tenant: service
            user: murano
            password: password
        message_queue:
            engine: rabbitmq
            members:
                - host: 192.168.1.13
                - host: 192.168.1.14
                - host: 192.168.1.15
            user: openstack
            password: supersecret
            virtual_host: '/openstack'
    murano_agent_queue:
        engine: rabbitmq
        port: 5672
        host: 192.168.1.10
        user: openstack
        password: supersecretcatalogpassword
```
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-murano/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-murano

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

OpenContrail Formula

Contrail Controller is an open, standards-based software solution that delivers network virtualization and service automation for federated cloud networks. It provides self-service provisioning, improves network troubleshooting and diagnostics, and enables service chaining for dynamic application environments across enterprise virtual private cloud (VPC), managed Infrastructure as a Service (IaaS), and Networks Functions Virtualization (NFV) use cases.

Package source

Formula support OpenContrail as well as Juniper Contrail package repository in the backend.

Differences within the configuration and state run are controlled by opencontrail.common.vendor: [opencontrail|juniper] pillar attribute.

Default value is set to opencontrail.

Juniper releases tested with this formula:

- 3.0.2.x

To use Juniper Contrail repository as a source of packages override pillar as in this example:

```yaml
opencontrail:
  common:
    vendor: juniper
```
Sample Pillars

Controller nodes

There are several scenarios for OpenContrail control plane.

All-in-one single

Config, control, analytics, database, web – altogether on one node.

```yaml
opencontrail:
  common:
    version: 2.2
    source:
      engine: pkg
      address: http://mirror.robotice.cz/contrail-havana/
  identity:
    engine: keystone
    host: 127.0.0.1
    port: 35357
    token: token
    password: password
  network:
    engine: neutron
    host: 127.0.0.1
    port: 9696
  config:
    version: 2.2
    enabled: true
    network:
      engine: neutron
      host: 127.0.0.1
      port: 9696
  discovery:
    host: 127.0.0.1
  analytics:
    host: 127.0.0.1
  bind:
    address: 127.0.0.1
  message_queue:
    engine: rabbitmq
    host: 127.0.0.1
    port: 5672
  database:
    members:
      - host: 127.0.0.1
        port: 9160
  cache:
    members:
      - host: 127.0.0.1
        port: 11211
  identity:
    engine: keystone
    version: '2.0'
    region: RegionOne
```

(continues on next page)
host: 127.0.0.1
port: 35357
user: admin
password: password
token: token
tenant: admin
members:
- host: 127.0.0.1
  id: 1
rootlogger: "INFO, CONSOLE"
control:
  version: 2.2
  enabled: true
  bind:
    address: 127.0.0.1
discovery:
  host: 127.0.0.1
master:
  host: 127.0.0.1
members:
- host: 127.0.0.1
  id: 1
collector:
  version: 2.2
  enabled: true
  bind:
    address: 127.0.0.1
master:
  host: 127.0.0.1
discovery:
  host: 127.0.0.1
data_ttl: 2
database:
  members:
- host: 127.0.0.1
  port: 9160
database:
  version: 2.2
cassandra:
  version: 2
  enabled: true
  minimum_disk: 10
  name: 'Contrail'
  original_token: 0
  compaction_throughput_mb_per_sec: 16
  concurrent_compactors: 1
  data_dirs:
- /var/lib/cassandra
member:
  host: 127.0.0.1
bind:
  host: 127.0.0.1
  port: 9042
  rpc_port: 9160
members:
- host: 127.0.0.1
id: 1

web:
  version: 2.2
  enabled: True
  bind:
    address: 127.0.0.1

analytics:
  host: 127.0.0.1

master:
  host: 127.0.0.1

bind:
  host: 127.0.0.1

cache:
  host: 127.0.0.1
  port: 6379

members:
  - host: 127.0.0.1
    id: 1

identity:
  engine: keystone
  version: '2.0'
  host: 127.0.0.1
  port: 35357
  user: admin
  password: password
  token: token
  tenant: admin

All-in-one cluster

Config, control, analytics, database, web – altogether, clustered on multiple nodes.

opencontrail:
  common:
    version: 2.2
    source:
      engine: pkg
      address: http://mirror.robotice.cz/contrail-havana/
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      token: token
      password: password
    network:
      engine: neutron
      host: 127.0.0.1
      port: 9696

config:
  version: 2.2
  enabled: true
  network:
    engine: neutron
    host: 127.0.0.1
    port: 9696
  discovery:
host: 127.0.0.1
analytics:
  host: 127.0.0.1
bind:
  address: 127.0.0.1
message_queue:
  engine: rabbitmq
  host: 127.0.0.1
  port: 5672
database:
  members:
  - host: 127.0.0.1
    port: 9160
  - host: 127.0.0.1
    port: 9160
  - host: 127.0.0.1
    port: 9160
cache:
  members:
  - host: 127.0.0.1
    port: 11211
  - host: 127.0.0.1
    port: 11211
  - host: 127.0.0.1
    port: 11211
identity:
  engine: keystone
  version: '2.0'
  region: RegionOne
  host: 127.0.0.1
  port: 35357
  user: admin
  password: password
  token: token
  tenant: admin
  members:
  - host: 127.0.0.1
    id: 1
  - host: 127.0.0.1
    id: 2
  - host: 127.0.0.1
    id: 3
control:
  version: 2.2
  enabled: true
  bind:
    address: 127.0.0.1
discovery:
  host: 127.0.0.1
master:
  host: 127.0.0.1
members:
  - host: 127.0.0.1
    id: 1
  - host: 127.0.0.1
    id: 2
  - host: 127.0.0.1
  - host: 127.0.0.1

(continues on next page)
id: 3

collector:
  version: 2.2
  enabled: true
  bind: address: 127.0.0.1
  master: host: 127.0.0.1
discovery:
  host: 127.0.0.1
data_ttl: 1
database:
  members:
  - host: 127.0.0.1
    port: 9160
  - host: 127.0.0.1
    port: 9160
  - host: 127.0.0.1
    port: 9160
database:
  version: 2.2
  cassandra:
    version: 2
    enabled: true
    name: 'Contrail'
    minimum_disk: 10
    original_token: 0
data_dirs:
  - /var/lib/cassandra
  id: 1
discovery:
  host: 127.0.0.1
bind:
  host: 127.0.0.1
  port: 9042
  rpc_port: 9160
members:
  - host: 127.0.0.1
    id: 1
  - host: 127.0.0.1
    id: 2
  - host: 127.0.0.1
    id: 3
web:
  version: 2.2
  enabled: True
  bind:
    address: 127.0.0.1
master:
  host: 127.0.0.1
analytics:
  host: 127.0.0.1
cache:
  engine: redis
  host: 127.0.0.1
  port: 6379
members:
Separated analytics from control and config

Config, control, database, web.

```yaml
opencontrail:
  common:
    version: 2.2
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      token: token
      password: password
    network:
      engine: neutron
      host: 127.0.0.1
      port: 9696
  config:
    version: 2.2
    enabled: true
    network:
      engine: neutron
      host: 127.0.0.1
      port: 9696
    discovery:
      host: 127.0.0.1
    analytics:
      host: 127.0.0.1
    bind:
      address: 127.0.0.1
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
    database:
      members:
      - host: 127.0.0.1
        port: 9160
      - host: 127.0.0.1
```
port: 9160
- host: 127.0.0.1
  port: 9160

cache:
  members:
  - host: 127.0.0.1
  - host: 127.0.0.1
  - host: 127.0.0.1

identity:
  engine: keystone
  version: '2.0'
  region: RegionOne
  host: 127.0.0.1
  port: 35357
  user: admin
  password: password
  token: token
  tenant: admin

members:
- host: 127.0.0.1
  id: 1
- host: 127.0.0.1
  id: 2
- host: 127.0.0.1
  id: 3

control:
  version: 2.2
  enabled: true
  bind:
    address: 127.0.0.1

discovery:
  host: 127.0.0.1

master:
  host: 127.0.0.1

members:
- host: 127.0.0.1
  id: 1
- host: 127.0.0.1
  id: 2
- host: 127.0.0.1
  id: 3

database:
  version: 127.0.0.1

  cassandra:
    version: 2
    enabled: true
    name: 'Contrail'
    minimum_disk: 10
    original_token: 0

data_dirs:
- /var/lib/cassandra

  id: 1
  discovery:
    host: 127.0.0.1
bind:
  host: 127.0.0.1
  port: 9042
  rpc_port: 9160
members:
- host: 127.0.0.1
  id: 1
- host: 127.0.0.1
  id: 2
- host: 127.0.0.1
  id: 3
web:
  version: 2.2
  enabled: True
  bind:
    address: 127.0.0.1
analytics:
  host: 127.0.0.1
master:
  host: 127.0.0.1
cache:
  engine: redis
  host: 127.0.0.1
  port: 6379
members:
- host: 127.0.0.1
  id: 1
- host: 127.0.0.1
  id: 2
- host: 127.0.0.1
  id: 3
identity:
  engine: keystone
  version: '2.0'
  host: 127.0.0.1
  port: 35357
  user: admin
  password: password
  token: token
  tenant: admin

Analytic nodes

Analytics and database on an analytic node(s)

opencontrail:
  common:
    version: 2.2
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      token: token
      password: password
      network:
        engine: neutron
        host: 127.0.0.1
port: 9696

collector:
  version: 2.2
  enabled: true
  bind:
    host: 127.0.0.1
  master:
    host: 127.0.0.1
  discovery:
    host: 127.0.0.1
  data_ttl: 1
  database:
    members:
    - host: 127.0.0.1
      port: 9160
    - host: 127.0.0.1
      port: 9160
    - host: 127.0.0.1
      port: 9160
database:
  version: 2.2
  cassandra:
    version: 2
    enabled: true
    name: 'Contrail'
    minimum_disk: 10
    original_token: 0
    data_dirs:
      - /var/lib/cassandra
    id: 1
discovery:
  host: 127.0.0.1
bind:
  host: 127.0.0.1
  port: 9042
  rpc_port: 9160
members:
- host: 127.0.0.1
  id: 1
- host: 127.0.0.1
  id: 2
- host: 127.0.0.1
  id: 3

Compute nodes

Vrouter configuration on a compute node(s)

opencontrail:
  common:
    version: 2.2
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357

(continues on next page)
Compute nodes with gateway_mode

Gateway mode: can be server/ vcpe (default is none)

```yaml
opencontrail:
  compute:
    gateway_mode: server
```

TSN nodes

Configure TSN nodes

```yaml
opencontrail:
  compute:
    enabled: true
  tor:
    enabled: true
    bind:
      port: 8086
    agent:
      tor01:
        id: 0
        port: 6632
        host: 127.0.0.1
        address: 127.0.0.1
```

Set up metadata secret for the Vrouter

In order to get cloud-init within the instance to properly fetch instance metadata, metadata_proxy_secret in the Vrouter agent config should match the value in nova.conf. The administrator should define it in the pillar:

```yaml
token: token
password: password
network:
  engine: neutron
  host: 127.0.0.1
  port: 9696
compute:
  version: 2.2
  enabled: True
  hostname: node-12.domain.tld
  discovery:
    host: 127.0.0.1
  interface:
    address: 127.0.0.1
    dev: eth0
    gateway: 127.0.0.1
    mask: /24
    dns: 127.0.0.1
    mtu: 9000
```
opencontrail:
  compute:
    metadata:
      secret: opencontrail

Add auth info for Barbican on compute nodes

opencontrail:
  compute:
    lbaas:
      enabled: true
      secret_manager:
        engine: barbican
        identity:
          user: admin
          password: "supersecretpassword123"
          tenant: admin

Keystone v3

To enable support for keystone v3 in opencontrail, there must be defined version for config and web role.

opencontrail:
  config:
    version: 2.2
    enabled: true
    ...
    identity:
      engine: keystone
      version: '3'
    ...

opencontrail:
  web:
    version: 2.2
    enabled: true
    ...
    identity:
      engine: keystone
      version: '3'
    ...

Without Keystone

opencontrail:
  ...
  common:
    ...
  identity:
    engine: none
    token: none

(continues on next page)
password: none
...
config:
...
identity:
  engine: none
  password: none
  token: none
...
web:
...
identity:
  engine: none
  password: none
  token: none
...

Kubernetes support

Kubernetes vrouter nodes

Vrouter configuration on a kubernetes node(s)

```yaml
opencontrail:
  ...
  compute:
  
  engine: kubernetes
...
```

vRouter with separated control plane

Separate XMPP traffic from dataplane interface.

```yaml
opencontrail:
  compute:
  bind:
  
  address: 172.16.0.50
...
```

Override RPF default in Contrail API

From MCP1.1 with OpenContrail >= 3.1.1 you can override RPF default for newly created virtual networks. This can be useful for usecases like running Calico and K8S in overlay. The `override_rpf_default_by` has valid values `disable`, `enable`. If not defined, the configuration fallbacks to Contrail default - currently `enable`.

```yaml
opencontrail:
  ...
  config:
  
  override_rpf_default_by: 'disable'
...
```
Cassandra GC logging

From Contrail version 3 you can set a way you want to handle Cassandra GC logs. The behavior is controlled by `cassandra_gc_logging`. Valid values are ‘rotation’ (default), ‘legacy’ and false.

- ‘rotation’ is supported by JDK 6u34 7u2 or later and handles rotation of log files automatically. - ‘legacy’ is a way to support older JDKs and you will need to handle logs by other means. This can be handled for example by using `- service.opencontrail.database.cassandra_log_cleanup` in your reclass model. - false will disable the cassandra gc logging

```yaml
opencontrail:
  ...
  database:
    cassandra_gc_logging: false
  ...
```

Disable Contrail API authentication

Contrail version must >= 3.0. It is useful especially for Keystone v3.

```yaml
opencontrail:
  ...
  config:
    multi_tenancy: false
  ...
```

Enable RBAC

```yaml
opencontrail:
  ...
  config:
    aaa_mode: rbac
    cloud_admin_role: admin
    global_read_only_role: member
  ...
```

Switch from on demand to periodic keystone sync

This can be useful when you want to sync projects from OpenStack to Contrail automatically. The period of sync is 60s.

```yaml
opencontrail:
  ...
  config:
    identity:
      sync_on_demand: false
  ...
```
Cassandra listen interface

database:
   ....
   bind:
      interface: eth0
      port: 9042
      rpc_port: 9160
   ....

OpenContrail WebUI version >= 3.1.1

For OpenContrail version >= 3.1.1 and Cassandra >= 2.1 we should override WebUI’s cassandra port from 9160 to 9042.

For appropriate node at class level:

opencontrail:
   ....
   web:
      database:
         port: 9042
   ....

RabbitMQ HA hosts

opencontrail:
   config:
      message_queue:
         engine: rabbitmq
         members:
            - host: 10.0.16.1
            - host: 10.0.16.2
            - host: 10.0.16.3
         port: 5672

database:
   ....
   bind:
      interface: eth0
      port: 9042
      rpc_port: 9160
   ....

DPDK vRouter

opencontrail:
   compute:
      dpdk:
         enabled: true
         taskset: "0x000000000000000C"
socket_mem: "1024,1024"
interface:
  mac_address: 90:e2:ba:7c:22:e1
  pci: 0000:81:00.1
...

Increase number of alarm-gen workers

Port prefix will increment used ports by workers starting with 5901.

collector:
  alarm_gen:
    workers: 1
    port_prefix: 59

Contraill client

Basic parameters with identity and host configs

opencontrail:
  client:
    identity:
      user: admin
      project: admin
      password: adminpass
      host: keystone_host
    config:
      host: contrail_api_host
      port: contrail_api_ort

Enforcing virtual routers

opencontrail:
  client:
    ...
    virtual_router:
      cmp01:
        ip_address: 172.16.0.11
        dpdk_enabled: True
      cmp02:
        ip_address: 172.16.0.12
        dpdk_enabled: True

Enforcing global system config

opencontrail:
  client:
    ...
    global_system_config:
      name: default-global-system-config
      asn: 64512
      grp:
        enable: true

(continued on next page)
restart_time: 60
end_of_rib_timeout: 30
bgp_helper_enable: false
xmpp_helper_enable: false
long_lived_restart_time: 300

Enforcing global vrouter config

opencontrail:
  client:
    ...
    global_vrouter_config:
      name: default-global-vrouter-config
      parent_type: global-system-config
      encap_priority: "MPLSoUDP,MPLSoGRE"
      vxlan_vn_id_mode: automatic
      fq_names:
        - 'default-global-system-config'
        - 'default-global-vrouter-config'

Enforcing control nodes

opencontrail:
  client:
    ...
    bgp_router:
      ntw01:
        type: control-node
        ip_address: 172.16.0.11
      nwt02:
        type: control-node
        ip_address: 172.16.0.12
      nwt03:
        type: control-node
        ip_address: 172.16.0.13

Enforcing edge BGP routers

opencontrail:
  client:
    ...
    bgp_router:
      mx01:
        type: router
        ip_address: 172.16.0.21
        asn: 64512
      mx02:
        type: router
        ip_address: 172.16.0.22
        asn: 64512
        key_type: md5
        key: password

Enforcing config nodes

opencontrail:
  client:
    (continues on next page)
Enforcing database nodes

opencontrail:
  client:
    ...
    database_node:
      ntw01:
        ip_address: 172.16.0.21
      ntw02:
        ip_address: 172.16.0.22

Enforcing analytics nodes

opencontrail:
  client:
    ...
    analytics_node:
      ntw01:
        ip_address: 172.16.0.21
      ntw02:
        ip_address: 172.16.0.22

Enforcing Link Local Services

opencontrail:
  client:
    ...
    linklocal_service:
      # example with dns name address (only one permitted)
      metal:
        llk_ip: 10.0.0.23
        llk_port: 80
        ipf_addresses: "meta.example.com"
        ipf_port: 80
      # example with multiple ip addresses
      meta2:
        llk_ip: 10.0.0.23
        llk_port: 80
        ipf_addresses:
          - 10.10.10.10
          - 10.20.20.20
          - 10.30.30.30
        ipf_port: 80
      # example with one ip address
      meta3:
        llk_ip: 10.0.0.23
        llk_port: 80
        ipf_addresses:
          - 10.10.10.10
        ipf_port: 80
# example with name override
lls_meta4:
  name: meta4
  lls_ip: 10.0.0.23
  lls_port: 80
  ipf_addresses:
  - 10.10.10.10
  ipf_port: 80

Configuring OpenStack default quotas

Enforcing physical routers

```yaml
opencontrail:
  client: ...
  physical_router:
    router1:
      interface:
        port1:
          name: port1
          logical_interface:
            port1_l:
              name: 'port1.0'
              vlan_tag: 0
              interface_type: L2
              virtual_machine_interface:
                port1_port:
                  name: port1_port
                  ip_address: 192.168.90.107
                  mac_address: '2e:92:a8:af:c2:21'
                  security_group: 'default'
                  virtual_network: 'virtual-network'
```

Enforcing virtual networks

```yaml
opencontrail:
  client:
    virtual_networks:
      net01:
        name: 'network01'
        ip_address: '172.16.111.0'
        ip_prefix: 24
        asn: 64512
        route_target: 10000
```

(continues on next page)
Enforcing floating ip pool settings.

Virtual network with flag external needs to be created before managing the floating ip pool. Param vn_name is the name of the external network.

```yaml
opencontrail:
  client:
    floating_ip_pools:
      pool1:
        vn_name: external-network
        vn_project: admin
        vn_domain: default-domain
        owner_access: 7
        global_access: 0
        list_of_projects:
          - [tenant1, 7]
          - [tenant2, 7]
          - [tenant3, 7]
      pool2:
        vn_name: floating ips
        vn_project: admin
        vn_domain: default-domain
        owner_access: 7
        global_access: 0
        list_of_projects:
          - [tenant3, 7]
```

If you want to remove all shares from the ip floating pool, define only empty list in list of projects, like this:

```yaml
opencontrail:
  client:
    floating_ip_pools:
      pool1:
        vn_name: external-network
        vn_project: admin
        vn_domain: default-domain
        owner_access: 7
        global_access: 0
        list_of_projects: []
```

2.1. Project Introduction 419
Contrail DNS custom forwarders

By default Contrail uses the /etc/resolv.conf file to determine the upstream DNS servers. This can have some side-affects, like resolving internal DNS entries on your public instances.

In order to overrule this default set, you can configure nameservers using pillar data. The formula is then responsible for configuring and generating an alternate resolv.conf file.

Note: this has been patched recently in the Contrail distribution of Mirantis: https://github.com/Mirantis/contrail-controller/commit/ed9a25ccbeeb7d079a93e955a5b1d65a4 https://github.com/Mirantis/contrail-controller/commit/94c844cf2e9bcfcd48587ae03d1b869e737ade

To change forwarders for the default-dns option (which is handled by compute nodes):

```
compute:
    ..... 
    dns:
        forwarders:
            - 8.8.8.8
            - 8.8.4.4
    ..... 
```

To change forwarders for vDNS zones (handled by control nodes):

```
control:
    ..... 
    dns:
        forwarders:
            - 8.8.8.8
            - 8.8.4.4
    ..... 
```

Usage

Basic installation

Add control BGP

```
python /etc/contrail/provision_control.py --api_server_ip 192.168.1.11 --api_server_port 8082 --host_name network1.contrail.domain.com --host_ip 192.168.1.11 --router_asn 64512
```

Install compute node

```
yum install contrail-vrouter contrail-openstack-vrouter
```

```
salt-call state.sls nova,opencontrail
```

Add virtual router

```
python /etc/contrail/provision_vrouter.py --host_name hostnode1.intra.domain.com --host_ip 10.0.100.101 --api_server_ip 10.0.100.30 --oper add --admin_user admin --admin_password cloudlab --admin_tenant_name admin
```

```
/etc/sysconfig/network-scripts/ifcfg-bond0 -- comment GATEWAY,NETMASK,IPADDR
```

(continues on next page)
Debugging

Display vhost XMPP connection status
You should see the correct controller_ip and state should be established.

\[http://<compute-node>:8085/Snh_AgentXmppConnectionStatusReq?\]

Display vrouter interface status
When vrf_name = —ERROR— then something goes wrong

\[http://<compute-node>:8085/Snh_IfReq?name=\]

Display IF MAP table
Look for neighbours, if VM has 2, it’s ok

\[http://<control-node>:8083/Snh_IFMapTableShowReq?table_name=\]

Trace XMPP requests

\[http://<compute-node>:8085/Snh_SandeshTraceRequest?x=XmppMessageTrace\]

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

\[http://salt-formulas.readthedocs.io/\]

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

\[https://github.com/salt-formulas/salt-formula-opencontrail/issues\]

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

\[https://launchpad.net/salt-formulas\]

You can also join salt-formulas-users team and subscribe to mailing list:

\[https://launchpad.net/~salt-formulas-users\]

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

\[https://github.com/salt-formulas/salt-formula-opencontrail\]

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Neutron Formula

Neutron is an OpenStack project to provide “networking as a service” between interface devices (e.g., vNICs) managed by other Openstack services (e.g., nova).
Starting in the Folsom release, Neutron is a core and supported part of the OpenStack platform (for Essex, we were an “incubated” project, which means use is suggested only for those who really know what they’re doing with Neutron).

Sample Pillars

Neutron Server on the controller node

```
neutron:
  server:
    enabled: true
    version: mitaka
    allow_pagination: true
    pagination_max_limit: 100
    api_workers: 2
    rpc_workers: 2
    rpc_state_report_workers: 2
    bind:
      address: 172.20.0.1
      port: 9696
    database:
      engine: mysql
      host: 127.0.0.1
      port: 3306
      name: neutron
      user: neutron
      password: pwd
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      user: neutron
      password: pwd
      tenant: service
      endpoint_type: internal
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    metadata:
      host: 127.0.0.1
      port: 8775
      password: pass
      workers: 2
    audit:
      enabled: false
```

Note: The pagination is useful to retrieve a large bunch of resources, because a single request may fail (timeout). This is enabled with both parameters `allow_pagination` and `pagination_max_limit` as shown above.

Configuration of policy.json file

```
neutron:
  server:
    ....
```
policy:
    create_subnet: 'rule:admin_or_network_owner'
    'get_network:queue_id': 'rule:admin_only'
    # Add key without value to remove line from policy.json
    'create_network:shared': 

Neutron LBaaSv2 enablement

```yaml
neutron:
    server:
        lbaas:
            enabled: true
            providers:
                octavia:
                    engine: octavia
                    driver_path: 'neutron_lbaas.drivers.octavia.driver.OctaviaDriver'
                    base_url: 'http://127.0.0.1:9876'
                avi_adc:
                    engine: avinetworks
                    driver_path: 'avi_lbaasv2.avi_driver.AviDriver'
                    controller_address: 10.182.129.239
                    controller_user: admin
                    controller_password: Cloudlab2016
                    controller_cloud_name: Default-Cloud
                avi_adc2:
                    engine: avinetworks
                    ...  
```

Note: If the Contrail backend is set, Opencontrail loadbalancer would be enabled automatically. In this case lbaas should disabled in pillar:

```yaml
neutron:
    server:
        lbaas:
            enabled: false  
```

Neutron FWaaSv1 enablement

```yaml
neutron:
    fwaas:
        enabled: true
        version: ocata
        api_version: v1  
```

Enable CORS parameters

```yaml
neutron:
    server:
        cors:
            allowed_origin: https:localhost.local,http:localhost.local  
```

(continues on next page)
Neutron VXLAN tenant networks with Network nodes

With DVR for East-West and Network node for North-South.

This use case describes a model utilising VxLAN overlay with DVR. The DVR routers will only be utilized for traffic that is router within the cloud infrastructure and that remains encapsulated. External traffic will be routed to via the network nodes.

The intention is that each tenant will require at least two (2) vrouters one to be utilised

Neutron Server

```
neutron:
  server:
    version: mitaka
    path_mtu: 1500
    bind:
      address: 172.20.0.1
      port: 9696
    database:
      engine: mysql
      host: 127.0.0.1
      port: 3306
      name: neutron
      user: neutron
      password: pwd
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      user: neutron
      password: pwd
      tenant: service
      endpoint_type: internal
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    global_physnet_mtu: 9000
    l3_ha: False # Which type of router will be created by default
    dvr: True # disabled for non DVR use case
    backend:
      engine: ml2
      tenant_network_types: "flat,vxlan"
      external_mtu: 9000
      mechanism:
```

(continues on next page)
Network Node

```yaml
neutron:
  gateway:
    enabled: True
    version: mitaka
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    local_ip: 192.168.20.20 \ # br-mesh ip address
dvr: True \ # disabled for non DVR use case
agent_mode: dvr_snat
metadata:
  host: 127.0.0.1
  password: pass
backend:
  engine: ml2
  tenant_network_types: "flat,vxlan"
mechanism:
  ovs:
    driver: openvswitch
```

Compute Node

```yaml
neutron:
  compute:
    enabled: True
    version: mitaka
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    local_ip: 192.168.20.20 \ # br-mesh ip address
dvr: True \ # disabled for non DVR use case
agent_mode: dvr
external_access: false \ # Compute node with DVR for east-west only, Network Node_→ has True as default
metadata:
  host: 127.0.0.1
  password: pass
backend:
  engine: ml2
  tenant_network_types: "flat,vxlan"
mechanism:
  ovs:
    driver: openvswitch
audit:
```
enabled: false

**Disable physnet1 bridge**

By default we have external access turned on, so among any physnets in your reclass there would be additional one: physnet1, which is mapped to br-floating.

If you need internal nets only without this bridge, remove br-floating and configurations mappings. Disable mappings for this bridge on neutron-servers:

```yaml
neutron:
  server:
    external_access: false
```

gateways:

```yaml
neutron:
  gateway:
    external_access: false
```

compute nodes:

```yaml
neutron:
  compute:
    external_access: false
```

**Add additional bridge mappings for OVS bridges**

By default we have external access turned on, so among any physnets in your reclass there would be additional one: physnet1, which is mapped to br-floating.

If you need to add extra non-default bridge mappings they can be defined separately for both gateways and compute nodes:

gateways:

```yaml
neutron:
  gateway:
    bridge_mappings:
      physnet4: br-floating-internet
```

compute nodes:

```yaml
neutron:
  compute:
    bridge_mappings:
      physnet4: br-floating-internet
```

**Specify different mtu values for different physnets**

Neutron Server
Neutron VXLAN tenant networks with Network Nodes (non DVR)

This section describes a network solution that utilises VxLAN overlay networks without DVR with all routers being managed on the network nodes.

Neutron Server

```yaml
neutron:
  server:
    version: mitaka
    bind:
      address: 172.20.0.1
      port: 9696
    database:
      engine: mysql
      host: 127.0.0.1
      port: 3306
      name: neutron
      user: neutron
      password: pwd
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      user: neutron
      password: pwd
      tenant: service
      endpoint_type: internal
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    global_physnet_mtu: 9000
    l3_ha: True
    dvr: False
    backend:
      engine: ml2
      tenant_network_types= "flat,vxlan"
    external_mtu: 9000
    mechanism:
      ovs:
        driver: openvswitch
```

Network Node
neutron:
  gateway:
    enabled: True
    version: mitaka
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    local_ip: 192.168.20.20 # br-mesh ip address
    dvr: False
    agent_mode: legacy
    availability_zone: az1
    metadata:
      host: 127.0.0.1
      password: pass
    backend:
      engine: m12
      tenant_network_types: "flat,vxlan"
    mechanism:
      ovs:
        driver: openvswitch

Compute Node

neutron:
  compute:
    enabled: True
    version: mitaka
    message_queue:
      engine: rabbitmq
      host: 127.0.0.1
      port: 5672
      user: openstack
      password: pwd
      virtual_host: '/openstack'
    local_ip: 192.168.20.20 # br-mesh ip address
    external_access: False
    dvr: False
    backend:
      engine: m12
      tenant_network_types: "flat,vxlan"
    mechanism:
      ovs:
        driver: openvswitch

Neutron VXLAN tenant networks with Network Nodes with DVR

With DVR for East-West and North-South, DVR everywhere, Network node for SNAT.

This section describes a network solution that utilises VxLAN overlay networks with DVR with North-South and East-West. Network Node is used only for SNAT.

Neutron Server
```
neutron:
    server:
        version: mitaka
        bind:
            address: 172.20.0.1
            port: 9696
        database:
            engine: mysql
            host: 127.0.0.1
            port: 3306
            name: neutron
            user: neutron
            password: pwd
        identity:
            engine: keystone
            host: 127.0.0.1
            port: 35357
            user: neutron
            password: pwd
            tenant: service
            endpoint_type: internal
        message_queue:
            engine: rabbitmq
            host: 127.0.0.1
            port: 5672
            user: openstack
            password: pwd
            virtual_host: '/openstack'
        global_physnet_mtu: 9000
        l3_ha: False
        dvr: True
        backend:
            engine: ml2
            tenant_network_types= "flat,vxlan"
            external_mtu: 9000
            mechanism:
                ovs:
                    driver: openvswitch

Network Node
```

```
neutron:
    gateway:
        enabled: True
        version: mitaka
        message_queue:
            engine: rabbitmq
            host: 127.0.0.1
            port: 5672
            user: openstack
            password: pwd
            virtual_host: '/openstack'
        local_ip: 192.168.20.20 # br-mesh ip address
        dvr: True
        agent_mode: dvr_snat
        availability_zone: az1
        metadata:
```

(continues on next page)
host: 127.0.0.1
password: pass
backend:
  engine: m12
tenant_network_types: "flat,vxlan"
mechanism:
  ovs:
    driver: openvswitch

Compute Node

neutron:
  compute:
    enabled: True
  version: mitaka
  message_queue:
    engine: rabbitmq
    host: 127.0.0.1
    port: 5672
    user: openstack
    password: pwd
    virtual_host: '/openstack'
  local_ip: 192.168.20.20
  # br-mesh ip address
dvr: True
  external_access: True
  agent_mode: dvr
  availability_zone: az1
  metadata:
    host: 127.0.0.1
    password: pass
  backend:
    engine: m12
tenant_network_types: "flat,vxlan"
  mechanism:
    ovs:
      driver: openvswitch

Sample Linux network configuration for DVR

linux:
  network:
    bridge: openvswitch
  interface:
    eth1:
      enabled: true
type: eth
mtu: 9000
proto: manual
eth2:
  enabled: true
type: eth
mtu: 9000
proto: manual
eth3:
  enabled: true
type: eth
mtu: 9000

(continues on next page)
proto: manual
br-int:
  enabled: true
  mtu: 9000
  type: ovs_bridge
br-floating:
  enabled: true
  mtu: 9000
  type: ovs_bridge
float-to-ex:
  enabled: true
  type: ovs_port
  mtu: 65000
  bridge: br-floating
br-mgmt:
  enabled: true
  type: bridge
  mtu: 9000
  address: ${_param:single_address}
  netmask: 255.255.255.0
  useInterfaces:
    - eth1
br-mesh:
  enabled: true
  type: bridge
  mtu: 9000
  address: ${_param:tenant_address}
  netmask: 255.255.255.0
  useInterfaces:
    - eth2
br-ex:
  enabled: true
  type: bridge
  mtu: 9000
  address: ${_param:external_address}
  netmask: 255.255.255.0
  useInterfaces:
    - eth3
    - float-to-ex

Additional VXLAN tenant network settings

The default multicast group of 224.0.0.1 only multicasts to a single subnet. Allow overriding it to allow larger underlay network topologies.

Neutron Server

neutron:
  server:
    vxlan:
      group: 239.0.0.0/8
      vni_ranges: "2:65535"
Neutron VLAN tenant networks with Network Nodes

VLAN tenant provider

Neutron Server only

```yaml
neutron:
  server:
    version: mitaka
    ...  
    global_physnet_mtu: 9000
    l3_ha: False
    dvr: True
    backend:
      engine: ml2
      tenant_network_types: "flat,vlan" # Can be mixed flat,vlan,vxlan
      tenant_vlan_range: "1000:2000"
      external_vlan_range: "100:200" # Does not have to be defined.
      external_mtu: 9000
      mechanism:
        ovs:
          driver: openvswitch
```

Compute node

```yaml
neutron:
  compute:
    version: mitaka
    ...  
    dvr: True
    agent_mode: dvr
    external_access: False
    backend:
      engine: ml2
      tenant_network_types: "flat,vlan" # Can be mixed flat,vlan,vxlan
    mechanism:
      ovs:
        driver: openvswitch
```

Advanced Neutron Features (DPDK, SR-IOV)

Neutron OVS DPDK

Enable datapath netdev for neutron openvswitch agent

```yaml
neutron:
  server:
    version: mitaka
    ...  
    dpdk: True
    ...
```

(continues on next page)
vhost_socket_dir: /var/run/openvswitch
backend:
  engine: ml2
  ...  
  mechanism:
    ovs:
      driver: openvswitch

Neutron OVS SR-IOV

neutron:
  server:
    version: mitaka
    backend:
      engine: ml2
      ...  
      mechanism:
        ovs:
          driver: openvswitch
        sriov:
          driver: sriovnicswitch
  compute:
    version: mitaka
    ...  
    backend:
      engine: ml2
      tenant_network_types: "flat,vlan" # Can be mixed flat,vlan,vxlan
      sriov:
        nic_one:
          devname: eth1
          physical_network: physnet3
        mechanism:
          ovs:
            driver: openvswitch

Neutron with VLAN-aware-VMs

neutron:
  server:
    vlan_aware_vms: true
    ....
  compute:
    vlan_aware_vms: true
    ....
  gateway:
    vlan_aware_vms: true

Neutron with OVN

Control node:
neutron:
    server:
        backend:
            engine: ovn
            mechanism:
                ovn:
                    driver: ovn
                    tenant_network_types: "geneve,flat"
            ovn_ctl_opts:
                db-nb-create-insecure-remote: 'yes'
                db-sb-create-insecure-remote: 'yes'

Compute node:

neutron:
    compute:
        local_ip: 10.2.0.105
        controller_vip: 10.1.0.101
        external_access: false
        backend:
            engine: ovn

Neutron Server

Neutron Server with OpenContrail

neutron:
    server:
        backend:
            engine: contrail
            host: contrail_discovery_host
            port: 8082
            user: admin
            password: password
            tenant: admin
            token: token

Neutron Server with Midonet

neutron:
    server:
        backend:
            engine: midonet
            host: midonet_api_host
            port: 8181
            user: admin
            password: password

Neutron Keystone region

neutron:
    server:
        enabled: true
        version: kilo
...
Client-side RabbitMQ HA setup

```yaml
identity:
  region: RegionTwo
...
compute:
  region: RegionTwo
...
```

### Configuring TLS communications

**Note:** by default system wide installed CA certs are used, so `cacert_file` param is optional, as well as `cacert`.

- **RabbitMQ TLS**

```yaml
neutron:
  server, gateway, compute:
    message_queue:
      engine: rabbitmq
      members:
        - host: 10.0.16.1
        - host: 10.0.16.2
        - host: 10.0.16.3
      user: openstack
      password: pwd
      virtual_host: '/openstack'
...
```

- **MySQL TLS**

```yaml
neutron:
  server:
    database:
      ssl:
        enabled: True
        (optional) cacert: cert body if the cacert_file does not exists
        (optional) cacert_file: /etc/openstack/mysql-ca.pem
        (optional) version: TLSv1_2
```

- **Openstack HTTPS API**

```yaml
neutron:
  server:
    identity:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
```
Enable auditing filter, ie: CADF

```
neutron:
  server:
    audit:
      enabled: true
      ...
    filter_factory: 'keystonemiddleware.audit:filter_factory'
    map_file: '/etc/pycadf/neutron_api_audit_map.conf'
  ...
compute:
  audit:
    enabled: true
    ...
  filter_factory: 'keystonemiddleware.audit:filter_factory'
  map_file: '/etc/pycadf/neutron_api_audit_map.conf'
  ...
```

Neutron with security groups disabled

```
neutron:
  server:
    security_groups_enabled: False
  ...
compute:
  security_groups_enabled: False
  ...
gateway:
  security_groups_enabled: False
```

**Neutron Client**

**Neutron networks**

```
neutron:
  client:
    enabled: true
  server:
    identity:
      endpoint_type: internalURL
    network:
      inet1:
        tenant: demo
        shared: False
        admin_state_up: True
        router_external: True
        provider_physical_network: inet
        provider_network_type: flat
        provider_segmentation_id: 2
        subnet:
          inet1-subnet1:
            cidr: 192.168.90.0/24
            enable_dhcp: False
      inet2:
        tenant: admin
        shared: False
```

Neutron routers

 neutron:
  client:
   enabled: true
  server:
   identity:
     endpoint_type: internalURL
   router:
     inet1-router:
       tenant: demo
       admin_state_up: True
       gateway_network: inet
       interfaces:
         - inet1-subnet1
         - inet1-subnet2
   identity1:
     router:
     ...

TODO: implement adding new interfaces to a router while updating it

Neutron security groups

 neutron:
  client:
   enabled: true
  server:
   identity:
     endpoint_type: internalURL
  security_group:
    security_group1:
      tenant: demo
      description: security group 1
      rules:
        - direction: ingress
          ethertype: IPv4
          protocol: TCP
          port_range_min: 1
          port_range_max: 65535
          remote_ip_prefix: 0.0.0.0/0
        - direction: ingress
          ethertype: IPv4
          protocol: UDP

(continues on next page)
port_range_min: 1
port_range_max: 65535
remote_ip_prefix: 0.0.0.0/0
- direction: ingress
  protocol: ICMP
  remote_ip_prefix: 0.0.0.0/0

identity1:
  security_group:
    ...

TODO: implement updating existing security rules (now it adds new rule if trying to --update existing one)

Floating IP addresses

neutron:
  client:
    enabled: true
  server:
    identity:
      endpoint_type: internalURL
    floating_ip:
      prx01-instance:
        server: prx01.mk22-lab-basic.local
        subnet: private-subnet1
        network: public-net1
        tenant: demo
      gtw01-instance:
        ...

Note: The network must have flag router:external set to True. Instance port in the stated subnet will be associated with the dynamically generated floating IP.

Enable Neutron extensions (QoS, DNS, etc.)

neutron:
  server:
    backend:
      extension:
        dns:
          enabled: True
          host: 127.0.0.1
          port: 9001
          protocol: http
          ...
        qos:
          enabled: True
Neutron with Designate

```yaml
neutron:
  server:
    backend:
      extension:
        dns:
          enabled: True
          host: 127.0.0.1
          port: 9001
          protocol: http
```

Enhanced logging with logging.conf

By default logging.conf is disabled.

That is possible to enable per-binary logging.conf with new variables:

- `openstack_log_appender` - set it to true to enable log_config_append for all OpenStack services;
- `openstack_fluentd_handler_enabled` - set to true to enable FluentHandler for all Openstack services.
- `openstack_ossyslog_handler_enabled` - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

```yaml
neutron:
  server:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
          enabled: true
        ossyslog:
          enabled: true

compute:
  logging:
    log_appender: true
    log_handlers:
      watchedfile:
        enabled: true
      fluentd:
        enabled: true
      ossyslog:
        enabled: true

gateway:
  logging:
    log_appender: true
    log_handlers:
      watchedfile:
        enabled: true
```

(continues on next page)
Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-neutron/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-neutron

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Nova Formula

OpenStack Nova provides a cloud computing fabric controller, supporting a wide variety of virtualization technologies, including KVM, Xen, LXC, VMware, and more. In addition to its native API, it includes compatibility with the commonly encountered Amazon EC2 and S3 APIs.

Sample Pillars

Controller nodes

Nova services on the controller node

```yaml
nova:
  controller:
    version: juno
    enabled: true
    security_group: true
    cpu_allocation_ratio: 8.0
    ram_allocation_ratio: 1.0
    disk_allocation_ratio: 1.0
    cross_az_attach: false
    workers: 8
```

(continues on next page)
report_interval: 60
bind:
  public_address: 10.0.0.122
  public_name: openstack.domain.com
  novncproxy_port: 6080
database:
  engine: mysql
  host: 127.0.0.1
  port: 3306
  name: nova
  user: nova
  password: pwd
identity:
  engine: keystone
  host: 127.0.0.1
  port: 35357
  user: nova
  password: pwd
  tenant: service
message_queue:
  engine: rabbitmq
  host: 127.0.0.1
  port: 5672
  user: openstack
  password: pwd
  virtual_host: '/openstack'
network:
  engine: neutron
  host: 127.0.0.1
  port: 9696
  extension_sync_interval: 600
identity:
  engine: keystone
  host: 127.0.0.1
  port: 35357
  user: neutron
  password: pwd
  tenant: service
metadata:
  password: password
audit:
  enabled: false
osapi_max_limit: 500
barbican:
  enabled: true

Nova services from custom package repository

nova:
  controller:
    version: juno
    source:
      engine: pkg
      address: http://...

Client-side RabbitMQ HA setup

2.1. Project Introduction
Enable auditing filter, ie: CADF

```yaml
nova:
  controller:
    audit:
      enabled: true
    ....
    filter_factory: 'keystonemiddleware.audit:filter_factory'
    map_file: '/etc/pycadf/nova_api_audit_map.conf'
    ....
```

Enable CORS parameters

```yaml
nova:
  controller:
    cors:
      allowed_origin: https:localhost.local,http:localhost.local
      expose_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_methods: GET,PUT,POST,DELETE,PATCH
      allow_headers: X-Auth-Token,X-Openstack-Request-Id,X-Subject-Token
      allow_credentials: True
      max_age: 86400
```

Configuration of policy.json file

```yaml
nova:
  controller:
    ....
    policy:
      context_is_admin: 'role:admin or role:administrator'
      'compute:create': 'rule:admin_or_owner'
      # Add key without value to remove line from policy.json
      'compute:create:attach_network':
```

Enable Barbican integration

```yaml
nova:
  controller:
    ....
    barbican:
      enabled: true
```
Configuring TLS communications

Note: by default system wide installed CA certs are used, so `cacert_file` param is optional, as well as `cacert`.

- **RabbitMQ TLS**

```python
nova:
  compute:
    message_queue:
      port: 5671
      ssl:
        enabled: True
        (optional) cacert: cert body if the cacert_file does not exists
        (optional) cacert_file: /etc/openstack/rabbitmq-ca.pem
        (optional) version: TLSv1_2
```

- **MySQL TLS**

```python
nova:
  controller:
    database:
      ssl:
        enabled: True
        (optional) cacert: cert body if the cacert_file does not exists
        (optional) cacert_file: /etc/openstack/mysql-ca.pem
```

- **Openstack HTTPS API**

Set the `https` as protocol at `nova:compute` and `nova:controller` sections:

```python
nova:
  controller:
    identity:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    network:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    glance:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem

nova:
  compute:
    identity:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    network:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    image:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
    ironic:
      protocol: https
      (optional) cacert_file: /etc/openstack/proxy.pem
```

Note: the barbican, cinder and placement url endpoints are discovering using service catalog.
Compute nodes

Nova controller services on compute node

```yaml
nova:
  compute:
    version: juno
    enabled: true
    virtualization: kvm
    cross_az_attach: false
    disk_cachemodes: network=writeback,block=none
    availability_zone: availability_zone_01
    aggregates:
      - hosts_with_fc
      - hosts_with_ssd
    security_group: true
    resume_guests_state_on_host_boot: False
    my_ip: 10.1.0.16
  bind:
    vnc_address: 172.20.0.100
    vnc_port: 6080
    vnc_name: openstack.domain.com
    vnc_protocol: http
  database:
    engine: mysql
    host: 127.0.0.1
    port: 3306
    name: nova
    user: nova
    password: pwd
  identity:
    engine: keystone
    host: 127.0.0.1
    port: 35357
    user: nova
    password: pwd
    tenant: service
  message_queue:
    engine: rabbitmq
    host: 127.0.0.1
    port: 5672
    user: openstack
    password: pwd
    virtual_host: '/openstack'
  image:
    engine: glance
    host: 127.0.0.1
    port: 9292
  network:
    engine: neutron
    host: 127.0.0.1
    port: 9696
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      user: neutron
```

(continues on next page)
Group and user to be used for QEMU processes run by the system instance

```
nova:
  compute:
    enabled: true
    ...
  qemu:
    user: nova
    group: cinder
    dynamic_ownership: 1
```

Group membership for user nova (upgrade related)

```
nova:
  compute:
    enabled: true
    ...
  user:
    groups:
      - libvirt
```

Nova services on compute node with OpenContrail

```
nova:
  compute:
    enabled: true
    ...
    networking: contrail
```

Nova services on compute node with memcached caching

```
nova:
  compute:
    enabled: true
    ...
    cache:
      engine: memcached
      members:
        - host: 127.0.0.1
          port: 11211
        - host: 127.0.0.1
          port: 11211
```

Client-side RabbitMQ HA setup

```
nova:
  compute:
    ....
    message_queue:
```

(continues on next page)
### Nova with ephemeral configured with Ceph

```
nova:
    compute:
        enabled: true
        ...
    ceph:
        ephemeral: yes
        rbd_pool: nova
        rbd_user: nova
        secret_uuid: 03006edd-d957-40a3-ac4c-26cd254b3731
        ...
```

### Nova with ephemeral configured with LVM

```
nova:
    compute:
        enabled: true
        ...
    lvm:
        ephemeral: yes
        images_volume_group: nova_vg

linux:
    storage:
        lvm:
            nova_vg:
                name: nova_vg
                devices:
                    - /dev/sdf
                    - /dev/sdd
                    - /dev/sdg
                    - /dev/sde
                    - /dev/sdc
                    - /dev/sdj
                    - /dev/sdh
```

### Enable Barbican integration

```
nova:
    compute:
        ...
    barbican:
        enabled: true
```

### Nova metadata custom bindings

```
```
Client role

Nova configured with NFS

Nova flavors

Availability zones

(continues on next page)
server:
    identity:
        availability_zones:
            - availability_zone_01
            - availability_zone_02

Aggregates

nova:
    client:
        enabled: true
    server:
        identity:
            aggregates:
                - aggregate1
                - aggregate2

Upgrade levels

nova:
    controller:
        upgrade_levels:
            compute: juno

nova:
    compute:
        upgrade_levels:
            compute: juno

SR-IOV

Add PciPassthroughFilter into scheduler filters and NICs on specific compute nodes.

nova:
    controller:
        sriov: true

nova:
    compute:
        sriov:
            nic_one:
                devname: eth1
                physical_network: physnet1

CPU pinning & Hugepages

CPU pinning of virtual machine instances to dedicated physical CPU cores. Hugepages mount point for libvirt.
SaltStack-Formulas Documentation, Release master

### Custom Scheduler filters

If you have a custom filter, that needs to be included in the scheduler, then you can include it like so:

```yaml
nova:
  controller:
    scheduler_custom_filters:
      - my_custom_driver.nova.scheduler.filters.my_custom_filter.MyCustomFilter

  scheduler_default_filters: "DifferentHostFilter,SameHostFilter,RetryFilter,
    AvailabilityZoneFilter,RamFilter,CoreFilter,DiskFilter,ComputeFilter,
    ComputeCapabilitiesFilter,ImagePropertiesFilter,ServerGroupAntiAffinityFilter,
    ServerGroupAffinityFilter,NUMATopologyFilter,AggregateInstanceExtraSpecsFilter"
```

### Hardware Trip/Unmap Support

To enable TRIM support for ephemeral images (thru nova managed images), libvirt has this option.

```yaml
nova:
  compute:
    libvirt:
      hw_disk_discard: unmap
```

In order to actually utilize this feature, the following metadata must be set on the image as well, so the SCSI unmap is supported.

```bash
glance image-update --property hw_scsi_model=virtio-scsi <image>
glance image-update --property hw_disk_bus=scsi <image>
```

### Scheduler Host Manager

Specify a custom host manager.
libvirt CPU mode

Allow setting the model of CPU that is exposed to a VM. This allows better support live migration between hypervisors with different hardware, among other things. Defaults to host-passthrough.

```yaml
nova:
  controller:
    scheduler_host_manager: ironic_host_manager

  compute:
    cpu_mode: host-model
```

Nova compute workarounds

Live snapshotting is disabled by default in nova. To enable this, it needs a manual switch.

From manual:

```bash
# When using libvirt 1.2.2 live snapshots fail intermittently under load
# (likely related to concurrent libvirt/qemu operations). This config
# option provides a mechanism to disable live snapshot, in favor of cold
# snapshot, while this is resolved. Cold snapshot causes an instance
# outage while the guest is going through the snapshotting process.
#
# For more information, refer to the bug report:
#
# https://bugs.launchpad.net/nova/+bug/1334398
```

Configurable pillar data:

```yaml
nova:
  compute:
    workaround:
      disable_libvirt_livesnapshot: False
```

Config drive options

See example below on how to configure the options for the config drive.

```yaml
nova:
  compute:
    config_drive:
      forced: True  # Default: True
      cdrom: True  # Default: False
      format: iso9660  # Default: vfat
      inject_password: False  # Default: False
```

Number of concurrent live migrates

Default is to have no concurrent live migrations (so 1 live-migration at a time).

Excerpt from config options page (https://docs.openstack.org/ocata/config-reference/compute/config-options.html):
Maximum number of live migrations to run concurrently. This limit is enforced to avoid outbound live migrations overwhelming the host/network and causing failures. It is not recommended that you change this unless you are very sure that doing so is safe and stable in your environment.

Possible values:

- 0: treated as unlimited.
- Negative value defaults to 0.
- Any positive integer representing maximum number of live migrations to run concurrently.

To configure this option:

```
nova:
  compute:
    max_concurrent_live_migrations: 1  # (1 is the default)
```

### Enhanced logging with logging.conf

By default logging.conf is disabled.

**That is possible to enable per-binary logging.conf with new variables:**

- openstack_log_appender - set it to true to enable log_config_append for all OpenStack services;
- openstack_fluentd_handler_enabled - set to true to enable FluentHandler for all Openstack services.
- openstack_ossyslog_handler_enabled - set to true to enable OSSysLogHandler for all Openstack services.

Only WatchedFileHandler, OSSysLogHandler and FluentHandler are available.

Also it is possible to configure this with pillar:

```
nova:
  controller:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
          enabled: true
        ossyslog:
          enabled: true
  compute:
    logging:
      log_appender: true
      log_handlers:
        watchedfile:
          enabled: true
        fluentd:
          enabled: true
        ossyslog:
          enabled: true
```
**Documentation and Bugs**

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-nova/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-nova

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Rally**

Rally is a Benchmark-as-a-Service project for OpenStack.

**Sample pillars**

```yaml
rally:
  benchmark:
    enabled: true
    source:
      engine: git
      address: git://github.com/stackforge/rally.git
      revision: master
    database:
      engine: mysql
      host: 10.10.20.20
      port: 3306
      name: rally
      user: rally
      password: password
    provider:
      example_cloud:
        auth:
          auth_url: http://example.net:5000/v2.0/
          username: admin
          password: myadminpass
          tenant_name: demo
          endpoint_type: internal
        tests:
```

(continues on next page)
- nova_volumes
- neutron_networks

Rally client with specified git scenarios

```yaml
rally:
  client:
    enabled: true
    source:
      engine: git
      address: git@repo.domain.com/heat-templates.git
      revision: master
```

Read more

- https://www.mirantis.com/blog/rally-openstack-tempest-testing-made-simpler/
- https://wiki.openstack.org/wiki/Rally
- https://wiki.openstack.org/wiki/Rally/HowTo
- https://launchpad.net/rally
- https://github.com/stackforge/rally
- https://trello.com/b/DoD8aeZy/rally

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-rally/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-rally

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Sahara formula

The Sahara project provides a simple means to provision a data-intensive application cluster (Hadoop or Spark) on top of OpenStack.

Sample pillars

```yaml
sahara:
  server:
    enabled: true
    version: kilo
    bind:
      host: 0.0.0.0
      port: 8386
  database:
    engine: mysql
    host: 127.0.0.1
    port: 3306
    name: sahara
    user: sahara
    password: password
  identity:
    engine: keystone
    protocol: http
    host: 127.0.0.1
    port: 35357
    tenant: sahara
    user: sahara
    password: password
  message_queue:
    engine: rabbitmq
    port: 5672
    members:
    - host: 192.168.1.13
    - host: 192.168.1.14
    - host: 192.168.1.15
    user: openstack
    password: supersecret
    virtual_host: '/openstack'
```

Usage

Get Vanilla glance images

- http://sahara-files.mirantis.com/sahara-juno-vanilla-1.2.1-centos-6.5.qcow2

Register image in sahara
sahara image-register --image-id $IMAGE_ID --username ubuntu
sahara image-add-tag --image-id $IMAGE_ID --tag vanilla
sahara image-add-tag --image-id $IMAGE_ID --tag 1.2.1

Make sure that image is registered correctly

sahara image-list

External links

- http://docs.openstack.org/developer/sahara/userdoc/vanilla_plugin.html
- http://docs.openstack.org/developer/sahara/devref/quickstart.html
- http://docs.openstack.org/developer/sahara/horizon/installation.guide.html

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-sahara/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-sahara

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Swift Formula

Swift is a highly available, distributed, eventually consistent object/blob store. Organizations can use Swift to store lots of data efficiently, safely, and cheaply.
Sample Metadata

Swift proxy

```
swift:
  common:
    cache:
      engine: memcached
      members:
      - host: 127.0.0.1
        port: 11211
      - host: 127.0.0.1
        port: 11211
    enabled: true
    version: kilo
    swift_hash_path_suffix: hash
    swift_hash_path_prefix: hash
  proxy:
    version: kilo
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8080
    identity:
      engine: keystone
      host: 127.0.0.1
      port: 35357
      user: swift
      password: pwd
      tenant: service
```

Swift storage

```
swift:
  common:
    cache:
      engine: memcached
      members:
      - host: 127.0.0.1
        port: 11211
      - host: 127.0.0.1
        port: 11211
    version: kilo
    enabled: true
    swift_hash_path_suffix: hash
    swift_hash_path_prefix: hash
  object:
    enabled: true
    version: kilo
    bind:
      address: 0.0.0.0
      port: 6000
    container:
      enabled: true
```

(continues on next page)
version: kilo
allow_versions: true
bind:
  address: 0.0.0.0
  port: 6001
account:
  enabled: true
version: kilo
bind:
  address: 0.0.0.0
  port: 6002

To enable object versioning feature

swift:
  ....
container:
  ....
  allow_versions: true
  ....

Ring builder

parameters:
  swift:
    ring_builder:
      enabled: true
      rings:
        - name: default
          partition_power: 9
          replicas: 3
          hours: 1
          region: 1
          devices:
            - address: ${_param:storage_node01_address}
              device: vdb
            - address: ${_param:storage_node02_address}
              device: vdc
            - address: ${_param:storage_node03_address}
              device: vdd
            - partition_power: 9
              replicas: 2
              hours: 1
              region: 1
              devices:
                - address: ${_param:storage_node01_address}
                  device: vdb
                - address: ${_param:storage_node02_address}
                  device: vdc

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:
In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-swift/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-swift

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

**Tempest Formula**

This is a set of integration tests to be run against a live OpenStack cluster. Tempest has batteries of tests for OpenStack API validation, Scenarios, and other specific tests useful in validating an OpenStack deployment.

**Sample Pillars**

```yaml
tempest:
  test:
    enabled: true
    source:
      engine: git
      address: git://github.com/openstack/tempest.git
      revision: master
    suite:
      identity:
        disable_ssl_certificate_validation: true
        auth_version: v3
        uri_v3:
          region: RegionOne
        identity-feature-enabled:
          trust: true
          api_v2: false
          api_v3: true
```

**More Information**

- [http://docs.openstack.org/developer/tempest/overview.html](http://docs.openstack.org/developer/tempest/overview.html)
- [http://www.slideshare.net/masayukiigawa/tempest-scenariotests-20140512?related=1](http://www.slideshare.net/masayukiigawa/tempest-scenariotests-20140512?related=1)
- [https://github.com/stackforge/puppet-tempest](https://github.com/stackforge/puppet-tempest)
Programming Languages

Support programming languages, libraries, environments.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>java</td>
<td><a href="https://github.com/salt-formulas/salt-formula-java">https://github.com/salt-formulas/salt-formula-java</a></td>
</tr>
<tr>
<td>nodejs</td>
<td><a href="https://github.com/salt-formulas/salt-formula-nodejs">https://github.com/salt-formulas/salt-formula-nodejs</a></td>
</tr>
<tr>
<td>php</td>
<td><a href="https://github.com/salt-formulas/salt-formula-php">https://github.com/salt-formulas/salt-formula-php</a></td>
</tr>
<tr>
<td>python</td>
<td><a href="https://github.com/salt-formulas/salt-formula-python">https://github.com/salt-formulas/salt-formula-python</a></td>
</tr>
<tr>
<td>ruby</td>
<td><a href="https://github.com/salt-formulas/salt-formula-ruby">https://github.com/salt-formulas/salt-formula-ruby</a></td>
</tr>
</tbody>
</table>

Java

Programming language environment.

Sample pillars

OpenJDK 8 environment with development libs

```
java:
  environment:
    enabled: true
    version: '8'
    platform: openjdk
    development: true
```

Oracle JAVA JDK 8

```
java:
  environment:
    enabled: true
    version: '8'
    platform: oracle-java
    development: true
```

Oracle JAVA JDK 9

```
java:
  environment:
    enabled: true
    version: '9'
    release: '0.1'
    build: '11'
```

(continues on next page)
Read more

- http://openjdk.java.net/install/
- http://www.wikihow.com/Install-Oracle-Java-on-Ubuntu-Linux
- https://github.com/saltstack-formulas/sun-java-formula
- https://github.com/saltstack-formulas/sun-java-formula
- https://github.com/saltstack-formulas/java-formula

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-java/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-java

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

NodeJS

Event-driven I/O server-side JavaScript environment based on V8. Includes API documentation, change-log, examples and announcements.

Sample pillars

Simplest environment
SaltStack-Formulas Documentation, Release master

```yaml
nodejs:
  environment:
    enabled: true
```

Pillar for development

```yaml
nodejs:
  environment:
    enabled: true
  development: true
```

Read more

- http://nodejs.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-nodejs/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-nodejs

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

PHP Formula

PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML.

Sample Pillars

```yaml
php:
  environment:
    enabled: true
  cache:
```

(continues on next page)
More Information


**Python formula**

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

**Available metadata**

```
service.environment.environment: Basic Python environment
service.environment.development: Python development environment
python.environment.django: Python Django environment
```

**Sample pillars**

Simple Python environment

```
python:
  environment:
    enabled: true
```

Development Python environment

```
python:
  environment:
    enabled: true
    module:
      development: true
```

Python django environment

```
python:
  environment:
    enabled: true
    module:
      django: true
```

Using offline mirrors
Read more

- https://www.python.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

    http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

    https://github.com/salt-formulas/salt-formula-python/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

    https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

    https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

    https://github.com/salt-formulas/salt-formula-python

Any questions or feedback is always welcome so feel free to join our IRC channel:

    #salt-formulas @ irc.freenode.net

Ruby programming language

Ruby is a dynamic, open source programming language with a focus on simplicity and productivity. It has an elegant syntax that is natural to read and easy to write.

Pillars

Ruby version 1.8
Ruby version 1.8

```ruby
enabled: true
version: '1.8'
development: true
```

Ruby version 1.9

```ruby
enabled: true
version: '1.9'
development: true
```

Ruby version 2.1

```ruby
enabled: true
version: '2.1'
development: true
```

Example gem deployment of Sensu plugin

```yaml
environment:
  managed: False
gem:
  sensu-plugins-elasticsearch:
    version: 0.4.3
    user: sensu
    executable: /opt/sensu/embedded/bin/gem
```

Read more


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-ruby/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-ruby

Any questions or feedback is always welcome so feel free to join our IRC channel:
Automated management of web-based applications.

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>flower</td>
<td><a href="https://github.com/salt-formulas/salt-formula-flower">https://github.com/salt-formulas/salt-formula-flower</a></td>
</tr>
<tr>
<td>jupyter</td>
<td><a href="https://github.com/salt-formulas/salt-formula-jupyter">https://github.com/salt-formulas/salt-formula-jupyter</a></td>
</tr>
<tr>
<td>leonardo</td>
<td><a href="https://github.com/salt-formulas/salt-formula-leonardo">https://github.com/salt-formulas/salt-formula-leonardo</a></td>
</tr>
<tr>
<td>mayan</td>
<td><a href="https://github.com/salt-formulas/salt-formula-mayan">https://github.com/salt-formulas/salt-formula-mayan</a></td>
</tr>
<tr>
<td>moodle</td>
<td><a href="https://github.com/salt-formulas/salt-formula-moodle">https://github.com/salt-formulas/salt-formula-moodle</a></td>
</tr>
<tr>
<td>openode</td>
<td><a href="https://github.com/salt-formulas/salt-formula-openode">https://github.com/salt-formulas/salt-formula-openode</a></td>
</tr>
<tr>
<td>redmine</td>
<td><a href="https://github.com/salt-formulas/salt-formula-redmine">https://github.com/salt-formulas/salt-formula-redmine</a></td>
</tr>
<tr>
<td>sentry</td>
<td><a href="https://github.com/salt-formulas/salt-formula-sentry">https://github.com/salt-formulas/salt-formula-sentry</a></td>
</tr>
<tr>
<td>suitecrm</td>
<td><a href="https://github.com/salt-formulas/salt-formula-suitecrm">https://github.com/salt-formulas/salt-formula-suitecrm</a></td>
</tr>
<tr>
<td>taiga</td>
<td><a href="https://github.com/salt-formulas/salt-formula-taiga">https://github.com/salt-formulas/salt-formula-taiga</a></td>
</tr>
<tr>
<td>wordpress</td>
<td><a href="https://github.com/salt-formulas/salt-formula-wordpress">https://github.com/salt-formulas/salt-formula-wordpress</a></td>
</tr>
</tbody>
</table>

**Flower Formula**

Flower is a web based tool for monitoring and administrating Celery clusters.

**Sample Pillars**

Flower single broker

```
flower:
  server:
    enabled: true
  bind:
    port: 5555
    address: 0.0.0.0
  broker:
    engine: redis
    host: localhost
    port: 6379
    number: 0
```

Flower with multiple brokers
Flower with rabbitmq broker

```yaml
flower:
  server:
    enabled: true
  message_queue:
    location_hklab01:
      bind:
        port: 5555
        address: 0.0.0.0
      broker:
        engine: rabbitmq
        host: localhost
        port: 5672
        virtual_host: /test
        user: test
        password: test
```

Flower with redis broker

```yaml
flower:
  server:
    enabled: true
  bind:
    port: 5555
    address: 0.0.0.0
  broker:
    engine: redis
    host: localhost
    port: 6379
    number: 0
```

More Information

- [https://github.com/mher/flower](https://github.com/mher/flower)

Jupyter notebook server

Open source, interactive data science and scientific computing across over 40 programming languages.

Sample pillars

Single jupyter server

```yaml
jupyter:
  server:
    enabled: true
    bind:
      address: 0.0.0.0
      port: 8888
    notebook_source:
      engine: git
      address: gitrepo
      revision: master
      requirements: true
```
Read more

- http://jupyter.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-jupyter/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-jupyter

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Django-Leonardo formula

Python/django based CMS.

Sample metadata

```yaml
leonardo:
  server:
    enabled: true
  app:
    example_app:
      enabled: true
      workers: 3
      # disable strict host check on nginx proxy at app node
      dev: true
      bind:
        address: 0.0.0.0 # ${linux:network:fqdn}
        port: 9754
        protocol: tcp
      source:
        type: 'git'
        address: 'git@repol.robotice.cz:python-apps/leonardo.git'
        rev: 'master'
```

(continues on next page)
secret_key: 'y5m^_^ak6+5(f.m^_^ak6+5(f.m^_^ak6+5(f.'
database:
    engine: 'postgresql'
    host: '127.0.0.1'
    name: 'leonardo'
    password: 'db-pwd'
    user: 'leonardo'
mail:
    host: 'mail.domain.com'
    password: 'mail-pwd'
    user: 'mail-user'
plugin:
    eshop: {}
    static: {}
    sentry: {}
my_site:
    site: true
blog:
    source:
        engine: 'git'
        address: 'git+https://github.com/django-leonardo/leonardo-module-blog._git#egg=leonardo_module_blog'

Site Name

Without setting formula produce something like this Example app from your site name site_name

leonardo:
    server:
        app:
            example_app:
                site_name: My awesome site

Site Language

leonardo:
    server:
        app:
            example_app:
                languages:
                    en:
                        default: true
cs: {}
de: {}

LDAP auth support

leonardo:
    server:
        app:

(continues on next page)
myapp:
  ldap:
    url: "ldaps://idm.example.com"
    binddn: "uid=apache,cn=users,cn=accounts,dc=example,dc=com"
    password: "secretpassword"
    basedn: "dc=example,dc=com"
    require_group: myapp-users
    flags_mapping:
      is_active: myapp-users
      is_staff: myapp-admins
      is_superuser: myapp-admins

This settings needs leonardo-auth-ldap installed.

Site Admins & Managers

leonardo:
  server:
    app:
      example_app:
        admins:
          mail@majklk.cz:
            name: majklk
          mail@newt.cz: {}
        managers:
          mail@majklk.cz:
            name: majklk
          mail@newt.cz:
            name: newt

Cache

without setting cache we get default localhost memcache with per site prefix

leonardo:
  server:
    enabled: true
  app:
    example_app:
      cache:
        engine: 'memcached'
        host: '192.168.1.1'
        prefix: 'CACHE_EXAMPLEAPP'

Workers

Leonardo uses Celery workers for long running backgrounds jobs which runs under supervisor.

Redis
leonardo:
    server:
        enabled: true
    app:
        example_app:
            worker: true
            broker:
                engine: redis
                host: 127.0.0.1
                port: 6379
                number: 0

AMQP
leonardo:
    server:
        enabled: true
    app:
        example_app:
            worker: true
            broker:
                engine: amqp
                host: 127.0.0.1
                port: 5672
                password: password
                user: example_app
                virtual_host: /

Sentry Exception Handling
leonardo:
    server:
        app:
            example_app:
                ...  
                logging:
                    engine: raven
                    dsn: http://pub:private@sentry1.test.cz/2

Backup and Initial Data
leonardo:
    server:
        enabled: true
    app:
        example_app:
            backup: true
            initial_data:
                engine: backupninja
                source: backup.com
                host: web01.webapp.prd.dio.backup.com
                name: example_app

for reinit data do this:
rm /root/postgresql/flags/leonardo_example_app-restored
su postgres
psql
drop database leonardo_example_app;
salt-call state.sls postgresql,leonardo

Gitversions

leonardo:
  server:
    enabled: true
  app:
    example_app:
      backup: true
      initial_data:
        engine: gitversions
        source: git@repol.robotice.cz:majklk/backup-test.git

You also need django-gitversions installed.

Development Mode

leonardo:
  server:
    enabled: true
  app:
    example_app:
      development: true

Init your site

experimental feature for advanced users, which provides easy way to start your site without site repository ready yet

leonardo:
  server:
    enabled: true
  app:
    example_app:
      init: true

This parameter says, run makemigrations command before other management commands.

note: In default state makemigrations generates migrations into main leonardo module(repository).

Whatever

Sometimes you need propagate plugin specifig config into your site, for this purpose we have simple but elegant solution for do this

leonardo:
  server:
    enabled: true
app:
  example_app:
    plugin:
      eshop:
        config:
          order: true

will be

ESHOP_CONFIG = {'order': True}

Note: App.config will be rendered as python object in EXAMPLE_APP_CONFIG = {'app_config': True}

More information

- https://launchpad.net/~tcpcloud
- https://github.com/django-leonardo/django-leonardo
- https://github.com/leonardo-modules/leonardo-auth-ldap

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-leonardo/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-leonardo

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Mayan Formula

Automated OCR of documents, automatic categorization, flexible metadata, extensive access control, Mayan EDMS has all this to offers and many more features to help you tame your documents.
**Sample pillars**

```yaml
mayan:
  server:
    enabled: true
    workers: 3
    bind:
      address: 0.0.0.0
      port: 9753
  source:
    type: git
    address: git@github.com:mayan-edms/mayan-edms.git
    rev: master
  database:
    engine: 'postgresql'
    host: 'localhost'
    port: 5672
    name: 'mayan'
    password: 'pass'
    user: 'mayan'
  api:
    enabled: true
    hmac_key: d2d00896183011e28eb950e5493b99d90
    uri_id: lsadfasfg468h7j9g7j9h78gk6g54fg6f
    bind:
      port: 33333
      host: 0.0.0.0
```

**Sample pillar with specific folder for documents**

```yaml
mayan:
  server:
    enabled: true
    workers: 3
    storage_location: "~/share"
    bind:
      address: 0.0.0.0
      port: 9753
  source:
    type: git
    address: git@github.com:mayan-edms/mayan-edms.git
    rev: master
  database:
    engine: 'postgresql'
    host: 'localhost'
    port: 5672
    name: 'mayan'
    password: 'pass'
    user: 'mayan'
  api:
    enabled: true
    hmac_key: d2d00896183011e28eb950e5493b99d90
    uri_id: lsadfasfg468h7j9g7j9h78gk6g54fg6f
    bind:
      port: 33333
      host: 0.0.0.0
```
More Information

- http://www.mayan-edms.com/
- http://openode.readthedocs.org/
- https://github.com/openode/mayan_pyro_api

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-mayan/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-mayan

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Moodle Formula

Moodle is a Course Management System (CMS), also known as a Learning Management System (LMS) or a Virtual Learning Environment (VLE). It is a Free web application that educators can use to create effective online learning sites.

Sample Pillars

```yaml
moodle:
  enabled: true
  apps:
    - enabled: true
      name: 'uni'
      prefix: 'uni_' # max 5 chars
      version: '2.5'
      database:
        engine: 'postgresql'
        host: '127.0.0.1'
```

(continues on next page)
name: 'moodle_uni'
password: 'pwd'
user: 'moodle_uni'
cache:
  engine: 'memcached'
  host: '127.0.0.1'
themes:
  - name: uni
    source:
      type: git
      address: git@repo.git.cz:domain/repo.git
      branch: master

More Information

- http://midact.com/content/moodle-how-enable-memcached
- http://docs.moodle.org/dev/The_Moodle_Universal_Cache_%28MUC%29
- http://docs.moodle.org/24/en/Cron

OPENODE

OPENode is open source web application for communities seeking answers for diverse problems in commercial, public or voluntary sectors. Based on flexible communication in nodes it helps to find solutions effectively and build smarter knowledgebase. It enables users to:

Ask questions and write answers Discuss specific topics in linear forums Group topics by tags Index and search documents & images using OCR technology Set public or private nodes and user rights.

Example pillar

<table>
<thead>
<tr>
<th>openode:</th>
</tr>
</thead>
<tbody>
<tr>
<td>server:</td>
</tr>
<tr>
<td>enabled: true</td>
</tr>
<tr>
<td>workers: 3</td>
</tr>
<tr>
<td>bind:</td>
</tr>
<tr>
<td>address: 0.0.0.0</td>
</tr>
<tr>
<td>port: 9753</td>
</tr>
<tr>
<td>source:</td>
</tr>
<tr>
<td>type: git</td>
</tr>
<tr>
<td>address: <a href="https://github.com/openode/openode.git">https://github.com/openode/openode.git</a></td>
</tr>
<tr>
<td>rev: master</td>
</tr>
<tr>
<td>database:</td>
</tr>
<tr>
<td>engine: 'postgresql'</td>
</tr>
<tr>
<td>host: 'localhost'</td>
</tr>
<tr>
<td>port: 5672</td>
</tr>
<tr>
<td>name: 'openode'</td>
</tr>
<tr>
<td>password: 'pass'</td>
</tr>
<tr>
<td>user: 'openode'</td>
</tr>
</tbody>
</table>

(continues on next page)
Read More

- http://openode.net/
- http://openode.readthedocs.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-openode/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-openode

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Redmine Formula

Redmine is a flexible project management web application. Written using the Ruby on Rails framework, it is cross-platform and cross-database.

Sample pillars

```yaml
redmine:
  server:
    enabled: true
    version: '2.3'
    apps:
      - name: majklk
        database:
          engine: postgresql
```
More Information

- [http://www.redmine.org/](http://www.redmine.org/)

Sentry formula

Sentry is a realtime event logging and aggregation platform. At its core it specializes in monitoring errors and extracting all the information needed to do a proper post-mortem without any of the hassle of the standard user feedback loop.

It’s important to note that Sentry should not be thought of as a log stream, but as an event aggregator. It fits somewhere in-between a simple metrics solution (such as Graphite) and a full-on log stream aggregator (like Logstash).

Sample pillars

Standalone server

```python
python:
  environment:
    enabled: true
  module:
    development: true
  sentry:
    server:
      enabled: true
      workers: 3
      secret_key: rfui34bt34bierbrebsbfhvbfdsv
      bind:
        name: sentry.domain.com
        address: 0.0.0.0
        port: 8080
    cache:
      engine: 'redis'
      host: '127.0.0.1'
      database:
        engine: 'postgresql'
        host: '127.0.0.1'
        name: 'sentry'
        password: 'pwd'
        user: 'sentry'
```
mail:
  host: domain.com
  password: pass
  user: robot@domain.com

Server behind proxy

python:
  environment:
    enabled: true
  module:
    development: true
sentry:
  server:
    enabled: true
    workers: 3
    secret_key: rfui34bt34bierbrebsbfhvbfdsv
    url: http://another.domain.cz
  bind:
    name: sentry.domain.com
    address: 0.0.0.0
    port: 8080
  cache:
    engine: 'redis'
    host: '127.0.0.1'
  database:
    engine: 'postgresql'
    host: '127.0.0.1'
    name: 'sentry'
    password: 'pwd'
    user: 'sentry'
  mail:
    host: domain.com
    password: pass
    user: robot@domain.com

More information

- https://github.com/getsentry/sentry
- https://docs.sentry.io/server/installation/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-sentry/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas
You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-sentry

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

SuiteCRM

SuiteCRM is SugarCRM, Supercharged! SuiteCRM is a fork of the popular open source SugarCRM Community Edition. This release features a host of additional open source modules, along with the standard features and functionality found within SugarCRM CE.

Sample pillars

Simple server with 1 app

```
suitecrm:
    server: enabled: true
    app:
        devel1: enabled: true
        version: '7.1.3'
        database:
            engine: 'postgresql'
            host: '127.0.0.1'
            name: 'suitecrm_devel'
            password: 'password'
            user: 'suitecrm_devel'
```

Read more

- https://suitecrm.com/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-suitecrm/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

2.1. Project Introduction
Taiga

Project management web application with scrum in mind! Built on top of Django and AngularJ.

Sample pillars

Simple taiga server

```yaml
# taiga

server:
  enabled: true
  server_name: 'taiga.domain.com'
  mail_from: 'taiga@domain.com'
  secret_key: 'y5m^_^ak6+5(f.m^_^ak6+5(f.m^_^ak6+5(f.'

cache:
  engine: 'memcached'
  host: '127.0.0.1'
  prefix: 'CACHE_TAIGA'

database:
  engine: 'postgresql'
  host: '127.0.0.1'
  name: 'taiga'
  password: 'password'
  user: 'taiga'

mail:
  host: localhost
  port: 25
  encryption: none
```

Simple taiga server with TLS mail and authentication

```yaml
# taiga

server:
  ...  
  mail:
    host: localhost
    port: 465
    user: taiga
    password: password
    encryption: tls
```

Simple taiga server with SSL mail

```yaml
# taiga

server:
  ...  
  mail:
    host: localhost
    port: 995
    user: taiga
```

(continues on next page)
Install ldap authentication plugin:

```
    taiga:
      server:
        plugin:
          taiga_contrib_ldap_auth:
            enabled: true
            source:
              engine: pip
              name: taiga-contrib-ldap-auth
            parameters:
              backend:
                ldap_server: "ldaps://idm.example.com/"
                ldap_port: 636
                bind_bind_dn: uid=taiga,cn=users,cn=accounts,dc=tcpcloud,dc=eu
                bind_bind_password: password
                ldap_search_base: "cn=users,cn=accounts,dc=tcpcloud,dc=eu"
                ldap_search_property: uid
                ldap_email_property: mail
                ldap_full_name_property: displayName
            frontend:
              loginFormType: ldap
```

Read more

- [https://github.com/taigaio](https://github.com/taigaio)

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

[https://github.com/salt-formulas/salt-formula-taiga/issues](https://github.com/salt-formulas/salt-formula-taiga/issues)

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

[https://launchpad.net/salt-formulas](https://launchpad.net/salt-formulas)

You can also join salt-formulas-users team and subscribe to mailing list:

[https://launchpad.net/~salt-formulas-users](https://launchpad.net/~salt-formulas-users)

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula:

[https://github.com/salt-formulas/salt-formula-taiga](https://github.com/salt-formulas/salt-formula-taiga)

Any questions or feedback is always welcome so feel free to join our IRC channel:
Wordpress formula

WordPress is web software you can use to create a beautiful website or blog.

Sample metadata

Simple site

```yaml
wordpress:
  server:
    app:
      app_name:
        enabled: true
        version: '4.0'
      url: example.com
      title: TCPisekWeb
      admin_user: admin
      admin_password: password
      admin_email: nikicresl@gmail.com
      core_update: false
      theme_update: false
      plugin:
        bbpress:
          engine: http
          version: latest
        git_plugin:
          engine: git
          address: git@git.domain.com:git-repo
          revision: master
      database:
        engine: mysql
        host: 127.0.0.1
        name: w_site
        password: password
        user: w_tcpisek
        prefix: tcpisek
```

Read more


Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

[http://salt-formulas.readthedocs.io/](http://salt-formulas.readthedocs.io/)

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:
For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-wordpress

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

<table>
<thead>
<tr>
<th>Formula</th>
<th>Repository</th>
</tr>
</thead>
<tbody>
<tr>
<td>ffmpeg</td>
<td><a href="https://github.com/salt-formulas/salt-formula-ffmpeg">https://github.com/salt-formulas/salt-formula-ffmpeg</a></td>
</tr>
<tr>
<td>kodi</td>
<td><a href="https://github.com/salt-formulas/salt-formula-kodi">https://github.com/salt-formulas/salt-formula-kodi</a></td>
</tr>
<tr>
<td>home-assistant</td>
<td><a href="https://github.com/salt-formulas/salt-formula-home-assistant">https://github.com/salt-formulas/salt-formula-home-assistant</a></td>
</tr>
<tr>
<td>octoprint</td>
<td><a href="https://github.com/salt-formulas/salt-formula-octoprint">https://github.com/salt-formulas/salt-formula-octoprint</a></td>
</tr>
</tbody>
</table>

### ffmpeg formula

A complete, cross-platform solution to record, convert and stream audio and video.

#### Sample pillars

```
ffmpeg:
  server:
    enabled: true
  input:
    video0:
      source: /dev/video0
      bind:
        host: 192.168.2.1
        port: 8888
        video_format: mjpeg
```

(continues on next page)
Read more

- https://www.ffmpeg.org/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-ffmpeg/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-ffmpeg

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

KODI formula

Kodi (formerly known as XBMC) is a software media center for playing videos, music, pictures, games, and more.

Sample pillars

```yaml
kodi:
  server:
    enabled: True
```
Usage

plugin repositories
wget http://kodi-czsk.github.io/repository/repo/repository.kodi-czsk/repository.kodi-czsk-1.0.0.zip
tvheadend

curl http://apt.tvheadend.org/repo.gpg.key | sudo apt-key add - apt-add-repository -r http://apt.tvheadend.org/stable
apt-add-repository http://apt.tvheadend.org/unstable apt-get update
apt-get update apt-get install tvb-t device firmware if necessary tvheadend ui - http://localhost:9981/

Read more

- https://code.google.com/p/dmd-xbmc/
- http://kodi-czsk.github.io/repository/
- https://tvheadend.org/projects/tvheadend/wiki/AptRepository
- https://kodi.tv/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-kodi/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-kodi

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

Home Assistant Formula

Home Assistant is an open-source home automation platform running on Python 3. Track and control all devices at home and automate control.
Sample Metadata

Single homeassistant service

```yaml
home_assistant:
    server:
        enabled: true
        bind:
            address: 0.0.0.0
            port: 8123
```

desktop-service service wit git based configuration

```yaml
home_assistant:
    server:
        enabled: true
        bind:
            address: 0.0.0.0
            port: 8123
    config:
        engine: git
        address: '${_param:home_assistant_config_repository}'
        branch: ${_param:home_assistant_config_revision}
```

References

- https://home-assistant.io/getting-started/

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-home-assistant/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula:

https://github.com/salt-formulas/salt-formula-home-assistant

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net
Octoprint formula

The web interface for your 3D printer.

Sample pillars

Single printer [deprecated]

```
octoprint:
  server:
    enabled: true
    source:
      engine: git
      address: 'https://github.com/foosel/OctoPrint.git'
      rev: "master"
  printer:
    engine: serial
    webcam: true
    webcam:
      host: localhost
      port: 1234
```

Multi printers setup

```
octoprint:
  server:
    enabled: true
    source:
      engine: git
      address: 'https://github.com/foosel/OctoPrint.git'
      rev: "master"
  printer:
    printer01:
      bind:
        address: 0.0.0.0
        port: 5001
      device:
        bus: serial
        port: /dev/ACM01
        model: prusa-mk2
      camera:
        protocol: mjpg
        url: localhost
        port: 1234
    printer02:
      device:
        bus: serial
        port: /dev/ACM02
        model: prusa-clone
      bind:
        address: 0.0.0.0
        port: 5002
```
More Information

- http://octoprint.org/
- https://github.com/foosel/OctoPrint

Documentation and Bugs

To learn how to install and update salt-formulas, consult the documentation available online at:

http://salt-formulas.readthedocs.io/

In the unfortunate event that bugs are discovered, they should be reported to the appropriate issue tracker. Use Github issue tracker for specific salt formula:

https://github.com/salt-formulas/salt-formula-octoprint/issues

For feature requests, bug reports or blueprints affecting entire ecosystem, use Launchpad salt-formulas project:

https://launchpad.net/salt-formulas

You can also join salt-formulas-users team and subscribe to mailing list:

https://launchpad.net/~salt-formulas-users

Developers wishing to work on the salt-formulas projects should always base their work on master branch and submit pull request against specific formula.

https://github.com/salt-formulas/salt-formula-octoprint

Any questions or feedback is always welcome so feel free to join our IRC channel:

#salt-formulas @ irc.freenode.net

2.2 Development Documentation

In this section, you will find documentation relevant to developing SaltStack formulas. How to change existing formula, how to create a new formula.
## 2.2.1 Extending

### Chapter 1. Extending

Home SaltStack-Formulas Development Documentation

### Creating New Formula with Cookiecutter

This guide shows how to use cookiecutter template to create new Salt formula.

#### Installation

Install in blank virtualenv.

```
pip install cookiecutter
cd cookiecutter
```

#### Usage

```
cookiecutter salt-formula
```

---

### Sync Multiple Repository with Myrepos

- Installation
- Clone Repositories
- Update Repositories
- More Information

Home SaltStack-Formulas Development Documentation
Installation

```bash
apt-get install myrepos
```

To add `gerrit` remote automatically, set your username:

```bash
git config --global gitreview.username johndoe
```

To avoid using `--trust-all` option, add this `.mrconfig` into trusts file:

```bash
echo $PWD/.mrconfig >> ~/.mrtrust
```

Clone Repositories

Simply run `checkout` tool without parameters or with formula names, eg.:

```bash
./checkout
./checkout nova freeipa salt
```

Or with some parallelism:

```bash
mr --trust-all --force -j 4 checkout
```

Update Repositories

Pull with rebase in each repo or only one

```bash
mr --trust-all update
mr --trust-all -d tcpcloud update
mr --trust-all -d tcpcloud/apache update
```

More Information

- [https://wiki.debian.org/Teams/Ruby/Packaging](https://wiki.debian.org/Teams/Ruby/Packaging)
- [https://myrepos.branchable.com/](https://myrepos.branchable.com/)

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation
Formula Authoring Guidelines

- **Formula Directory Structure**
- **Salt state files**
  - service/map.jinja
  - service/init.sls
  - service/role1.sls
  - service/role2/init.sls
- **Coding styles for state files**
  - Line length above 80 characters
  - Single line declaration
  - No newline at the end of the file
  - Trailing whitespace characters
- **Reclass metadata files**
  - metadata/service/role1/local.yml
  - metadata/service/role1/single.yml
  - metadata/service/role1/cluster.yml
- **Debian packaging**
  - debian/changelog
  - debian/copyright
  - debian/docs
  - debian/install
  - debian/control
- **Supplemental files**
  - README.rst
  - LICENSE
  - VERSION
  - CHANGELOG.rst
  - Versioning
- **Formula unit testing**
- **More information**

Salt formulas encapsulate specific services. This document contains guidelines to salt formula creation and maintenance.
Formula Directory Structure

Formulas follow the same directory structure as Salt official conventions and best practices described in the SaltStack documentation.

Every formula should have the following directory layout:

```
service-formula
|-- _grains/
 `-- service.yml
|-- _modules/
 `-- service.yml
|-- _states/
 `-- service.yml
|-- service/
 `-- files/
    |-- service.conf
    `-- service-systemd
 `-- meta/
    |-- sphinx.yml
    `-- colleted.yml
 `-- map.jinja
 `-- init.sls
 `-- _common.sls
 `-- role1.sls
 `-- role2/
    |-- init.sls
    `-- service.sls
 `-- more.sls
|-- debian/
    |-- changelog
    |-- compat
    |-- control
    |-- copyright
    |-- docs
    |-- install
    |-- rules
    `-- source
        `-- format
|-- metadata/
 `-- service/
    |-- role1/
    | `-- deployment1.yml
    | `-- deployment2.yml
    | `-- deployment3.yml
    `-- role2/
    |-- deployment4.yml
    `-- deployment5.yml
|-- CHANGELOG.rst
|-- LICENSE
|-- pillar.example
|-- README.rst
 `-- VERSION
```

Content of the formula directories in more detail.

_-grains/_ Optional grain modules
_-modules/_ Optional execution modules
_-states/_ Optional state modules
Salt state files

Salt state files are located in `service` directory.

**service/map.jinja**

Map file helps to clean the differences among operating systems and provides default values so there’s no need to provide default value in state files.

Following snippet uses YAML to serialize the data and is the recommended way to write `map.jinja` file as YAML can be easily extended in place.

```jinja
{%- load_yaml as role1_defaults %}
Debian:
  pkgs:
    - python-psycopg2
  dir:
    base: /srv/service/venv
    home: /var/lib/service
RedHat:
  pkgs:
    - python-psycopg2
  dir:
    base: /srv/service/venv
    home: /var/lib/service
    workspace: /srv/service/workspace
{%- endload %}

{%- set role1 = salt['grains.filter_by'](role1_defaults, merge=salt['pillar.get']('service:role1')) %}
```

Following snippet uses JSON to serialize the data and was favored in past.

```jinja
{% set api = salt['grains.filter_by']({
  'Debian': {
    'pkgs': ['salt-api'],
    'service': 'salt-api',
  },
  'RedHat': {
    'pkgs': ['salt-api'],
    'service': 'salt-api',
  },
}, merge=salt['pillar.get']('salt:api')) %}
```

Following snippet sets different common role parameters according to `service:role:source:engine pillar` variable of given service role.

---

2.2. Development Documentation
({%- set source_engine = salt['pillar.get']('service:role:source:engine') %}

({%- load_yaml as base_defaults %})

({%- if source_engine == 'git' %})

Debian:
  pkgs:
    - python-psycopg2
  dir:
    base: /srv/service/venv
    home: /var/lib/service
    workspace: /srv/service/workspace

({%- else %})

Debian:
  pkgs:
    - helpdesk
  dir:
    base: /usr/lib/service

({%- endif %})

({%- endload %})

**service/init.sls**

Conditional include of individual service roles. Basically this is essential piece that makes the usage of formulas truly model-driven. You have catalog of services and this determines according to present metadata what roles get started.

Using `service/init.sls` file allows the service catalog to be role independent.

```
include:
  {% if pillar.service.role1 is defined %}
  - service.role1
  {% endif %}

{% if pillar.service.role2 is defined %}
  - service.role2
  {% endif %}
```

You can use one file as `role1.sls` for simple roles. For more complex roles handling many resources, use individual directories as `role2`.

```
service-formula/
  `-- service/
      |-- role1.sls
      `-- role2/
          |-- init.sls
          |-- service.sls
          |-- resource1.sls
          `-- resource2.sls
```

Then you can verify the full service catalog on node by following command:

```
root@web01:~# salt-call state.show_top
[INFO ] Loading fresh modules *for* state activity
local:
  ---------
  base:
    - linux
    - openssh

(continues on next page)
- ntp
- salt
- backupninja
- git
- sphinx
- python
- nginx
- nodejs
- postgresql
- rabbitmq
- redis
- ruby

Service metadata are stored also as services grain.

```bash
root@web01:~# salt-call grains.item services
local: _---------
    services:
      - linux
      - openssh
      - ntp
      - salt
      - backupninja
      - git
      - sphinx
      - python
      - nginx
      - nodejs
      - postgresql
      - rabbitmq
      - redis
      - ruby
```

And each service roles metadata is stored as detailed roles grain.

```bash
root@web01:~# salt-call grains.item roles
local: _---------
    roles:
      - git.client
      - postgresql.server
      - nodejs.environment
      - ntp.client
      - linux.storage
      - linux.system
      - linux.network
      - redis.server
      - rabbitmq.server
      - python.environment
      - backupninja.client
      - nginx.server
      - openssh.client
      - openssh.server
      - salt.minion
      - sphinx.server
```
Note: It is recommended to run `state.sls salt` prior the `state.highstate` command as grains may not be generated properly and some configuration parameters may not be set at all.

**service/role1.sls**

Actual salt state resources that enforce service existence. Common production and recommended pattern is to install packages, setup configuration files and ensure the service is up and running.

```jinja
{% from "redis/map.jinja" import server with context %}
{% if server.enabled %}

redis_packages:
  pkg.installed:
    - names: {{ server.pkgs }}

{{ server.dir.conf }}/redis.conf:
  file.managed:
    - source: salt://redis/files/redis.conf
    - template: jinja
    - user: root
    - group: root
    - mode: 644
    - require:
      - pkg: redis_packages

redis_service:
  service.running:
    - enable: true
    - name: {{ server.service }}
    - watch:
      - file: {{ server.dir.conf }}/redis.conf

{% endif %}

{% else %}

For development purposes other installation than s

Note: The role for `role.enabled` condition is to restrict the give service role from execution with default parameters, the single error is thrown instead. You can optionally add `else` statement to disable or completely remove given service role.

**service/role2/init.sls**

This approach is used with more complex roles, it is similar to `service/init.sls`, but uses conditions to further limit the inclusion of unnecessary files.

For example Linux network role includes conditionally hosts and interfaces.

```jinja
{% from "linux/map.jinja" import network with context %}

include:
  - linux.network.hostname

{% if network.host|length > 0 %}

(continues on next page)
Coding styles for state files

Good styling practices for writing salt state declarations.

Line length above 80 characters

As a ‘standard code width limit’ and for historical reasons - IBM punch card had exactly 80 columns.

Single line declaration

Avoid extending your code by adding single-line declarations. It makes your code much cleaner and easier to parse / grep while searching for those declarations.

The bad example:

```python
pkg:
  - installed
```

The correct example:

```python
pkg.installed
```

No newline at the end of the file

Each line should be terminated in a newline character, including the last one. Some programs have problems processing the last line of a file if it isn’t newline terminated.

Trailing whitespace characters

Trailing whitespaces take more spaces than necessary, any regexp based searches won’t return lines as a result due to trailing whitespace(s).

Reclass metadata files

Each of these files serve as default metadata set for given deployment. Each service role can have several deployments. For example rabbitmq server role has following deployments:

- metadata/rabbitmq/server/local.yml
- metadata/rabbitmq/server/single.yml
metadata/rabbitmq/server/cluster.yml

metadata/service/role1/local.yml

```
applications:
  - rabbitmq
parameters:
  _param:
    rabbitmq_admin_user: admin
  rabbitmq:
    server:
      enabled: true
      secret_key: ${_param:rabbitmq_secret_key}
      bind:
        address: 127.0.0.1
        port: 5672
    plugins:
      - amqp_client
      - rabbitmq_management
    admin:
      name: ${_param:rabbitmq_admin_user}
      password: ${_param:rabbitmq_admin_password}
```

metadata/service/role1/single.yml

```
applications:
  - rabbitmq
parameters:
  _param:
    rabbitmq_admin_user: admin
  rabbitmq:
    server:
      enabled: true
      secret_key: ${_param:rabbitmq_secret_key}
      bind:
        address: 0.0.0.0
        port: 5672
    plugins:
      - amqp_client
      - rabbitmq_management
    admin:
      name: ${_param:rabbitmq_admin_user}
      password: ${_param:rabbitmq_admin_password}
```

metadata/service/role1/cluster.yml

```
applications:
  - rabbitmq
parameters:
  rabbitmq:
    server:
      enabled: true
```

(continues on next page)
secret_key: ${_param:rabbitmq_secret_key}
bind:
  address: ${_param:cluster_local_address}
  port: 5672
plugins:
- amqp_client
- rabbitmq_management
admin:
  name: admin
  password: ${_param:rabbitmq_admin_password}
host:
  '/openstack':
    enabled: true
    user: openstack
    password: ${_param:rabbitmq_openstack_password}
policies:
  - name: HA
    pattern: '^(?!amq\.).*$'
    definition: '{"ha-mode": "all"}'
cluster:
  enabled: true
  name: openstack
  role: ${_param:rabbitmq_cluster_role}
  master: ${_param:cluster_node01_hostname}
  mode: disc
  members:
  - name: ${_param:cluster_node01_hostname}
    host: ${_param:cluster_node01_address}
  - name: ${_param:cluster_node02_hostname}
    host: ${_param:cluster_node02_address}
  - name: ${_param:cluster_node03_hostname}
    host: ${_param:cluster_node03_address}

Parameters like ${_param:rabbitmq_secret_key} are interpolation of common parameter passed from higher system or cluster levels.

**Debian packaging**

Use of debian packaging is preferable way for deploying production salt masters and it’s formulas. Take basic structure of debian directory from some existing formula and modify to suit your formula.

Description of most important files follows.

**debian/changelog**

salt-formula-salt (0.1) trusty; urgency=medium

+ Initial release

-- Ales Komarek <ales.komarek@tcpcloud.eu> Thu, 13 Aug 2015 23:23:41 +0200
**debian/copyright**

Licensing informations of the package.

| Format: | http://www.debian.org/doc/packaging-manuals/copyright-format/1.0/ |
| Upstream-Name: | salt-formula-salt |
| Upstream-Contact: | Ales Komarek <ales.komarek@tcpcloud.eu> |
| Source: | https://github.com/tcpcloud/salt-formula-salt |

Files: *

License: Apache-2.0

- Copyright (C) 2014-2015 tcp cloud a.s.
- Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License.
- On a Debian system you can find a copy of this license in /usr/share/common-licenses/Apache-2.0.

**debian/docs**

Files listed here will be available in /usr/share/doc. Don’t put COPYRIGHT or LICENSE files here as they are handled in a different way.

- README.rst
- CHANGELOG.rst
- VERSION

**debian/install**

Defines what is going to be installed in which location.

| salt/* | /usr/share/salt-formulas/env/salt/ |
| metadata/service/* | /usr/share/salt-formulas/reclass/service/salt/ |

**debian/control**

This file keeps metadata of source and binary package.

| Source: | salt-formula-salt |
| Maintainer: | tcpcloud Packaging Team <pkg-team@tcpcloud.eu> |
| Section: | admin |
| Priority: | optional |
| Build-Depends: | debhelper (>= 9) |
| Standards-Version: | 3.9.6 |
| Homepage: | http://www.tcpcloud.eu |
| Vcs-Browser: | https://github.com/tcpcloud/salt-formula-salt |
| Vcs-Git: | https://github.com/tcpcloud/salt-formula-salt.git |

Package: salt-formula-salt
Architecture: all

(continues on next page)
Supplemental files

Files that are required to complete information about given formula.

**README.rst**

A sample skeleton of the README.rst file:

```rst
=======
service
=======
Install and configure the Specific service.

.. note::
    See the full `Salt Formulas installation and usage instructions <https://docs.saltstack.com/en/latest/topics/development/conventions/formulas.html>`_.

Available states
================

.. contents::
   :local:
   ``service``

``service.role1``
-----------------
Setup individual role.

Available metadata
===================

.. contents::
   :local:
   ``metadata.service.role.single``

Setup from system packages.
```
```
```
``metadata.service.role.development``
--------------------------------------
Setup from git repository.

Configuration parameters
========================

``service_secret_key``
-------------------------------
``rabbitmq_service_password``
--------------------------------------
``postgresql_service_password``
---------------------------------------
If development is setup.

``service_source_revision``
-------------------------------
If development is setup.

Example reclass
==============

Production setup

```
.. code-block:: yaml

  service-single:
    name: service-single
    domain: dev.domain.com
    classes:
    - system.service.server.single
    params:
      rabbitmq_admin_password: cwerfwefzdcdsf
      rabbitmq_secret_key: fdsfdwdfsdfsfs
      rabbitmq_service_password: fdsf24fsdfsdacaf
      keystone_service_password: fdaasdfdasfDasfda
      postgresql_service_password: dfsfdfdasfda
      nginx_site_service_host: $\{linux:network:fqdn\}
      service_secret_key: fda32r
```

Development setup

```
.. code-block:: yaml

  service-single:
    name: service-single
    domain: dev.domain.com
    classes:
```
- system.service.server.development
  params:
  rabbitmq_admin_password: cwerfwefzdcdsf
  rabbitmq_secret_key: fsdfwfsfdfsdf
  rabbitmq_service_password: fdsf24fsdfsdfcadfd
  keystone_service_password: fdasdfsdfdasdfsdfda
  postgresql_service_password: dfdasdfsdfda
  nginx_site_service_host: ${linux:network: fqdn}
  service_secret_key: fda32r
  service_source_repository: git@git.tcpcloud.eu:python-apps/service.git
  service_source_revision: feature/243

Example pillar
===============

Install from specific branch of Git

.. code-block:: yaml

    service:
      server:
        source:
          engine: 'git'
          address: 'git@git.tcpcloud.eu:python-apps/service.git'
          revision: 'feature/214'

To enable debug logging for both Django and Gunicorn and raise number of Gunicorn workers

.. code-block:: yaml

    service:
      server:
        log_level: 'debug'
        workers: 8

To change where Django listens

.. code-block:: yaml

    service:
      server:
        bind:
          address: 'not-localhost'
          port: 9755

Read more
========

* http://doc.tcpcloud.eu/
LICENSE

Contains license information and terms & conditions how you are allowed to use and distribute the files of the underlying directories.

Copyright (c) 2014-2015 Your name

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

VERSION

Latest version number, git repository tag, package version as well.

0.0.2

CHANGELOG.rst

The CHANGELOG.rst file should detail the individual versions, their release date and a set of bullet points for each version highlighting the overall changes in a given version of the formula.

A sample skeleton of the CHANGELOG.rst file:

CHANGELOG.rst:

```
service formula
===============
0.0.2 (2014-01-01)
- Re-organized formula file layout
- Fixed filename used for upstart logger template
- Allow for pillar message to have default if none specified

0.0.1 (2013-01-01)
- Initial formula setup
```

Versioning

Formula are versioned according to Semantic Versioning, http://semver.org/.

Note: Given a version number MAJOR.MINOR.PATCH, increment the:
1. MAJOR version when you make incompatible API changes,
2. MINOR version when you add functionality in a backwards-compatible manner, and
3. PATCH version when you make backwards-compatible bug fixes.

Additional labels for pre-release and build metadata are available as extensions to the MAJOR.MINOR.PATCH format.

Formula versions are tracked using Git tags as well as the VERSION file in the formula repository. The VERSION file should contain the currently released version of the particular formula.

**Formula unit testing**

A smoke-test for invalid Jinja, invalid YAML, or an invalid Salt state structure can be performed by with the `state.show_sls` function:

```
salt '*' state.show_sls service-name
```

Salt Formulas can then be tested by running each `.sls` file via `state.sls` and checking the output for the success or failure of each state in the Formula. This should be done for each supported platform.

```
salt '*' state.sls sls-file-name test
```

**More information**


**Contributor Guidelines**

- **Bugs**
  - **Tags**
  - **Status**
  - **Importance**
  - **Triaging Bugs**
- **Submitting Code**
Bugs

Bugs should be filed on Bug Launchpad for SaltStack-formulas.

When submitting a bug, or working on a bug, please ensure the following criteria are met:

- The description clearly states or describes the original problem or root cause of the problem.
- Include historical information on how the problem was identified.
- Any relevant logs are included.
- If the issue is a bug that needs fixing in a branch other than master, please note the associated branch within the launchpad issue.
- The provided information should be totally self-contained. External access to web services/sites should not be needed.
- Steps to reproduce the problem if possible.

Tags

If it’s a bug that needs fixing in a branch in addition to master, add a `<release>-backport-potential` tag (e.g. `kilo-backport-potential`). There are predefined tags that will auto-complete.

Status

Please leave the status of an issue alone until someone confirms it or a member of the bugs team triages it. While waiting for the issue to be confirmed or triaged the status should remain as New.

Importance

Should only be touched if it is a Blocker/Gating issue. If it is, please set to High, and only use Critical if you have found a bug that can take down whole infrastructures. Once the importance has been changed the status should be changed to Triaged by someone other than the bug creator.

Triaging Bugs

Reported bugs need prioritization, confirmation, and shouldn’t go stale. If you care about OpenStack stability but aren’t wanting to actively develop the roles and playbooks used within the “salt-formulas” project consider contributing in the area of bug triage, which helps immensely. The whole process is described in the upstream Bug Triage Documentation.

Submitting Code

- Write good commit messages. We follow the OpenStack “Git Commit Good Practice” guide. If you have any questions regarding how to write good commit messages please review the upstream OpenStack documentation.
• Changes to the project should be submitted for review via the Gerrit tool, following the workflow documented here.

• Pull requests submitted through GitHub will be ignored and closed without regard.

• All feature additions/deletions should be accompanied by a blueprint/spec. ie: adding additional active agents to neutron, developing a new service role, etc...

• Before creating blueprint/spec an associated issue should be raised on launchpad. This issue will be triaged and a determination will be made on how large the change is and whether or not the change warrants a blueprint/spec. Both features and bug fixes may require the creation of a blueprint/spec. This requirement will be voted on by core reviewers and will be based on the size and impact of the change.

• All blueprints/specs should be voted on and approved by core reviewers before any associated code will be merged. For more information on blueprints/specs please review the upstream OpenStack Blueprint documentation. At the time the blueprint/spec is voted on a determination will be made whether or not the work will be backported to any of the “released” branches.

• Patches should be focused on solving one problem at a time. If the review is overly complex or generally large the initial commit will receive a “-2” and the contributor will be asked to split the patch up across multiple reviews. In the case of complex feature additions the design and implementation of the feature should be done in such a way that it can be submitted in multiple patches using dependencies. Using dependent changes should always aim to result in a working build throughout the dependency chain. Documentation is available for advanced gerrit usage too.

• All patch sets should adhere to the Salt Style Guide listed here as well as adhere to the Salt best practices when possible.

• All changes should be clearly listed in the commit message, with an associated bug id/blueprint along with any extra information where applicable.

• Refactoring work should never include additional “rider” features. Features that may pertain to something that was re-factored should be raised as an issue and submitted in prior or subsequent patches.

Backporting

• Backporting is defined as the act of reproducing a change from another branch. Unclean/squashed/modified cherry-picks and complete reimplementations are OK.

• Backporting is often done by using the same code (via cherry picking), but this is not always the case. This method is preferred when the cherry-pick provides a complete solution for the targeted problem.

• When cherry-picking a commit from one branch to another the commit message should be amended with any files that may have been in conflict while performing the cherry-pick operation. Additionally, cherry-pick commit messages should contain the original commit SHA near the bottom of the new commit message. This can be done with cherry-pick –x. Here’s more information on Submitting a change to a branch for review.

• Every backport commit must still only solve one problem, as per the guidelines in Submitting Code.

• If a backport is a squashed set of cherry-picked commits, the original SHAs should be referenced in the commit message and the reason for squashing the commits should be clearly explained.

• When a cherry-pick is modified in any way, the changes made and the reasons for them must be explicitly expressed in the commit message.

• Refactoring work must not be backported to a “released” branch.
Style Guide

When creating tasks and other roles for use in Salt please create them using the YAML dictionary format.

Example YAML dictionary format:

```yaml
- name: The name of the tasks
  module_name:
    thing1: "some-stuff"
    thing2: "some-other-stuff"
  tags:
    - some-tag
    - some-other-tag
```

Example what **NOT** to do:

```yaml
- name: The name of the tasks
  module_name: thing1="some-stuff" thing2="some-other-stuff"
  tags: some-tag
```

```yaml
- name: The name of the tasks
  module_name: >
    thing1="some-stuff"
    thing2="some-other-stuff"
  tags: some-tag
```

Usage of the “>” and “|” operators should be limited to Salt conditionals and command modules such as the Salt shell or command.

• Documentation Home
• Project Introduction
• Installation and Operations Manual
• Development Documentation

2.2.2 Testing

Chapter 2. Testing

Home SaltStack-Formulas Development Documentation
Testing Coding Style

- Using Double Quotes with no Variables
- Line Length Above 80 Characters
- Single Line Declarations
- No Newline at the End of the File
- Trailing Whitespace Characters

Formulas are pre-written Salt States. They are as open-ended as Salt States themselves and can be used for tasks such as installing a package, configuring, and starting a service, setting up users or permissions, and many other common tasks. They have certain rules that needs to be adhered.

**Using Double Quotes with no Variables**

In general - it’s a bad idea. All the strings which does not contain dynamic content (variables) should use single quote instead of double.

**Line Length Above 80 Characters**

As a ‘standard code width limit’ and for historical reasons - [IBM punch card](http://en.wikipedia.org/wiki/Punched_card) had exactly 80 columns.

**Single Line Declarations**

Avoid extending your code by adding single-line declarations. It makes your code much cleaner and easier to parse / grep while searching for those declarations.

**No Newline at the End of the File**

Each line should be terminated in a newline character, including the last one. Some programs have problems processing the last line of a file if it isn’t newline terminated. [Stackoverflow thread](http://stackoverflow.com/questions/729692/why-should-files-end-with-a-newline)

**Trailing Whitespace Characters**

Trailing whitespaces take more spaces than necessary, any regexp based searches won’t return lines as a result due to trailing whitespace(s).
Pillars are tree-like structures of data defined on the Salt Master and passed through to the minions. They allow confidential, targeted data to be securely sent only to the relevant minion. Pillar is therefore one of the most important systems when using Salt.

**Testing Scenarios**

Testing plan tests each formula with the example pillars covering all possible deployment setups:

The first test run covers `state.show_sls` call to ensure that it parses properly with debug output.

The second test covers `state.sls` to run the state definition, and run `state.sls` again, capturing output, asserting that `^Not Run:` is not present in the output, because if it is then it means that a state cannot detect by itself whether it has to be run or not and thus is not idempotent.

**File metadata.yml**

```yaml
name: "service"
version: "0.2"
source: "https://github.com/tcpcloud/salt-formula-service"
```

---

**Testing Salt Formulas**

- Generate Test Structures in Formula
- Formula Testing with Test Kitchen
  - Using Test Kitchen
  - How it Works
  - Verifying Deployment
Each formula contains `Makefile` with at least `test` target. Under `tests` directory are located resources for test execution.

Test target executes "smoke test" implemented by `tests/run_tests.sh` capable to fetch dependencies in python virtual environment by executing `salt-call state.show_sls` with provided `tests/pillar` data.

The purpose of the smoke test is to find syntax, typo issues and verify example pillar data against the formula.

Initial content of `tests` folder contains test pillars and a `run_tests.sh` as generated by `cookiecutter`.

```
$ ls tests
  pillar
     client_single.sls
     server_single.sls
  run_tests.sh
```

Create or update pillars in `tests/pillar/\*\.sls` with test data.

### Generate Test Structures in Formula

There is and salt-formulas `cookiecutter` template. to generate initial repository structure for new formula.

For existing formulas there is an convenient script capable to generate initial structures from available content. For more details follow the README in the above linked repository. To simply generate test structures according specification stated in this document simply run `kitchen-init.sh`.

**tl;dr:**

```
```

### Formula Testing with Test Kitchen

Test Kitchen with forked `kitchen-salt` provisioner plugin may be used for local development as well as CI scenario.

Test Kitchen is a test harness tool to execute your configured code on one or more platforms in isolation. There is a `.kitchen.yml` in main directory that defines `platforms` to be tested and `suites` to execute on them.

Kitchen CI can spin instances locally or remote, based on used `driver`. For example `.kitchen.yml` may define a `docker` or `vagrant` driver.

For more, explore it’s rich `ecosystem` of supported drivers/provisioners/verifiers/…

### Using Test Kitchen

A listing of scenarios to be executed:
The **Busser Verifier** is used to setup and run tests implemented in `<repo>/test/integration`. It installs the particular driver to tested instance (Serverspec, InSpec, Shell, Bats, ...) prior the verification is executed.

Example workflow:

```bash
# list instances and status
kitchen list

# manually execute integration tests
kitchen [test || [create|converge|verify|exec|login|destroy|...]] [instance] -t tests/integration

# use with provided Makefile (ie: within CI pipeline)
make kitchen
```

**How it Works**

Kitchen spin an instances in Docker, Vagrant, OpenStack environment, etc. based on configured driver. Instance is configured as salt minion, where the configuration is defined by `.kitchen.yml` and `tests/pillar/*.sls`

Override your specific needs with `.kitchen.<backend|local>.yml` that you may load as: `KITCHEN_LOCAL_YAML=.kitchen.<driver>.yml kitchen <action> <suite>`.

Example: `KITCHEN_LOCAL_YAML=.kitchen.local kitchen verify server-ubuntu-1404 -t tests/integration`.

Test Kitchen then allows you execute several action to perform your testing under configured conditions:

1. **create**, provision an test instance (VM, container)
2. **converge**, run a provisioner (shell script, kitchen-salt)
3. **verify**, run a verification (inspec, other may be added)
4. **destroy**

**Verifying Deployment**

There is couple of verifier plugins that are shipped with Test Kitchen. They allow to run simple bash scripts and checking it’s exit codes to run specific purpose based frameworks.

The **Busser Verifier** goes with test-kitchen by default. It is used to setup and run tests implemented in `<repo>/test/integration`. It guess and installs the particular driver to tested instance. By default **InSpec** is expected.

You may avoid to install busser framework if you configure specific verifier in `.kitchen.yml`:

```yaml
verifier:
  name: inspec
```
For default Inspec **Verifier** implement your scripts directly in `<repo>/test/integration/<suite>` directory with `_spec.rb` suffix.

If you would to write another verification scripts than InSpec store them in `<repo>/tests/integration/<suite>/<verifier>`. Busser <https://github.com/test-kitchen/busser> is a test setup and execution framework under test kitchen.

Implement integration tests under `<repo>/tests/integration/<suite>/<verifier>` directory with `_spec.<verifier suffix>` filename suffix.

**InSpec**

InSpec is native validation framework for Test Kitchen and as such don’t require usage of `<verifier>` folder. Thus the tests may be stored directly under `<repo>/tests/integration/<suite>`

Additional resources.

- https://inspec.io
- https://github.com/chef/inspec
- https://github.com/chef/kitchen-inspec

Example verification scripts under `tests/integration` folder of the formula:

```
tests
  integration
    default
      default_testcase_spec.rb # Written in InSpec
    backupmx
      serverspec # <Verifier framework>
        backupmx_spec.rb # Written in ServerSpec
    helpers
      serverspec
        spec_helper.rb
    relay
      serverspec
        relay_spec.rb
    server
      serverspec
        aliases_spec.rb
        server_spec.rb
  pillar
    backupmx.sls
    relay.sls
    server.sls
    run_tests.sh
```

**Requirements**

Use latest stable [kitchen-salt](https://github.com/saltstack/kitchen-salt) and kitchen-test.

**TL;DR**

First you have to install ruby package manager `gem`.

Install required gems:

```
# Ruby side:
gem install <gem name from the list below>
```

(continues on next page)
# Isolated w/Bundler

gem install bundler

cat > Gemfile <-EOF
    source 'https://rubygems.org'
    gem 'rake'
    gem 'test-kitchen'
    gem 'kitchen-docker'
    gem 'kitchen-inspec'
    gem 'inspec'
    gem 'kitchen-salt', :git => 'https://github.com/salt-formulas/kitchen-salt.git'
EOF

bundle install [--path $PWD/.vendor/bundle]

# use with prefix 'bundle kitchen':
# bundle exec kitchen list

Create aliases:

cat > ~/.${SHELL}rc <-EOF
    alias bk='nocorrect bundle exec kitchen'
    alias kl='nocorrect bundle exec kitchen list'
EOF


## Install procedure

One may be satisfied installing ruby and gems system-wide right from OS package manager.

If you are an ruby/chef developer you will probably want to use [ChefDK](https://downloads.chef.io/chefdk).

For advanced users or the sake of complex environments you may use [rbenv](https://github.com/rbenv/rbenv) for user side ruby installation.

- [https://github.com/rbenv/rbenv](https://github.com/rbenv/rbenv)
- [http://kitchen.ci/docs/getting-started/installing](http://kitchen.ci/docs/getting-started/installing)

An example steps to install user side ruby and prerequisites:

# Use package manager to install rbenv and ruby-build

sudo apt-get install rbenv ruby-build

# list all available versions:

    rbenv install -l

# install a Ruby version of your choice or pick latest

    rbenv install $(rbenv install -l|grep -E '^[ ]*\[0-9]\.[0-9]'+|tail -1)

# activate

    rbenv local 2.4.0

# it's usually a good idea to update rubygems first

    rbenv exec gem update --system
# install test kitchen
rbenv exec gem install bundler
rbenv exec gem install test-kitchen

Continue with the optional Gemfile in the formula main directory to fetch fine tuned dependencies. If you use Gemfile and Bundler for local dependencies prepend all command with `rbenv exec bundler exec` and possibly set an alias in your ~/.bashrc, etc.

```
cat >> ~/.bashrc <<EOF
    alias rk="rbenv exec kitchen"
    alias bk="rbenv exec bundler exec kitchen"
EOF
```

With such alias set, you should be able to execute `rbenv exec bundler exec make kitchen` and see test results.

## Sample Configurations

For advanced configs have a look at `.kitchen*.yml` examples in cookiecutter template <https://github.com/salt-formulas/salt-formulas/tree/master/cookiecutter/salt-formula/%7B%7Bcookiecutter.service_name%7D%7D>_.

```
#.kitchen.yml
---
driver:
    name: docker
    hostname: opencontrail
    use_sudo: true

provisioner:
    name: salt_solo
    salt_install: bootstrap
    salt_bootstrap_url: https://bootstrap.saltstack.com
    salt_version: latest
    require_chef: false
    log_level: error
    formula: opencontrail
    grains:
        noservices: True
    dependencies:
        - name: linux
          repo: git
          source: https://github.com/salt-formulas/salt-formula-linux
    state_top:
        base:
          ":":
            - linux
            - opencontrail
        pillars:
            top.sls:
                base:
                    ":":
                        - linux_repo_openstack
                        - linux_repo_cassandra
                        - linux_repo_opencontrail
```

(continues on next page)
- linux_repo_mos
- linux
- opencontrail
- opencontrail_juniper

```yaml
linux.sls:
  linux:
    system:
      enabled: true
      name: opencontrail
    opencontrail_juniper.sls: {}

pillars-from-files:
  linux_repo_mos.sls: tests/pillar/repo_mos8.sls
  linux_repo_cassandra.sls: tests/pillar/repo_cassandra.sls
  linux_repo_openstack.sls: tests/pillar/repo_openstack.sls
  linux_repo_opencontrail.sls: tests/pillar/repo_opencontrail.sls
```

verifier:
  name: inspec
  sudo: true

platforms:
- name: <%= ENV['PLATFORM'] || 'ubuntu-xenial' %>
  driver_config:
    image: <%= ENV['PLATFORM'] || 'trevorj/salty-whales:xenial' %>
    platform: ubuntu

suites:
- name: <%= ENV['SUITE'] || 'single' %>
  provisioner:
    pillars-from-files:
      opencontrail.sls: tests/pillar/<%= ENV['SUITE'] || 'single' %>.sls

- name: cluster
  provisioner:
    pillars-from-files:
      opencontrail.sls: tests/pillar/cluster.sls

- name: analytics
  provisioner:
    pillars-from-files:
      opencontrail.sls: tests/pillar/analytics.sls

- name: control
  provisioner:
    pillars-from-files:
      opencontrail.sls: tests/pillar/control.sls

- name: vendor-juniper
  provisioner:
    vendor_repo:
      type: apt
      url: http://aptly.local/contrail
      key_url: http://aptly.local/public.gpg
      components: main
      distribution: trusty
      pillars-from-files:
Continous Integration with Travis

Salt-formulas uses Travis CI to run smoke and integration tests. To generate `.travis.yml` follow *Generate test structures in formula*.

Sample Configurations

`.travis.yml`

```yaml
sudo: required
services:
  - docker

# PREREQUISITES
install:
  - pip install PyYAML
  - pip install virtualenv
  - |
    test -e Gemfile || cat <<EOF > Gemfile
    source 'https://rubygems.org'
    gem 'rake'
    gem 'test-kitchen'
    gem 'kitchen-docker'
    gem 'kitchen-inspec'
    gem 'inspec'
    gem 'kitchen-salt', :git => 'https://github.com/salt-formulas/kitchen-salt.git
  - bundle install

# BUILD MATRIX
env:
  - PLATFORM=trevorj/salty-whales:trusty
  - PLATFORM=trevorj/salty-whales:xenial
  - PLATFORM=trevorj/salty-whales:xenial-2016.3

# SMOKE TEST
before_script:
  - set -o pipefail
  - make test | tail

# KITCHEN TEST
script:
  - bundle exec kitchen test -t tests/integration
```

2.2. Development Documentation
Common Practices

noservices
At some rare cases execution of given state in the formula is not possible or required. For these cases set grain noservices: True and wrap corresponding code as in the example below:

```powershell
%- if not grains.get('noservices', False) %}
  mysql_database_{{ database_name }}:
    mysql_database.present:
      - name: {{ database_name }}
      - character_set: {{ database.get('encoding', 'utf8') }}
      - connection_user: {{ connection.user }}
      - connection_pass: {{ connection.password }}
      - connection_charset: {{ connection.charset }}
%- endif %}
```

As the mysql database might not be available in the given test environment (travis/docker, etc.). In .kitchen.yml we set grain noservices: True by default.

```
grains:
  noservices: True
```

** formula dependencies **

Formula dependencies might be specified in <formula repo>/metadata.yml

```
name: "galera"
version: "1.0"
source: "https://github.com/salt-formulas/salt-formula-galera"
dependencies:
  - name: mysql
    source: "https://github.com/salt-formulas/salt-formula-mysql"
```

While using test-kitchen formula dependencies must be specified in .kitchen.yml as well. Dependencies may be installed from git, spm or even apt repository.

```
provisioner::
  dependencies:
    - name: mysql
      repo: git
      source: https://github.com/salt-formulas/salt-formula-mysql.git
    - name: linux
      repo: git
      source: https://github.com/salt-formulas/salt-formula-linux.git
```

For convenience kitchen-salt will read metadata.yml of these dependencies and install their dependencies in case you omit them in .kitchen.yml.

** build matrix **

To simplify local CI we ship .kitchen.yml with limited number of platforms. (ie: latest ubuntu as a fallback option if no ENV variable PLATFORM is specified)

However this is later extended on Travis CI while using ENV variables in build matrix.

.travis.yml snippet:
In order to test your model you may use kitchen-salt again. To validate model use:

**Kitchen-salt to validate mode**

With the below approach you may validate or even deploy your model in any platform the *kitchen-test* support.

Expected repository structure:

```
tree -L 3
.
  classes
  -- service
  -- system
  -- cluster
    -- k8s-aio-calico
    -- k8s-aio-contrail
    -- k8s-ha-calico
    -- k8s-ha-calico-cloudprovider
    -- k8s-ha-calico-syndic
```

(continues on next page)
Place this `kitchen.yml` and `verify.sh` to your model repo.

Example `kitchen.yml`:

```yaml
---
driver:
  name: docker
  use_sudo: false
  volume:
    - <%= ENV['PWD'] %>/tmp/kitchen

provisioner:
  name: shell
  script: verify.sh

platforms:
  <% `find classes/cluster -maxdepth 1 -mindepth 1 -type d | tr '_' '-' |sort -u`.  
  -> split().each do |cluster| %>
  - name: <%= cluster %>
    driver_config:
      #image: ubuntu:16.04
      image: tcpcloud/salt-models-testing # With preinstalled dependencies (faster)
      platform: ubuntu
      hostname: cfg01.<%= cluster %>.local
    provision_command:
      - apt-get update
      - apt-get install -y git curl python-pip
      - pip install --upgrade pip
      - git clone https://github.com/salt-formulas/salt-formulas-scripts /srv/salt/
      -> scripts
      - cd /srv/salt/scripts; git pull -r; cd -
      # NOTE: Configure ENV options as needed, example:
      - echo "
        export Bootstrap=1;\n        export cluster_name=<%= cluster %>;\n        export FORMULAS_SOURCE=pkg;\n        export RECLASS_VERSION=master;\n        export RECLASS_IGNORE_CLASS_NOTFOUND=True;\n        export RECLASS_IGNORE_CLASS_REGEXP='service.*';\n        export EXTRA_FORMULAS="";\n        " > /kitchen.env
```

(continues on next page)
Example verify.sh:

```bash
#!/bin/bash

#export HOSTNAME=${`hostname -s`}
#export DOMAIN=${`hostname -d`}

cd /srv/salt/scripts; git pull -r || true; source bootstrap.sh || exit 1

# BOOTSTRAP
if [[ $BOOTSTRAP =~ ^(True|true|1|yes)$ ]]; then
    # workarounds for kitchen
    test ! -e /tmp/kitchen || (mkdir -p /srv/salt/reclass; rsync -avh /tmp/kitchen/ /srv/salt/reclass)
    cd /srv/salt/reclass
    # clone latest system-level if missing
    if [[ ! -e .gitmodules ]] && [[ ! -e classes/system/linux ]]; then
        git submodule update --init --recursive --remote || true
    fi
    source_local_envs
    /srv/salt/scripts/bootstrap.sh &&
    if [[ -e /tmp/kitchen ]]; then sed -i '/BOOTSTRAP=/d' /kitchen.env; fi
fi

# VERIFY
export RECLASS_IGNORE_CLASS_NOTFOUND=False

cd /srv/salt/reclass &&
if [[ -z "$1" ]] ; then
    verify_salt_master &&
    verify_salt_minions
else
    verify_salt_minion "$1"
fi
```

Usage:

```
$ kitchen list
```

<table>
<thead>
<tr>
<th>Instance</th>
<th>Driver</th>
<th>Provisioner</th>
<th>Verifier</th>
<th>Transport</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster-k8s-aio-calico</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-k8s-aio-contrail</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-k8s-ha-calico</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-k8s-ha-calico-cloudprovider</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cluster-k8s-ha-calico-syndic</td>
<td>Docker</td>
<td>Shell</td>
<td>Busser</td>
<td>Ssh</td>
</tr>
<tr>
<td>&lt;Not Created&gt;</td>
<td>&lt;None&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Once all requirements are set, use tests/runtests.py to run all of the tests included in Salt’s test suite. For more information, see –help.

Running the Tests

• Documentation Home
• Project Introduction
• Installation and Operations Manual
• Development Documentation

2.2.3 Maintenance

Chapter 3. Maintenance

Home SaltStack-Formulas Development Documentation

Formula Versioning

• Creating New Release
Current versioning system is date based same as Saltstack versioning using format YYYY-MM-R (year-month-revision) where revision is minor release that increments of 1 starting at 0.

Creating New Release

Releasing is currently not automatic and is up to maintainer of individual formula.

To automate the tasks needed to make a new release, there are unified targets in Makefile that should be present in each formula repository.

See make help for more information, there are release-major and release-minor targets. First one will create new major release by current date. Second will raise revision of current major release.

Example use and output:

```
$ make release-minor
Current version is 2017.2, new version is 2017.2.1
echo "2017.2.1" > VERSION
sed -i 's,version: .*,version: "2017.2.1",g' metadata.yml
[ ! -f debian/changelog ] || dch -v 2017.2.1 -m --force-distribution -D `dpkg-parsechangelog -S Distribution` "New version"
makeshangelog-2017.2.1
make[1]: Entering directory '/home/filip/src/salt-formulas/formulas/letsencrypt'
{echo "=========
Changelog
=========
"; 
 (echo 2017.2.1;git tag) | sort -r | grep -E '^[0-9.]+' | while read i; do \ 
  cur=$i; \ 
  test $i = 2017.2.1 && i=HEAD; \ 
  prev=`(echo 2017.2.1;git tag)|sort|grep -E '^[0-9.]+$'|head -1`;
  echo "Version $cur
=============================
"; 
  git log --pretty=short --invert-grep --grep="Merge pull request" --decorate $prev.
  "$i"; \ 
  echo; \ 
 done) > CHANGELOG.rst
make[1]: Leaving directory '/home/filip/src/salt-formulas/formulas/letsencrypt'
{git add -u; git commit -m "Version 2017.2.1"}
[master 4859e22] Version 2017.2.1
4 files changed, 81 insertions(+), 13 deletions(-)
rewrite CHANGELOG.rst (98%)
git tag -s -m 2017.2.1 2017.2.1
git show ...
$ git push origin master
$ git push origin --tags
```

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home SaltStack-Formulas Development Documentation
Formula Packaging

- Debian
  - Native Packages
  - Quilt Packages
- More Information

This section describes process of building distribution packages for various distributions.

Debian

Debian packaging ecosystems is very diversified, there’s many ways how to build and maintain a package.

We have decided to use git-buildpackage (aka gbp) and support two source formats depending on formula needs: 3.0 (native) and 3.0 (quilt)

Native Packages

Native source format is for applications made especially for Debian, it doesn’t distinguish upstream vs. debian distribution. As it’s the easiest format available, it’s currently used by most of the formulas. The only requirement is to have debian directory in formula’s git repository and building the package is as simple as:

dpkg-buildpackage -uc -us

or building source package and using cowbuilder:

dpkg-buildpackage -S -nc -uc -us
sudo cowbuilder --build "./salt-formula-somthin_*.dsc"

Disadvantages of using native format is that it’s not possible to maintain stable versions and therefore maintain formula package in Debian distribution.

Quilt Packages

Quilt format adds more complexity as it distinguish upstream vs. debian distribution.

Upstream is original unmodified source code, originating from Git repository, Pypy, or some source tarball provided by upstream, etc. Such distribution doesn’t care about debian packaging and doesn’t ship debian directory.

Debian consists of actual debian directory with everything needed similar to native format but as an additional it supports quilt patches. This feature allows package maintainer to maintain patches to specific upstream version separately (eg. to backport new features, fixes, etc.). In this way it’s possible to maintain stable versions of software even if it’s no longer supported upstream.

This format doesn’t solve way how debian packaging is done, whether it’s tracked in a Git repository, SVN, etc. Then git-buildpackage comes into play.

With gbp it’s possible to have separate branch for packaging (eg. debian/unstable) and upstream (usually master) and this is what we are using to maintain packages for some formulas.

Example branches in such formula can be following:
• master
  - formulas itself
• debian/unstable
  - packaging for Debian, uploaded into unstable
  - if it’s needed to patch formula in particular stable release (eg. stretch), according branch can be created, eg. debian/stretch
• debian/trusty
  - packaging for specific Ubuntu version
  - uploaded on Launchpad into ~salt-formulas/ppa
• debian/xenial
  - packaging for specific Ubuntu version
  - uploaded on Launchpad into ~salt-formulas/ppa

This mechanism also utilizes Git tags to mark specific release, eg. debian/1.0-1.

To build package, checkout into debian branch and run:

```
gbp buildpackage --git-ignore-new --git-ignore-branch -S -uc -us
```

**More Information**

Debian packaging is complex topic so it’s good to check some external resources as well:

Configuration Node Setup

• **Configuring the Operating System**
  – Setting up package repository
  – Configuring Secure Shell (SSH) keys

**Configuring the Operating System**

The configuration files will be installed to `/etc/salt` and are named after the respective components, `/etc/salt/master` and `/etc/salt/minion`.

By default the Salt master listens on ports 4505 and 4506 on all interfaces (0.0.0.0). To bind Salt to a specific IP, redefine the “interface” directive in the master configuration file, typically `/etc/salt/master`, as follows:

```
- #interface: 0.0.0.0
+ interface: 10.0.0.1
```

After updating the configuration file, restart the Salt master. For more details about other configurable options. Make sure that mentioned ports are open by your network firewall.

Open salt master config

```
vim /etc/salt/master.d/master.conf
```

And set the content to the following, enabling dev environment and reclass metadata source.

```
file_roots:
  base:
  - /srv/salt/env/dev
  - /srv/salt/env/base

pillar_opts: False

reclass: &reclass
  storage_type: yaml_fs
  inventory_base_uri: /srv/salt/reclass

ext_pillar:
  - reclass: &reclass

master_tops:
  - reclass: &reclass
```

And set the content to the following to setup reclass as salt-master metadata source.

```
vim /etc/reclass/reclass-config.yml
```

```
stORAGE_TYPE: yaml_fs
pretty_print: True
output: yaml
inventory_base_uri: /srv/salt/reclass
```

Configure the master service
# Ubuntu

```
service salt-master restart
```

# Redhat

```
systemctl enable salt-master.service
systemctl start salt-master
```

See the master configuration reference for more details about other configurable options.

## Setting up package repository

Use `curl` to install your distribution’s stable packages. Examine the downloaded file `install_salt.sh` to ensure that it contains what you expect (bash script). You need to perform this step even for salt-master installation as it adds official saltstack package management PPA repository.

```
apt-get install vim curl git-core
curl -L https://bootstrap.saltstack.com -o install_salt.sh
sudo sh install_salt.sh
```

Install the Salt master from the apt repository with the apt-get command after you installed salt-minion.

```
sudo apt-get install salt-minion salt-master reclass
```

**Note:** Installation is tested on Ubuntu Linux 14.04/16.04, but should work on any distribution with python 2.7 installed.

You should keep Salt components at current stable version.

## Configuring Secure Shell (SSH) keys

Generate SSH key file for accessing your reclass metadata and development formulas.

```
mkdir /root/.ssh
ssh-keygen -b 4096 -t rsa -f /root/.ssh/id_rsa -q -N ""
chmod 400 /root/.ssh/id_rsa
```

Create SaltStack environment file root, we will use `dev` environment.

```
mkdir /srv/salt/env/dev -p
```

Get the reclass metadata definition from the git server.

```
git clone git@github.com:tcpcloud/workshop-salt-model.git /srv/salt/reclass
```

Get the core formulas from git repository server needed to setup the rest.

```
git clone git@github.com:tcpcloud/salt-formula-linux.git /srv/salt/env/dev/linux -b develop
git clone git@github.com:tcpcloud/salt-formula-salt.git /srv/salt/env/dev/salt -b develop
git clone git@github.com:tcpcloud/salt-formula-openssh.git /srv/salt/env/dev/openssh -b develop
git clone git@github.com:tcpcloud/salt-formula-git.git /srv/salt/env/dev/git -b develop
```
On most distributions, you can set up a Salt Minion with the Salt Bootstrap.

**Note:** In every two-step example, you would be well-served to examine the downloaded file and examine it to ensure that it does what you expect.

Using `curl` to install latest git:

```
curl -L https://bootstrap.saltstack.com -o install_salt.sh
sudo sh install_salt.sh git develop
```

Using `wget` to install your distribution’s stable packages:

```
wget -O install_salt.sh https://bootstrap.saltstack.com
sudo sh install_salt.sh
```

Install a specific version from git using `wget`:

```
wget -O install_salt.sh https://bootstrap.saltstack.com
sudo sh install_salt.sh -P git v2015.5
```

On the above example we added `-P` which will allow PIP packages to be installed if required but it’s no a necessary flag for git based bootstraps.

**Basic minion Configuration**

Salt configuration is very simple. The only requirement for setting up a minion is to set the location of the master in the minion configuration file.

The configuration files will be installed to `/etc/salt` and are named after the respective components, `/etc/salt/master`, and `/etc/salt/minion`. 
### Setting Salt Master host

Although there are many Salt Minion configuration options, configuring a Salt Minion is very simple. By default a Salt Minion will try to connect to the DNS name “salt”; if the Minion is able to resolve that name correctly, no configuration is needed.

If the DNS name “salt” does not resolve to point to the correct location of the Master, redefine the “master” directive in the minion configuration file, typically `/etc/salt/minion`, as follows:

```bash
- #master: salt
+ master: 10.0.0.1
```

### Setting Salt minion ID

Then explicitly declare the ID for this minion to use. Since Salt uses detached IDs it is possible to run multiple minions on the same machine but with different IDs.

```bash
id: foo.bar.com
```

After updating the configuration files, restart the Salt minion.

```bash
# Ubuntu
service salt-minion restart

# Redhat
systemctl enable salt-minion.service
systemctl start salt-minion
```

See the minion configuration reference for more details about other configurable options.

---

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Home Installation and Operations Manual

### Install Infrastructure Services

- **Support infrastructure deployment**

First execute basic states on all nodes to ensure Salt minion, system and OpenSSH are set up.

```bash
salt '*' state.sls linux,salt,openssh,ntp
```

---

2.3. Installation and Operations Manual 529
Support infrastructure deployment

Metering node is deployed by running highstate:

```bash
salt 'mtr*' state.highstate
```

On monitoring node, git needs to be setup first:

```bash
salt 'mon*' state.sls git
salt 'mon*' state.highstate
```

• Documentation Home
• Project Introduction
• Installation and Operations Manual
• Development Documentation

Validate Configuration Node

Now it’s time to validate your configuration infrastructure.

Check validity of reclass data for entire infrastructure:

```bash
reclass-salt --top
```

It will return service catalog of entire infrastructure.

Get reclass data for specific node:

```bash
reclass-salt --pillar ctl01.workshop.cloudlab.cz
```

Verify that all salt minions are accepted at master:

```bash
root@cfg01:~# salt-key
Accepted Keys:
cfg01.workshop.cloudlab.cz
mtr01.workshop.cloudlab.cz
Denied Keys:
Unaccepted Keys:
Rejected Keys:
```

Verify that all Salt minions are responding:

```bash
root@cfg01:~# salt '*workshop.cloudlab.cz' test.ping
cfg01.workshop.cloudlab.cz:
  True
mtr01.workshop.cloudlab.cz:
  True
web01.workshop.cloudlab.cz:
  True
cmp02.workshop.cloudlab.cz:
  True
```

(continues on next page)
Get IP addresses of minions:

```
root@cfg01:~# salt "*.workshop.cloudlab.cz" grains.get ipv4
```

Show top states (installed services) for all nodes in the infrastructure.

```
root@cfg01:~# salt '*' state.show_top
[INFO ] Loading fresh modules for state activity
nodeXXX:
   --------
   base:
   - git
   - linux
   - ntp
   - salt
   - collectd
   - openssh
   - reclass
```
Initial Environment Configuration

- **Linux system setup**
  - Basic linux box
  - Linux with defined users (optionaly with password)
  - Linux package installation
  - Linux cron job
  - Linux security limits
  - Enable autologin on tty1
- **Linux Kernel setup**
- **Linux repositories setup**
- **Linux prompt setup**
- **Linux network setup**
  - Linux interface/route setup
  - Linux network bridges
  - Other network related configuration
- **Linux storage setup**
- **OpenSSH client**
- **OpenSSH server**
- **Salt minion configuration**
- **NTP client**

### Linux system setup

#### Basic linux box

```yaml
linux:
  system:
    enabled: true
    name: 'node1'
    domain: 'domain.com'
    cluster: 'system'
    environment: prod
    timezone: 'Europe/Prague'
    utc: true
```

#### Linux with defined users (optionaly with password)
Linux:
  system:
    ...
  user:
    jdoe:
      name: 'jdoe'
      enabled: true
      shell: /bin/bash
      full_name: 'Jonh Doe'
      home: '/home/jdoe'
      email: 'jonh@doe.com'
    jsmith:
      name: 'jsmith'
      enabled: true
      full_name: 'Password'
      home: '/home/jsmith'
      password: userpassword

Linux package installation

Install latest version

linux:
  system:
    ...
  package:
    package-name:
      version: latest

Linux package with specified version and repository

linux:
  system:
    ...
  package:
    package-name:
      version: 2132.323
      repo: 'custom-repo'
      hold: true

Linux package with specified version and repository - disable GPG check

linux:
  system:
    ...
  package:
    package-name:
      version: 2132.323
      repo: 'custom-repo'
      verify: false
Linux cron job

```
linux:
  system:
    ...
  job:
    cmd1:
      command: '/cmd/to/run'
      enabled: true
      user: 'root'
      hour: 2
      minute: 0
```

Linux security limits

Limit sensu user maximum memory usage to 1GB

```
linux:
  system:
    ...
  limit:
    sensu:
      enabled: true
      domain: sensu
      limits:
        - type: hard
          item: as
          value: 1000000
```

Enable autologin on tty1

```
linux:
  system:
    console:
      tty1:
        autologin: root
```

Linux Kernel setup

Install always up to date LTS kernel and headers from Ubuntu trusty

```
linux:
  system:
    kernel:
      type: generic
      lts: trusty
      headers: true
```

Install specific kernel version and ensure all other kernel packages are not present. Also install extra modules and headers for this kernel
Linux repositories setup

RedHat based Linux with additional OpenStack repo

```yaml
linux:
  system:
    kernel:
      type: generic
      extra: true
      headers: true
      version: 4.2.0-22

repo:
  rdo-icehouse:
    enabled: true
    source: 'https://repos.fedorapeople.org/repos/openstack/openstack-kilo/el7/
    pgpcheck: 0
```

Ensure system repository to use czech Debian mirror (default: true) Also pin it’s packages with priority 900

```yaml
linux:
  system:
    repo:
      debian:
        default: true
        source: "deb http://ftp.cz.debian.org/debian/ jessie main contrib non-free"
        key_url: "http://dummy.com/public.gpg"
        pin:
          - pin: 'origin "ftp.cz.debian.org"
            priority: 900
            package: '*/'
```

rc.local example

```bash
#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.
exit 0
```
Linux prompt setup

Setting prompt is implemented by creating /etc/profile.d/prompt.sh. Every user can have different prompt

```
linux:
  system:
    prompt:
      root: \n\n\x1b[0;37m\D{%y/%m/%d %H:%M:%S} $(hostname -f)\x1b[0m\n\x1b[1;31m\x1b[0m\u@\h:\w
      default: \n\D{%y/%m/%d %H:%M:%S} $(hostname -f)\n\x1b[0m\u@\h:\w
```

Linux network setup

Linux interface/route setup

Linux with default static network interfaces, default gateway interface and DNS servers

```
linux:
  network:
    enabled: true
    interface:
      eth0:
        enabled: true
        type: eth
        address: 192.168.0.102
        netmask: 255.255.255.0
        gateway: 192.168.0.1
        name_servers:
          - 8.8.8.8
          - 8.8.4.4
        mtu: 1500
```

Linux with bonded interfaces and disabled NetworkManager

```
linux:
  network:
    enabled: true
    interface:
      eth0:
        type: eth
      ...
      eth1:
        type: eth
      ...
    bond0:
      enabled: true
      type: bond
      address: 192.168.0.102
      netmask: 255.255.255.0
      mtu: 1500
      use_in:
        - interface: ${linux:interface:eth0}
        - interface: ${linux:interface:eth0}
    network_manager:
      disable: true
```
Linux with vlan interface_params

```yaml
linux:
  network:
    enabled: true
    interface:
      vlan69:
        type: vlan
        use_interfaces:
          - interface: ${linux:interface:bond0}
```

Linux networks with routes defined

```yaml
linux:
  network:
    enabled: true
    gateway: 10.0.0.1
    default_interface: eth0
    interface:
      eth0:
        type: eth
        route:
          default:
            address: 192.168.0.123
            netmask: 255.255.255.0
            gateway: 192.168.0.1
```

**Linux network bridges**

Native linux bridges

```yaml
linux:
  network:
    interface:
      eth1:
        enabled: true
        type: eth
        proto: manual
        up_cmds:
          - ip address add 0/0 dev $IFACE
          - ip link set $IFACE up
        down_cmds:
          - ip link set $IFACE down
      br-ex:
        enabled: true
        type: bridge
        address: ${linux:network:host:public_local:address}
        netmask: 255.255.255.0
        use_interfaces:
          - eth1
```

OpenVSwitch bridges

```yaml
linux:
  network:
    bridge: openvswitch
```

(continues on next page)
interface:
    eth1:
      enabled: true
      type: eth
      proto: manual
      up_cmds:
      - ip address add 0/0 dev $IFACE
      - ip link set $IFACE up
      down_cmds:
      - ip link set $IFACE down
    br-ex:
      enabled: true
      type: bridge
      address: ${linux:network:host:public_local:address}
      netmask: 255.255.255.0
      use_interfaces:
      - eth1

Other network related configuration

Linux with network manager

```
linux:
  network:
    enabled: true
    network_manager: true
```

/etc/hosts configuration

```
linux:
  network:
    ...
    host:
      node1:
        address: 192.168.10.200
        names:
        - node2.domain.com
        - service2.domain.com
      node2:
        address: 192.168.10.201
        names:
        - node2.domain.com
        - service2.domain.com
```

/etc/resolv.conf configuration

```
linux:
  network:
    resolv:
      dns:
      - 8.8.4.4
      - 8.8.8.8
      domain: my.example.com
      search:
```
Linux storage setup

Linux with mounted Samba

```yaml
linux:
  storage:
    enabled: true
    mount:
      samba1:
        path: /media/myuser/public/
        device: //192.168.0.1/storage
        file_system: cifs
        options: guest,uid=myuser,iocharset=utf8,file_mode=0777,dir_mode=0777,noperm
```

Linux with file swap

```yaml
linux:
  storage:
    enabled: true
    swap:
      file:
        enabled: true
        engine: file
        device: /swapfile
        size: 1024
```

LVM group vg1 with one device and data volume mounted into /mnt/data

```yaml
linux:
  storage:
    mount:
      data:
        device: /dev/vg1/data
        file_system: ext4
        path: /mnt/data
  lvm:
    vg1:
      enabled: true
      devices:
        - /dev/sdb
      volume:
        data:
          size: 40G
          mount: $(linux:storage:mount:data)
```

OpenSSH client

OpenSSH client with shared private key
OpenSSH client with individual private key and known host

```yaml
openssh:
  client:
    enabled: true
    user:
      root:
        enabled: true
        private_key: ${private_keys:vaio.newt.cz}
        user: ${linux:system:user:root}

OpenSSH server with configuration parameters

```yaml
openssh:
  server:
    enabled: true
    permit_root_login: true
    public_key_auth: true
    password_auth: true
    host_auth: true
    banner: Welcome to server!

OpenSSH server with auth keys for users

```yaml
openssh:
  server:
    enabled: true
    ...
  user:
    user1:
      enabled: true
      user: ${linux:system:user:user1}
      public_keys:
        - ${public_keys:user1}
    root:
      enabled: true
      user: ${linux:system:user:root}
      public_keys:
        - ${public_keys:user1}

OpenSSH server for use with FreeIPA

```yaml
SaltStack-Formulas Documentation, Release master

```
openssh:
    server:
        enabled: true
        public_key_auth: true
        authorized_keys_command:
            command: /usr/bin/sss_ssh_authorizedkeys
            user: nobody
```

Salt minion configuration

Simple Salt minion

```
salt:
    minion:
        enabled: true
        master:
            host: master.domain.com
```

Multi-master Salt minion

```
salt:
    minion:
        enabled: true
        masters:
            - host: master1.domain.com
            - host: master2.domain.com
```

Salt minion with salt mine options

```
salt:
    minion:
        enabled: true
        master:
        mine:
            interval: 60
            module:
                grains.items: []
                network.interfaces: []
```

Salt minion with graphing dependencies

```
salt:
    minion:
        enabled: true
        graph_states: true
        master:
```

NTP client

```
ntp:
    client:
        enabled: true
```

(continues on next page)
Monitoring, Metering and Logging

The overall health of the systems is measured continuously. The metering system collects metrics from the systems and store them in time-series database for further evaluation and analysis. The log collecting system collects logs from all systems, transforms them to unified form and stores them for analysis. The monitoring system checks for functionality of separate systems and raises events in case of threshold breach. The monitoring systems may query log and time-series databases for accident patterns and raise an event if anomaly is detected.

The difference between monitoring and metering systems

Monitoring is generally used to check for functionality on the overall system and to figure out if the hardware for the overall installation and usage needs to be scaled up. With monitoring, we also do not care that much if we have lost some samples in between. Metering is required for information gathering on usage as a base for resource utilisation. Many monitoring checks are simple meter checks with threshold definitions.
Event Monitoring

The overall health of the systems is measured continuously. The metering system collects metrics from the systems and store them in time-series database for further evaluation and analysis. The log collecting system collects logs from all systems, transforms them to unified form and stores them for analysis. The monitoring system checks for functionality of separate systems and raises events in case of threshold breach. The monitoring systems may query log and time-series databases for accident patterns and raise an event if anomaly is detected.

The difference between monitoring and metering systems

Monitoring is generally used to check for functionality on the overall system and to figure out if the hardware for the overall installation and usage needs to be scaled up. With monitoring, we also do not care that much if we have lost some samples in between. Metering is required for information gathering on usage as a base for resource utilisation. Many monitoring checks are simple meter checks with threshold definitions.

Monitoring Service (Sensu)

Sensu is often described as the “monitoring router”. Essentially, Sensu takes the results of “check” scripts run across many systems, and if certain conditions are met, passes their information to one or more “handlers”. Checks are used, for example, to determine if a service like Apache is up or down. Checks can also be used to collect data, such as MySQL query statistics or Rails application metrics. Handlers take actions, using result information, such as sending an email, messaging a chat room, or adding a data point to a graph. There are several types of handlers, but the most common and most powerful is “pipe”, a script that receives data via standard input. Check and handler scripts can be written in any language, and the community repository continues to grow!

Sensu properties:

- Written in Ruby, using EventMachine
- Great test coverage with continuous integration via Travis CI
- Can use existing Nagios plugins
- Configuration all in JSON
- Has a message-oriented architecture, using RabbitMQ and JSON payloads
- Packages are “omnibus”, for consistency, isolation, and low-friction deployment

Sensu embraces modern infrastructure design, works elegantly with configuration management tools, and is built for the cloud.
Collecting Telemetry Data

• Collectd/Graphite
  – Graphite Metrics Functions

Gathering metrics and other values. There are three basic types of meters that are stored in the time-series database.

Cumulative
Increasing over time (network or disk usage counters)

Gauge
Discrete items (number of connected users) and fluctuating values (system load)

Delta
Values changing over time (bandwidth)

Collectd/Graphite

Collectd gathers statistics about the system it is running on and stores this information. Those statistics can then be used to find current performance bottlenecks (i.e. performance analysis) and predict future system load (i.e. capacity planning). It’s written in C for performance and portability, allowing it to run on systems without scripting language or cron daemon, such as embedded systems. At the same time it includes optimizations and features to handle hundreds of thousands of data sets. It comes with over 90 plugins which range from standard cases to very specialized and advanced topics. It provides powerful networking features and is extensible in numerous ways.

Graphite is an enterprise-scale monitoring tool that runs well on cheap hardware. It was originally designed and written by Chris Davis at Orbitz in 2006 as side project that ultimately grew to be a foundational monitoring tool. In 2008, Orbitz allowed Graphite to be released under the open source Apache 2.0 license. Since then Chris has continued to work on Graphite and has deployed it at other companies including Sears, where it serves as a pillar of the e-commerce monitoring system. Today many large companies use it.

What Graphite does not do is collect data for you, however there are some tools out there that know how to send data to graphite. Even though it often requires a little code, sending data to Graphite is very simple.

Graphite consists of 3 software components:
• carbon - a Twisted daemon that listens for time-series data
• whisper - a simple database library for storing time-series data (similar in design to RRD)
• graphite - A Django webapp that renders graphs on-demand using Cairo

Graphite Metrics Functions

The metrics can be adjusted by applying functions on them within the Graphite composer. Aside the ability to store time-series data Graphite has a lot of additional functions that can be used to alter time-series data to more appropriate form, if we want to get the delta from the cumulative metrics or ad vice versa.

integral(seriesList)

This will show the sum over time, sort of like a continuous addition function. Useful for finding totals or trends in metrics that are collected per minute.

Example:
\texttt{\&target=integral(company.sales.perMinute)}

This would start at zero on the left side of the graph, adding the sales each minute, and show the total sales for the time period selected at the right side, (time now, or the time specified by `\&until=`).

\textbf{derivative(seriesList)}

This is the opposite of the integral function. This is useful for taking a running total metric and calculating the delta between subsequent data points.

This function does not normalize for periods of time, as a true derivative would. Instead see the \texttt{perSecond()} function to calculate a rate of change over time.

Example:

\texttt{\&target=derivative(company.server.application01.ifconfig.TXPackets)}

\textbf{sumSeries(*seriesLists)}

Short form: \texttt{sum()}

This will add metrics together and return the sum at each datapoint. (See integral for a sum over time)

Example:

\texttt{\&target=sum(company.server.application*.requestsHandled)}

This would show the sum of all requests handled per minute (provided requestsHandled are collected once a minute). If metrics with different retention rates are combined, the coarsest metric is graphed, and the sum of the other metrics is averaged for the metrics with finer retention rates.

Read more about functions at \url{http://graphite.readthedocs.org/en/latest/functions.html#module-graphite.render.functions}
Heka

Heka is an open source stream processing software system developed by Mozilla. Heka is a “Swiss Army Knife” type tool for data processing, useful for a wide variety of different tasks, such as:

- Loading and parsing log files from a file system.
- Accepting statsd type metrics data for aggregation and forwarding to upstream time series data stores such as graphite or InfluxDB.
- Launching external processes to gather operational data from the local system.
- Performing real time analysis, graphing, and anomaly detection on any data flowing through the Heka pipeline.
- Shipping data from one location to another via the use of an external transport (such as AMQP) or directly (via TCP).
- Delivering processed data to one or more persistent data stores.

ElasticSearch

Elasticsearch is a search server based on Lucene. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schema-free JSON documents.

Kibana Dashboard

Kibana is an open source data visualization plugin for Elasticsearch. It provides visualization capabilities on top of the content indexed on an Elasticsearch cluster. Users can create bar, line and scatter plots, or pie charts and maps on top of large volumes of data.

2.3.4 Use cases

Chapter 4. Use Cases

Home Installation and Operations Manual
Use Case: Kubernetes

TODO

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation

Use Case: Openstack

TODO

- Documentation Home
- Project Introduction
- Installation and Operations Manual
- Development Documentation
CHAPTER 3

Indices and tables

• genindex
• modindex
• search