Outline

- Heat
- Architecture
- Template, Environment, Nested Templates
- WaitConditions
- Autoscaling
Heat: IaaS automation

- Orchestration service for OpenStack
- Uses templating mechanism
- Controls complex groups of cloud resources
- Huge potential and multiple use cases
Heat components

- Heat consists of several components:
  - heat-api: provides the REST API
  - heat-api-cfn: provides EC2 CloudFormation API
  - heat-engine: orchestrates the launching of templates and provides events back to the API consumer.
  - heat: a CLI which communicates with the heat-api
Logical architecture
Templates

```
# This is required.
heat_template_version: 2013-05-23

parameters:
  # parameters go here

resources:
  # resources go here (this section is required)

outputs:
  # outputs go here
```

**Parameters.** The specification of any arguments that the user might be required to provide.

**Resources** section specifies what resources Heat should create.

**Outputs.** Any expected values that are to be returned once the template has been processed.
Resources

** Autoscaling:
AWS::AutoScaling::LaunchConfiguration
AWS::AutoScaling::AutoScalingGroup
AWS::AutoScaling::ScalingPolicy'
OS::Heat::CWLiteAlarm
OS::Ceilometer::Alarm

** Virtual Machines
AWS::EC2::Instance
OS::Nova::Server
AWS::CloudFormation::Stack

** Object storage
OS::Swift::Container

** Volumes:
OS::Cinder::Volume
OS::Cinder::VolumeAttachment

** High Availability:
OS::Heat::HARestarter

*** Neutron SDN:
OS::Neutron::FloatingIP
OS::Neutron::FloatingIPAssociation
OS::Neutron::Port
OS::Neutron::Router
OS::Neutron::RouterInterface
OS::Neutron::RouterGateway
OS::Neutron::Subnet

*** Neutron Load balancer:
OS::Neutron::HealthMonitor
OS::Neutron::Pool
OS::Neutron::LoadBalancer
Heat Icehouse: New resources

- OS::Heat::CloudConfig
- OS::Heat::MultipartMime
- OS::Heat::SoftwareConfig
- OS::Heat::SoftwareDeployment
- OS::Heat::StructuredConfig
- OS::Heat::StructuredDeployment
- OS::Heat::RandomString
- OS::Heat::ResourceGroup
- OS::Heat::AutoScalingGroup
- OS::Heat::ScalingPolicy
- OS::Neutron::SecurityGroup
- OS::Neutron::MeteringLabel
- OS::Neutron::MeteringRule
- OS::Neutron::ProviderNet
- OS::Neutron::NetworkGateway
- OS::Neutron::PoolMember
- OS::Nova::KeyPair
- OS::Nova::FloatingIP
- OS::Nova::FloatingIPAssociation
- OS::Trove::Instance
Environment

• An environment file is a YAML file with a parameters section containing values for parameters declared in your template

```bash
$ heat stack-create -f mytemplate.yml -e local.yaml stack-test
```
Contextualization

• user_data

resources:
  my_instance:
    type: OS::Nova::Server
    properties:
      # general properties ...
      user_data:
        str_replace:
          template: |
          #!/bin/bash
          echo "Hello world"
          echo "Setting MySQL root password"
          mysqladmin -u root password $db_rootpassword
          # do more things ...
          params:
            $db_rootpassword: { get_param: DBRootPassword }
Wait condition

- Most resources (like OS::Nova::Server) transition state automatically (CREATE_IN_PROGRESS -> CREATE_COMPLETE)

- A wait condition (AWS::CloudFormation::WaitCondition) is a resource that only transitions upon receiving an external signal.

- This permits us to make parts of our Heat template wait for an external event before they are created:

```yaml
wordpress_server:
  type: "OS::Nova::Server"
  depends_on: mysql_wait_condition
```
Nested stack

- Heat can treat a template as a resource primitive.
- Allows you to reuse complex configurations in other stacks.
- Create a library of components appropriate to your environment

```yaml
my_server:
  # Here is our nested stack.
  type: wp-nested-server.yaml
  properties:
```
**Autoscaling**

*AutoScalingGroup* is a resource that can create and destroy other resources on demand.

*ScalingPolicy* defines an action that Heat can take on an AutoScalingGroup.

*Ceilometer Alarms*. An alarm allows Ceilometer to POST to a URL when a metric matches certain values.
Autoscaling

"CPUAlarmHigh": {
    "Type": "OS::Metering::Alarm",
    "Properties": {
        "meter_name": "cpu_util", threshold: "75",
        "evaluation_periods": "5", "period": "60",
        "statistic": "avg", "comparison_operator": "gt",
        "description": "Scale-up if CPU > 75% for 300s",
        "alarm_actions": ["ScaleUpPolicy", "AlarmUrl"...],
        "matching_metadata": {
            "metadata.user_metadata.server_group": "MyWebServerGroup"
        }
    }
}
Autoscaling: use-case

Heat stack creates a variable number of Wordpress servers depending on CPU load, and manages a load balancer to provide access to this service.

"OS::Neutron::LoadBalancer"

"OS::Heat::AutoScalingGroup"
Application Software Configuration

New heat software config and deployment resources

OS::Heat::SoftwareConfig

OS::Heat::SoftwareDeployment

Integrating configuration tools

Templates are well integrated with Puppet, Chef
Multi-region support

- 1. Region Silos (current implementation)
- 2. Stack based Multi region
- 3. Master Orchestrator
- 4. Multi-region resources
- 5. Global Orchestrator
References

- http://docs.openstack.org/developer/heat/template_guide/
- https://wiki.openstack.org/wiki/Heat
- http://docs.openstack.org/developer/heat/template_guide/hot_spec.html
- https://wiki.openstack.org/wiki/Heat/Environments