Orchestration with Openstack Heat

July 24, 2017

sawangpong@itbakery.net

Topic

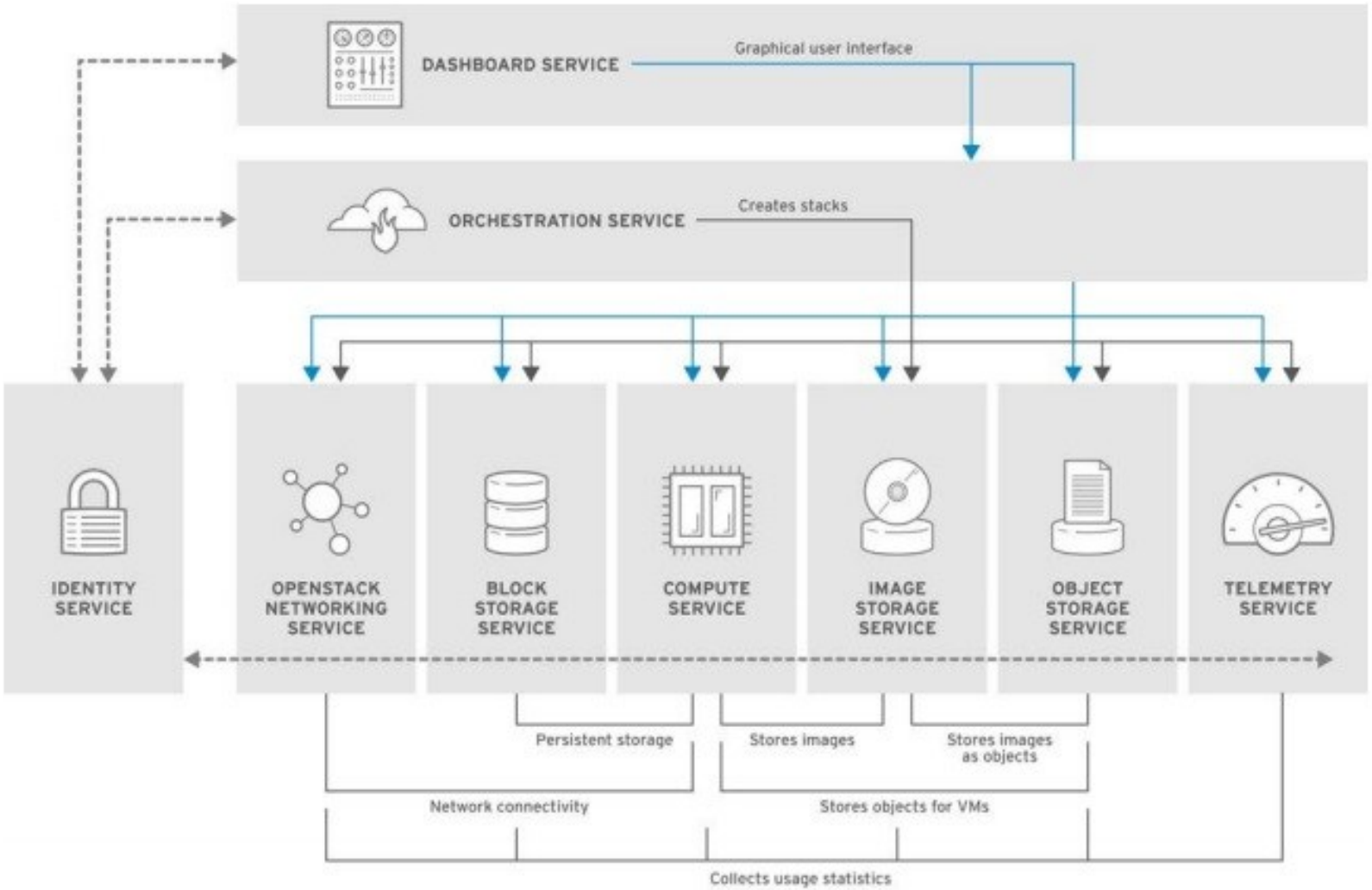
- What is Orchestration, Why we need it?
- Set up Environment
- Heat Architecture && Heat Template

- Workshop (9-10 September)
  - Raw
  - Cloud-init
  - Cloud-config
  - Application Autoscale + load balance + mysql cluster

Orchestration

Orchestration is the automated arrangement, coordination, and management of computer systems, middleware, and services.

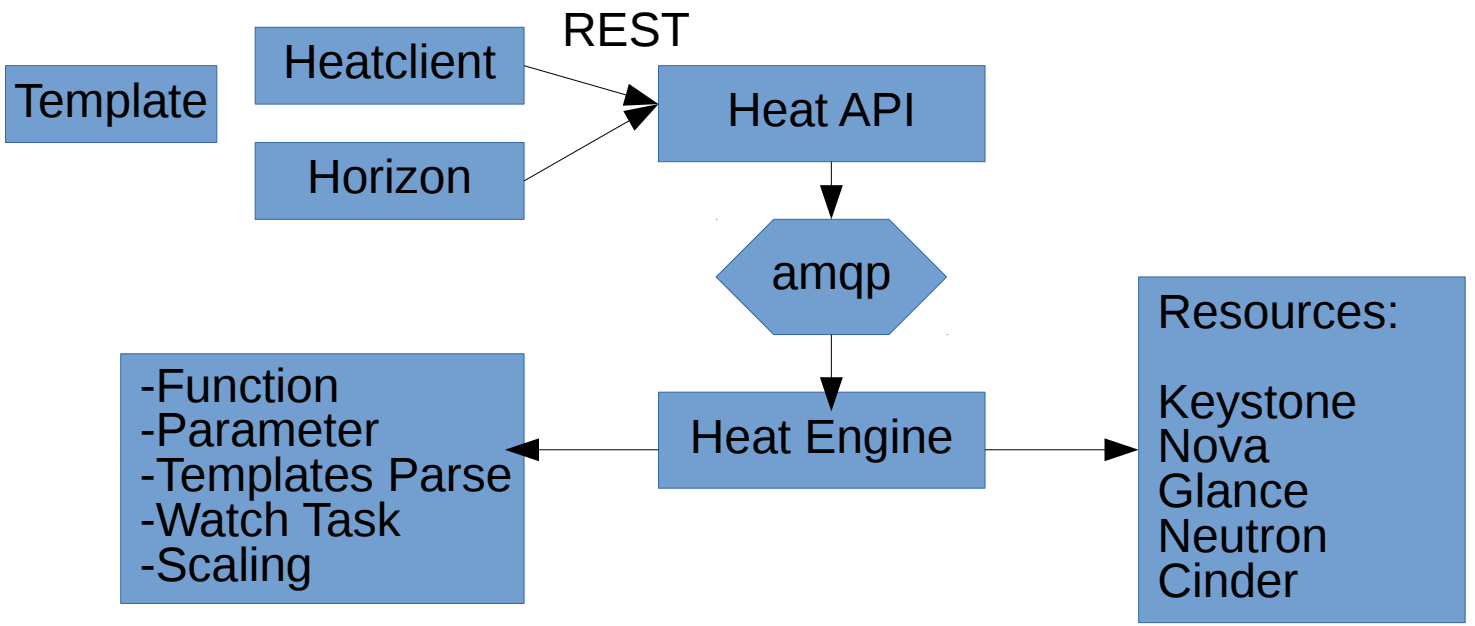
Heat overview



Heat

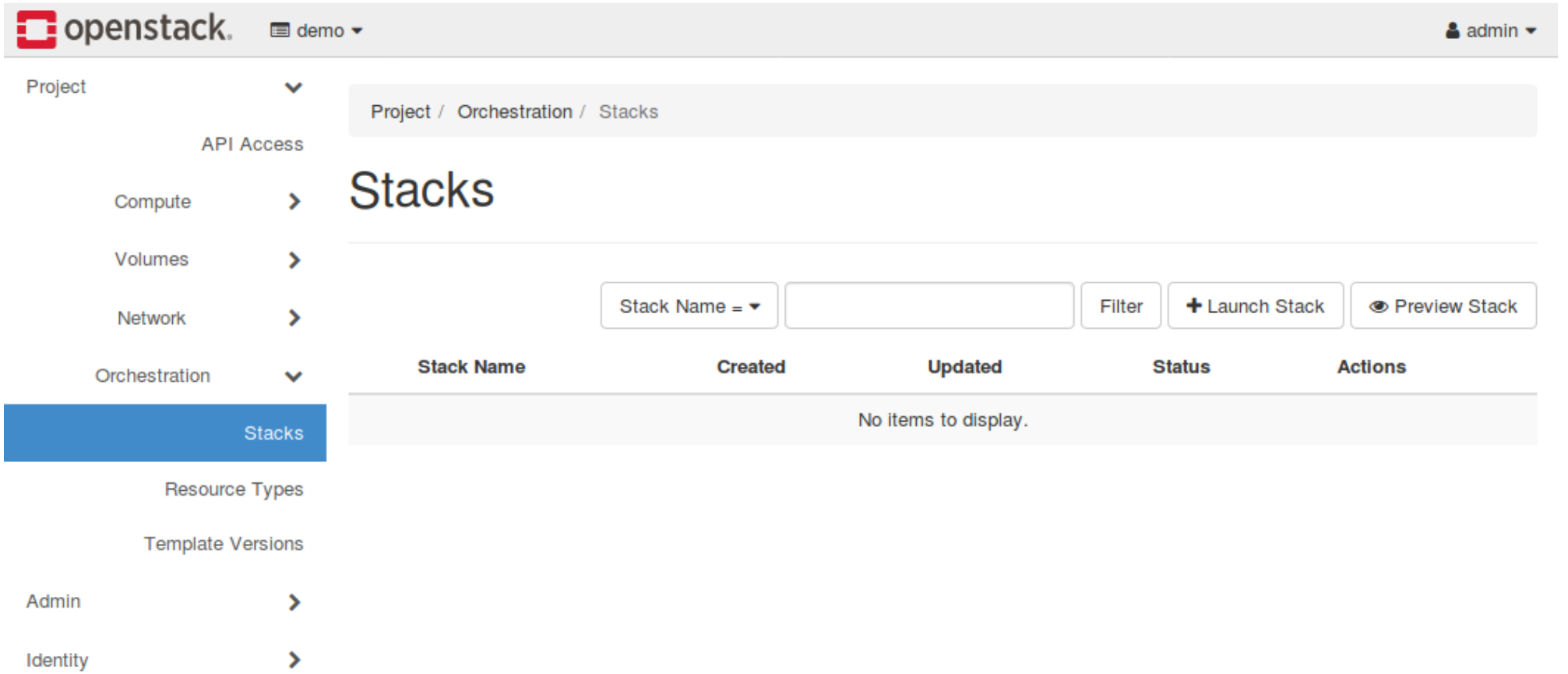
- Heat provides a mechanism for orchestrating Openstack resources via modular templates
- HOT Template written in YAML (recommented) , json
- Heat allows you to create multiple instances, logical network, and other services in automation process
- Support by Horizon (GUI), CLI (Heat Client Tool)
- Services
  - Heat-api: Openstack API
  - Heat-api-cfn: Cloud Formation Compatible API
  - Heat-engine: Sends resources creation requests to openstacks

Heat Architecture



- Heat engine manage resources

Horizon support Heat



Resource type

openstack. demo admin

Project / Orchestration / Resource Types

API Access

Compute

Volumes

Network

Orchestration

Stacks

Resource Types

Template Versions

Admin

Identity

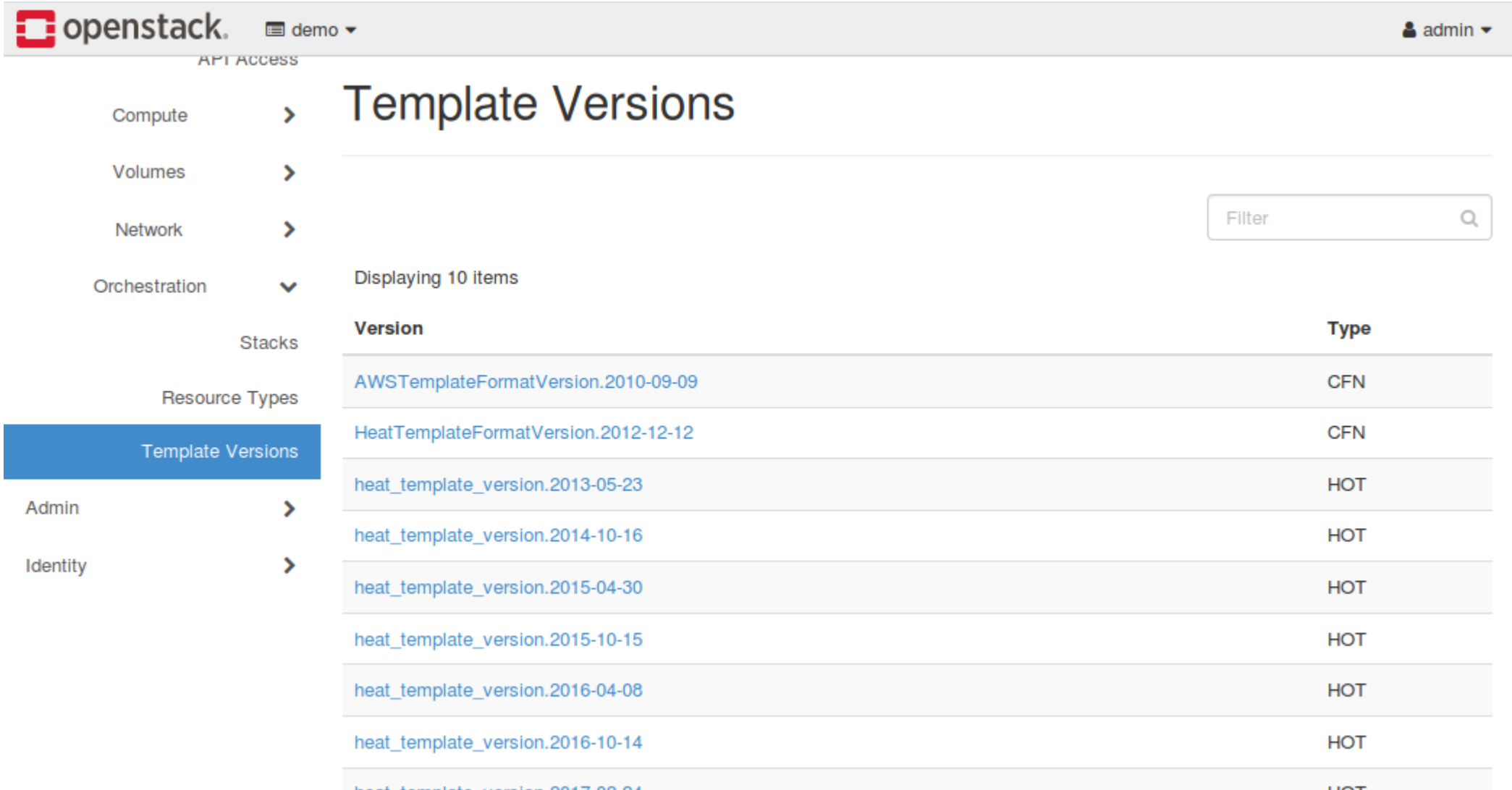
Resource Types

Type = Filter

Displaying 101 items

Type
AWS::AutoScaling::AutoScalingGroup
AWS::AutoScaling::LaunchConfiguration
AWS::AutoScaling::ScalingPolicy
AWS::CloudFormation::Stack
AWS::CloudFormation::WaitCondition
AWS::CloudFormation::WaitConditionHandle

Support Template Version



Heat Document

https://docs.openstack.org/heat/latest/

openstack. SEARCH SOFTWARE USERS COMMUNITY MARKETPLACE EVENTS LEARN DOCS JOIN LOG IN

OpenStack Documentation

heat 9.0.0.0b3.dev156

Welcome to the Heat documentation!

UPDATED: 'SUN JUL 23 18:58:04 2017, COMMIT BA41249'

Heat is a service to orchestrate composite cloud applications using a declarative template format through an OpenStack-native REST API.

Heat's purpose and vision

- Heat provides a template based orchestration for describing a cloud application by executing appropriate OpenStack API calls to generate running cloud applications.
- A Heat template describes the infrastructure for a cloud application in text files which are readable and writable by humans, and can be managed by version control tools.

Basic Heat Orchestration Template (HOT)

- Written in YAML Syntax
- Templates define a stack
- Stack – group of openstack resources (Nova, Volume)
- Support CloudFormation Compatible (CFN)

Understanding Yaml

- YMAL is markup language
- YMAL provide data structure compile into Python data structure
- Rule #1: Indentation 2 space. Each level show data collection
- Rule #2: Colons. Key-value pairs represent python dictionary format (hash, associative arrays)
  my_key: my_value

Key-value map to python

Key value

```
my_key:  my_value
```

```
{my_key:  my_value}
```

```
first_level_dict_key:
  second_level_dict_key: value_in_second_level_dict
```

```
{
    'first_level_dict_key': {
        'second_level_dict_key': 'value_in_second_level_dict'
    }
}
```

- Rule #3: Dashes Represent lists of items

```
- list_value_one
- list_value_two
- list_value_three
```

- List can be the value of key-value pair

```
my_dictionary:
  - list_value_one
  - list_value_two
  - list_value_three
```

```
{'my_dictionary': ['list_value_one',
'list_value_two',
'list_value_three']}
```

YAML Parser Practice

- http://yaml-online-parser.appspot.com/

Online YAML Parser

```
- just: write some
- yaml:
  - [here, and]
  - {it: updates, in: real-time}
```

Output

```
[
  {
    "just": "write some"
  },
  {
    "yaml": [
      [
        "here",
        "and"
      ],
      {
        "it": "updates",
        "in": "real-time"
      }
    ]
  }
]
```

Output: ◉ json ○ python ○ canonical yaml Link to this page

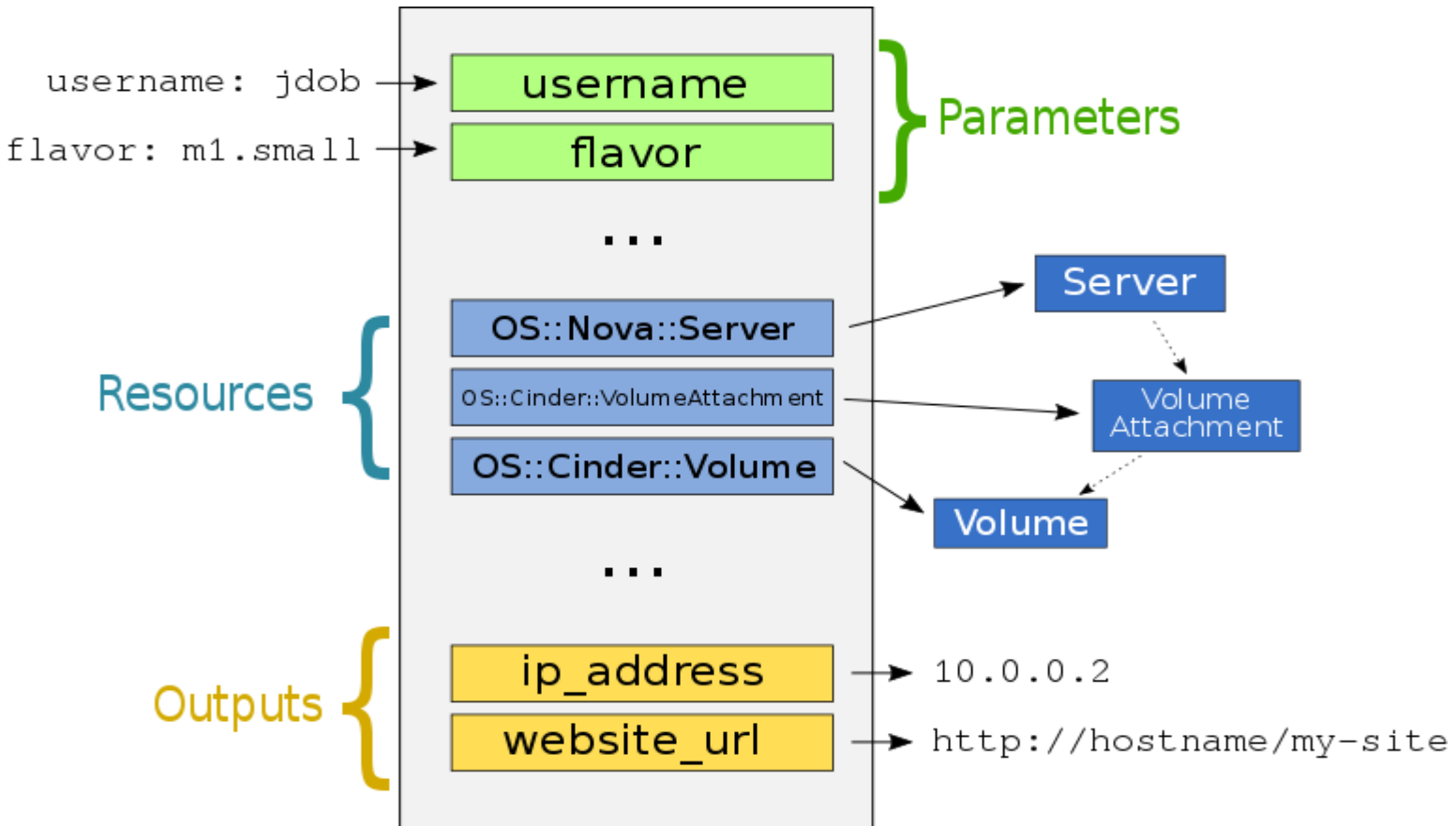
Examples from YAML 1.2 Spec

Template Format 6 sections

```
heat_template_version: 2015-04-30

description:

parameters: #Input parameters that have to provide when
              instantiating the template

resources: #the resource to be create

output: #data to pass back to user stack creation

Condition: #used to restrict stack creation based on resource
           properties
```

Ref:
https://docs.openstack.org/heat/latest/template_guide/hot_guide.html

Stack visualization



Basic Template

```
heat_template_version: 2015-04-30

description: Simple template to deploy a single compute instance

resources:
  my_instance:
    type: OS::Nova::Server
    properties:
      key_name: my_key
      image: F18-x86_64-cfntools
      flavor: m1.small
```

Set default value for parameter

```
parameters:
  instance_type:
    type: string
    label: Instance Type
    description: Type of instance (flavor) to be used
    default: m1.small
```

```
parameters:
  database_password:
    type: string
    label: Database Password
    description: Password to be used for database
    hidden: true
```

Template with parameter

```
heat_template_version: 2015-04-30

description: Simple template to deploy a single compute instance

parameters:
  key_name:
    type: string
    label: Key Name
    description: Name of key-pair to be used for compute instance
  image_id:
    type: string
    label: Image ID
    description: Image to be used for compute instance
  instance_type:
    type: string
    label: Instance Type
    description: Type of instance (flavor) to be used

resources:
  my_instance:
    type: OS::Nova::Server
    properties:
      key_name: { get_param: key_name }
      image: { get_param: image_id }
      flavor: { get_param: instance_type }
```

Restrict user input

```
parameters:
  instance_type:
    type: string
    label: Instance Type
    description: Type of instance (flavor) to be used
    constraints:
      - allowed_values: [ m1.medium, m1.large, m1.xlarge ]
        description: Value must be one of m1.medium, m1.large or m1.xlarge.
```

Restrict user input with pattern

```
parameters:
  database_password:
    type: string
    label: Database Password
    description: Password to be used for database
    hidden: true
    constraints:
      - length: { min: 6, max: 8 }
        description: Password length must be between 6 and 8 characters.
      - allowed_pattern: "[a-zA-Z0-9]+"
        description: Password must consist of characters and numbers only.
      - allowed_pattern: "[A-Z]+[a-zA-Z0-9]*"
        description: Password must start with an uppercase character.
```

Template output

```
resources:
  my_instance:
    type: OS::Nova::Server
    # ...

outputs:
  instance_ip:
    description: IP address of the deployed compute instance
    value: { get_attr: [my_instance, first_address] }
  instance_private_ip:
    description: Private IP address of the deployed compute instance
    value: { get_attr: [my_instance, networks, private, 0] }
```

Ref:
https://docs.openstack.org/heat/latest/template_guide/hot_spec.html#hot-spec-intrinsic-functions

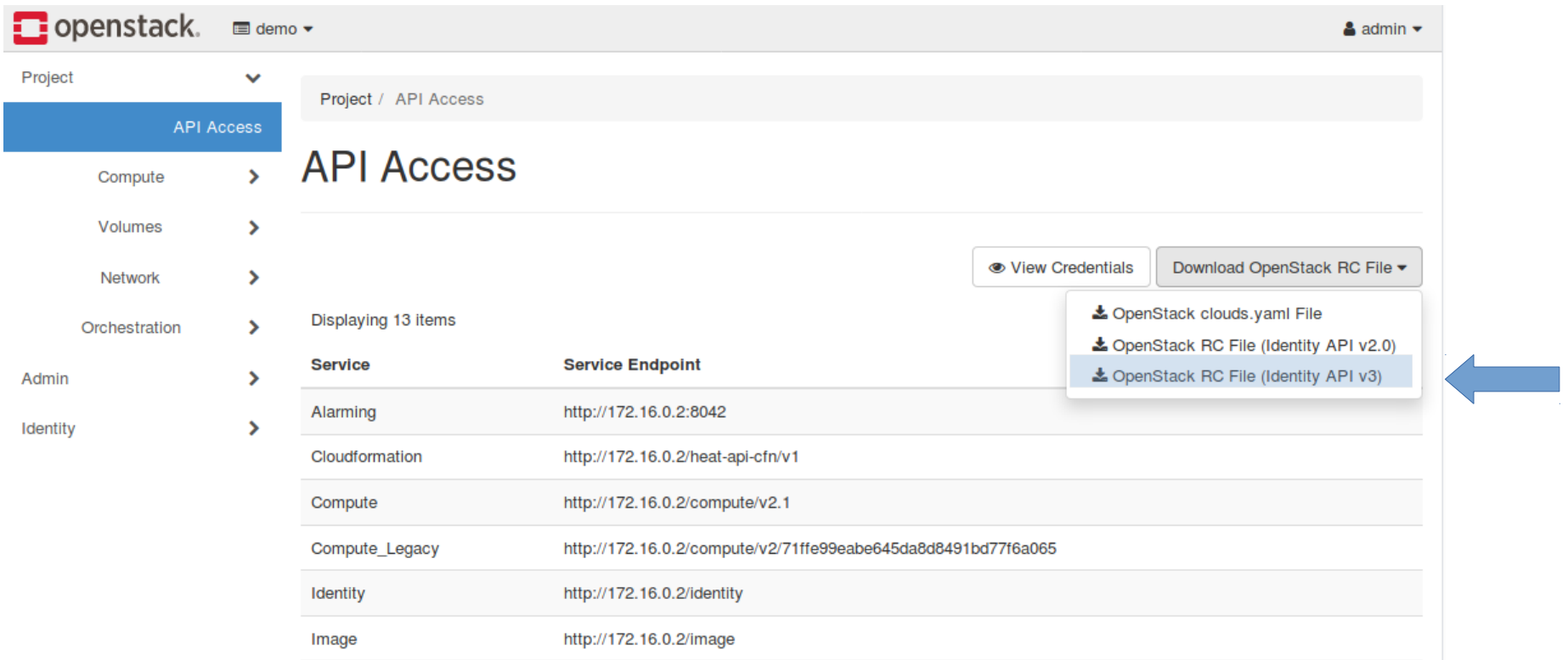
Create your first stack

- Build environment devstack
- https://github.com/itbakery/heatdemo

```
$ git clone https://github.com/itbakery/heatdemo
$ cd heatdemo
$ vagrant up  --provider=libvirt

$ cd /vagrant/
$ bash devstack.sh
```

Login to Horizon

- Login as ‘admin’
- Download openstack RC file , rename ‘admin-openrc.sh’



Create your first stack

- Source admin-openrc.sh

```
$ source admin-openrc.sh
$ openstack stack list
$ openstack flavor list
$ openstack image list
$ openstack keypair create heat_key > heat_key.priv
$ chmod 600 heat_key.priv
```

- Create stack

```
$ openstack stack create -t demo1/stack.yaml \
--parameter super_secret_text='password' \
--parameter demo_text="demo text"  demo1
```

Create your first stack

- Running

```
+---------------------+--------------------------------------+
| Field               | Value                                |
+---------------------+--------------------------------------+
| id                  | f4d49e5d-2ca2-4031-b7a4-6594f71a7a9f |
| stack_name          | demo1                                |
| description         | No description                       |
| creation_time       | 2017-07-24T04:52:43Z                 |
| updated_time        | None                                 |
| stack_status        | CREATE_IN_PROGRESS                   |
| stack_status_reason | Stack CREATE started                 |
+---------------------+--------------------------------------+

$ openstack stack list

$ openstack stack resource list demo1
+--------------+--------------------------------------+------------------+----------------+---------------------+
| resource_name | physical_resource_id                | resource_type    | resource_status | updated_time        |
+--------------+--------------------------------------+------------------+----------------+---------------------+
| my_instance   | 5703ea60-51e2-4d9c-a6f8-caad06bee496 | OS::Nova::Server | CREATE_COMPLETE | 2017-07-24T05:34:39Z |
+--------------+--------------------------------------+------------------+----------------+---------------------+

$ openstack stack delete  demo1
$ openstack stack delete  demo1 Are you sure you want to delete this stack(s)
[y/N]? y
```

Result

192.168.121.151/dashboard/project/stacks/

openstack heat template

openstack. demo admin

Project
API Access
Compute
Volumes
Network
Orchestration
Stacks
Resource Types
Template Versions
Admin
Identity

Project / Orchestration / Stacks

Stacks

Stack Name = | Filter | + Launch Stack | Preview Stack | Delete Stacks | More Actions

Displaying 1 item

Stack Name	Created	Updated	Status	Actions
demo1	14 minutes	Never	Create Complete	Check Stack

Displaying 1 item

Use env.yaml pass parameter

```
$ openstack stack create -t demo2/stack.yaml  -e demo2/env.yaml  demo2
$ openstack stack list
$ openstack stack show demo2
```

Use get_file

```
$ openstack stack create -t demo2/stack.yaml  -e demo2/env.yaml  demo2
$ openstack stack list
$ openstack stack show demo2
```

Heat networks

```
$ openstack stack create -t demo3/stack.yaml    demo3
$ openstack stack list
$ openstack stack show demo3
```

Metering auto scale