Deploying an Oracle Database Server on Linux on z Systems using OpenStack Heat by Converting an IBM Cloud Orchestrator 2.4 Virtual System Pattern

October 2015

IBM Washington Systems Center zGrowth Team

Mike Bonett
Executive I/T Specialist

Special Notices

Trademarks

Acknowledgements

Many thanks to the following people for providing input and/or reviewing this paper:

- Carol Davis, IBM ATG Team
- Art Eisenhour, IBM zGrowth Team
- John Goodyear, IBM zGrowth Team

Introduction

The **OpenStack Heat Orchestration Engine** (referred to as Heat in the rest of this document) is a component of OpenStack which provides the creation and deployment of a "pattern" of operating system images and associated software onto supported hypervisors. The term "pattern" is a construct around the deployment of one or more operating system images with installed application/middleware software in a connected topology. This moves beyond basic image provisioning to provide greater deployment options.

A **Heat Orchestration Template**, also referred to as a HOT or Heat template, is a structured text description of the pattern provisioning steps to be executed. A Heat template structure:

- Defines the template parameters to be set at runtime
- Defines the resources to be deployed, and their proper sequence. The resources include:
    - Deployed instances (servers)
    - Software components (software configurations)
    - Network configurations for the deployed instances
    - Storage for the deployed instance (beyond the image used to create the instance)
- Define the outputs to be presented at the completion of the template deployment (such as the IP address of a created instance, or the URL of an application on an instance)

A complete reference for the Heat template structure and contents is documented on the OpenStack website at http://docs.openstack.org/developer/heat .

IBM Cloud Manager with OpenStack and **IBM Cloud Orchestrator** both use OpenStack to deploy Linux instances under z/VM, and provide support for using Heat templates. IBM Cloud Manager with OpenStack provides basic cloud provisioning and self-service capabilities. IBM Cloud Orchestrator provides advanced capabilities, such as extensive self-service catalog functions and integrating of instance provisioning and lifecycle management with external business and operational processes via IBM Business Process Manager.

The products provide these options for deploying Heat templates:

- IBM Cloud Manager with OpenStack deploys them from its Dashboard (OpenStack Horizon component) interface.
- IBM Cloud Orchestrator deploys them from its Dashboard (OpenStack Horizon) interface or its self-service user interface. Within the self-service interface the template can be deployed by an administrator, or by a user as part of a self-service catalog offering.

IBM Cloud Orchestrator version 2.4 also provides a pattern engine based on IBM Workload Deployer (IWD). This pattern engine allows the creation of "virtual systems patterns". The white paper *Creating and Deploying an Oracle Database Server Pattern on Linux on z Systems using IBM Cloud Orchestrator*, available on IBM TechDocs at http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102535, uses an IBM Cloud Orchestrator 2.4 virtual system pattern to deploy an Oracle 11gr2 database server. IBM Cloud Orchestrator 2.5 no longer supports IWD patterns. Any existing IWD patterns but be converted to Heat templates for use by IBM Cloud Orchestrator 2.5.

This paper will cover an example of using IBM Cloud Manager with OpenStack (4.2 or 4.3) or IBM Cloud Orchestrator (2.4.0.1 or 2.5) to deploy an Oracle version 12c database via a Heat template. It will take the IWD virtual system pattern used in the above mentioned white paper and provide an example of the process to convert the virtual system pattern to a Heat template, and a set of Chef recipes invoked by the template, to deploy an Oracle 12c database server.

The major topics that are in this paper covered are:

- Preparing the Linux source image
- Mapping the virtual system pattern script packages to Chef cookbooks and recipes
- Including recipes within a Heat template
- Deploying the Heat template from the administrative interface of IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator
- Deploying the Heat template from the IBM Cloud Orchestrator self-service interface

Preparing the Linux operating system image

The same Linux under z/VM image used for the IBM Cloud Orchestrator pattern was used with the Heat template. Patterns prepared for IBM Cloud Orchestrator can also be used with IBM Cloud Manager with OpenStack. Both require:

- the **xcatconf4z** script provided by the z/VM 6.3 Extreme Cloud Administration Toolkit (xCAT)
- The **cloud-init** package, open source software used for passing information to deployed instances from OpenStack (including Heat templates) to carry out image customization actions after deployment.

Linux guests used for creating images must adhere to a specific configuration, which is fully documented (along with the required xcatconf4z and cloud-init setup) in the *Enabling z/VM for OpenStack* manual. The Linux images are created by capturing an existing Linux guest that has the specific configuration, and storing it as an OpenStack image.

For this pattern, the z/VM environment where new instances were being deployed only had 3390-9 ECKD DASD volumes available. To support the installation of the Oracle database, the image was created with the following configuration:

- A 6GB root volume minidisk, as this is the largest GB multiple that will fit on a single 3390-9.
- Two partitions on the volume :
    - Partition 1: a non-LVM partition, mounted at /.
    - Partition 2: a LVM physical volume within a volume group.
        - Three logical volumes defined and associated with the following mount points:
            - /opt (this is where the Oracle code and database will be installed
            - /var
            - /tmp

Having the second partition defined as a LVM allows the logical volume storage to be expanded, by adding additional LVM physical volumes to the volume group. This will be described in the next section.

Red Hat Enterprise Linux 6.5 was installed on the image, using the "basic server" installation option. The image was captured by XCAT on z/VM and then imported into OpenStack for use by either IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator.

Mapping IBM Cloud Orchestrator 2.4 script packages to Chef cookbooks

The IBM Cloud Orchestrator virtual system pattern used in the *Creating and Deploying an Oracle Database Server Pattern on Linux on z Systems using IBM Cloud Orchestrator* white paper must be converted to a Chef cookbook and appropriate recipes. In IBM Cloud Orchestrator 2.4, a virtual system pattern consists of one or more operating system images, each with a set of associated script packages. The script packages are executed after the image is deployed, to perform further image customization and/or software installation actions.

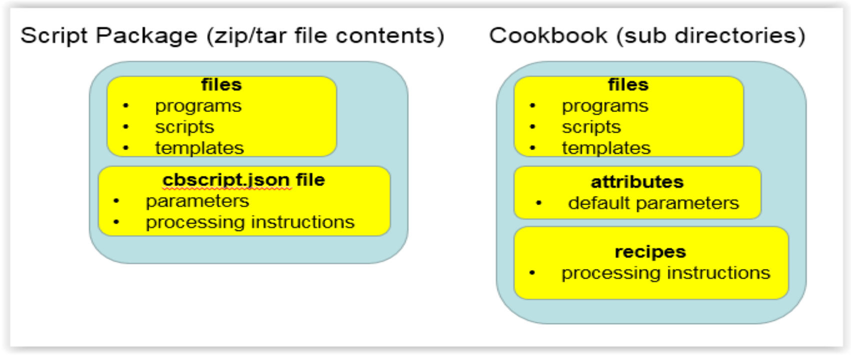
A script package is a .zip or .tar file that contains:

- One or more files that are copied to the deployed instances. These files are generally:
  - Programs (shell scripts or any programming language supported by the image operating system) that are to be executed.
  - Configuration or template files used by the programs that are executed when the image(s) are deployed.
- A **cbscript.json** file which describes
  - Script package attributes (name, release level, etc.)
  - The initial command to be executed on the instance (for Linux usually a shell script)
  - The parameters required by the shell script. These are passed in from the virtual system pattern definition or when the pattern is executed. Default values can be set in the cbscript.json file.

A Chef cookbook is a directory structure. The top level directory is the cookbook name. There are multiple subdirectories; for this example focuses on three of them:

- **files** – contains files used by the recipes within the cookbook. These are generally:
  - Programs (shell scripts or any programming language supported by the image operating system) that are to be executed
  - Configuration or template files used by the programs that are executed when the image(s) are deployed.
- **recipes** – contains .rb files that define a recipe. The name of the .rb file is the name of the recipe. The contents describe the actions to be taken for a specific recipe.
- **attributes** – contains .rb files to define parameters used in the recipes. The default.rb file defines default values for the parameters. The actual values used by a recipe can be overridden in various ways; for this paper the values will be passed in from the chef client executing the recipe.

The following picture shows the script package and cookbook structures side-by-side:



From the picture, a basic mapping from the script package components to a Chef cookbook can be seen:

- The files in the script package can be moved to the cookbook files subdirectory.
- The parameters defined in the cbscript.json file can be defined in the cookbook attributes directory.
- The cbscript.json processing instructions (primarily the initial program/script to run and its associated parameters) can be defined in a recipe.

Since recipes support multiple types of processing actions, multiple script packages can be combined into a single recipe. For example, the IBM Cloud Orchestrator 2.4 Oracle virtual system pattern documented in the above referenced white papers contained the following script packages:

1. **Add ECKD volume**: add an additional minidisk volume of the requested cylinder size, and activates it in the Linux guest. The script uses SMAPI calls to perform those functions. 2 3390-9 volumes were added to provide enough storage. Optional if the source image is built with enough storage.
2. **Format volume as an LVM physical volume and add to volume group**: place the added volume online, create a single partition formatted as a LVM physical volume, and add it to a volume group. Required if “Add ECKD volume” script package was used.
3. **Expand mount point**: expand the space under a logical volume mount point by adding the requested storage from the volume group to the logical volume. The pattern added storage to the logical volumes mounted at /opt, /var, and /tmp to meet the Oracle installation requirements. Optional if those directories already have enough space.
4. **Add swap space**: checks to determine if the image has enough swap space defined; if not, adds additional space to meet the Oracle installation requirement.
5. **Add yum repository**: Define a red hat yast update manager (yum) repository. Optional if the source image is built with a working yum repository defined as a based configuration.
6. **Install required updates**: This script uses the yum repository to verify the required Linux packages as documented in the product installation guide, and installs/upgrades packages as needed.

7. **Install Oracle Database Server**: installs the Oracle software by performing the following steps:
    - Updates the Linux configuration files with values required by/recommended for running an Oracle database environment.
    - Defines the Linux user IDs and groups required by the Oracle database.
    - Executes a silent Oracle installation by building dynamically a response file, using some values specific to the instance (such as the instance hostname).
    - Creates a default database (with its Oracle SID equal to the hostname) and listener, and starts both.
    - Generates a Linux shell script that is copied to /etc/init.d. The script can be used to manage the Oracle database to be run as a service, and start the database automatically whenever the instance is booted.

The following table shows how each script package was mapped to a Chef cookbook and recipe:

Source Script Package	**Number of times called in ICO pattern**	**Target Chef Cookbook (recipe)**	**Number of times called in Heat template**
Add ECKD Volumes	2	Add ECKD (add ECKD)	1
Format volume as an LVM physical volume and add to volume group	2	Add ECKD (add ECKD)	1
Expand mount point	3	Add ECKD (expand mount point)	3
Add swap space	1	Oracle install (create swapfile)	1
Add yum repository	1	Oracle install (add yum repository)	1
Install required updates	1	Oracle install (install Oracle server)	1
Install Oracle server	1	Oracle install (install Oracle server)	1

The additional capabilities Chef provides allows for the following improvements:

- 7 script packages were consolidated into 2 Chef cookbooks and 5 recipes.
- The **Add ECKD** recipe was modified to add multiple volumes in a single invocation.
- The **Oracle Install** cookbook recipes were updated to support Oracle server version 12c.

Creating the pattern in a Heat template that uses the Chef components

A heat template can be coded in any text editor. It uses the **yet another markup language** (yaml or yml) syntax; an editor which recognizes that syntax, such as Notepad++, is most useful to detect syntax problems, and was used in this example.

> **Note:** IBM UrbanCode Deploy with Patterns is a product that can be used to graphically generate Heat templates and directly deploy them into either IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator.

The Heat template contains these key definitions:

- A image resource:

```
 oracle_server:
    type: OS::Nova::Server
    properties:
      key_name: { get_param: key_name }
      image: { get_param: image_id }
      flavor: { get_param: flavor }
      user_data_format: RAW
      user_data: { get_resource: install_components }
```

  The image, flavor, and key name properties identify (via passed parameters) the image to be used, the OpenStack "flavor" (memory, CPU, disk allocation) for the deployed instance, and a ssh key to access the instance once deployed. The user_data property points to an **install_components** resource.

- The **install_components** resource identifies the software components to install and the order of installation:

```
install_components:
    type: OS::Heat::MultipartMime
    properties:
      parts:
      - config: { get_resource: dnslookuptohosts }
      - config: { get_resource: chef_client_install }
      - config: { get_resource: add_eckd_volumes }
      - config: { get_resource: increase_opt_space }
      - config: { get_resource: increase_var_space }
      - config: { get_resource: increase_tmp_space }
      - config: { get_resource: oracle_12c_install }
```

  install_components is a MultpartMime template object and passes the software components in properties:parts:config to the image, in a format that cloud-init can understand. Each config object identifies a specific software component resource (SoftwareConfig template objects) defined in the template:

  - **dns_lookuptohosts** and **chef_client_install** software components will invoke scripts to validate/set a resolvable hostname for the image, and install the chef client.

- The other software components take the input parameters passed to the heat template at runtime and invoke the chef client to use the chef cookbooks to carry out the software activities. Each resource generates a json file defining the parameters to use and the recipe to run, and then invokes the chef client to install the software. Here is an example of the oracle_12_install software component:

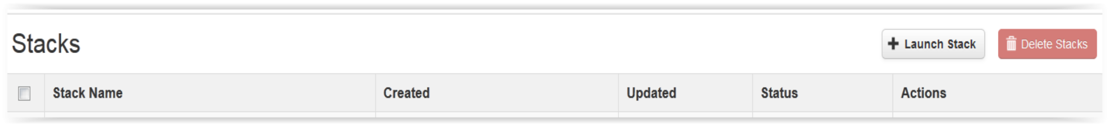
```
oracle_12c_install:
   type: OS::Heat::SoftwareConfig
   properties:
     group: script
     config:
       str_replace:
         template: |
            #!/bin/sh
            echo "starting $0 ..."
            rm -f /tmp/tmp/oracle-12c.json
            cat << EOF >> /tmp/oracle_12c.json
            {
             "swapfile":
               {
                "name": "$swname",
                "sizekb": "$swsize"
               },
             "yumrepofile":
               {
             "url": "$yumrpf"
               },
             "orcl":
               {
                "nfsmount": "$oranfs",
                "installcode_dir": "$orainst",
                "base_dir": "$orabase",
                "home_subdir": "$orahome",
                "inventory_dir": "$orainv",
                "defaultpw": "$dfpw"
               },
             "run_list" : [
                  "recipe[oracle-12c-linux-z::create_swapfile]",
                  "recipe[oracle-12c-linux-z::add_yum_repo_file]",
                  "recipe[oracle-12c-linux-z::install_oracle_12c_s390x_redhat]"
                 ]
            }
            EOF
            echo "json generated:"
            echo "----------------------"
            echo `cat /tmp/oracle_12c.json`
            echo "----------------------"
            #
            echo "invoking chef to install oracle 12c database..."
            chef-client -j /tmp/oracle_12c.json -l debug
            echo "chef-client ended."

         params:
           $swname: { get_param: swapfile_name }
           $swsize: { get_param: swapfile_sizekb }
           $yumrpf: { get_param: yumrepofile_url}
           $oranfs: { get_param: orcl_nfsmount }
           $orainst: { get_param: orcl_installcode_dir}
           $orabase: { get_param: orcl_base_dir }
           $orahome: { get_param: orcl_home_subdir }
           $orainv: { get_param: orcl_inventory_subdir}
           $dfpw:  { get_param: orcl_defaultpw }
```

This sample SoftwareConfig component generates a json file containing the recipes that will be invoked from the oracle-12c-linux-z cookbook and the parameters for those recipes. It then invokes the chef client to execute the json file and perform the software installation.

Deploying the pattern from the IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator dashboards (OpenStack Horizon)

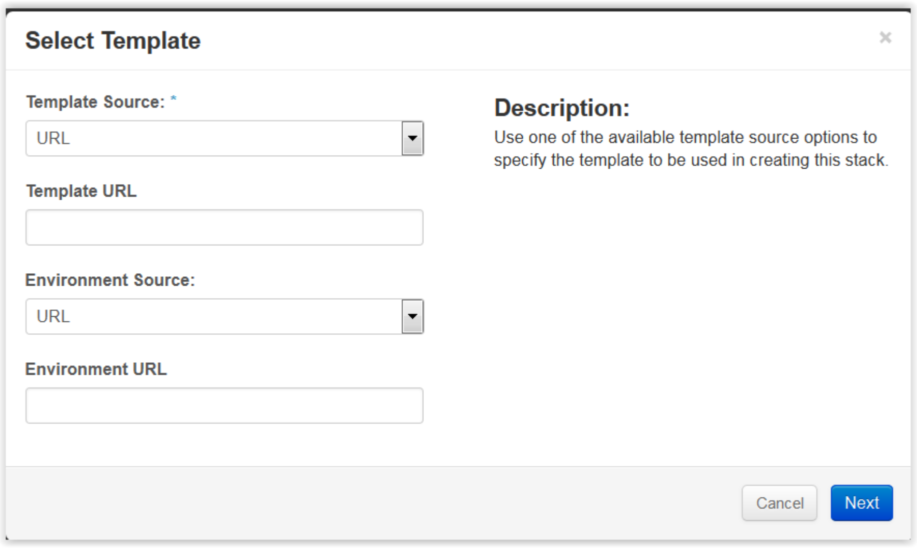
After the Heat template is created, either IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator can be used to deploy it. Both products have an administrative dashboard based on the OpenStack Horizon user interface. To deploy the template from this dashboard:

- Select **Project->Orchestration->Stacks**, to display the **Stacks** menu. This will have a heading with a list of any deployed stacks underneath it:



	Stack Name	Created	Updated	Status	Actions

- Click **Launch Stack** to display a menu to select the Heat template to be used:



Template Source: * URL

Template URL

Environment Source: URL

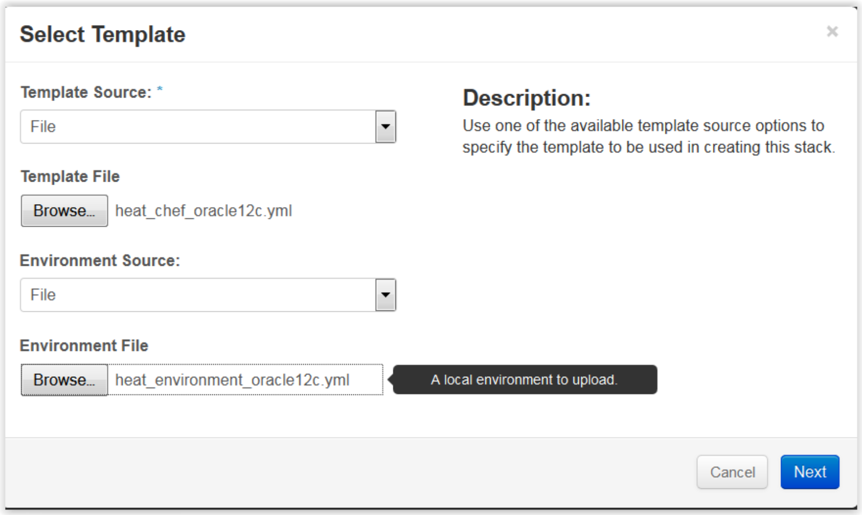
Environment URL

Description:
Use one of the available template source options to specify the template to be used in creating this stack.

- The Template Source drop down allows the Heat template file to be retrieved from three sources::
  - URL: Retrieved from a HTTP or Web Server
  - File: Uploaded from the workstation where the dashboard interface is being used
  - Direct Input: Typed or pasted directly into an input box

- The **Environment Source** drop down allows an environment file to be retrieved in the same manner as the Heat template file. An Environment file contains name-value pairs that map to the input parameters of the heat template; this allows the parameter values to be entered without having to manually type them in.

  For example, if the source of both the template file and the environment file was the workstation using the dashboard interface, the Select Template menu contents would be similar to the following:



Select Template

Template Source: * File

Template File: heat_chef_oracle12c.yml

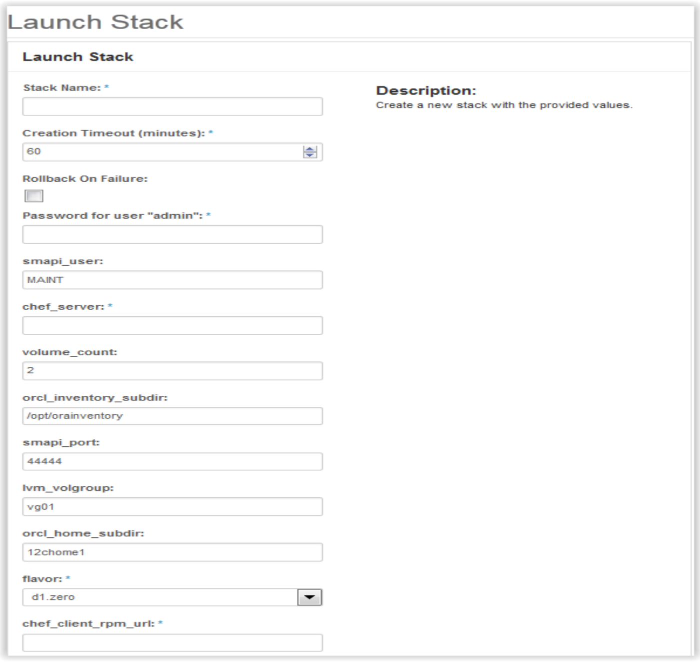
Environment Source: File

Environment File: heat_environment_oracle12c.yml

A local environment to upload.

Description:
Use one of the available template source options to specify the template to be used in creating this stack.

- Click **Next** to bring up the **Launch Stack** menu containing the template parameters. Some parameters will be prefilled based on defaults defined in the heat template, or values provided in the environment file. All values can be overridden if necessary:



Launch Stack

Launch Stack

Stack Name: *

Creation Timeout (minutes): * 60

Rollback On Failure:

Password for user "admin": *

smapi_user: MAINT

chef_server: *

volume_count: 2

orcl_inventory_subdir: /opt/orainventory

smapi_port: 44444

lvm_volgroup: vg01

orcl_home_subdir: 12chome1

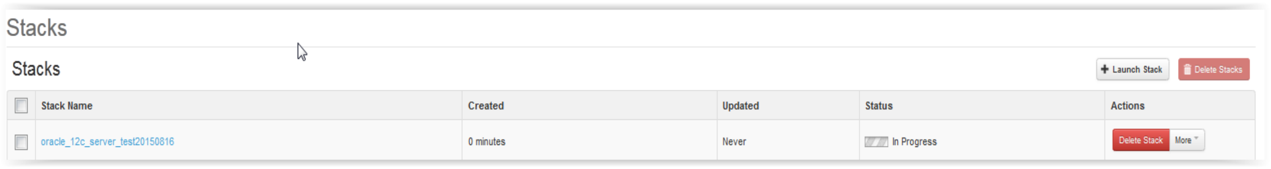
flavor: * d1.zero

chef_client_rpm_url: *

Description:
Create a new stack with the provided values.

Note: the above is just a partial listing of the parameters menu.

- At the bottom of the Launch Stack menu click **Launch**. The Main Stacks menu will display with a message that the template has been successfully launched and the deployment will have an initial status of **in progress**:

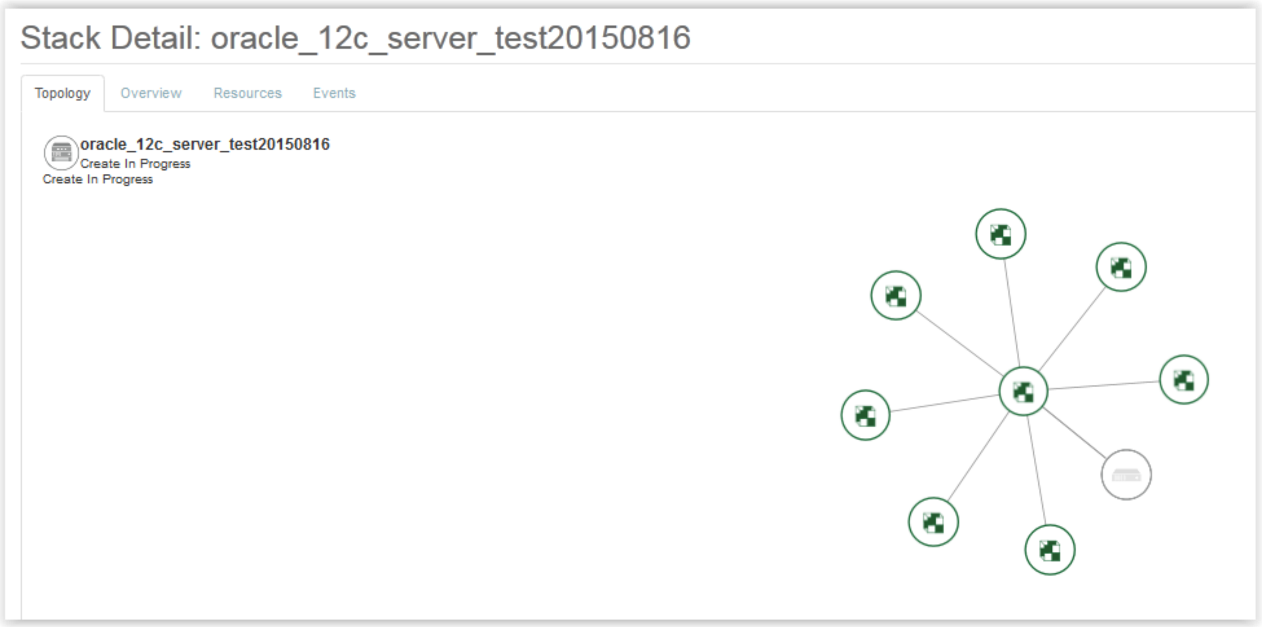


Stacks

Stacks

	Stack Name	Created	Updated	Status	Actions
☐	oracle_12c_server_test20150816	0 minutes	Never	In Progress	Delete Stack More

- Clicking the Stack Name will display a menu providing more details about the orchestration status:

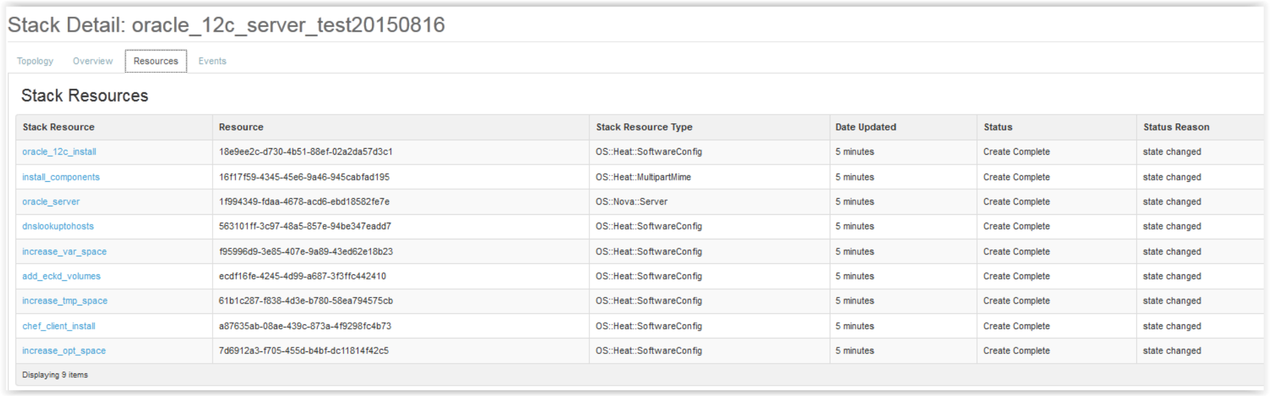


Stack Detail: oracle_12c_server_test20150816

oracle_12c_server_test20150816
Create In Progress
Create In Progress

- The graphical view on the right shows the status of the components. Moving the cursor over them will provide more details. The greyed out icon represents the server, which is being built. The icon in the center represents the MultipartMime template resource, and the icons connected to it represent the individual software resource components. They are solid since their build is complete, but they have not yet been deployed since the server is not yet completely built.

- The **Resources** tab shows similar template information as the topology view, but in tabular format:



Stack Resources

Stack Resource	Resource	Stack Resource Type	Date Updated	Status	Status Reason
oracle_12c_install	18e9ee2c-d730-4b51-88ef-02a2da57d3c1	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed
install_components	16f17f59-4345-45e6-9a46-945cabfad195	OS::Heat::MultipartMime	5 minutes	Create Complete	state changed
oracle_server	1f994349-fdaa-4678-acd6-ebd18582fe7e	OS::Nova::Server	5 minutes	Create Complete	state changed
dnslookuptohosts	563101ff-3c97-48a5-857e-94be347eadd7	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed
increase_var_space	f95996d9-3e85-407e-9a89-43ed62e18b23	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed
add_eckd_volumes	ecdf16fe-4245-4d99-a687-3f3ffc442410	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed
increase_tmp_space	61b1c287-f838-4d3e-b780-58ea794575cb	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed
chef_client_install	a87635ab-08ae-439c-873a-4f9298fc4b73	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed
increase_opt_space	7d6912a3-f705-455d-b4bf-dc11814f42c5	OS::Heat::SoftwareConfig	5 minutes	Create Complete	state changed

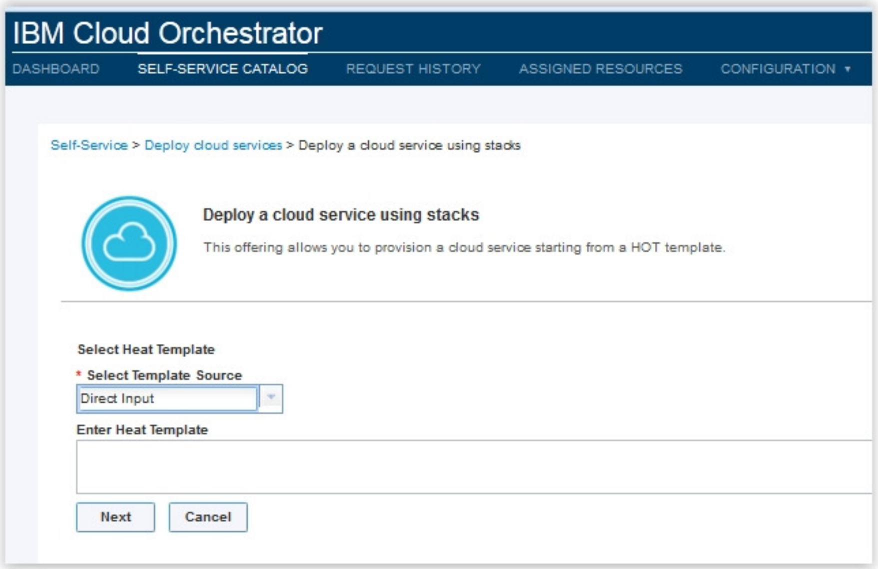
Displaying 9 items

Clicking on any of the component names will provide more details.

Deploying a pattern from the IBM Cloud Orchestrator Self-Service Interface

In IBM Cloud Orchestrator, Heat template patterns can be deployed from its Self-Service interface. Non-administrative users are more likely to deploy the pattern from the Self-Service Catalog section of the Self-service interface. Access to the pattern can be limited as desired - to particular users, projects (groups of users) or domains:

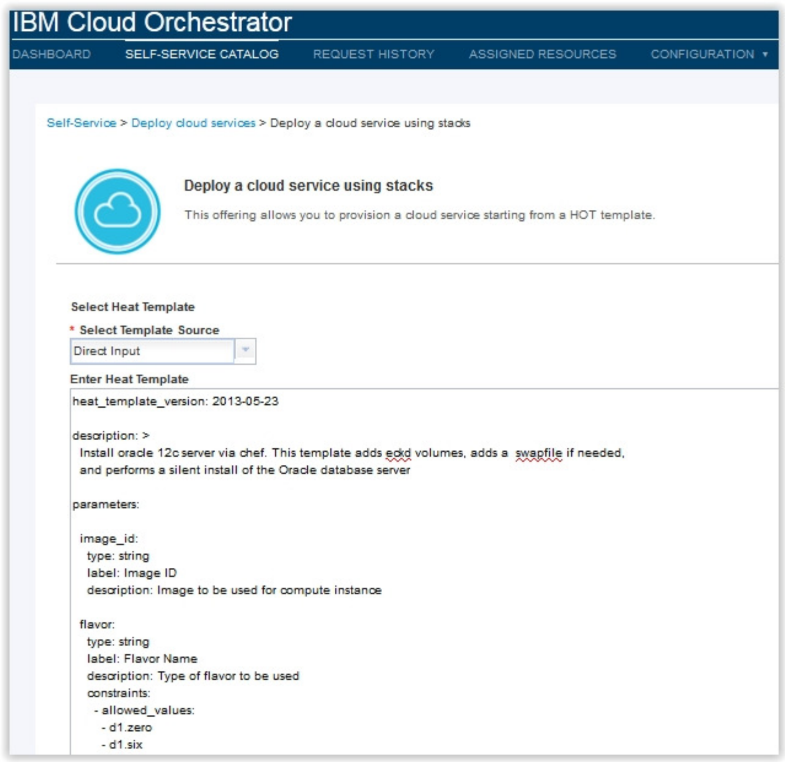
- After logging into the interface, select **Self-Service->Deploy Cloud Services->Deploy Cloud Services using Stacks**:



This offering allows you to provision a cloud service starting from a HOT template.

Select Template Source: Direct Input

- In this example the Heat template text is cut and pasted into the input box on this menu:



Self-Service > Deploy cloud services > Deploy a cloud service using stacks

Deploy a cloud service using stacks

This offering allows you to provision a cloud service starting from a HOT template.

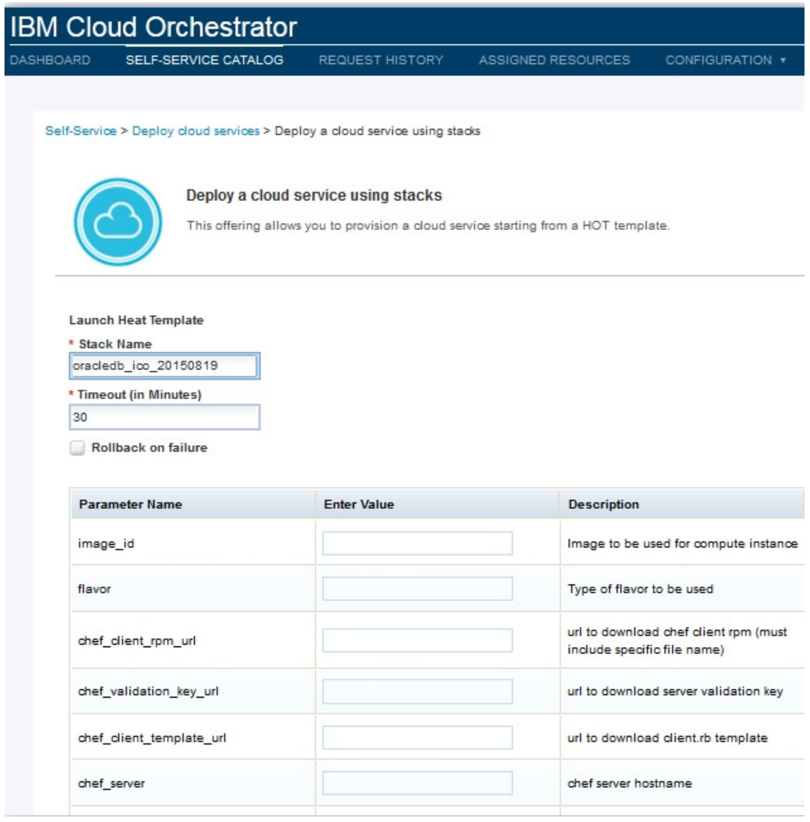
Select Heat Template

* Select Template Source: Direct Input

Enter Heat Template

```
heat_template_version: 2013-05-23

description: >
 Install oracle 12c server via chef. This template adds eckd volumes, adds a  swapfile if needed,
 and performs a silent install of the Oracle database server

parameters:

 image_id:
   type: string
   label: Image ID
   description: Image to be used for compute instance

 flavor:
   type: string
   label: Flavor Name
   description: Type of flavor to be used
   constraints:
    - allowed_values:
     - d1.zero
     - d1.six
```

- Clicking **Next** then shows the variables associated with the template:



Self-Service > Deploy cloud services > Deploy a cloud service using stacks

Deploy a cloud service using stacks

This offering allows you to provision a cloud service starting from a HOT template.

Launch Heat Template

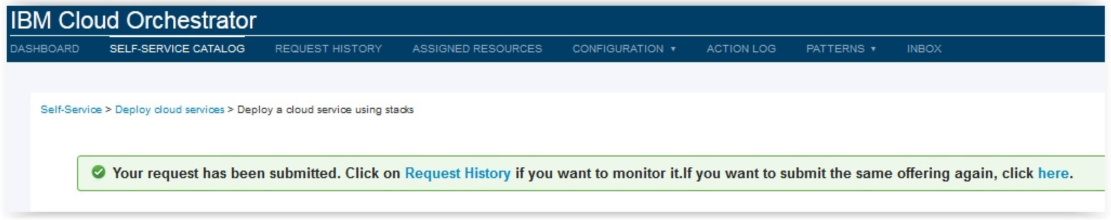
* Stack Name: oracledb_ico_20150819

* Timeout (in Minutes): 30

Rollback on failure

Parameter Name	Enter Value	Description
image_id		Image to be used for compute instance
flavor		Type of flavor to be used
chef_client_rpm_url		url to download chef client rpm (must include specific file name)
chef_validation_key_url		url to download server validation key
chef_client_template_url		url to download client.rb template
chef_server		chef server hostname

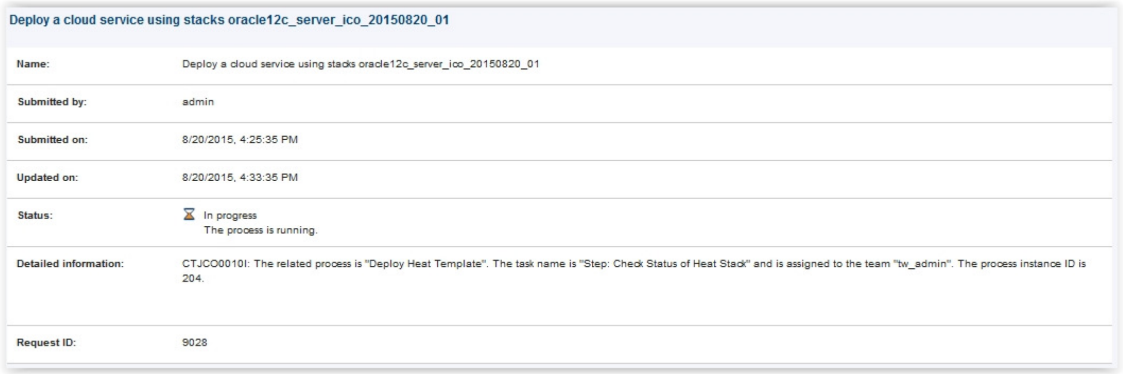
- After the values are entered or overridden, clicking Submit to launch the template. Successful mapping of the input values to the template and submission of the request to the OpenStack Heat engine will result in the following message:



Self-Service > Deploy cloud services > Deploy a cloud service using stacks

Your request has been submitted. Click on Request History if you want to monitor it.If you want to submit the same offering again, click here.

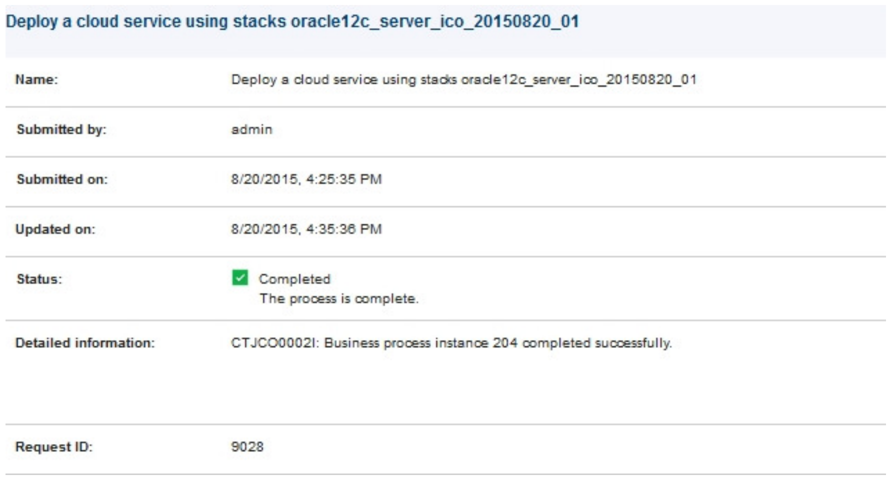
- Click **Request History** to monitor the status:



Deploy a cloud service using stacks oracle12c_server_ico_20150820_01

Name:	Deploy a cloud service using stacks oracle12c_server_ico_20150820_01
Submitted by:	admin
Submitted on:	8/20/2015, 4:25:35 PM
Updated on:	8/20/2015, 4:33:35 PM
Status:	In progress The process is running.
Detailed information:	CTJCO0010I: The related process is "Deploy Heat Template". The task name is "Step: Check Status of Heat Stack" and is assigned to the team "tw_admin". The process instance ID is 204.
Request ID:	9028

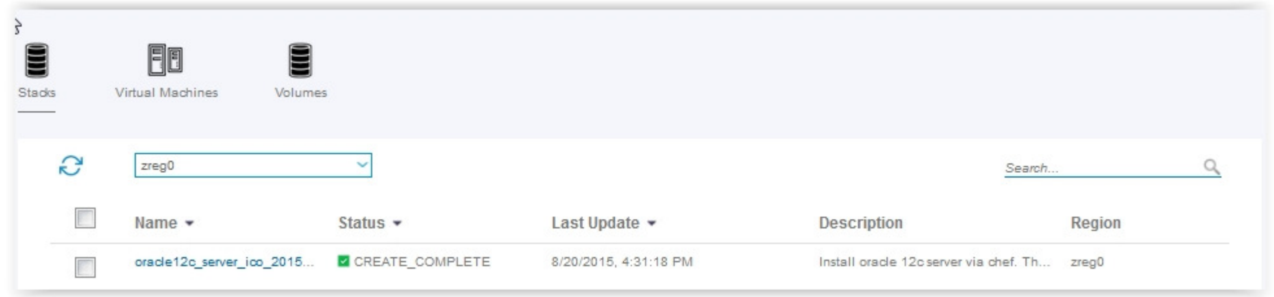
- The Status value will change when the deployment completes:



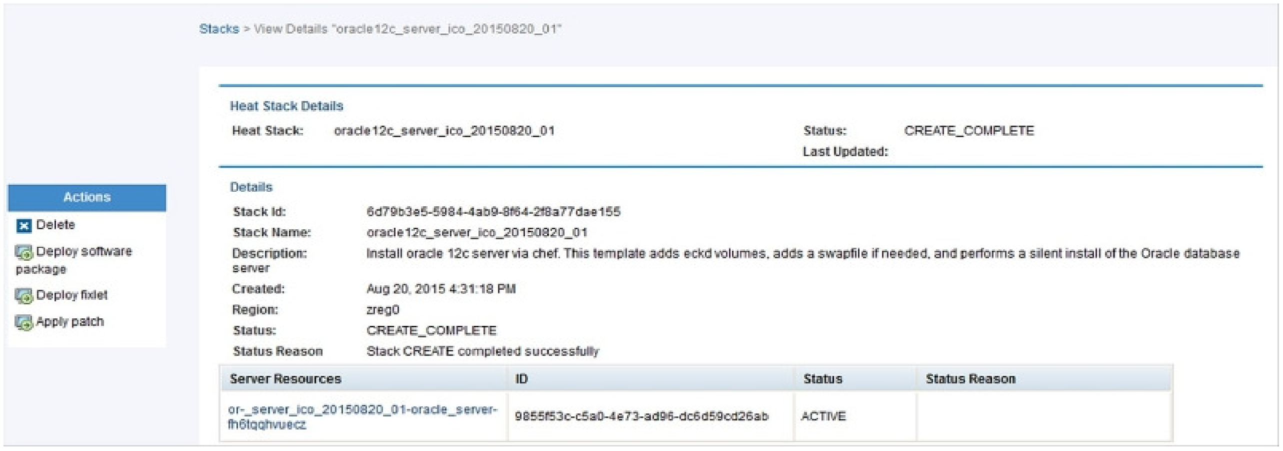
Deploy a cloud service using stacks oracle12c_server_ico_20150820_01

Name:	Deploy a cloud service using stacks oracle12c_server_ico_20150820_01
Submitted by:	admin
Submitted on:	8/20/2015, 4:25:35 PM
Updated on:	8/20/2015, 4:35:36 PM
Status:	✓ Completed The process is complete.
Detailed information:	CTJCO0002I: Business process instance 204 completed successfully.
Request ID:	9028

- On the main menu, under **Assigned Resources**, the deployment instance details can be seen and its state managed:



	Name	Status	Last Update	Description	Region
☐	oracle12c_server_ico_2015...	✓ CREATE_COMPLETE	8/20/2015, 4:31:18 PM	Install oracle 12c server via chef. Th...	zreg0



Stacks > View Details "oracle12c_server_ico_20150820_01"

Heat Stack Details

Heat Stack: oracle12c_server_ico_20150820_01

Status: CREATE_COMPLETE

Last Updated:

Details

Stack Id:	6d79b3e5-5984-4ab9-8f64-2f8a77dae155
Stack Name:	oracle12c_server_ico_20150820_01
Description:	Install oracle 12c server via chef. This template adds eckd volumes, adds a swapfile if needed, and performs a silent install of the Oracle database server
Created:	Aug 20, 2015 4:31:18 PM
Region:	zreg0
Status:	CREATE_COMPLETE
Status Reason	Stack CREATE completed successfully

Server Resources	ID	Status	Status Reason
or-_server_ico_20150820_01-oracle_server-fh6tqqhvuecz	9855f53c-c5a0-4e73-ad96-dc6d59cd26ab	ACTIVE	

At this point the instance deployment is complete and the Oracle database server is ready for use. Detailed information on the Heat template processing and chef recipe execution will be available in the /var/log/cloud-init-output.log on the instance.

Summary and further information

This example is only one way to create an Oracle Heat template and deploy it; each environment has their own unique standards and requirements that must be taken into consideration. For example:

- An environment might be using a different automated software configuration standard than Chef. The Heat template can still be used, instead of calling Chef it can be customized to invoke any function callable from the Linux command line. Heat provides built-in support for some software configurations standards (such as Puppet).
- Some of the items implemented in the Heat template can be implemented instead as part of the source image. For example, if storage is not constrained, a root volume minidisk large enough to support the Oracle installation can be used to create the source image. In addition, the source image can be preconfigured with a working yum repository, so that only an update has to be invoked at deployment time.

- The Heat template can be extended to include application deployment. For example:

    - Adding script packages to install database tables into the database.

- Adding an additional image to be deployed as a web application server (the number of which can be automatically scaled) with script packages to install the web server and an application, and have the application connect to tables installed in the database.

Ultimately, the result is a usable, working Oracle database pattern, which can be deployed by both IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator, in a way to support the concept of a self-service cloud using Linux on z Systems.

For more information on setting up and using IBM Cloud Manager with OpenStack or IBM Cloud Orchestrator with z/VM, see the following resources:

- Enabling z/VM for OpenStack: http://www.vm.ibm.com/sysman/openstk.html
- IBM Cloud Manager with OpenStack Knowledge Center:
  - https://www.ibm.com/support/knowledgecenter/SST55W/welcome
- IBM Cloud Manager with OpenStack - z/VM Integration Considerations:
  - http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102549
- IBM Cloud Orchestrator Knowledge Center:
  - http://www.ibm.com/support/knowledgecenter/SS4KMC/welcome
- IBM Cloud Orchestrator - z/VM Integration Considerations:
  http://www.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/WP102494