APP DEVELOPMENT WITH OPENSTACK SDK

FRANCESCO VOLLERO

FVOLLERO@REDHAT.COM

TABLE OF CONTENTS

WHAT IS OPENSTACK

- One of the biggest Opensource project after Linux Kernel
- Private Cloud Platform

also known as the Opensource version of AWS

HOW WE COMMUNICATE WITH OPENSTACK

- API
  - CLI
  - WEbUI

API

- REST Architectural Style on HTTP
- Use the HTTP Verbs (GET, POST, PUT, DELETE, etc.)
- Performs the processing in response to the request,
- Available if program corresponding to HTTP.
  - Possible automation of the process.

API EXAMPLE (IE: SERVER REBOOT)

```
http://localhost:5000/v2/{tenant_id}/servers/{server_id}/action
```

Header

```
X-Auth-Token: 93e34d80815d4c78a80182ad7cb565a6
```

Request

```
{
     "Reboot": { "type": "SOFT"}
}
```

Consuming the API with curl

```
curl http://host/action -XPOST -H "X-Auth-Token: 324e" -d '{ "reboot": {
```

API WEAKNESS

- Too awkward to learn fastly
    - There is a need to enter the authentication token every time
    - There is a need to make the header and body
    - It is necessary to store the ID assigned to the resource to be executed, such as ID of the server

CLI (COMMAND LINE INTERFACE)

- CLI allow the user to intuitively run actions
- The user can run the command from the shell.
- Depending on the command, CLI will issue an HTTP request,
- Returns a result received from the API to the user.
- Rather than the json format, we get an human readable format.

CLI EXAMPLE

```
nova reboot [--hard] [--poll] instance_name
```

SDK

- In order to use the API in various languages (python, ruby and so on),
- Processing of API (methods) and usage data (property)
- Server, such as Object conforming to the entity as unique thing.
- Native integration with the language

SDK

```
import server_manager
server = server_manager.create (name = xxxx, image = yyyyy, ...)
server_id = server.id
server.reboot()
```

SDK (ADVANTAGES)

- Compared to the CLI + shell, the SDK + various languages is easy to use with a high function.
  - Object-Oriented
  - Be able to assign an object to a variable.
  - The basic syntax is high-function (for, if, calculation, substitution)
  - Easy-to-use multi-threaded, inter-process communication, daemon

API ~ SDK ~ CLI

- Ease of use: API < SDK < CLI
- Degrees of freedom: API> SDK> CLI

SDK ECOSYSTEM

PYTHON

- **openstack native clients**
- **python-openstacksdk**
- **pyrax** - Wroted by Rackspace for their public cloud
- **libcloud** - Multi cloud provider library
- **shade** - Simple and easy to use client library

RUBY

- fog
- Aviator

JAVA

- Apache Jclouds
- Openstack4j

OTHER LANGUAGES

- Keystone and Swift libraries(C)
- lfe-openstack(Erlang)
- OpenStack-SDK-Go(Go)

NATIVE PYTHON CLIENT AS SDK

- Own implementation in each project
  - Design (Object) different for each project.
- Correspondence situation of the API version is the back-end
  - Nova: v2 only correspondence.
    - Even if you specify the v1.1 and v3 is SDK for v2 will be used
  - Neutron: v2 support
  - Keystone: v2 / v3 support
  - Glance: v1 / v2 support
  - Cinder: v1 / v2 support

INTRODUCTION TO PYTHON CLIENTS

- Using ipython
  - Processing a line at a time of python, executable as REPL

```
pip install ipython
### Or python command
ipython
In [1]: print ("Hello World")
Hello World
```

INTRODUCTION TO PYTHON CLIENTS FOR NOVA AND NEUTRON

```
import os
### Import from SDK
from novaclient.client import Client as NovaClient
from neutronclient.v2_0.client import Client as NeutronClient
from keystoneclient.auth.identity import v2 as keystone_v2
from keystoneclient import session
### Create keystone auth information object
auth = keystone_v2.Password (auth_url = os.environ [ "OS_AUTH_URL"],
                   username = os.environ [ "OS_USERNAME"],
                   password = os.environ [ "OS_PASSWORD"],
                   tenant_name = os.environ [ "OS_TENANT_NAME"])
### Create keystone session and access to nova and neutron.
nova_client = NovaClient (2, session = session.Session (auth = auth))
neutron_client = NeutronClient (session = session.Session (auth = auth))
```

USE NOVA AND NEUTRON CLIENT AS SDK EASILY

- The methods and properties an object has,
- It can be accessed in the "obj.method" Ya "obj.property".
  - Ipython is us lists the methods and properties.

```
In [25]: nova_client.
nova_client.agents nova_client.fping nova_client.security_group_rules
nova_client.aggregates nova_client.get_timings nova_client.security_group
nova_client.authenticate nova_client.hosts nova_client.server_groups
nova_client.availability_zones nova_client.hypervisor_stats nova_client.s
nova_client.certs nova_client.hypervisors nova_client.services
nova_client.client nova_client.images nova_client.set_management_url
nova_client.cloudpipe nova_client.keypairs nova_client.tenant_id
nova_client.dns_domains nova_client.limits nova_client.usage
```

USE NOVA AND NEUTRON CLIENT AS SDK EASILY

- Server list ? `nova_client.servers.list()`
  - A result, Server object came back at 3 array.

```
In [24]: nova_client.servers
Out [24]: <novaclient.v2.servers.ServerManager at 0x26d3210>
In [25]: nova_client.servers.list ()
Out [25]: [<Server: demo_server2>, <Server: demo_server1>, <Server: demo_server0
```

USE NOVA AND NEUTRON CLIENT AS SDK EASILY

- Get the name by turning the server list in a for statement.
  - Assignment to the servers to change the return value of the server list processing
  - Turn the servers array in a for statement.
    - Ipython is convenient for us to supplement the automatic also indent
    - Processing until you enter all the for statement does not begin

```
In [10]: servers = nova_client.servers.list ()
In [11]: for server in servers:
   ....: print server.name
   ....: If server.name == "instance1":
   ....: print "I am" + server.name
instance-123
instance1
I am instance1
demo_server0
```

BALAGAN?

- Heavily tested since used by the CLI
- Need to use ipython (for tab completion) or read the man thousand of times
- Not really straightforward

PYTHON-OPENSTACKSDK

- Start from the connection class, from the connection object, get a client to talk with all services
- For clients of the acquisition as of existing python-clients, separately, there is no need to pass authentication information.

```
# Gather informations about the flavors and routers
conn = connection.Connection (** auth_args)
flavors = conn.compute.list_flavors ()
routers = conn.network.list_routers ()
```

DEMO

CONCLUSION

- If you use the SDK
  - More easily than the CLI implementation is difficult branch processing
  - Be able to take advantage of the rich library
- Python-clients
  - Because it is its own implementation for each project requires a separate investigation
- Python-openstacksdk
  - In the manners of operation is constant, it is unified.
  - On the other hand, due to his youngness are still expected in the future some bugs.

Q & A