Orchestrierung einer Application Server Cloud mit OpenStack

Detlef Drewanz
Master Principal Sales Consultant
Systems Sales Consulting

Supported by Michael Bräuer
with WLST Scripting
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Disclaimer

• This is not an Oracle WebLogic Presentation!

• This is about Software Deployment with OpenStack
  – The used Software as Example is Oracle WebLogic Server
  – The Test Platform for this Presentation was OpenStack on Oracle Solaris 11

• It is a deep dive technical presentation
Agenda

1. Motivation
2. Oracle WebLogic Server Concepts
3. OpenStack Concepts
4. Plain Instance Orchestration
5. OpenStack Orchestration with Heat Templates
6. Software Configuration with OpenStack Metadata
Motivation for an Application Server Cloud

• Need a dynamic Platform for elastic Workload with high Memory and CPU Demand

• Segregation in a Cloud Architecture
  – Isolated Production Setups
  – Secured Penetration Testing Environments

• Self-Service supports Rapid Deployment for Dev, Test and Prod

• Unified Application Deployment with common API
  – Enable Multicloud Setup
Oracle WebLogic Server Concepts

Domain

- WebLogic Console
  - Admin Server
  - Node Manager

Application Deployment
- Server Management

Clusters:
- Cluster 1
- Cluster 2
- Cluster 3

Managed Servers
- Start/Stop
- Node Manager

Application Access
Agenda

1. Motivation
2. Oracle WebLogic Server Concepts
3. OpenStack Concepts
4. Plain Instance Orchestration
5. OpenStack Orchestration with Heat Templates
6. Software Configuration with OpenStack Metadata
What is OpenStack?

• Open Source Cloud Software
  – Generic solution for IaaS, PaaS, and SaaS
• Originally created by NASA and Rackspace
  – Supported by many Vendors
• Combines Compute, Network and Storage Resources
  – Self-service Dashboard
  – Mostly written in Python
  – Services exposed through RESTful APIs
• More info at http://www.openstack.org
OpenStack Logical Architecture

A Short Recap
Oracle OpenStack for Oracle Solaris
Havana (11.2), Juno (11.2 SRU10), Kilo (11.3 SRU9)
General (OpenStack) Converged Cloud Architecture

- Public Network
  - Network Node

- Cloud Network
  - Compute Node
  - Compute Node
  - Compute Node
  - Compute Node
  - Compute Node

- Storage Network
  - Shared Storage

- OpenStack Controller
- Utilities Controller
Moving a WebLogic Domain into an OpenStack Private Cloud
Agenda

1. Motivation
2. Oracle WebLogic Server Concepts
3. OpenStack Concepts
4. Plain Instance Orchestration
5. OpenStack Orchestration with Heat Templates
6. Software Configuration with OpenStack Metadata
Orchestration

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

• Automate the Deployment and Initial Configuration
  – Compose Stacks out of multiple Applications, running in different VMs
  – Allocate Infrastructure Resources
  – Create VMs
  – Install Applications

• Orchestration is not Configuration Management
Plain creation of a WebLogic Server Domain

1. Install the Machine (VM)
   A. Allocate a Storage Volume
   B. Get an OS Installation Image
   C. Allocate a Network Port
   D. Get a Network Address
   E. Install Machine with A, B, C, D, E
   F. Connect the Machine to C

2. Install the WebLogic Software

3. Configure WebLogic
   A. Domain and Cluster
   B. Node Manager
   C. Admin Server or Managed Server

4. Repeat 1, 2, 3 for each Machine

5. Deploy an Application only on the Admin Server

6. Using the Domain

7. Run all this in reverse Order to remove the Domain from Cloud
Creating one WebLogic Server Machine with OpenStack

1. Install the Machine (VM)
   A. Allocate a Storage Volume
   B. Get an OS Installation Image
   C. Allocate a Network Port
   D. Get a Network Address
   E. Install Machine with A, B, C, D, E
   F. Connect the Machine to C

2. Install the WebLogic Software

3. Configure WebLogic
   A. Domain and Cluster
   B. Node Manager
   C. Admin Server or Managed Server

1. Create an OpenStack Instance
   A. Allocate a cinder Volume
   B. Select a glance Image
   C. Allocate a neutron Port
   D. Allocate a neutron Port Address
   E. Create a nova instance
   F. ... Done by OpenStack ...

2. Pre-Created glance Image (1.B)

3. Execute in OpenStack Instance
   A. A (jython) WLST script configures the WebLogic Server
      based on a domain.properties file
Orchestration in a Cloud

**Prepare**
- Prepare the Component Installation Image
- Add specific Configuration Utilities

**Create**
- Get the Resources in the Cloud
- Install the Stack Components

**Configure**
- Get dynamic Config Data of Application and Cloud Resources
- Use Configuration Utilities to configure Components
Prepare the Glance Image

wls-install.sh

• Prepare the Image after the OS with an installed JDK
• Add the weblogic user
• Install, but do not configure WLS

# Create user and group
create_group $ADMIN_GROUP
create_user $ADMIN_USER $ADMIN_GROUP

... # Install JDK
su - $ADMIN_USER -c "cd $JDK_HOME;tar xzf $SOFTWARE_TMP/$JDK_FILE_NAME"

... # Create response file
sed "s|ORACLE_HOME=|ORACLE_HOME=${WLS_HOME}|g" $RESPONSE_FILE > $SOFTWARE_TMP/response_tmp.rsp

... # Silent WLS installation
su - $ADMIN_USER -c "$JDK_HOME/$JDK_INSTALL_DIR/bin/java -jar $SOFTWARE_TMP/$WLS_INSTALL_JAR \n -silent -invPtrLoc $SOFTWARE_TMP/oraInst.loc -responseFile $SOFTWARE_TMP/response_tmp.rsp"
Create WebLogic Machines using OpenStack Horizon
Further Automation of **Create** and **Configure**

- Automate the creation **and removal** of multiple depending Instances?
  
  A. Use OpenStack CLI with Parameters to create and remove
  
  B. Use Orchestration with Heat and Templates
Agenda

1. Motivation
2. Oracle WebLogic Server Concepts
3. OpenStack Concepts
4. Plain Instance Orchestration
5. OpenStack Orchestration with Heat Templates
6. Software Configuration with OpenStack Metadata
Cloud Orchestration with Heat

- Manages complex Group of connected Cloud Resources
- Tiered Application Deployments spread across multiple VMs
- Provide MetaData and UserData to Nova Instances
- Control Orchestration Flow
- Clean-up the Stack on removal
- Described through Templates
Heat Templates

heat_template_version: 2015-04-30
description: Sample Heat Template

parameters:
... resources:
... outputs:
...

# heat stack-create wls-cluster -f wls-hot.yaml -P 'private_network=my_network'

... parameters:
  private_network:
    type: string
    label: Private Network
    default: Tenant_Network
... resources:
  managed_server:
    type: OS::Nova::Server
    properties:
      image: { get_param: image }
      flavor: { get_param: flavor }
      networks:
        - network: { get_param: private_network }
  managed_floatingip:
    type: OS::Nova::FloatingIP
    depends_on: managed_server
    properties:
      pool: { get_param: external_network }
  managed_association:
    type: OS::Nova::FloatingIPAssociation
    properties:
      floating_ip: { get_resource: managed_floatingip }
      server_id: { get_resource: managed_server }
... outputs:
  Admin_Server_Console_URL:
    description: URL of the Admin Server Console
    value:
      str_replace:
        template: http://$ip:$admin_port/console
        params:
          $ip: { get_attr: [admin_networkport, fixed_ips, 0, ip_address] }
          $admin_port: { get_param: admin_port }
Create a Heat Stack using OpenStack Horizon

1. OpenStack Dashboard
2. Launch Stack
3. Select Template
4. Instances

Admin Password
Admin Listen Port
Admin Username
To Deploy Application Name

Benefits
Created Stack - Details
Next: **Configure** the WebLogic Machines 1/2

**wls-config.py**

- Get Machine specific Configuration Data and create domain.properties File
- Execute WLST Script (jython) to configure a WLS Server Machine

```python
# Configure Admin Server Machine based on domain.properties file
...
create(MACHINE_NAME, 'UnixMachine')
cd('UnixMachine/' + MACHINE_NAME)
create(MACHINE_NAME, 'NodeManager')
cd('NodeManager/' + MACHINE_NAME)
set('ListenAddress', MACHINE_ADDRESS)
set('ListenPort', int(NODEMGR_PORT))
...
cd('/Servers/AdminServer')
set('ListenAddress',ADMIN_ADDRESS)
set('ListenPort',int(ADMIN_PORT))
set('Machine',MACHINE_NAME)
...
cd('/')
create(CLUSTER_NAME,'Cluster')
...

# Write Server template

# Configure Managed Server Machine
...
connect(ADMIN_USERNAME, ADMIN_USERPWD, ADMIN_SERVER_URL)
edit(SERVER_NAME)
startEdit()
cd('/')
cmo.createServer(SERVER_NAME)
cmo.setListenAddress(MANAGED_ADDRESS)
cmo.setListenPort(int(MANAGED_PORT))
...
Next: **Configure** the WebLogic Machines 2/2

**wls-config.py**

- Start NodeManager
- Use Node Manager to start Admin Server or Manager Server

```python
... # Start node manager
output = os.system(DOMAIN_HOME + 'bin/startNodeManager.sh > ' + DOMAIN_HOME + '/nodemanager/nodemanager.out 2>&1 &')
...
# Connect to Node Manager
nmConnect(username=ADMIN_USERNAME, password=ADMIN_USERPWD, host=MACHINE_ADDRESS, port=NODEMGR_PORT,
domainName=DOMAIN_NAME, domainDir=DOMAIN_HOME, nmType='Plain')

# Start Server.
# On AdminServer this is a local start.
# On a Managed Server, this is a controlled-by-AdminServer start.
if ADMIN_SERVER:
    nmStart(serverName=SERVER_NAME, domainDir=DOMAIN_HOME)
else:
    prps = makePropertiesObject("AdminURL=http://" + ADMIN_ADDRESS + ":" + ADMIN_PORT + ";Username=" + ADMIN_USERNAME + ";Password=" + ADMIN_USERPWD)
    nmStart(serverName=SERVER_NAME, domainDir=DOMAIN_HOME, props = prps)

... nmDisconnect()
...```
Follow-up: **Deploy** a WebLogic Server Application

wls-deploy.py

- Deploy a WebLogic Application from the Admin Server to the Cluster
- The Application can be loaded from shared Application Repository or is pre-delivered with the glance Image

```python
# Deploy an app based on domain.properties file
...
ADMIN_SERVER_URL = 't3://' + ADMIN_ADDRESS + ':' + ADMIN_PORT
connect(ADMIN_USERNAME, ADMIN_USERPWD, ADMIN_SERVER_URL)
...
deploy(appName=APPLICATION_NAME, path=APPLICATION_FILE, targets=CLUSTER_NAME, stageMode='stage')
....
disconnect()
```
Further Automation of **Create** and **Configure**

• Automate the creation **and removal** of multiple depending Instances?
  – Use Orchestration with Heat and Templates

• Deliver Domain Properties?
  – Server Type, Application, IP Address, Hostnames, Domain Name, Cluster Name, ...
  A. Pre-Create and share through a Configuration Repository into Tenant Networks
  B. Use MetaData delivered from OpenStack to nova Instances
Agenda

1. Motivation
2. Oracle WebLogic Server Concepts
3. OpenStack Concepts
4. Plain Instance Orchestration
5. OpenStack Orchestration with Heat Templates
6. Software Configuration with OpenStack Metadata
MetaData Concepts in OpenStack

How to provide configuration data from the "Undercloud" to the Instance

Heat Template

```
# curl http://169.254.169.254/openstack/latest/meta_data.json
{...
"uuid": "0c859d0a-adaa-43ad-a0b2-d185c3707081",
"availability_zone": "nova",
"hostname": "wls-cluster-admin-server-vlmwnkg4asu.novalocal",
"launch_index": 0,
"name": "wls-cluster-admin_server-vlmwnkg4asu"
}
```

```
# curl http://169.254.169.254/latest/meta_data
...
block-device-mapping/
hostname
instance-action
instance-id
instance-type
kernel-id
local-hostname
local-ipv4
placement/
public-hostname
public-ipv4
```

```
192.168.200.51
```

```
# curl http://169.254.169.254/openstack/latest/user_data
...
Heat user_data in OS::Nova::Server Resource ...
```
Nova MetaData in Heat Templates

- Put MetaData into Heat Template

```yaml
... resources:  
  managed_server:  
    type: OS::Nova::Server  
    properties:  
      metadata:  
        ...  
        machine.name: wls-adminserver  
        managed.address: { get_attr: [admin_networkport, fixed_ips, 0, ip_address] }  
        managed.port: { get_param: admin_port }  
        image: { get_param: image }  
        flavor: { get_param: flavor }  
        networks:  
          - network: { get_param: private_network }
```

- Read from Instance

```bash
# curl http://169.254.169.254/openstack/latest/meta_data.json
{...
  "hostname": "wls-cluster-admin-server-ymayayjmz7pp.novalocal", ...
  "meta": {...
    "managed.port": "7001", ...
    "machine.name": "wls-adminserver", "managed.address": "192.168.200.54", ...
    "name": "wls-cluster-admin_server-ymayayjmz7pp"}
}
```

- Simple Dump "MetaData" Script

```python
#!/usr/bin/python
import getopt, json, sys, urllib2
...
def main():
  # check script parameters
  META='http://169.254.169.254/openstack/latest/meta_data.json'
  DUMP_FILE=domain.properties
  # Read Metadata
  meta = json.loads(urllib2.urlopen(META).read())
  dmp = open(DUMP_FILE, "w")
  for key in meta['meta']:
    dmp.write(str(key) + '=' + str(meta['meta'][key]) + 'n')
  dmp.close()
...
if __name__ == '__main__':
  main()
```

- Pre-install "Dump MetaData" in the glance Image

```bash
# curl http://169.254.169.254/openstack/latest/meta_data.json
...
```

Copyright © 2016, Oracle and/or its affiliates. All rights reserved. DOAG 2016
Further Automation of **Create** and **Configure**

- Automate the creation **and removal** of multiple depending Instances?
  - Use Orchestration with Heat and Templates

- Deliver Domain Properties?
  - Use MetaData delivered from OpenStack to nova Instances

- Execute Boot Configuration Scripts in Instances
  - Use pre-installed Utility to pull MetaData and User_Data and execute User_Data
  - Alternative: Generate Domain Properties File by a template User_Data Script
Nova User_Data in Heat Templates

• Put user_data into Heat Templates

... resources:
  managed_server:
    type: OS::Nova::Server
    properties:
      image: { get_param: image }
      flavor: { get_param: flavor }
      networks:
        - network: { get_param: private_network }
  user_data_format: RAW
  user_data: |
    str_replace:
      params:
      ...
      $admin_username: { get_param: admin_username }
      ...
      $props: wls-config/domain.properties
      template: |
        #!/bin/sh
        ...
        echo "admin.username=$admin_username" >> $props
        ...
        # Configure and install the Node Manager and the Admin Server
        su - weblogic -c 'cd wls-config;$wls_home/oracle_common/common/bin/wlst.sh wls-cluster.py -p ~weblogic/domain.properties'
        ...

• Read from Instance

  # curl http://169.254.169.254/openstack/latest/user_data
  #!/bin/sh
  ...
  echo "admin.username=$admin_username" >> $props
  ...
  # Configure and install the Node Manager and the Admin Server
  su - weblogic -c /opt/wls/oracle_common/common/bin/wlst.sh wls-cluster.py -p ~weblogic/domain.properties'
  ...

• Pull and execute the boot-script by an Utility, pre-installed in the glance Image, e.g. in Solaris cloudbase-init
Putting it all together: Orchestration with OpenStack

- **Prepare**
  - Prepare the glance Image and the Heat Template

- **Create**
  - Create the Stack with a Heat Template

- **Configure**
  - The instances pull MetaData and execute Heat provided configuration on first boot
Summary

• Create a WLS-Domain in Minutes

• Needed parts:
  – One Image
  – One OpenStack Heat Template File
  – One Heat Parameter File

• OpenStack Heat enables highly agile WLS-Cloud Setups
  – Full Stack Orchestration
  – Full Stack Removal
More information

• openstack.org
  – Heat Documentation
  – Cloudbase-init Documentation

• Oracle Solaris OpenStack
  – Oracle.com/openstack
  – Oracle Technology Network

• Whitepaper: OpenStack Infrastructure
  https://community.oracle.com/docs/DOC-1001147
Do you want to see a Life Demo or try it out?

- Today Live Demo in Demo Kino

  Über den Wolken - OpenStack live

  Thursday, 17.11.2016
  Foyer Tokio
  13:00 - 13:45

- See DOAG 2016 Download Area for the used Scripts to Download
Q & A
Detlef.Drewanz@oracle.com
Integrated Cloud
Applications & Platform Services