

# Building a Heat Template from the Ground up

**Johannes Graßler**

Cloud Developer

[johannes.grassler@suse.com](mailto:johannes.grassler@suse.com)



# Introduction

# What is Heat?

- Orchestration Service controlled by instructions in HOT resource description language (or CloudFormation CFN templates)
- Introduced in OpenStack Icehouse
- Really usable as of OpenStack Kilo (some teething trouble)
- Terminology: Heat *Templates* (written in HOT) describe a setup consisting of one or more resources and are instantiated as Heat *Stacks*.

# This Workshop

- Heat Services and Architecture
- Anatomy of a Heat Template
- Hands on: Let's try and break things...

# Services and Architecture

- heat-api: OpenStack API
- heat-api-cfn: Cloud Formation Compatible API
- heat-engine: sends resource creation requests to other services (Nova, Neutron, ...)



# Anatomy of a Heat Template

# Overview

- Two possible formats for data payload: YAML or JSON (YAML recommended)
- Sections:
  - `heat_template_version`: Governs available features
  - `parameters`: Parameters that can be supplied through command line or web interface
  - `resources`: The resources to be created
  - `outputs`: data to pass back to the user upon stack creation

# Command Line Heat Client (1)

*Prerequisite: Environment with valid OpenStack credentials*

## **Create stack `mystack` from `/tmp/stack.yaml`**

```
heat stack-create -f /tmp/stack.yaml  
mystack
```

## **With parameter**

```
heat stack-create -f /tmp/stack.yaml -P  
flavor=m1.tiny mystack
```

## **Delete stack `mystack`**

```
heat stack-delete mystack
```



# Command Line Heat Client (2)

## Detailed Information about stack `mystack`

```
heat stack-show mystack
```

Error messages (if any) in field `stack_status_reason` (more readable in `heat-engine.log`, if you have access)

## List resources of stack `mystack`

```
heat resource-list -n 5 mystack
```

## List outputs for stack `mystack`

```
heat output-list mystack
```

## Get Output `floating_ip` from stack `mystack`

```
heat output-show mystack floating_ip
```

# Heat Resources: An example

```
resources:
  myserver:
    type: OS::Nova::Server
    properties:
      name: myserver
      key_name: { get_param: key_name }
      image: { get_param: image }
      flavor: { get_param: flavor }
      networks:
        - port:
            get_resource:
              myport
```

# Heat Functions

## **get\_param**

```
key_name: { get_param: key_name }
```

## **get\_resource**

```
networks:  
  - port: { get_resource: myport }
```

## **get\_attr**

```
outputs:  
  floating_ip_address:  
    description: The server's floating IP address.  
    value:  
      get_attr:  
        - myfloating_ip          # resource  
        - floating_ip_address    # attribute
```

# Hands-on: Creating a Heat Stack

# Preparations

## Get a local copy of slides

```
git clone https://github.com/jgrassler/talks.git
cd talks/heat-workshop
```

## Install Heat command line client

```
zypper install python-heatclient      # SUSE
yum install python-heatclient         # RedHat
aptitude install python-heatclient    # Debian/Ubuntu
```

## Prepare openrc

Log in to the cloud dashboard using one of the credentials sheets we handed out and download an openrc to ~/openrc.workshop. Source that openrc:

```
source ~/openrc.workshop
```

# A minimal template

## with-errors/01-minimal.yaml

```
heat_template_version: 2525-01-01
```

## Try to create stack

```
. /root/openrc.myuser  
heat stack-create --poll -f \  
    /tmp/stack.yaml mystack
```

*Partial:* partial01-broken.yaml



# Error 1: heat\_template\_version

*Partial:* partial01-broken.yaml

## Error Message

```
ERROR: The template version is invalid:  
"heat_template_version: 2525-01-01".  
"heat_template_version" should be one of:  
2013-05-23, 2014-10-16, 2015-04-30, 2015-10-15
```

## Resolution

Pick one of the template versions supported by your cloud (2015-10-15 or lower for the cloud we provided) from the list above.

*Partial:* partial01.yaml

# Adding parameters to your template

## 02-parameters.yaml

```
parameters:
  floating_network:
    type: string
    default: floating
  image:
    type: string
    default: cirros-0.3.4-x86_64
  flavor:
    type: string
    default: m1.tiny
  key_name:
    type: string
    default: root
```

## Create the stack

```
heat stack-create --poll -f /tmp/stack.yaml mystack
```

*Partial:* partial02.yaml

# Error 2: Tabs versus spaces

*Partial:* partial02-broken.yaml

## Error Message

```
Error parsing template
file:///crypt/home/johannes/src/talks/heat-
workshop/partial/partial02-broken.yaml while scanning for the
next token
found character that cannot start any token
  in "<unicode string>", line 13, column 1
```

## Resolution

Check the problematic line for a mix of tabs and spaces. Depending on your tabstop setting it may not be obvious at first. The same error may also occur due to a key starting with non-alphanumeric characters.

## Retry stack creation

```
heat stack-create --poll -f /tmp/stack.yaml mystack
```

*Partial:* partial02.yaml

# Creating a Network

**Begin resources section after parameters section**

```
resources:
```

## **03-network.yaml**

```
resources:
```

```
  mynetwork:  
    type: OS::Neutron::Net  
    properties:  
      name: mynet
```

## **Create the stack**

```
heat stack-create --poll -f /tmp/stack.yaml mystack
```

*Partial:* partial03.yaml

# Creating a Subnet

## 04-subnet.yaml

```
mysubnet:  
  type: OS::Neutron::Subnet  
  properties:  
    cidr: 10.0.0.1/24  
    name: mysubnet  
    network:  
      get_resource: mynetwork
```

## Create the stack

```
heat stack-create --poll -f /tmp/stack.yaml  
mystack
```

*Partial:* partial04.yaml

# A Port on your Network

## 05-port.yaml

```
myport:
  type: OS::Neutron::Port
  properties:
    network:
      get_resource: mynetwork
```

## Create the stack

```
heat stack-create --poll -f
/tmp/stack.yaml mystack
```

*Partial:* partial05.yaml



# Creating an Instance

## 06-server.yaml

```
myserver:
  type: OS::Nova::Server
  properties:
    name: myserver
    config_drive: true
    flavor: { get_param: flavor }
    image: { get_param: image }
    key_name: { get_param: key_name }
    networks:
      - port: { get_resource: myport }
    user_data_format: RAW
    user_data: |
      #!/bin/sh
      echo 'Hello, World' >> /etc/motd
```

## Create the stack

```
heat stack-create --poll -f /tmp/stack.yaml mystack
```

*Partial:* partial06.yaml

# Error 4: SSH key not found

## Error Message

```
ERROR: Property error: :  
resources.myserver.properties.key_name: :  
Error  
validating value 'mykey': The Key (mykey)  
could not be found.
```

## Resolution

```
nova keypair-add --pub-key  
~/.ssh/id_rsa.pub mykey
```

*Partial:* partial06.yaml

# Error 5: No valid host was found

## Modified command

```
heat stack-create --poll -f /tmp/stack.yaml -P  
flavor m1.ginormous mystack
```

## Error message

```
resources.myserver: Went to status ERROR due to  
"Message: No valid host was  
found. There are not enough hosts available., Code:  
500"
```

## Resolution

Have enough resources available, either by picking a smaller flavor or by having more/bigger compute nodes.

*Partial:* partial06.yaml

# Attempt to associate a floating IP

## 09-float.yaml

```
myfloatingip:  
  type: OS::Neutron::FloatingIP  
  properties:  
    port_id: { get_resource: myport }  
    floating_network:  
      get_param: floating_network
```

## Create the stack

```
heat stack-create --poll -f /tmp/stack.yaml  
mystack
```

*Partial:* partial09-broken.yaml

# Error 6: External network not reachable

## Check stack status

```
heat stack-show mystack
```

## Error message

```
NotFound: resources.floatingip: External network  
27011e4a-1727-499a-9c1f-b372a62071a9 is not reachable  
from subnet  
dbe382bd-0ccf-4107-9f96-8cecad3797f1 Therefore, cannot  
associate Port  
162b6498-13ce-46b9-a436-76fb50f06c67 with a Floating IP.
```

## Resolution

Comment/Remove the `floatingip` resource for now. We need a router first.

*Partial:* `partial06.yaml`

# Creating a Router

## 07-router.yaml

```
router:
  type: OS::Neutron::Router
  properties:
    external_gateway_info:
      network:
        get_param: floating_network
```

## Create the stack

```
heat stack-create --poll -f /tmp/stack.yaml
mystack
```

*Partial:* partial07.yaml



# Adding A RouterInterface

## 08-interface.yaml

```
router_interface:
  type: OS::Neutron::RouterInterface
  properties:
    router: { get_resource: router }
    subnet: { get_resource: mysubnet }
```

## Create the stack

```
heat stack-create --poll -f
/tmp/stack.yaml mystack
```

*Partial:* partial08.yaml

# Associating a Floating IP

## 09-float.yaml

```
myfloatingip:  
  type: OS::Neutron::FloatingIP  
  properties:  
    port_id: { get_resource: myport }  
    floating_network:  
      get_param: floating_network
```

## Create the stack and display floating IP

```
heat stack-create --poll -f /tmp/stack.yaml  
mystack  
heat resource-show mystack myfloatingip
```

*Partial:* partial09.yaml

# Error 7: Floating IP unreachable

## Error

*Pings or SSH to floating IP address time out*

## Resolution

Create a security group that allows inbound traffic (default is to block everything).

*Partial:* `partial09.yaml`

# Security Groups to the Rescue

## 10-group.yaml

```
allow_inbound:
  type: OS::Neutron::SecurityGroup
  properties:
    description: "Allow inbound SSH and ICMP traffic"
    name: allow SSH and ICMP from anywhere
    rules:
      - direction: ingress
        remote_ip_prefix: 0.0.0.0/0
        protocol: tcp
        port_range_min: 22
        port_range_max: 22
      - remote_ip_prefix: 0.0.0.0/0
        protocol: icmp
```

## Create the stack and display floating IP

```
heat stack-create --poll -f /tmp/stack.yaml mystack
heat resource-show mystack myfloatingip
```

*Partial:* partial10.yaml

# Error 8: Security Group not associated

## Error

Floating IP not reachable despite security group.

## Resolution

Associate the security group with the port it is supposed to apply to.

*Partial:* `partial10.yaml`

# Port revisited

## 11-port.yaml

```
myport:
  type: OS::Neutron::Port
  properties:
    network:
      get_resource: mynetwork
    security_groups: # NEW
      - get_resource: allow_inbound # NEW
```

## Create the stack and display floating IP

```
heat stack-create --poll -f /tmp/stack.yaml mystack
heat resource-show mystack floatingip
```

*Partial:* partial11.yaml



# Outputs: What's my Floating IP?

## 12-outputs.yaml

```
outputs:
  floating_ip:
    value:
      get_attr:
        - myfloatingip
        - floating_ip_address
```

## Create the stack and display floating IP

```
heat stack-create --poll -f /tmp/stack.yaml
mystack
heat output-show mystack floating_ip
```

*Partial:* partial12.yaml

# Error 9: Can't delete stack

## Error message upon stack-delete

```
Unable to complete operation on subnet  
d5bda832-2816-4453-932e-c739cc0f1152. One or more  
ports  
have an IP allocation from this subnet.
```

## Resolution

```
neutron port-list | grep d5bda832-2816-4453-932e-  
c739cc0f1152 | awk '{ print $2}'
```

This will give you a list of Neutron port IDs. One of them belongs to `myserver` from the Heat stack, the rest are freeloaders. You need to get rid of the freeloaders somehow (better talk to their owners first).

# Further Reading: Template Guide

Thank you.







**Corporate Headquarters**  
Maxfeldstrasse 5  
90409 Nuremberg  
Germany

+49 911 740 53 0 (Worldwide)  
[www.suse.com](http://www.suse.com)

Join us on:  
[www.opensuse.org](http://www.opensuse.org)