INFN

Working with OpenStack

«OpenStack Dashboard & OpenStackClient»

«OpenStack Administration 101» , 30 Nov. – 3 Dec. 2021

Doina Cristina Duma & Alessandro Costantini

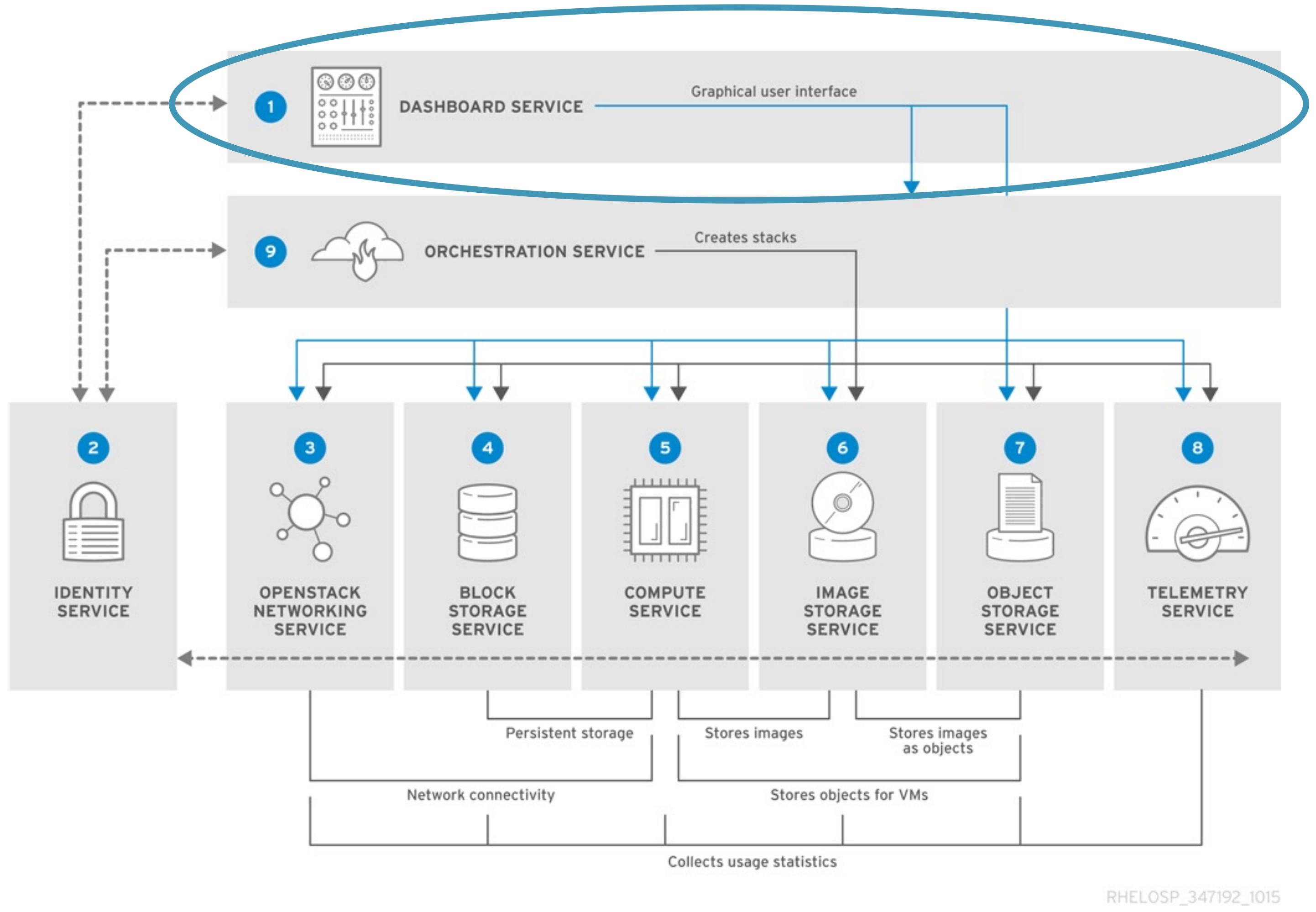
openstack CLOUD SOFTWARE

29/11/2021

Overview

- OpenStack Dashboard – aka Horizon
  - What it is & what it is used for
  - Main tabs/modules
- OpenStack Client
- Hands-on

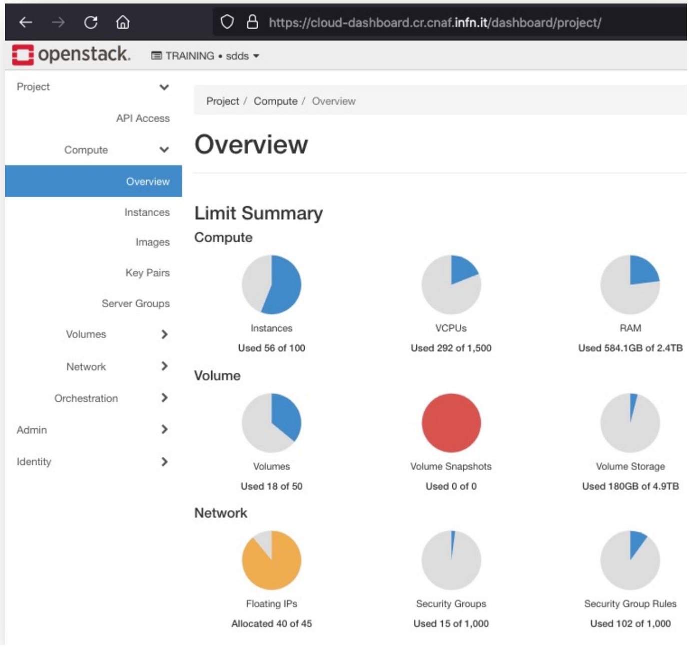
High-level Overview of Core Services



Horizon

- The **Horizon** project, also known as the **OpenStack Dashboard**, provides a **web based user interface** to an OpenStack cloud for both **cloud operators/administrators** and those who access and use the cloud's resources.
  - designed to be **easily skin-able** so that OpenStack software vendors and possibly even the cloud operators can change the look of the dashboard for their users.
  - Docs
  - Latest code source release
    - Django-based application that provides access to OpenStack services
    - Typically deployed as an Apache WSGI application
    - Leverages well known existing technologies – Bootstrap, jQuery, AngularJS
  - First appeared in OpenStack 'Essex' release

Depends on


https://cloud-dashboard.cr.cnaf.infn.it/dashboard/project/

openstack. TRAINING • sdds

Project / Compute / Overview

Overview

Limit Summary

Compute

Instances Used 56 of 100

VCPUs Used 292 of 1,500

RAM Used 584.1GB of 2.4TB

Volume

Volumes Used 18 of 50

Volume Snapshots Used 0 of 0

Volume Storage Used 180GB of 4.9TB

Network

Floating IPs Allocated 40 of 45

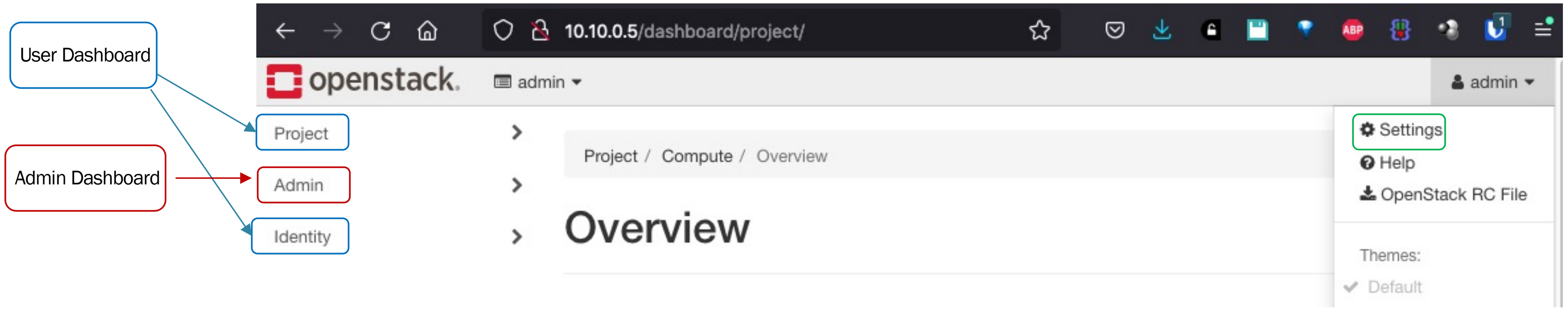
Security Groups Used 15 of 1,000

Security Group Rules Used 102 of 1,000

Horizon Basics

"Think simple" => make it easy => features

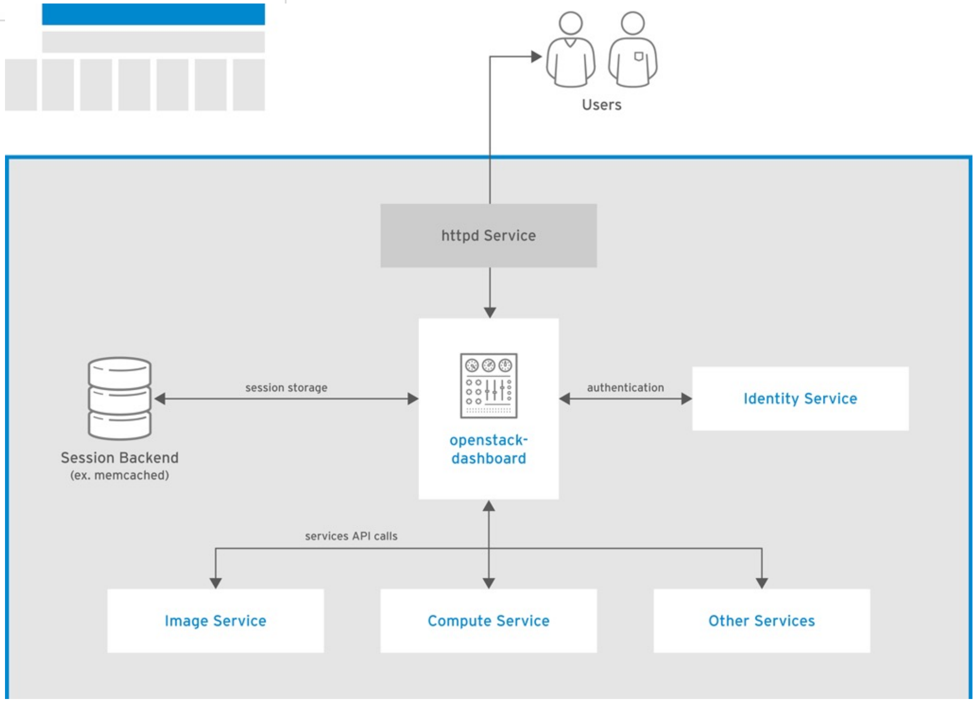
- Core Support: Out-of-the-box support for all core OpenStack projects. It ships with:
  - three central dashboards, a **"User Dashboard"**, a **"System Dashboard"**, and a **"Settings"** dashboard. Between these three they cover the core OpenStack applications and deliver on Core Support.
  - a set of API abstractions for the core OpenStack projects in order to provide a consistent, stable set of reusable methods for developers
- Extensible: Anyone can add a new component as a "first-class citizen".
  - based around the Dashboard class that provides a consistent API and set of capabilities for core OpenStack dashboard apps and also third-party apps
- Manageable: The core codebase should be simple and easy-to-navigate.
- Consistent: Visual and interaction paradigms are maintained throughout apps.
  - providing the necessary core classes to build from, as well as a solid set of reusable templates
- Stable: A reliable API with an emphasis on backwards-compatibility.
- Usable: Providing an *awesome* interface that people *want* to use.



Horizon Components & Architecture

Component	Description
openstack-dashboard	Django Web application that provides access to the dashboard from any Web browser.
Apache HTTP server (httpd service)	Hosts the application.

- The OpenStack Identity service authenticates and authorizes users
- The session backend provides database services
- The httpd service hosts the Web application and all other OpenStack services for API calls



Installation & Configuration guide (hints)

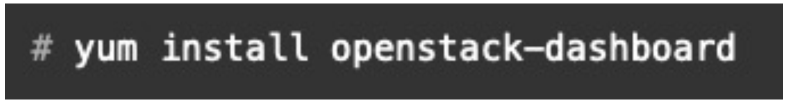
The only core service required by the dashboard is the Identity service

Requirements:

- Python (3.6/3.7)
- Django (2.2)
- Keystone endpoint – if available, it is automatically detected.
- Other services via plugins, available in Plugin Registry.

Installation:

- Note1: a proper installation, configuration, and operation of the Identity service using the Apache HTTP server and Memcached service is needed
- Note2: Default configuration files **vary with distribution**
- **Steps:**
    1. Install the packages:



```
# yum install openstack-dashboard
```

Installation & configuration guide (hints)

Installation:

- **Steps:**
  2. Edit the */etc/openstack-dashboard/local_settings:*
     - Configure the dashboard to use OpenStack services on the controller node:
     - Allow your hosts to access the dashboard
       - ALLOWED_HOSTS can also be ['*'] to accept all hosts
     - Configure the memcached session storage service
     - Enable the Identity API version 3:**
     - Enable support for domains:**
     - Configure API versions**
     - Configure Default as the default domain for users that you create via the dashboard
     - Configure user as the default role for users that you create via the dashboard:***
     - disable support for layer-3 networking services

  3. Edit */etc/httpd/conf.d/openstack-dashboard.conf and set:*

     ```
     WSGIApplicationGroup %{GLOBAL}
     ```

  4. Restart the web server and session storage service:

     ```
     # systemctl restart httpd.service memcached.service
     ```

```
OPENSTACK_HOST = "controller"
```

```
ALLOWED_HOSTS = ['one.example.com', 'two.example.com']
```

```
SESSION_ENGINE = 'django.contrib.sessions.backends.cache'

CACHES = {
    'default': {
         'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
         'LOCATION': 'controller:11211',
    }
}
```

```
OPENSTACK_KEYSTONE_URL = "http://%s/identity/v3" % OPENSTACK_HOST
```

```
OPENSTACK_API_VERSIONS = {
    "identity": 3,
    "image": 2,
    "volume": 3,
}
```

```
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = "Default"
```

```
OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"
```

```
OPENSTACK_NEUTRON_NETWORK = {
    'enable_distributed_router': False,
    'enable_ha_router': False,
    'enable_lb': False,
    'enable_quotas': True,
    'enable_security_group': True,
    'enable_vpn': False,
}
```

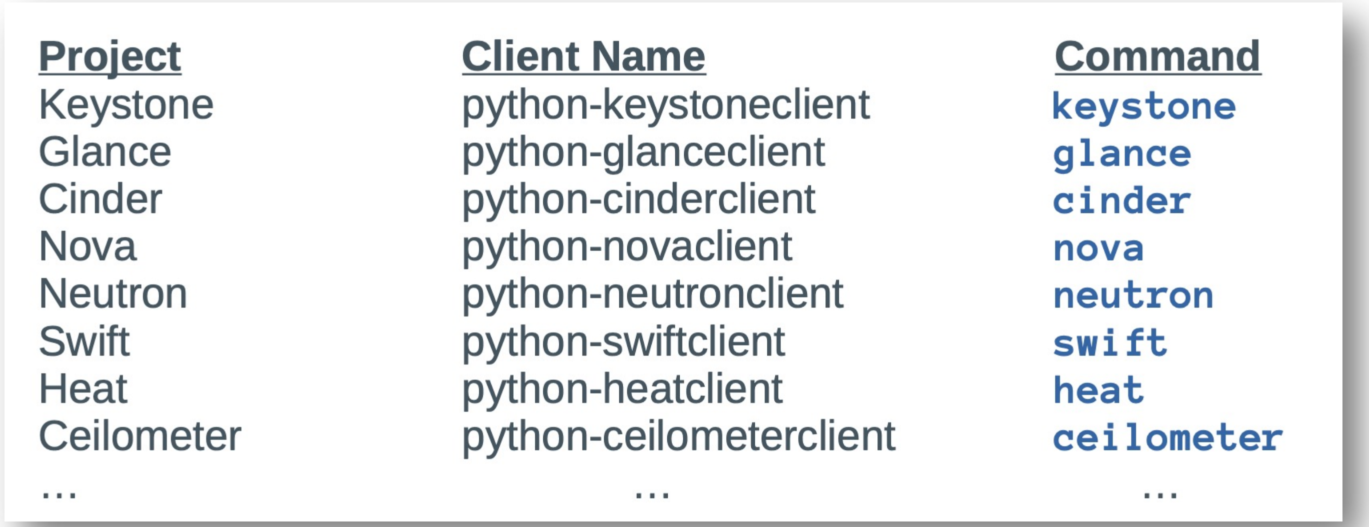
Navigation through OpenStack Dashboard

- Log in to the Dashboard – http://ipAddress_of_oa101-0X-ctrl/
  - Admin and/or user (private browsing) - the visible tabs and functions in the dashboard depend on the access permissions, or roles
    - If you are logged in as an end user, the Project tab and Identity tab are displayed.
    - If you are logged in as an administrator, the Project tab and Admin tab and Identity are displayed.
  - Project tab (leys's explore it together)
    - Projects are *organizational units* in the cloud and are also known as *tenants*.
    - Each user is a member of one or more projects.
    - Within a project, a user creates and manages instances
  - Admin tab (leys's explore it together)
    - Allows to view usage and to manage instances, volumes, flavors, images, networks,
  - Identity tab:
    - User => Project, Users, Application Credentials
    - Admin => Domains, Projects, Users, Groups, Roles
  - Settings tab:
    - User Settings, Change Password

We stop here with the Dashboard => will follow the hands-on **on creating all the elements needed for launching an instance**

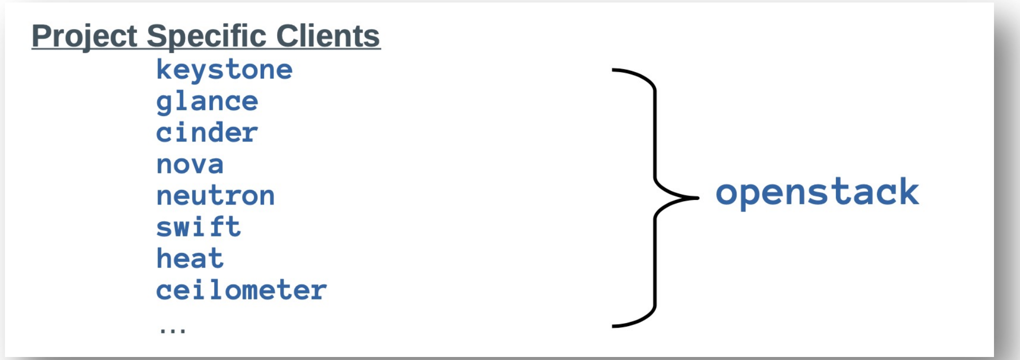
OpenStack Command Line Clients

- Python command line clients for managing OpenStack services
- Can use rc files to provide endpoint and authentication
- Communicate with each project's APIs
- Typically all are installed on Controller nodes
- Can be installed and run on any Linux machine



Project	Client Name	Command
Keystone	python-keystoneclient	`keystone`
Glance	python-glanceclient	`glance`
Cinder	python-cinderclient	`cinder`
Nova	python-novaclient	`nova`
Neutron	python-neutronclient	`neutron`
Swift	python-swiftclient	`swift`
Heat	python-heatclient	`heat`
Ceilometer	python-ceilometerclient	`ceilometer`
…	…	…

OpenStackClient

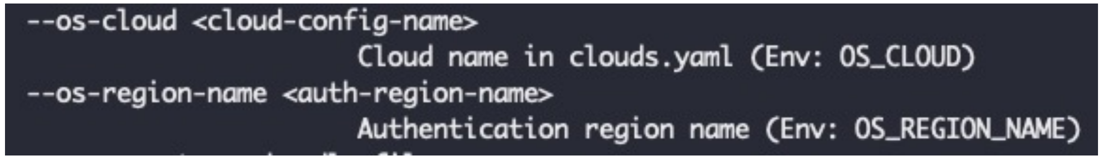


- OpenStackClient (aka OSC) is a command-line client for OpenStack that brings the command set for Compute, Identity, Image, Object Storage and Block Storage APIs together in a single shell with a uniform command structure.
  - The **openstack** command combines most of the features of the project specific CLI client into a single CLI client
  - Although most of the project specific command functionality can be replicated with the *openstack* command, there are some gaps.
  - There is documentation that shows the mapping between project specific commands and the *openstack* command. The map is located at: https://docs.openstack.org/python-openstackclient/latest/cli/decoder.html

- Goals
  - Use the OpenStack Python API libraries, extending or replacing them as required
  - Use a consistent naming and structure for commands and arguments
  - Provide consistent output formats with optional machine parseable formats
  - Use a single-binary approach that also contains an embedded shell that can execute multiple commands on a single authentication
  - Independence from the OpenStack project names; only API names are referenced (to the extent possible)

OpenStackClient

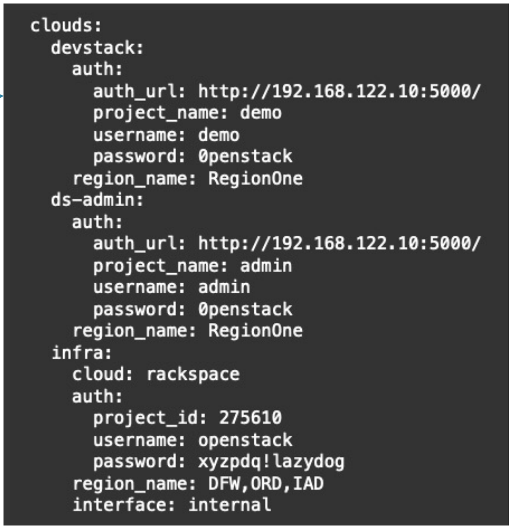
- **Installation**:
  - Ensure you have the proper repository for the Openstack version of your cloud infrastructure, for ex. *centos-release-openstack-wallaby-1-1.el8.noarch*
  - Install using "yum install python-openstackclient"
- **Configuration**, various methods:
  - Primarily configured using *command line options* and *environment variables*
  - There is a **relationship between the global options, environment variables and keywords** used in the configuration files that should make translation between these three areas simple
    - global options have a corresponding **environment variable** that may also be used to set the value
    - If both are present, the **command-line option takes priority**
    - environment variable names are derived from the option name by **dropping the leading dashes (--)**, converting each ***embedded dash (-) to an underscore (_)***, and **converting to upper case**
    - *keyword names* in the configurations files are ***derived from the global option*** names by *dropping the --os-* prefix if present



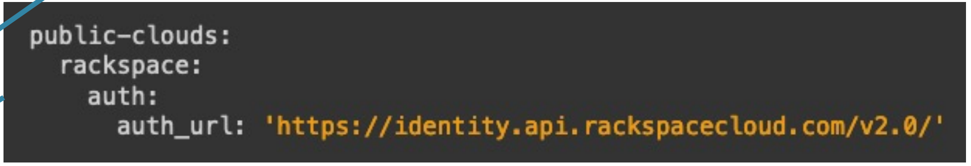
```
--os-cloud <cloud-config-name>
                        Cloud name in clouds.yaml (Env: OS_CLOUD)
--os-region-name <auth-region-name>
                        Authentication region name (Env: OS_REGION_NAME)
```

OpenStackClient

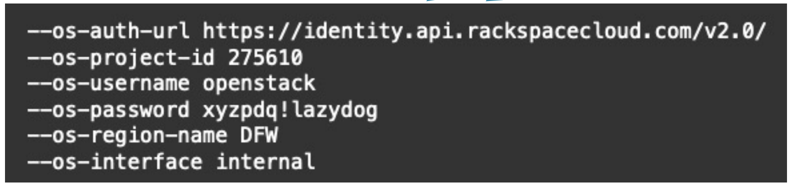
- Configuration, various methods:
  - Most of the settings can also be *placed into a configuration file* to simplify managing multiple cloud configurations:
    - ***clouds.yaml*** - contains everything needed to connect to one or more clouds.
      - may contain ***private information*** and is generally considered ***private to a user***.
      - Locations (first found wins!)
        - *current directory*
        - *~/.config/openstack*
        - */etc/openstack*
    - ***clouds-public.yaml*** - is intended to ***contain public information*** about clouds that are ***common across a large number of users***.
      - could easily be shared among users to simplify public cloud configuration
      - Same as above for the locations



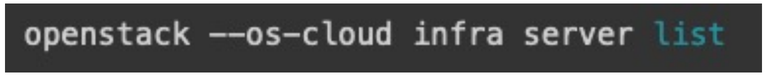
```
clouds:
  devstack:
    auth:
      auth_url: http://192.168.122.10:5000/
      project_name: demo
      username: demo
      password: 0penstack
    region_name: RegionOne
  ds-admin:
    auth:
      auth_url: http://192.168.122.10:5000/
      project_name: admin
      username: admin
      password: 0penstack
    region_name: RegionOne
  infra:
    cloud: rackspace
    auth:
      project_id: 275610
      username: openstack
      password: xyzpdq!lazydog
    region_name: DFW,ORD,IAD
    interface: internal
```



```
public-clouds:
  rackspace:
    auth:
      auth_url: 'https://identity.api.rackspacecloud.com/v2.0/'
```



```
--os-auth-url https://identity.api.rackspacecloud.com/v2.0/
--os-project-id 275610
--os-username openstack
--os-password xyzpdq!lazydog
--os-region-name DFW
--os-interface internal
```

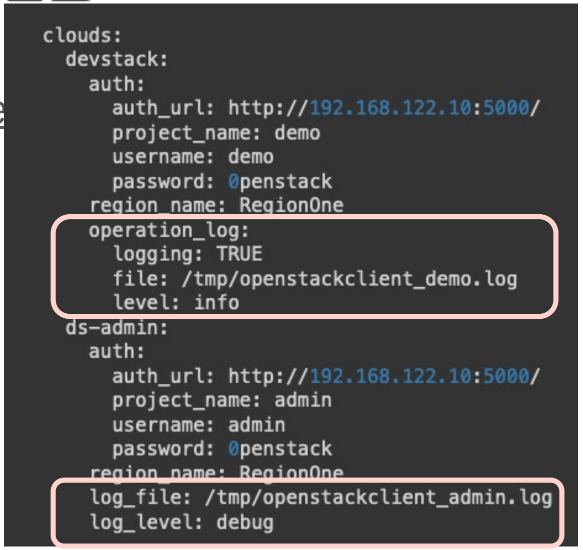


```
openstack --os-cloud infra server list
```

OpenStack Client

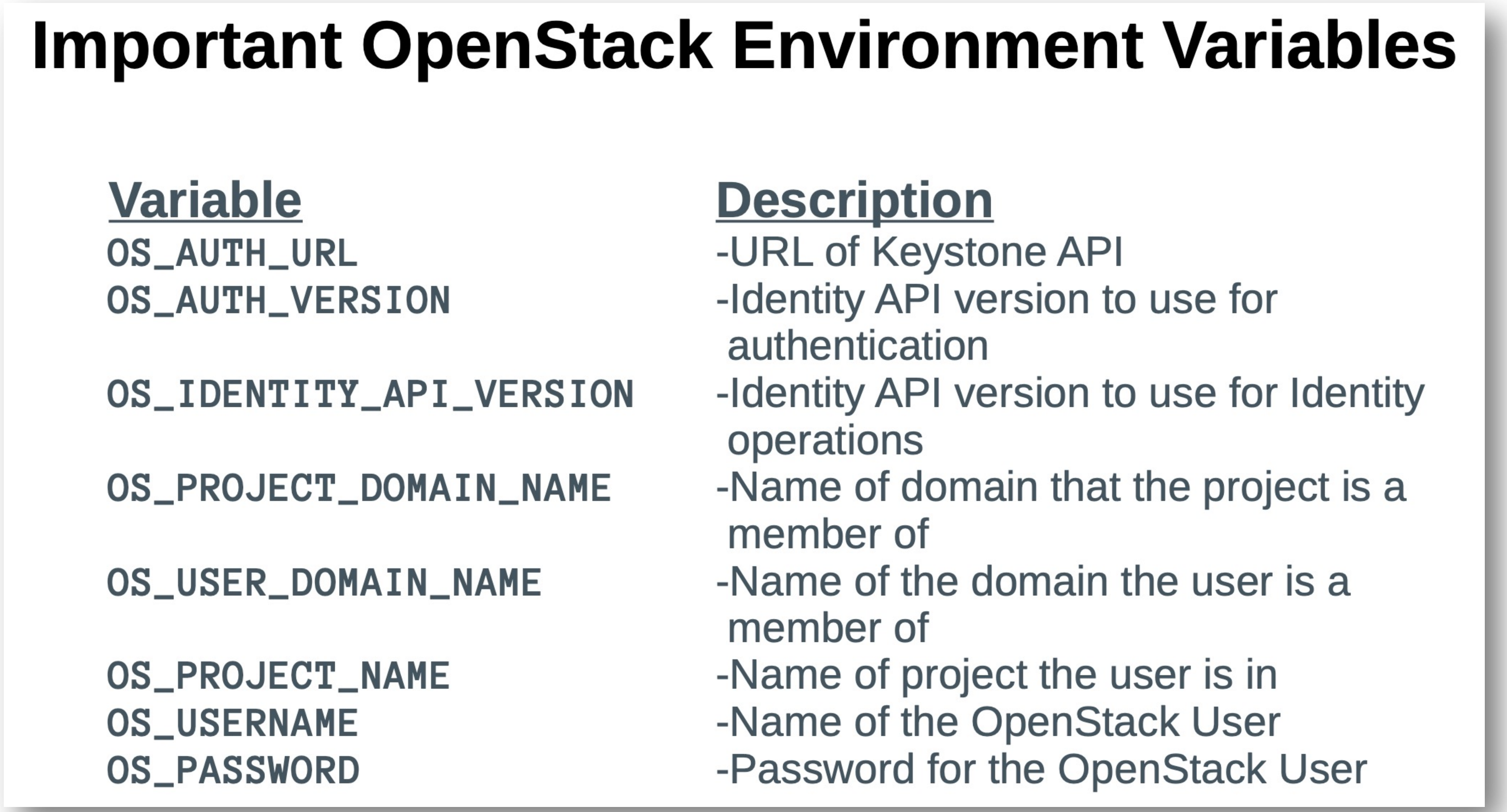
Logging settings:

- For the multiple clouds (accounts) case (**clouds.yaml**), set log_file, *log_leve*
  - *log_file*: </path/file-name>
    - Full path to logging file.
  - *log_level*: error | info | debug
    - If log level is not set, *warning* will be used
  - When a command is executed, these logs are saved **every time**
- If saving the output of a single command use the *- -log-file* option instead.
  - *–log-file <LOG_FILE>*
- The logging level for *- - log-file* can be set by using following options.
  - *-v, –verbose*
  - *-q, –quiet*
  - *–debug*



```
clouds:
  devstack:
    auth:
      auth_url: http://192.168.122.10:5000/
      project_name: demo
      username: demo
      password: 0penstack
    region_name: RegionOne
    operation_log:
      logging: TRUE
      file: /tmp/openstackclient_demo.log
      level: info
  ds-admin:
    auth:
      auth_url: http://192.168.122.10:5000/
      project_name: admin
      username: admin
      password: 0penstack
    region_name: RegionOne
    log_file: /tmp/openstackclient_admin.log
    log_level: debug
```

Understand OpenStack Credentials (rc) files

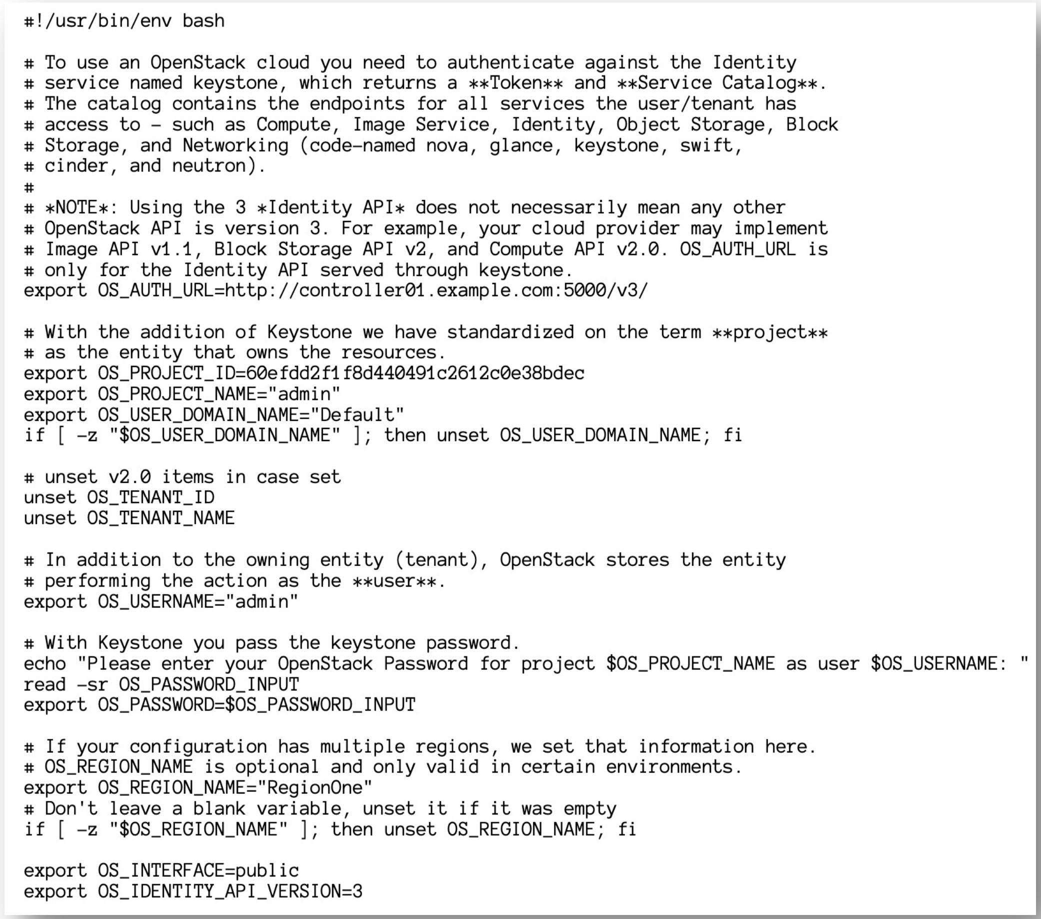


Important OpenStack Environment Variables

Variable	Description
OS_AUTH_URL	-URL of Keystone API
OS_AUTH_VERSION	-Identity API version to use for authentication
OS_IDENTITY_API_VERSION	-Identity API version to use for Identity operations
OS_PROJECT_DOMAIN_NAME	-Name of domain that the project is a member of
OS_USER_DOMAIN_NAME	-Name of the domain the user is a member of
OS_PROJECT_NAME	-Name of project the user is in
OS_USERNAME	-Name of the OpenStack User
OS_PASSWORD	-Password for the OpenStack User

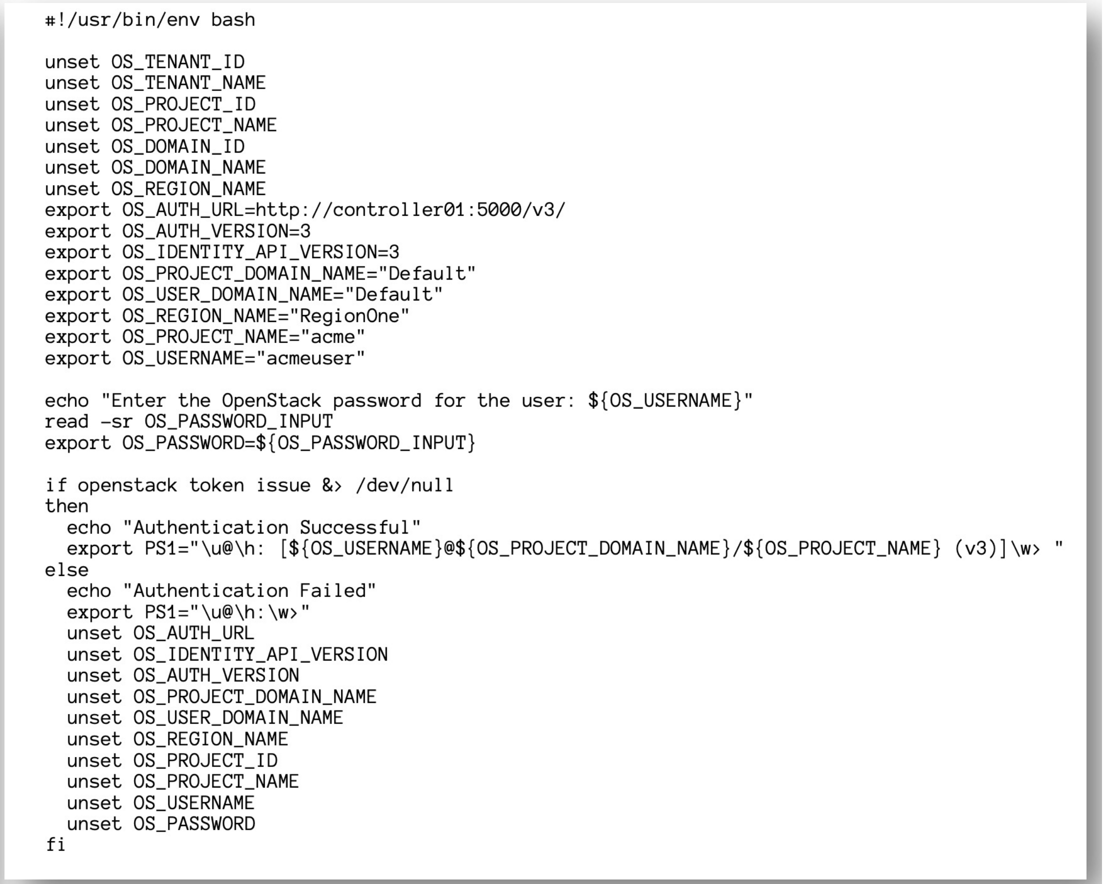
Example OpenStack RC Files

Downloadable



```
#!/usr/bin/env bash

# To use an OpenStack cloud you need to authenticate against the Identity
# service named keystone, which returns a **Token** and **Service Catalog**.
# The catalog contains the endpoints for all services the user/tenant has
# access to - such as Compute, Image Service, Identity, Object Storage, Block
# Storage, and Networking (code-named nova, glance, keystone, swift,
# cinder, and neutron).
#
# *NOTE*: Using the 3 *Identity API* does not necessarily mean any other
# OpenStack API is version 3. For example, your cloud provider may implement
# Image API v1.1, Block Storage API v2, and Compute API v2.0. OS_AUTH_URL is
# only for the Identity API served through keystone.
export OS_AUTH_URL=http://controller01.example.com:5000/v3/

# With the addition of Keystone we have standardized on the term **project**
# as the entity that owns the resources.
export OS_PROJECT_ID=60efdd2f1f8d440491c2612c0e38bdec
export OS_PROJECT_NAME="admin"
export OS_USER_DOMAIN_NAME="Default"
if [ -z "$OS_USER_DOMAIN_NAME" ]; then unset OS_USER_DOMAIN_NAME; fi

# unset v2.0 items in case set
unset OS_TENANT_ID
unset OS_TENANT_NAME

# In addition to the owning entity (tenant), OpenStack stores the entity
# performing the action as the **user**.
export OS_USERNAME="admin"

# With Keystone you pass the keystone password.
echo "Please enter your OpenStack Password for project $OS_PROJECT_NAME as user $OS_USERNAME: "
read -sr OS_PASSWORD_INPUT
export OS_PASSWORD=$OS_PASSWORD_INPUT

# If your configuration has multiple regions, we set that information here.
# OS_REGION_NAME is optional and only valid in certain environments.
export OS_REGION_NAME="RegionOne"
# Don't leave a blank variable, unset it if it was empty
if [ -z "$OS_REGION_NAME" ]; then unset OS_REGION_NAME; fi

export OS_INTERFACE=public
export OS_IDENTITY_API_VERSION=3
```

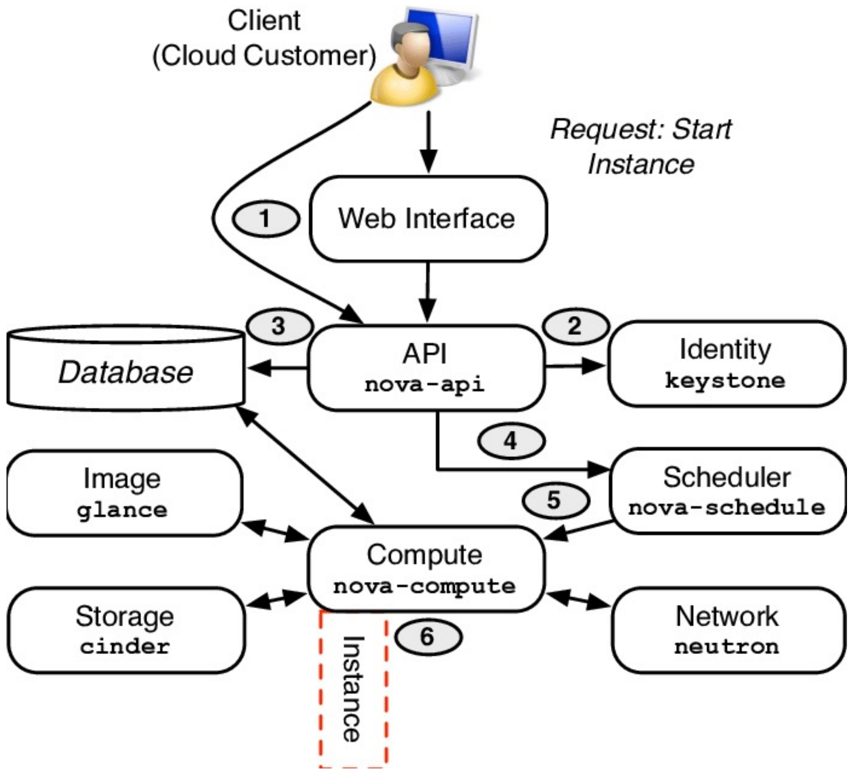
Custom



```
#!/usr/bin/env bash

unset OS_TENANT_ID
unset OS_TENANT_NAME
unset OS_PROJECT_ID
unset OS_PROJECT_NAME
unset OS_DOMAIN_ID
unset OS_DOMAIN_NAME
unset OS_REGION_NAME
export OS_AUTH_URL=http://controller01:5000/v3/
export OS_AUTH_VERSION=3
export OS_IDENTITY_API_VERSION=3
export OS_PROJECT_DOMAIN_NAME="Default"
export OS_USER_DOMAIN_NAME="Default"
export OS_REGION_NAME="RegionOne"
export OS_PROJECT_NAME="acme"
export OS_USERNAME="acmeuser"

echo "Enter the OpenStack password for the user: ${OS_USERNAME}"
read -sr OS_PASSWORD_INPUT
export OS_PASSWORD=${OS_PASSWORD_INPUT}

if openstack token issue &> /dev/null
then
  echo "Authentication Successful"
  export PS1="\u@\h: [${OS_USERNAME}@${OS_PROJECT_DOMAIN_NAME}/${OS_PROJECT_NAME} (v3)]\w> "
else
  echo "Authentication Failed"
  export PS1="\u@\h:\w>"
  unset OS_AUTH_URL
  unset OS_IDENTITY_API_VERSION
  unset OS_AUTH_VERSION
  unset OS_PROJECT_DOMAIN_NAME
  unset OS_USER_DOMAIN_NAME
  unset OS_REGION_NAME
  unset OS_PROJECT_ID
  unset OS_PROJECT_NAME
  unset OS_USERNAME
  unset OS_PASSWORD
fi
```

Hands-on time

VM creation



https://corso_oa101.baltig-pages.infn.it/hands-on/

References

- https://docs.openstack.org/horizon/wallaby/
- https://docs.openstack.org/python-openstackclient/latest/