Configuration files in RDO

do's, dont's, myths, gotchas, and rfc

Ihar Hrachyshka
@Red Hat

Basics: INI files

- Known since 80s, but no formal standard
- Lots of libraries and tools to use
- Everyone is special

Basics: INI files: /etc/my.cnf

```
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock

[mysqld_safe]
log-error=/var/log/mariadb/mariadb.log
pid-file=/var/run/mariadb/mariadb.pid

!includedir /etc/my.cnf.d
```

oslo.config: intro

- reads INI files (*.ini, *.conf, *.<anything>)
- used by all OpenStack services
- provides global ConfigOpts object

oslo.config: intro: neutron/agent/dhcp_agent.py

```
from oslo_config import cfg


def main():
    register_options(cfg.CONF)
    common_config.init(sys.argv[1:])
    config.setup_logging()
    server = neutron_service.Service.create(...)
    service.launch(cfg.CONF, server).wait()
```

oslo.config: intro: neutron/agent/dhcp_agent.py

```
from oslo_config import cfg

def main():
    register_options(cfg.CONF)
    common_config.init(sys.argv[1:])
    config.setup_logging()
    server = neutron_service.Service.create(...)
    service.launch(cfg.CONF, server).wait()
```

oslo.config: intro: neutron/agent/dhcp_agent.py

```
from oslo_config import cfg

def main():
    register_options(cfg.CONF)
    common_config.init(sys.argv[1:])
    config.setup_logging()
    server = neutron_service.Service.create(...)
    service.launch(cfg.CONF, server).wait()
```

oslo.config: intro: neutron/agent/dhcp_agent.py

```
from oslo_config import cfg


def main():
    register_options(cfg.CONF)
    common_config.init(sys.argv[1:])
    config.setup_logging()
    server = neutron_service.Service.create(...)
    service.launch(cfg.CONF, server).wait()
```

oslo.config: intro: neutron/common/config.py

```
def init(args, **kwargs):
    cfg.CONF(args=args, project='neutron',
             version=<...>,
             **kwargs)

    n_rpc.init(cfg.CONF)
```

oslo.config: intro: neutron/common/config.py

```
def init(args, **kwargs):
    cfg.CONF(args=args, project='neutron',
             version=<...>,
             **kwargs)

    n_rpc.init(cfg.CONF)
```

oslo.config: intro: neutron/common/config.py

```
def init(args, **kwargs):
    cfg.CONF(args=args, project='neutron',
             version=<...>,
             **kwargs)

    n_rpc.init(cfg.CONF)
```

oslo.config: intro: neutron/common/config.py

```
def init(args, **kwargs):
    cfg.CONF(args=args, project='neutron',
             version=<...>,
             **kwargs)

    n_rpc.init(cfg.CONF)
```

oslo.config: sys.argv

- --config-file and --config-dir
- multiple options allowed
- order is guaranteed: last beats first

oslo.config: sys.argv

```
$service \
    --config-dir /usr/share/$service \
    --config-dir /etc/$service \
    --config-file /opt/etc/$service.conf \
    --config-dir /opt/etc/$service
```

oslo.config: autodiscovery

```
class ConfigOpts(collections.Mapping):

    def _pre_setup(self, project, prog, version, usage,
                   default_config_files):
        <...>
        if default_config_files is None:
            default_config_files = find_config_files(project, prog)
        <...>
```

oslo.config: autodiscovery

```
def find_config_files(project=None, prog=None, extension='.conf'):
    <...>
    cfg_dirs = _get_config_dirs(project)
    <...>
    config_files.append(_search_dirs(cfg_dirs, prog, extension))
```

oslo.config: autodiscovery

```
def _get_config_dirs(project=None):

    cfg_dirs = [
        _fixpath(os.path.join('~', '.' + project)) if project else None,
        _fixpath('~'),
        os.path.join('/etc', project) if project else None,
        '/etc'
    ]
    return list(moves.filter(bool, cfg_dirs))
```

oslo.config: autodiscovery: RHEL-OSP

```
Subject: [PATCH] add /usr/share/$project/*-dist.conf to the default config set
--- a/oslo_config/cfg.py
+++ b/oslo_config/cfg.py
     config_files = []
     if project:
+        config_files.append(_search_dirs(['/usr/share/%s/' % project],
+                                         project, '-dist%s' % extension))
+        config_files.append(_search_dirs(['/usr/share/%s/' % project],
+                                         prog, '-dist%s' % extension))
         config_files.append(_search_dirs(cfg_dirs, project, extension))
     config_files.append(_search_dirs(cfg_dirs, prog, extension))
```

oslo.config: autodiscovery: $service-dist.conf

- distribution specific defaults
- RDO specific
- diverge from upstream defaults

oslo.config: autodiscovery: neutron-dist.conf

```
[DEFAULT]
verbose = True
lock_path = $state_path/lock
notification_driver = neutron.openstack.common.notifier.rpc_notifier
allow_overlapping_ips = True
use_stderr = False
api_paste_config = /usr/share/neutron/api-paste.ini

[agent]
root_helper = sudo neutron-rootwrap /etc/neutron/rootwrap.conf
root_helper_daemon = sudo neutron-rootwrap-daemon /etc/neutron/rootwrap.conf
```

...speaking of defaults

- upstream defaults
- distribution (package) defaults
- ‘your deployment tool of choice’ defaults

oslo.config: not just a .ini parsing library

- rich type library
- deprecation management
- config files generator

oslo.config: types

- BoolOpt, StrOpt
- MultiStrOpt
- PortOpt, IPOpt
- <your-type-of-interest>

oslo.config: types: MultiStrOpt

```
[service_providers]
service_provider=LOADBALANCER:Haproxy:...
service_provider=VPN:Libreswan:...
```

oslo.config: deprecation management

```
cfg.StrOpt(‘user-name’,
        help='...',
        deprecated_name=’username’,
        deprecated_group='keystone_authtoken'),
```

```
2016-01-20 18:10:21.688 WARNING oslo_config.cfg [req-116e5f9e-1994-4f98-
b914-86853a874730 None None] Option "username" from group
"keystone_authtoken" is deprecated. Use option "user-name" from group
"keystone_authtoken".
```

oslo.config: config files generator

- historically, files were stored under git
- generator allows for single source of truth
- nova, neutron, glance, … switched

oslo.config: config files generator (per file)

```
[DEFAULT]
output_file = etc/dhcp_agent.ini.sample
wrap_width = 79

namespace = neutron.base.agent
namespace = neutron.dhcp.agent
namespace = oslo.log
```

oslo.config: config files generator (setup.cfg)

```
oslo.config.opts =
    neutron = neutron.opts:list_opts
    neutron.agent = neutron.opts:list_agent_opts
    neutron.base.agent = neutron.opts:list_base_agent_opts
    neutron.db = neutron.opts:list_db_opts
    neutron.dhcp.agent = neutron.opts:list_dhcp_agent_opts
...
```

oslo.config: config files generator (contents)

```
def list_dhcp_agent_opts():
    return [
        ('DEFAULT',
         itertools.chain(
             neutron.agent.dhcp.config.DHCP_AGENT_OPTS,
             neutron.agent.dhcp.config.DHCP_OPTS,
             neutron.agent.dhcp.config.DNSMASQ_OPTS)
         )
    ]
```

oslo.config: future

- configuration databases
- autodoc for configuration options
- config option reloading (in cooperation with oslo.service)

myths

“packaged config files are somehow special”

myths

“packaged config files are somehow special”

“config files should have $service names”

myths

“packaged config files are somehow special”

“config files should have $service names”

“a single config file is always enough”

multiple config files: use case

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

multiple config files: use case

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

multiple config files: use case

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

multiple config files: use case

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

multiple config files: use case

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

multiple config files: use case

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

gotchas: don’t reimplement oslo.config, ever

- Neutron did it before for service_providers
- loaded the option from ‘special’ files
- Magic!

gotchas: don’t reimplement oslo.config, ever

Why not?

- hard for developers
- adds complexity to operations
- there is simply no good reason

gotchas: testing: autodiscovery

```
class BaseTestCase(DietTestCase):
    @staticmethod
    def config_parse(conf=None, args=None):
        if args is None:
            args = []
        args += ['--config-file', etcdir('neutron.conf')]
        if conf is None:
            config.init(args=args)
        else:
            conf(args)
```

gotchas: testing: autodiscovery

```
ROOTDIR = os.path.dirname(__file__)
ETCDIR = os.path.join(ROOTDIR, 'etc')


def etcdir(*p):
    return os.path.join(ETCDIR, *p)
```

^ returns file path from neutron/tests/etc

gotchas: testing: autodiscovery

```
[testenv:pep8]
basepython = python2.7
deps =
  {[testenv]deps}
commands=
...
  neutron-db-manage --config-file neutron/tests/etc/neutron.
conf check_migration
```

gotchas: testing: autodiscovery

```
neutron/tests/base.py:
        self.useFixture(fixtures.MonkeyPatch(
            'oslo_config.cfg.find_config_files',
            lambda project=None, prog=None,
            extension=None: []))
```

gotchas: testing: autodiscovery

oslo_config.fixture.Config.set_config_files

conf.d: rationale

In Neutron:

- plugin.ini
- *aas repo split, 3rd parties
- we don't have a predefined list of files to load

conf.d: solution

- per-service and common conf.d dirs
- document and maintain the order

conf.d: solution

```
ExecStart=/usr/bin/neutron-server \
    --config-file /usr/share/neutron/neutron-dist.conf \
    --config-dir /usr/share/neutron/server \
    --config-file /etc/neutron/neutron.conf \
    --config-file /etc/neutron/plugin.ini \
    --config-dir /etc/neutron/conf.d/common \
    --config-dir /etc/neutron/conf.d/neutron-server \
    --log-file /var/log/neutron/server.log
```

conf.d: solution

```
# conffile=/etc/neutron/conf.d/neutron-server/debug.conf
# echo “[DEFAULT]\ndebug=True” >> $conffile
# systemctl restart neutron-server
```

...now collect logs and…

```
# rm $conffile
# systemctl restart neutron-server
```

conf.d: benefits

- Undefined list of config files to load
- Not managed as .rpmsave/.rpmnew
- Easier to grasp local specific settings

conf.d: drawbacks

- It’s only RDO
- It’s only Neutron and Octavia

conf.d: adoption

- <your project>
- ansible books, puppet modules?
- RHEL-OSP: 3rd party integration

Questions?