County of Los Angeles
Internal Services Department (ISD)

OpenShift Storage Automation with Container-native Storage

Roya Vafadar, IT Specialist
Annette Clewett, Red Hat Storage Sr. Architect
May 10, 2018
County of Los Angeles on Forbes Lists
#133 America’s Best Employers
LACounty Video
Internal Services Department (ISD)

- Centrally support services for LA County Departments
- Provide IT services, including managing the Countywide Data Center facilities, with 1100+ IT professionals
- Objective – To service all citizens of Los Angeles County
- Focus - Provide our customers (County Department’s) with highest quality of service and reliable access to applications, data and technology
- Our Mission - Trusted Partner and Provider of Choice
Internal Services Dept. Supports

- 39 County Departments and supporting agencies who provide services to more than 11 million users
- 3500+ Virtual Machines
- 500+ Websites
We are responsible for establishing and enforcing support orders when a child receives public assistance or when a parent or legal caretaker requests our services. We act in the “public interest” and do not represent any individual. Our services include: opening a case; finding parents; establishing a legal parent-child relationship; establishing child and medical support orders; and enforcing support orders. We provide these services at no or little cost to you.

All issues regarding a child support case can often be resolved without you having to go to court. This is one reason it is important that you stay in contact with our office once a case has been opened.
Las Enfermedades Mentales
Nos Afectan a Todos
Para Ayuda Llame al

1-800-854-7771
24/7 Mental Health Helpline

If you need mental health help, don't wait
Life-Threatening Emergencies
CALL 9-1-1

For other mental health needs:
Get Help 24/7
800-854-7771

LOCATE MENTAL HEALTH SERVICES

#redhat #rhubsummit
LOS ANGELES COUNTY
OFFICE OF
CANNABIS MANAGEMENT

7 things you should know

Los Angeles County
The Los Angeles County HOMELESS INITIATIVE
REAL HELP. LASTING CHANGE.

Los Angeles County Homeless Initiative Impact Dashboard

Thousands of individuals and families have been helped through our dynamic Countywide movement to combat and prevent homelessness. See the significant steps being made to advance our efforts through the Impact Dashboard.

#redhat #rhsummit
Hosting Websites on Mainframe

- Hosting with WebSphere started in 2008
- Challenges
  - Stop all services to apply updates
  - Code Deployment scheduled once a week
  - Onboarding process is typically 1 month
Hosting Websites on VMs

• Hosting with LAMP stack started in 2015

• Challenges

  Need for multiple VMs (Web, DB) for production
  Patching, upgrades of multiple servers
  Cost of other environments (Dev/Test/QA,...)
  Takes minimum of one week to deliver to customer
Why OpenShift?

The need for automation

• Faster provisioning and deployment
• Easy to use
• Quick recovery
• Standardizing images
• Self-service offerings
Speed up Service Delivery with OpenShift

- Consistent Environment
- On-demand Infrastructure
- Automated Build & Deploy
- Delivery Pipeline
- Security First
- Unified Ops Model
Deployment with OpenShift

SCM

OPENSHIFT CI/CD PIPELINE (JENKINS)

IMAGE BUILD & DEPLOY

OPENSHIFT IMAGE REGISTRY

OPENSHIFT CLUSTER

NON-PROD

DEV

OPENSHIFT CLUSTER

PROD
Deployment with OpenShift

SCM

**OPENSHIFT CI/CD PIPELINE (JENKINS)**

**IMAGE BUILD & DEPLOY**

**PROMOTE TO TEST**

**OPENSHIFT IMAGE REGISTRY**

**OPENSHIFT CLUSTER**

- **NON-PROD**
- **DEV**
- **TEST**

**OPENSHIFT CLUSTER**

- **PROD**
Deployment with OpenShift

SCM

OPENSHIFT CI/CD PIPELINE (JENKINS)

IMAGE BUILD & DEPLOY

PROMOTE TO TEST

PROMOTE TO QA

OPENSHIFT IMAGE REGISTRY

OPENSHIFT CLUSTER

NON-PROD

DEV

TEST

QA

PROD

OPENSHIFT CLUSTER

OPENSHIFT IMAGE REGISTRY
Deployment with OpenShift

SCM

OPENSHIFT CI/CD PIPELINE (JENKINS)

IMAGE BUILD & DEPLOY

PROMOTE TO TEST

PROMOTE TO QA

CUSTOMER VALIDATION

GO LIVE?

NON-PROD

DEV TEST QA PROD

OPENSHEET IMAGE REGISTRY

OPENSHEET CLUSTER
Deployment with OpenShift

<table>
<thead>
<tr>
<th>OPENSHIFT CI/CD PIPELINE (JENKINS)</th>
<th>OPENSHIFT IMAGE REGISTRY</th>
<th>OPENSHIFT CLUSTER</th>
<th>CUSTOMER VALIDATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCM -&gt; IMAGE BUILD &amp; DEPLOY</td>
<td>PROMOTE TO TEST</td>
<td>PROMOTE TO QA</td>
<td>GO LIVE?</td>
</tr>
<tr>
<td></td>
<td>PROMOTE TO PROD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NON-PROD</th>
<th>DEV</th>
<th>TEST</th>
<th>QA</th>
<th>PROD</th>
</tr>
</thead>
</table>

#redhat #rhsummit
OpenShift Cluster with NFS Storage

- Manually create/delete NFS exports for each environment (DEV, TEST, QA, PROD)
- Manually create corresponding OpenShift persistent volumes
- Single point of failure
OpenShift Cluster with GlusterFS

- Persistent volumes are automatically created/deleted using GlusterFS
- Define storage class in the OpenShift template with GlusterFS
- High availability
What are we working on?

Using Jenkins and GlusterFS:

• 1-Click Project Provisioning
• More frequent updates
• Blue/Green Deployment
## Stage View

### Average stage times:
(Average full run time: ~1min 52s)

<table>
<thead>
<tr>
<th>Stage</th>
<th>Copy SCM Creds</th>
<th>Creating Application</th>
<th>Sending out email</th>
</tr>
</thead>
<tbody>
<tr>
<td>#80</td>
<td>1s</td>
<td>1min 27s</td>
<td>1s</td>
</tr>
<tr>
<td>#79</td>
<td>730ms</td>
<td>1min 38s</td>
<td>719ms</td>
</tr>
<tr>
<td>#78</td>
<td>905ms</td>
<td>1min 55s</td>
<td>5s</td>
</tr>
<tr>
<td>#80</td>
<td>839ms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

# redhat #rhsummit
## Persistent Volume Claims

<table>
<thead>
<tr>
<th>Name</th>
<th>Status</th>
<th>Capacity</th>
<th>Access Modes</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>mysql-data</td>
<td>Bound to volume pvc-d1f7c8e6cf2-11eb-859a-00595665c07</td>
<td>2 GB</td>
<td>RWX (Read-Write-Many)</td>
<td>2 minutes</td>
</tr>
<tr>
<td>wo-content</td>
<td>Bound to volume pvc-d1f7c8e6cf2-11eb-859a-00595665c07</td>
<td>8 GB</td>
<td>RWX (Read-Write-Many)</td>
<td>2 minutes</td>
</tr>
</tbody>
</table>
Other Resources

- **DEPLOYMENT mysql, #1**
  - 1 pod

- **DEPLOYMENT wordpress-mysql**
  - Builds 1
  - No deployments for wordpress-mysql
Other Resources

DEPLOYMENT
mysql #1

CONTAINER: MYSQL
- Image: rhos/mysql-56-rhel7
- Ports: 3306/TCP

Networking
- SERVICE internal traffic
  - mysql: 3306/TCP (mysql) → 3306

DEPLOYMENT
wordpress-mysql #1

CONTAINER: WORDPRESS-MYSQL
- Image: summit-deno-qa/wordpress-mysql 38fc492 506.6 MB
- Build: wordpress-mysql #1
- Source: Merge tag '49.5' of https://github.com/Wordress/Wordress into 4.3.5-branch, 3Tb4d75
- Ports: 8040/TCP

Networking

1 pod

2 pods
Build #80 (Apr 20, 2018 9:26:52 PM)

Started by user

This run spent 1 ms waiting in the queue.
Your WordPress instance is ready!

You can access your instance here: http://summit-demo/wp-login.php

username: admin
password: 4yqklghlg4uiy4IH)&*

Once logged in, you can import your site using the all-in-one-migration plugin tool.
WordPress 4.9.5 is available! Please update now.

Welcome to WordPress!
We've assembled some links to get you started:

Get Started
- Customize Your Site
  or, change your theme completely

Next Steps
- Write your first blog post
- Add an About page
- View your site

More Actions
- Manage widgets or menus
- Turn comments on or off
- Learn more about getting started

At a Glance
- 1 Post
- 1 Page
- 1 Comment

Quick Draft

Title

What's on your mind?

#redhat #rhsummit
Help those who cannot speak for themselves.

Report animal cruelty
Website Deployment

- OpenShift: 15 Days
- LAMP: 20 Days
- Mainframe: 25 Days

Days
OpenShift Needs Two Kinds of Storage

**Persistent Storage**

**Ephemeral Storage**

**OCP Infrastructure**
- Registry
- Metrics
- Logging

**Stateful Applications**

**Stateless Applications**
- pod image storage
OpenShift Persistent Storage Framework

A request for storage
Provider: ABC
Capacity: 10 GiB
Features: XYZ

DEVELOPER

"submits"

PersistentVolumeClaim

“A request for storage”
Provider: ABC
Capacity: 10 GiB
Features: XYZ
OpenShift Persistent Storage Framework

DEVELOPER

"submits"

PersistentVolumeClaim

"submits to"

StorageClass

“sets up”

OPERATIONS

“A request for storage”
Provider: ABC
Capacity: 10 GiB
Features: XYZ

“A provider of storage”
Provider URL: …
Credentials: …
Options: …
OpenShift Persistent Storage Framework

- **PersistentVolumeClaim**: A request for storage
  - Provider: ABC
  - Capacity: 10 GiB
  - Features: XYZ

- **StorageClass**: A provider of storage
  - Provider URL: ...
  - Credentials: ...
  - Options: ...

- **Storage Backend**: Instruction

**Operations**
- “sets up”
- “submitted to”
- “instructs”

**Developer**
- “submits”
OpenShift Persistent Storage Framework

Developer submits a request for storage:
- Provider: ABC
- Capacity: 10 GiB
- Features: XYZ

Operations set up a PersistentVolumeClaim:
- "A request for storage"

The storage provider is provisioned:
- Provider URL: ...
- Credentials: ...
- Options: ...

A PersistentVolume is created:
- "Provisioned Storage"
  - Name: ...
  - Size: ...
  - AccessMode: ...

Storage Backend is instructed to provision the PersistentVolume.
OpenShift Persistent Storage Framework

OPERATIONS

DEVELOPER

"submits"

PersistentVolumeClaim

“A request for storage”
Provider: ABC
Capacity: 10 GiB
Features: XYZ

“submitted to”

StorageClass

“A provider of storage”
Provider URL: …
Credentials: …
Options: …

“creates”

PersistentVolume

“Provisioned Storage”
Name: …
Size: …
AccessMode: …

“mounted by”

APPLICATION POD(S)

Storage Backend

“instructs”

“provisions”

APPLICATION POD(S)
Container Native Storage (CNS) with OpenShift

Storage Class Examples:
- Fast Pool = 3x OCP nodes with SSDs
- Slow Pool = 3x OCP nodes with HDDs

Pods on OCP nodes without local storage can mount volumes from Fast and Slow Storage Classes.
How OpenShift Requests Persistent Volume

Steps:
• OpenShift calls Heketi through RESTful API found in Storage Class object
• Heketi provisions volume on Gluster
• 3x Gluster PODs in CNS on 3x OCP nodes
• 3x minimum due to 3-way replication for Gluster volumes
Storage Class Use And Definition

Steps:
• Done after CNS or CRS cluster is created
• API call using Heketi route, Heketi clusterID and credentials

Example: OpenShift Storage Class YAML File
```yaml
# cat cns-fast-storageclass.yaml
apiVersion: storage.k8s.io/v1beta1
kind: StorageClass
metadata:
  name: cns01-vmdk-gluster-fast
provisioner:
  kubernetes.io/glusterfs
parameters:
  resturl:
    http://heketi-storage.apps.syseng.com
  clusterid: d0a035dc9022343480fcb0ec9de307
  restauthenabled: "true"
  restuser: "admin"
  secretNamespace: "default"
  secretName: "heketi-secret"
```

GlusterFS is POSIX-compliant distributed file system.

GlusterFS is user-space software running on Linux that takes local filesystem directories of standard servers and federates them in a pool across the network, to make them accessible to clients from a single mountpoint.
NFS versus CNS Storage Demo
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
facebook.com/redhatinc
twitter.com/RedHat
youtube.com/user/RedHatVideos